



ULPGC
Universidad de
Las Palmas de
Gran Canaria

Escuela de
Ingeniería Informática



Diseño y despliegue de una infraestructura de clúster que ofrece un servicio web basado en Apache en alta disponibilidad y con balanceo de carga en Fedora

Grado en Ingeniería Informática

Ariadna Del Pino Domínguez

TUTORIZADO POR:
Francisco Alexis Quesada Arencibia
Carmelo Rubén García Rodríguez

Julio 2022

Agradecimientos

En esta instancia, quisiera agradecer a todas las personas que me han prestado su ayuda durante todo el recorrido hacia mi titulación.

A mi familia y amigos. Sin ellos, quienes me han apoyado, acompañado y aguantado durante todo este camino, este trabajo no sería lo mismo.

A mis tutores, Dr. Alexis Quesada y Dr. Rubén García, por todo lo que me han enseñado durante este proceso y por su seguimiento constante en el desarrollo de este proyecto. Gracias por haber creído en mí.

A los compañeros del Instituto Universitario de Ciencias y Tecnologías Cibernéticas por su acogida, ayuda y atención durante mi alojamiento en el centro.

Muchas gracias de corazón.

Resumen

Tras el comunicado de la discontinuidad de la distribución de Linux CentOS a partir de la versión 8, en este proyecto se analizan las posibilidades que tiene *Fedora Workstation 35* para el diseño y despliegue de una infraestructura de clúster que ofrezca un servicio web basado en Apache y que haga uso de una base de datos.

Una vez realizado el estudio, se desarrolla dicha infraestructura en la que, tanto el servicio web como la base de datos, puede proporcionar alta disponibilidad y balanceo de carga a la hora de atender las peticiones que puedan realizar sus clientes. El servicio web que se ofrece es el CMS (sistema de gestión de contenidos) de *WordPress*.

El clúster también dispondrá de un espacio de almacenamiento compartido y distribuido basado en la tecnología iSCSI para almacenar los ficheros comunes que tendrán los nodos (equipos) que forman el clúster. El software que se ha utilizado para el desarrollo del proyecto es de distribución libre.

Palabras clave: Infraestructura de clúster, Apache, alta disponibilidad, balanceo de carga, *Wordpress* e iSCSI.

Abstract

After the announcement of the discontinuity of the Linux distribution CentOS from version 8 onwards, this project analyses the possibilities that Fedora Workstation 35 has for the design and use of a cluster infrastructure that offers a web service based on Apache and that make use of a database.

Once the study has been carried out, this infrastructure is developed in which both the web service and the database can provide high availability and load balancing when it comes to attending the requests made by its clients. The web service offered is the WordPress CMS (content management system).

The cluster will also have a shared and distributed storage space based on iSCSI technology to store the common files that the cluster nodes (computers) will have. The software that has been used for the development of the project is freely distributed.

Keywords: cluster infrastructure, Apache, high availability, load balancing, WordPress and iSCSI.

Índice general

1. Introducción	1
2. Estado del arte y objetivos iniciales	4
2.1. Estado del arte	4
2.2. Objetivos iniciales	5
3. Competencias específicas y aportaciones del trabajo	6
3.1. Competencias propias del título	6
3.2. Aportaciones del trabajo	9
4. Normativa y legislación	10
4.1. Licencia Apache	10
4.2. Licencia GNU GPL	10
4.3. Licencia GNU LGPL	11
4.4. Licencia pública de Mozilla	11
4.5. Licencia PHP	11
5. Metodología y planificación	12
6. Herramientas y tecnologías utilizadas	14
7. Desarrollo	18
7.1. Análisis	18
7.1.1. Requerimientos del sistema para la virtualización	18
7.1.2. Requerimientos del servicio WordPress	19
7.1.3. Requerimientos del servicio de base de datos <i>MariaDB</i>	20
7.1.4. Requerimientos de Fedora para la computación en clúster	20
7.2. Acondicionamiento del sistema	21
7.2.1. Preparación de la interfaz gráfica	21
7.2.2. Configuraciones y operaciones	22
7.2.2.1. Actualización del sistema	22
7.2.2.2. Estado de la utilidad <i>NetworkManager</i>	22
7.2.2.3. Instalación de la plataforma de virtualización	24
7.2.2.4. Operaciones con máquinas virtuales	24

7.2.2.5.	Utilización de recursos de almacenamiento virtual	24
7.2.2.6.	Infraestructura de red virtual	25
7.3.	Diseño e implementación del clúster	26
7.3.1.	Diseño del clúster	26
7.3.2.	Implementación del clúster	28
7.3.2.1.	Creación de la infraestructura básica del clúster	28
7.3.2.2.	Instalación y configuración del servidor Apache	45
7.3.2.3.	Instalación y configuración del HAProxy	48
7.3.2.4.	Instalación y configuración del software <i>High Availability Add-On</i>	52
7.3.2.5.	Creación y configuración del clúster	54
7.3.2.6.	Establecimiento de un mecanismo de aislamiento de nodos	56
7.3.2.7.	Creación del recurso HAProxy	61
7.3.2.8.	Instalación y configuración de MariaDB Galera Cluster	63
7.3.2.9.	Inclusión de MariaDB Galera Cluster y Estadísticas del HAProxy en el HAProxy	67
7.3.2.10.	Instalación y activación del cliente de MariaDB	72
7.3.2.11.	Configuración del espacio de almacenamiento compartido y distribuido	72
7.3.2.12.	Instalación y configuración del servidor LAMP	81
7.3.2.13.	Instalación y configuración de <i>WordPress</i>	81
7.3.2.14.	Creación y configuración de volúmenes lógicos	87
7.4.	Evaluación	94
8.	Conclusiones y trabajo futuro	99
	Bibliografía	106
	Anexo	107
	ANEXO I: Comprobación de los requerimientos del sistema anfitrión	107
	ANEXO II: Instalación de la herramienta <i>Gnome Tweaks</i>	110
	ANEXO III: Instalación del paquete de virtualización KVM	112
	ANEXO IV: Creación de una máquina virtual mediante la utilidad <i>virt-manager</i>	115
	ANEXO V: Creación de máquinas virtuales mediante distintas utilidades	123
	ANEXO VI: Creación de recursos de almacenamiento para las máquinas virtuales	131
	ANEXO VII: Creación de una red en modo <i>bridge</i>	152
	ANEXO VIII: Instalación y configuración de la pila LAMP	156
	ANEXO IX: Instalación de <i>WordPress</i>	159

Índice de figuras

7.1. Estado del NetworkManager	23
7.2. Infraestructura del clúster	27
7.3. Creación de la red aislada de control del clúster	30
7.4. Creación de la red aislada de almacenamiento del clúster	31
7.5. Resumen de todas las redes	31
7.6. Paso 3 - Ajustes de la memoria RAM y la CPU	32
7.7. Paso 4 - Ajustes de la imagen de disco	33
7.8. Paso 5 - Agregar nombre y seleccionar la red	33
7.9. Clonación de la máquina base	34
7.10. Asignación de la red de Almacenamiento al Nodo1-servicio	35
7.11. Comprobación de las redes asignadas al Nodo1-servicio	36
7.12. Configuración de la interfaz enp7s0 en el Nodo1-servicio	37
7.13. Configuración de la interfaz enp8s0 en el Nodo1-servicio	38
7.14. Verificación de las interfaces enp7s0 y enp8s0 en el Nodo1-servicio	39
7.15. Verificación de las direcciones IP del Nodo1-servicio	39
7.16. Demostración de la conexión de las redes aisladas del Nodo1-servicio	40
7.17. Demostración de la conexión de la interfaz enp7s0 del Nodo1-servicio	40
7.18. Demostración de la conexión de la interfaz enp8s0 del Nodo1-servicio	41
7.19. Demostración de la conectividad del anfitrión mediante la interfaz enp7s0 al Nodo1-servicio	41
7.20. Demostración de la conectividad del anfitrión mediante la interfaz enp8s0 al Nodo1-servicio	41
7.21. Verificación de la conectividad entre las máquinas de la red de Almacenamiento	42
7.22. Verificación de la conectividad entre las máquinas de la red de Control	42
7.23. Modificación del fichero /etc/hosts del “ Nodo1-servicio ”	44
7.24. Modificación del fichero /etc/hosts de la máquina iSCSI-almacenamiento	45
7.25. Listado del <i>firewall</i> por defecto en el “ Nodo1-servicio ”	46
7.26. Eliminación del rango de puertos por defecto del <i>firewall</i> en el Nodo1-servicio	47
7.27. Añadiendo el servicio Apache al <i>firewall</i> en el Nodo1-servicio	47
7.28. Verificación del funcionamiento de Apache en el anfitrión	48
7.29. Modificación del fichero haproxy.cfg para agregar HTTP	49
7.30. Abriendo el puerto del HAProxy al <i>firewall</i> en el Balanceador-de-carga	50
7.31. Verificación del correcto funcionamiento de Apache	51
7.32. Creación del fichero index.html en el Nodo1-servicio	51

7.33. Creación del fichero index.html en el Nodo2-servicio	51
7.34. Verificación del balanceo de carga con el Nodo1-servicio	52
7.35. Verificación del balanceo de carga con el Nodo2-servicio	52
7.36. Verificación del balanceo de carga con el nombre de dominio	52
7.37. Listado del <i>firewall</i> de todos los nodos	53
7.38. Establecimiento de la contraseña del usuario hacluster	54
7.39. Autenticación del usuario “hacluster” en todos los nodos del clúster	54
7.40. Habilitando los servicios del clúster	55
7.41. Estado del clúster	55
7.42. Primera parte de la generación del fichero fence_virt.conf	57
7.43. Segunda parte de la generación del fichero fence_virt.conf	58
7.44. Creación del fichero que contiene la clave compartida	59
7.45. Copia del fichero que contiene la clave compartida en los nodos	60
7.46. Comprobación de la conectividad <i>multicast</i>	60
7.47. Estado del fencing y del clúster	61
7.48. Estado del clúster y su recurso	62
7.49. Estado del clúster tras parar el nodo Balancedor-de-carga	63
7.50. Primera parte del fichero galera.cnf	64
7.51. Segunda parte del fichero galera.cnf	65
7.52. Verificación del tamaño del clúster	66
7.53. Probando la replicación de la base de datos	66
7.54. Modificación del fichero haproxy.cfg para agregar Galera Cluster	67
7.55. Modificación del fichero haproxy.cfg para agregar las estadísticas de HAProxy	67
7.56. Comprobación del booleano de SELinux	68
7.57. Prueba de balanceo de carga de la base de datos	70
7.58. Autenticación en la página de estadísticas	70
7.59. Prueba de la página de estadísticas de HAProxy	71
7.60. Creación y asociación de un nuevo volumen en iSCSI-almacenamiento	73
7.61. Verificación del nuevo volumen	73
7.62. Creación de un LVM en iSCSI-almacenamiento	75
7.63. Verificación del nuevo volumen con LVM en iSCSI-almacenamiento	75
7.64. Cambio del nombre del <i>initiator</i>	75
7.65. Pasos para crear el almacenamiento compartido	76
7.66. Verificación del almacenamiento compartido recién creado	77
7.67. Descubriendo el <i>target</i> en todos los nodos	78
7.68. Iniciando sesión en todos los nodos	78
7.69. Comprobación del disco en todos los nodos	79
7.70. Creación del volumen físico y el grupo de volúmenes	79
7.71. Comprobación del volumen físico y el grupo de volúmenes creados	80
7.72. Modificación del parámetro use_lvmlckd del fichero lvm.conf	80
7.73. Verificación de la carpeta descomprimida de WordPress	82
7.74. Cambio de permisos de la carpeta wordpress	83
7.75. Modificación del fichero wordpress.conf	83
7.76. Modificación del fichero wordpress.conf	83
7.77. Contextos de SELinux en la carpeta wordpress	84

7.78. Booleano de SELinux activado	84
7.79. Verificación de la réplica de la base de datos	85
7.80. Modificación del tema de <i>WordPress</i>	86
7.81. Verificación del Blog TFTP en el Nodo1-servicio	86
7.82. Verificación del Blog TFTP en el Nodo2-servicio	87
7.83. Creación del volumen lógico para el contenido Apache	87
7.84. Comprobación del volumen lógico para el contenido Apache	87
7.85. Formateando el volumen lógico para el contenido Apache	88
7.86. Montaje del dispositivo de almacenamiento	90
7.87. Verificación de la replicación del contenido	90
7.88. Cambio de contextos	91
7.89. Enlaces simbólicos	91
7.90. Montaje del dispositivo de almacenamiento 2	92
7.91. Aplicando contextos	93
7.92. Verificación de los enlaces simbólicos	94
7.93. Primera prueba de escritura en la base de datos	95
7.94. Verificación de la primera prueba de escritura en la base de datos	95
7.95. Segunda prueba de escritura en la base de datos	96
7.96. Verificación de la segunda prueba de escritura en la base de datos	96
7.97. Verificación del balanceo de carga del servicio	97
7.98. Verificación del balanceo de carga del servicio después de recargar	97
7.99. Verificación del balanceo de carga de la base de datos	98
1. Verificación del espacio libre en disco del sistema anfitrión	107
2. Verificación de la RAM del sistema anfitrión	107
3. Verificación de la información de la CPU del sistema anfitrión	108
4. Verificación de las extensiones de la CPU del sistema anfitrión	109
5. Activación de la extensión “ <i>Dash to Panel</i> ”	110
6. Activación de los botones de la barra de título	111
7. Paquetes de virtualización instalados en el sistema anfitrión	112
8. Estado del servicio libvirtd	113
9. Comprobación de los módulos del kernel KVM	113
10. Interfaz gráfica de KVM (<i>virt-manager</i>)	114
11. Crear nueva máquina virtual con (<i>virt-manager</i>)	115
12. Paso 1 - Selección del medio de instalación	116
13. Paso 2 - Selección de la imagen ISO	116
14. Paso 3 - Ajustes de la memoria y CPU	117
15. Paso 4 - Ajustes de la imagen de disco	117
16. Paso 5 - Establecimiento del nombre de la máquina	118
17. Elección del método de uso del sistema operativo	118
18. Paso 1 de la instalación - Selección de idioma	119
19. Paso 2 de la instalación - Ajustes generales	120
20. Examinar la dirección IP asignada a la máquina	121
21. Ping desde la máquina virtual	122
22. Ping desde el host anfitrión hacia la máquina virtual	122

23.	Búsqueda de ficheros de configuración XML y de disco de la máquina virtual	124
24.	Copia de los ficheros principales de la máquina virtual	124
25.	Modificación del nombre de la máquina virtual en el fichero XML	125
26.	Modificación de la ruta del disco de la máquina virtual en el fichero XML . .	125
27.	Eliminación de la MAC de la máquina virtual en el fichero XML	125
28.	Creación de la máquina virtual a partir del fichero XML	126
29.	Iniciar la máquina virtual	126
30.	Comprobación de la generación automática del UUID	127
31.	Comprobación de la generación automática de la MAC	127
32.	Clonar la máquina virtual con <i>virt-manager</i>	128
33.	Cambiar nombre de la máquina virtual a clonar	128
34.	Clonar la máquina virtual con <i>virt-clone</i>	129
35.	Comprobación de la máquina virtual creada con <i>virt-clone</i>	129
36.	Creación de la máquina virtual con <i>virt-install</i>	130
37.	Creación del volumen en el contenedor por defecto	132
38.	Comprobación de los volúmenes asignados a la máquina virtual	132
39.	Examinar los discos y particiones	133
40.	Paso 1 - Crear partición en el nuevo volumen	134
41.	Verificación de la partición creada en el nuevo volumen	134
42.	Paso 2 - Agregar formato a la partición del nuevo volumen	134
43.	Paso 3 - Crear y montar nuevo directorio	135
44.	Verificación de la partición creada, su formato y su punto de montaje	135
45.	Paso 4 - Creación del fichero en el nuevo volumen	135
46.	Creación y asignación de un nuevo volumen a una nueva máquina virtual . .	136
47.	Creación de la partición extendida en el sistema anfitrión	137
48.	Creación de la partición lógica en el sistema anfitrión	138
49.	Creación del fichero de configuración del disco	139
50.	Asociación del disco a la máquina virtual	139
51.	Verificación de la nueva partición en el sistema anfitrión	140
52.	Agregar formato ext4 a la partición en el sistema anfitrión	140
53.	Crear un nuevo contenedor sobre una partición lógica	141
54.	Creación de un nuevo volumen en el Contenedor_Particion	142
55.	Añadir el nuevo volumen al contenedor	142
56.	Asociar el volumen a la máquina virtual de prueba	143
57.	Elaboración del fichero XML para la creación del contenedor	144
58.	Definir, construir e iniciar el Contenedor_Particion_Virsh	145
59.	Estado del servicio nfs-server	146
60.	Estado del servicio rpcbind	146
61.	Añadir los servicios necesarios para utilizar NFS como cliente	146
62.	Verificación del contenido del directorio <i>/images</i> en el simulador NFS	147
63.	Creación del contenedor de tipo NFS	148
64.	Verificación del contenido del contenedor	149
65.	Creación del segundo contenedor de tipo NFS	150
66.	Verificación del segundo contenedor de tipo NFS	151
67.	Creación de un volumen en el segundo contenedor NFS	151

68.	Verificación del volumen en la máquina del simulador NFS	151
69.	Creación de una interfaz de red tipo puente	152
70.	Configuración de la interfaz de red tipo puente	152
71.	Activación de la red y verificación del estado	153
72.	Configuración de otros ajustes	153
73.	Verificación de la interfaz <i>bridge</i>	153
74.	Verificación de la conexión a internet con la <i>bridge</i>	154
75.	Agregando la <i>bridge</i> a la máquina virtual	154
76.	Verificación de la dirección IP en la máquina virtual	155
77.	Verificación de la conectividad de la máquina hacia el exterior	155
78.	Versión de PHP instalado	157
79.	Información del fichero info.php	157
80.	Comprobación del servidor LAMP	157
81.	Primera parte de la modificación del fichero ini.php	158
82.	Segunda parte de la modificación del fichero ini.php	158
83.	Página inicial de configuración de WordPress	159
84.	Paso 1 - Formulario sobre la conexión de la base de datos	160
85.	Paso 2 - Finalización de la primera parte de la instalación	160
86.	Paso 3 - Configuración de la información del sitio web	161
87.	Paso 4 - Finalización de la instalación de <i>WordPress</i>	161
88.	Paso 5 - Página de inicio de sesión de <i>WordPress</i>	162
89.	Página oficial de administración de <i>WordPress</i>	162

Índice de cuadros

5.1. Planificación inicial	13
7.1. Resumen de las interfaces de red de cada máquina virtual.	43

Capítulo 1

Introducción

“La fuerza no proviene de la capacidad física sino de una voluntad indomable.”

Mahatma Gandhi

Una vez notificado el nuevo enfoque de *CentOS Project* en la versión 8 de *CentOS Linux* (bifurcación de la distribución *Red Hat Enterprise Linux*), en otras palabras, la remodelación de *Red Hat Enterprise Linux* (RHEL) a *CentOS Stream* [1], se pretende realizar una investigación sobre la distribución de **Fedora Workstation 35** con el propósito de diseñar y desplegar una infraestructura de clúster de manera virtualizada.

Una infraestructura de clúster es un conjunto de servidores que están unidos entre sí por redes de alta velocidad y que trabajan juntos, llegando a comportarse como un único servidor. Cada uno de los servidores se configuran para realizar una misma tarea (normalmente ofrecer un servicio), siendo controlada y gestionada por un *software* para mejorar el rendimiento respecto a un sistema.

Uno de los elementos principales de un clúster son los **nodos**, es decir, aquellos equipos o servidores que están interconectados entre sí mediante una **infraestructura de red** de alta velocidad, comúnmente por medio de redes de área local (LAN). Por lo general, cada nodo posee un **hardware** similar y ejecuta el mismo **sistema operativo**. Sin embargo, existen clústeres que cuentan con nodos que disponen de *hardware* diferente y que ejecutan distintos sistemas operativos.

Otros componentes importantes para que un clúster funcione son los **protocolos de comunicación y servicios** (TCP/IP, HTTP, etc.), las **aplicaciones**, el **almacenamiento** y el **Middleware**. El *Middleware* es un software que trabaja entre el sistema operativo y las aplicaciones del clúster para permitir que los usuarios tengan control sobre el clúster. El almacenamiento puede ser el interno del servidor, una NAS (*Network Attached Storage*, en español, Almacenamiento conectado a red) o una SAN (*Storage Area Network*, en español, Red de área de almacenamiento).

Este proyecto comprende el diseño y despliegue de una **infraestructura de clúster** que

ofrece un **servicio web** basado en **Apache** y que hace uso de una base de datos. Apache es un servidor web HTTP que permite ejecutar sitios web. El servicio web seleccionado para ofrecer es **WordPress**, un sistema de gestión de contenidos (CMS) enfocado en la creación y gestión de sitios webs. Además, se proporciona **alta disponibilidad y balanceo de carga** para el servicio web y para la base de datos.

Con la alta disponibilidad se procura que el servicio esté accesible la mayor parte del tiempo y con el balance de carga, aparte de proporcionar alta disponibilidad, se pretende repartir la carga de trabajo de los nodos que ofrecen dichos servicios para garantizar la calidad de servicio en tiempo/respuesta.

El tipo de almacenamiento que tendrá el clúster es un SAN, pues se construye un espacio de almacenamiento compartido y distribuido basado en tecnología **iSCSI** para almacenar todo el contenido común y generado en el clúster. Para proporcionar este tipo de almacenamiento, se requiere hacer uso del sistema de archivos **GFS2** (*Global File System 2*).

El desarrollo de este proyecto se lleva a cabo utilizando software de distribución libre. Para simular dicha infraestructura de clúster, se utiliza un software de virtualización integrado en Linux, **KVM** (*Kernel-based Virtual Machine*). Esta tecnología permite ejecutar máquinas virtuales de diferentes sistemas operativos. Cada una de ellas posee un hardware virtualizado específico (tarjetas de red, dispositivos de almacenamiento, tarjetas gráficas, etc.).

Las **máquinas virtuales** que tendrán el sistema operativo en el que se basa este proyecto, Fedora Workstation 35, actuarán como los **nodos del clúster**. El sistema donde se instala dicha tecnología se denomina “sistema anfitrión” o “host anfitrión”.

En relación con la **estructura del documento**, se encuentra organizado en siete capítulos donde se detalla toda la información contemplada durante el desarrollo del proyecto. Esta información se descompone tal y como se precisa a continuación:

★ **Capítulo 1 - Introducción**

Este capítulo contiene la introducción del trabajo donde se pone en contexto el proyecto; se detallan los conceptos básicos y la motivación del desarrollo del trabajo, se exponen los objetivos que se deben alcanzar y finalmente, se describe la estructura del documento.

★ **Capítulo 2 - Estado actual y objetivos iniciales**

En este capítulo se describe el estado actual de la técnica del trabajo desarrollado y se desglosan los objetivos inicialmente programados.

★ **Capítulo 3 - Competencias específicas y aportaciones del trabajo**

El tercer capítulo enumera y justifica las competencias generales y específicas de la titulación durante el desarrollo del trabajo por medio de las diferentes operaciones y configuraciones realizadas. Además, se identifican las diferentes aportaciones del proyecto en distintos

ámbitos.

★ **Capítulo 4 - Normativa y legislación**

En el cuarto capítulo se describen las licencias correspondientes de las herramientas y tecnologías utilizadas.

★ **Capítulo 5 - Metodología y planificación**

Este capítulo especifica la metodología y la planificación de trabajo utilizada, explicando detalladamente las diferentes tareas llevadas a cabo.

★ **Capítulo 6 - Herramientas y tecnologías utilizadas**

En el quinto capítulo se recopilan las herramientas y tecnologías utilizadas para llevar a cabo el desarrollo del proyecto, definiendo y justificando su uso.

★ **Capítulo 7 - Desarrollo**

En este capítulo se desarrolla detalladamente cada una de las fases y tareas realizadas para diseñar e implementar la infraestructura de clúster. Se descompone en el análisis de los requerimientos, el acondicionamiento del sistema anfitrión, el diseño e implementación del clúster y la evaluación final de la infraestructura de clúster.

★ **Capítulo 8 - Conclusiones y trabajo futuro**

En este último capítulo se exponen las conclusiones que se han obtenido tras el desarrollo del proyecto y las posibles mejoras de la infraestructura de clúster para un futuro desarrollo.

★ **Bibliografía**

En esta sección se describen todas las fuentes de información utilizadas para apoyar el desarrollo del proyecto.

★ **Anexos**

En este apartado se incluyen todas las actividades realizadas para complementar el proyecto.

Capítulo 2

Estado del arte y objetivos iniciales

2.1. Estado del arte

En la actualidad, debido a la alta demanda de servicios y comercios en internet, es indiscutible que los sistemas que se encargan de ofrecerlos deban funcionar de manera ininterrumpida, inmediata e íntegra la mayor parte del tiempo. Estas características las ofrece los sistemas clústeres, unos sistemas distribuidos y conectados entre sí a través de redes que trabajan juntos para proveer un servicio con un mayor rendimiento que el uso de sistemas usuales.

Existen diferentes tipos de clúster en función del objetivo perseguido. Si se necesita un gran procesamiento de datos, es decir, un máximo rendimiento posible del servicio, se utiliza el clúster de **Alto rendimiento**. Si se requiere confiabilidad y disponibilidad la mayor parte del tiempo (evitar caídas y paradas), se diseña un clúster de **Alta disponibilidad**. Si el servicio ofrecido tiene una gran demanda de tráfico, se hace uso del **Balanceo de carga**, el cual permite que un conjunto de servidores del clúster balancee la carga de trabajo y de tráfico del servicio que ofrece a sus clientes [2].

Estos sistemas de clústeres son cada vez más usados por las empresas, no solo por las que ofrecen servicios webs, sino por todas aquellas que también necesitan un sistema de almacenamiento más seguro para acceder desde cualquier lugar (**Clúster en la nube**).

También existen clústeres según la **configuración** del nodo, el **número** de nodos y el **campo** donde se apliquen. Con respecto a la configuración del nodo, se denominan homogéneos si todos los nodos ejecutan el mismo sistema operativo y heterogéneos si ejecutan diferentes sistemas operativos [2]. Según el número de nodos y el campo donde se apliquen, se denominan: departamentales, corporativos, nacionales, internacionales, comerciales, gubernamentales, científicos, etc.

Las ventajas de estos sistemas son indudables, pues ofrece disponibilidad, confiabilidad, flexibilidad, escalabilidad (ya que los clústeres permiten agregar nodos a medida que vaya siendo necesario), incremento de velocidad de procesamiento y del número de transacciones

y una seguridad algo mejorada.

2.2. Objetivos iniciales

El principal objetivo de este proyecto es poner en funcionamiento un servicio web apoyado en Apache que proporcione sus servicios en alta disponibilidad, es decir, que se ejecute en varios sistemas para evitar la caída del servicio por cualquier tipo de fallo en los sistemas que se ejecutan, y que además, las peticiones realizadas por los clientes se atiendan de manera balanceada por los diferentes equipos, garantizando tiempos de respuesta adecuados para este tipo de servicio.

Para lograr este objetivo principal, se han planteado los siguientes objetivos básicos que se deben alcanzar:

- Identificar los elementos, hardware y software requeridos para proporcionar un servicio web basado en Apache en alta disponibilidad y con tiempos de respuesta adecuados y uniformes.
- Diseñar un clúster para soportar la alta disponibilidad y el balanceo de carga.
- Instalar los elementos de infraestructura requeridos para desplegar el clúster.
- Instalar los componentes software del sistema que proporcionen un clúster en alta disponibilidad y con balanceo de carga.
- Instalación del servicio de base de datos y del servicio web basado en Apache en alta disponibilidad y con balanceo de carga.

Capítulo 3

Competencias específicas y aportaciones del trabajo

Las competencias que adquieren los estudiantes durante el estudio del Grado en Ingeniería Informática son las competencias generales recogidas según el Real Decreto 1393/2007, las competencias nucleares de la Universidad de Las Palmas de Gran Canaria (ULPGC) y las competencias propias del título. En estas se comprenden las competencias en formación básica, de trabajo de título, las comunes a la Ingeniería Informática y las específicas en cada una de las intensificaciones [3].

Las competencias cubiertas en este proyecto son: FB04, CII05, CII011, CII012, CII013, TI01, TI02, TI04, TI05, TI06 y TI07. A continuación, se desglosan y se justifican todas por categoría:

3.1. Competencias propias del título

1. Formación básica (FB)

FB04. “Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación de la ingeniería.”

Esta competencia se cubrió con el uso del ordenador para realizar dicho proyecto, con la gestión e instalación del sistema operativo de Fedora Workstation 35 y con el uso y administración de bases de datos en MariaDB.

2. Trabajo fin de grado

TFG01. “Ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas

de la Ingeniería en Informática de naturaleza profesional, en el que se sinteticen e integren las competencias adquiridas en las enseñanzas.”

Esta competencia se ha adquirido mediante la realización y defensa del proyecto individual.

3. Común a la Ingeniería Informática (CII)

CII05. “Conocimiento, administración y mantenimiento sistemas, servicios y aplicaciones informáticas.”

Esta competencia se justifica con el estudio, administración y mantenimiento del sistema y servicios utilizados para realizar el proyecto.

CII011. “Conocimiento y aplicación de las características, funcionalidades y estructura de los Sistemas Distribuidos, las Redes de Computadores e Internet y diseñar e implementar aplicaciones basadas en ellas.”

Esta competencia se cubre con el análisis de los sistemas distribuidos para diseñar y desplegar la infraestructura de clúster y, con la creación e implementación de las diferentes redes que interconectan los nodos del clúster.

CII012. “Conocimiento y aplicación de las características, funcionalidades y estructura de las bases de datos, que permitan su adecuado uso, y el diseño y el análisis e implementación de aplicaciones basadas en ellos.”

Esta competencia se justifica con la gestión de las bases de datos para permitir un uso adecuado de ellas y poder implementarlas en *WordPress*.

CII013. “Conocimiento y aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a los Sistemas de información, incluidos los basados en web.”

Esta competencia se cubre con la investigación y puesta en marcha de herramientas pertinentes para el almacenamiento de los archivos comunes de los nodos del clúster.

4. Tecnologías de la Información (TI)

TI01. “Capacidad para comprender el entorno de una organización y sus necesidades en el ámbito de las tecnologías de la información y las comunicaciones.”

Esta competencia se justifica con el estudio de los diferentes sistemas y servicios necesarios que requiere el diseño y despliegue de la infraestructura de clúster.

TI02. “Capacidad para seleccionar, diseñar, desplegar, integrar, evaluar, construir, gestionar, explotar y mantener las tecnologías de hardware, software y redes, dentro de los parámetros de coste y calidad adecuados.”

Esta competencia se cubre con el desarrollo del proyecto mediante los recursos proporcionados por el IUCTC (Instituto Universitario de Ciencias y Tecnologías Cibernéticas) y las herramientas y tecnologías de software libre.

TI04. “Capacidad para seleccionar, diseñar, desplegar, integrar y gestionar redes e infraestructuras de comunicaciones en una organización.”

Esta competencia ha sido cubierta con el diseño y despliegue de la infraestructura de clúster, creando y gestionando infraestructuras de red para la interconexión de los nodos del clúster.

TI05. “Capacidad para seleccionar, desplegar, integrar y gestionar sistemas de información que satisfagan las necesidades de la organización, con los criterios de coste y calidad identificados.”

Esta competencia se justifica con el desarrollo del proyecto en su totalidad con la calidad esperada y sin ningún coste adicional.

TI06. “Capacidad de concebir sistemas, aplicaciones y servicios basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.”

Esta competencia se ha adquirido con el servicio web instalado para ser ofrecido por la infraestructura de clúster diseñada a través de Internet.

TI07. “Capacidad para comprender, aplicar y gestionar la garantía y seguridad de los sistemas informáticos.”

Esta competencia se justifica con las distintas configuraciones llevadas a cabo para asegurar el sistema anfitrión, todos los nodos del clúster y las máquinas creadas para el correcto funcionamiento del clúster.

3.2. Aportaciones del trabajo

En este proyecto se desarrolla y prepara un entorno virtual de alta disponibilidad y con balanceo de carga donde se despliega un blog en el CMS de *WordPress* (a modo de prueba) y se presupone que los clientes realizarán una gran cantidad de peticiones. En el caso de este tipo de servicio con una **alta demanda**, es esencial que los sistemas que se encargan de ofrecer los servicios estén disponibles de manera continuada, sin errores y que sean rápidos.

Las aportaciones que puede ofrecer este trabajo son variadas y se describen detalladamente a continuación.

Respecto al entorno **económico**, al hacer uso de la virtualización para desplegar un servicio, se utilizan los servidores disponibles y se lleva a cabo un uso más eficiente de los recursos que ofrecen, dejando de necesitar sistemas físicos. Esto supone una reducción de la inversión en capital y gastos operativos de la empresa que pretende ofrecer el servicio, una reducción del consumo energético, una reducción del tiempo de inactividad del servicio y una mayor disponibilidad del servidor. Además, esta infraestructura de clúster se ha diseñado y desarrollado en su totalidad con software libre, lo cual proporciona un mayor ahorro de costes.

En cuanto al entorno **técnico**, los clústeres de servidores son capaces de soportar la escalabilidad horizontalmente, es decir, permiten agregar más nodos al clúster según se vaya necesitando a causa de un posible aumento en la demanda del servicio por parte de los usuarios. Este proceso es muy simple y se logra clonando una de las máquinas virtuales que ofrece el servicio, configurando las direcciones de red y añadiéndola al clúster. Además, existe un mecanismo dentro del clúster que garantiza que el servicio esté siempre disponible a pesar de algún fallo en las máquinas que lo ofrecen y sin que los usuarios alcancen a percatar nada de lo ocurrido.

Al ser un clúster que ofrece balanceo de carga de los servicios de *WordPress* y de la base de datos, se evitan las **saturaciones** y como consecuencia la **ralentización** del sitio web, ganando en calidad de servicio, considerando la calidad de servicio como la garantía de unos tiempos de respuesta adecuados al servicio proporcionado y dentro de unos márgenes tolerables.

Capítulo 4

Normativa y legislación

En este capítulo se exponen las diferentes licencias a las que se protegen las diferentes tecnologías y herramientas utilizadas para el desarrollo del proyecto.

4.1. Licencia Apache

La Licencia Apache es una licencia de *software* libre permisiva elaborada por *Apache Software Foundation* (ASF). Esta licencia únicamente obliga a mantener el aviso de derecho de autor y el descargo de responsabilidad, no es una licencia *copyleft*. Durante su disponibilidad ha tenido tres versiones diferentes: 1.0, 1.1 y 2.0 [4].

La licencia actualmente acogida es la licencia **Apache 2.0** y fue desarrollada para mejorar la compatibilidad con software que posea licencias GPL (GNU General Public License).

Bajo los términos de esta licencia se ha utilizado el software del servidor web **Apache**.

4.2. Licencia GNU GPL

La *GNU General Public License* (GNU GPL), en español, Licencia Pública General de GNU es una licencia de derecho de autor (*copyleft*), de *software* libre y código abierto, permitiendo que se pueda compartir y manipular el *software* por cualquier usuario y de manera totalmente libre. Esta licencia fue creada por Richard Stallman, fundador de *Free Software Foundation* (FSF). Durante su disponibilidad ha tenido tres versiones diferentes: GPLv1, GPLv2 y GPLv3 [5].

Esta licencia está prevista para manuales, libros de textos, entre otros, para permitir que cualquier tipo de usuario pueda copiar, compartir y modificar con total libertad [6].

Bajo los términos de esta licencia con la versión **GPLv2** se ha utilizado las siguientes tecnologías: el sistema operativo **Fedora Workstation**, la herramienta de gestión de **fire-**

wall, el servicio de **HAProxy**, la tecnología **KVM**, el servicio de **MariaDB**, la tecnología de MariaDB Galera Cluster, el servicio de **NetworkManager** y el CMS de **WordPress**.

4.3. Licencia GNU LGPL

La *GNU Lesser General Public License* (GNU LGPL), en español, Licencia Pública General Reducida de GNU, es una licencia de *software* que permite compartir y modificar el software que acoge por cualquier usuario. Esta licencia fue desarrollada por *Free software General*. Durante su disponibilidad ha tenido tres versiones diferentes: LGPLv2.0, LGPLv2.1 y LGPLv3 [7].

La diferencia entre la licencia GPL y la LGPL es que esta puede vincularse con cualquier programa que no sea GPL, pudiendo ser software libre o no.

Bajo los términos de esta licencia se ha utilizado la tecnología **KVM**.

4.4. Licencia pública de Mozilla

La Licencia pública de Mozilla es una licencia de *software* libre y de código abierto, siendo una combinación de la Licencia *BSD* (Berkeley Software Distribution) y la *GNU General Public License* (GPL). Esta licencia fue desarrollada y es mantenida por la Fundación Mozilla [8].

Bajo los términos de esta licencia se ha utilizado, evidentemente, el navegador web **Firefox**.

4.5. Licencia PHP

La Licencia de PHP es una licencia de *software* libre no *copyleft* y de código abierto. Esta licencia no es compatible con la licencia GPL. Durante su disponibilidad ha tenido cinco versiones: 2.01, 2.02, 3.0, 3.01 y 4. La **versión 3.01** es la que se utiliza actualmente, ya que está formada por una licencia dual (bajo la licencia de PHP y GNU GPL) y a partir de la versión 4, esta dejó de usarse por las restricciones impuestas del *copyleft* [9].

Bajo los términos de esta licencia, se ha utilizado, evidentemente, el lenguaje de programación **PHP** con la **versión 3.01**.

Capítulo 5

Metodología y planificación

Al tratarse de un proyecto de diseño y despliegue de una infraestructura de un sistema de información, para el desarrollo del proyecto se ha seguido una **metodología secuencial**, la cual se divide en diferentes etapas y cada fase comienza cuando se haya completado la anterior. Estas fases son las que se han especificado en la planificación inicial del proyecto: análisis, diseño, desarrollo y evaluación. A continuación, se describen todas las etapas realizadas.

En primer lugar, se analizan las posibilidades de Fedora para la instalación y configuración de la plataforma de virtualización KVM (*Kernel-based Virtual Machine*) y se llevan a cabo diferentes operaciones y configuraciones básicas en dicha plataforma para verificar que se efectúan adecuadamente para su posterior uso en el desarrollo del clúster.

En segundo lugar, se analizan los requerimientos del servicio web basado en Apache y del servicio de base de datos a utilizar y los requerimientos de la plataforma Fedora para la computación en clúster, es decir, para proporcionar alta disponibilidad y balanceo de carga.

En tercer lugar, se diseña el clúster que se va a desarrollar posteriormente, indicando el número de nodos por el que está formado el clúster, el tipo de almacenamiento, la infraestructura de red, el servicio que ofrece, etc.

Una vez diseñado el clúster, se instalan los elementos de infraestructura para soportar la configuración clúster diseñada (nodos de cómputo, nodos de almacenamiento e infraestructura de red), se crea y configura el clúster y se instala *WordPress*, Apache, el servicio de base de datos (MariaDB), el servicio que ofrece el balanceo de carga (HAproxy) y el software necesario para configurar el espacio de almacenamiento.

A modo de resumen, se incluye la tabla 5.1 que describe la planificación inicial por fases, indicando la duración estimada y el nombre y descripción de las tareas. En esta, se puede apreciar una ligera diferencia con las fases recién detalladas, ya que en la primera fase (Estudio previo / Análisis) no aparece como tal el análisis realizado para la instalación y configuración de la plataforma de virtualización KVM y las diferentes operaciones realizadas en ella. Esto ocurre porque la tarea 1.2. se desglosa en este análisis para una mayor comprobación de los componentes de Fedora para la computación en clúster, realizando cada una de las posibles

operaciones para verificar que la distribución las efectúa correctamente y así, posteriormente, poder configurar la infraestructura de clúster con todos los elementos requeridos.

Fases	Duración estimada (horas)	Tareas (nombre y descripción)
Estudio previo / Análisis	30	Tarea 1.1: Estudio de los sistemas software para desplegar un servicio web basado en Apache y de un servicio de base de datos.
		Tarea 1.2: Estudio de los componentes de Fedora para proporcionar alta disponibilidad con balanceo de carga.
Diseño / Desarrollo / Implementación	190	Tarea 2.1: Diseño del clúster.
		Tarea 2.2: Instalación de los elementos de infraestructura del clúster.
		Tarea 2.3. Instalación del clúster.
		Tarea 2.4. Instalación del servicio web basado en Apache y del servicio de base de datos.
Evaluación / Validación / Prueba	20	Tarea 3.1: Pruebas para verificar la alta disponibilidad del servicio.
		Tarea 3.2: Pruebas para verificar que las peticiones se atienden de manera balanceada.
Documentación / Presentación	60	Tarea 4.1: Desarrollo de la documentación de la memoria del trabajo de fin de título y manuales de usuario.
		Tarea 4.2: Preparación de presentación y defensa del TFT.

Cuadro 5.1: Planificación inicial

El logro de los objetivos planteados ha sido gracias al largo análisis de las tecnologías empleadas y al estudio de su aplicación en la distribución de **Fedora Workstation 35**. Estas tecnologías están en continua evolución y la documentación actualizada sobre su aplicación en dicha distribución es notoriamente escasa. Esto implica una mayor inversión de tiempo en esta fase, resultando finalmente una duración aproximada de 40 horas.

También se debe tener en cuenta que existe un número considerable de horas empleadas en los momentos de ejecución de pruebas, en los momentos en los que ocurren errores y se tardan en solventarlos, en la instalación y configuración del sistema anfitrión y de las máquinas virtuales, en la realización de copias de seguridad, en el reinicio de las máquinas, en las actualizaciones de las máquinas, etc., aunque indudablemente están vinculados a este estilo de proyectos.

Capítulo 6

Herramientas y tecnologías utilizadas

Las principales tecnologías utilizadas para el desarrollo del proyecto se detallan a continuación.

1. Apache

Apache es un servidor web HTTP de código abierto. Está desarrollado y es mantenido por una comunidad de usuarios con la supervisión de *Apache Software Foundation* [10]. Está disponible para plataformas Windows, Unix y macOS.

Apache es uno de los componentes principales del **servidor LAMP**, sistema que usa las herramientas Linux, Apache, MySQL/MariaDB y PHP. Gracias a esto, es compatible con una gran variedad de CMS (*WordPress*, *Drupal*, etc.).

Este servidor web es instalado en los nodos que ofrecerán el servicio de *WordPress* (**Nodo1-servicio** y **Nodo2-servicio**).

2. Fedora

Fedora es una distribución GNU/Linux de *software* libre y código abierto que cuenta con el apoyo y patrocinio de Red Hat, siendo un proyecto comunitario donde la rama RHEL (*Red Hat Enterprise Linux*), deriva de las versiones de Fedora [11].

Existen tres ediciones del sistema operativo de Fedora: Fedora Workstation, Fedora Server y Fedora IoT. La primera edición está diseñada para ordenadores de sobremesa y portátiles con herramientas específicas preinstaladas para toda clase de desarrolladores. La segunda edición es un sistema operativo de servidor que incluye las mejores y últimas tecnologías. Y la última edición, es un sistema operativo de código abierto para su uso en dispositivos de IoT (*Internet of Things*, en español, Internet de las Cosas) [12].

En este caso se ha instalado la versión estable de Workstation del momento en el que se comenzó con el proyecto, Fedora Workstation 35. Sin embargo, ya está disponible la nueva versión estable de Fedora Workstation (36).

3. FirewallD

Firewalld es una herramienta de gestión de *firewall* (cortafuegos) para las plataformas de Linux. Es la manera en la que se protegen los sistemas del tráfico involuntario del exterior. Da la posibilidad de controlar el tráfico de red entrante con un conjunto de reglas de firewall que puede permitir bloquearlo o permitirlo.

Firewalld es el demonio del servicio de *firewall* dinámico, esto significa que se puede gestionar las reglas sin tener que reiniciar dicho demonio. Este demonio emplea dos ideas para gestionar el tráfico: zonas (conjuntos de reglas predeterminadas) y servicios (servicios que se pueden añadir al *firewall* y que utilizan uno o más puertos) [13].

Se utiliza para abrir puertos y servicios a las diferentes zonas existentes.

4. GFS2

GFS2 es un sistema de archivos distribuido nativo de Red Hat para clúster de servidores que permite que todos los nodos que forman el clúster tengan acceso de manera simultánea al dispositivo de almacenamiento que comparten, controlando la coherencia entre ellos [14].

Se ha utilizado este sistema de archivos distribuido para el espacio de almacenamiento compartido que poseen los nodos.

5. HAProxy

HAProxy es un *software* que proporciona un servicio de equilibrador de carga, alta disponibilidad y proxy para servicios HTTP y TCP. Para servicios webs con un tráfico muy alto, el HAProxy distribuye las multitudes de solicitudes entre los diferentes servidores que posee el servicio web[15].

En este caso se ha utilizado para balancear la carga del servicio web basado en Apache y del servicio de la base de datos (MariaDB).

6. High Availability Add-On

High Availability Add-On es un sistema disponible en clúster de servidores que permite la confiabilidad, escalabilidad y disponibilidad de los servicios que ofrece monitorizando los nodos del clúster [16].

Se ha hecho uso de este complemento para configurar y ver el aspecto del clúster.

7. KVM

KVM (*Kernel-based Virtual Machine*, en español, Máquina virtual basada en el núcleo) es una tecnología de código abierto que permite convertir a Linux en un hipervisor para ejecutar en un sistema anfitrión varios entornos virtuales de este (máquinas virtuales). Esta tecnología permite ejecutar máquinas virtuales de diferentes sistemas operativos y a través de diferentes medios (local, manual, por red y por una imagen de disco existente). Cada una de ellas posee un hardware virtualizado específico (tarjetas de red, dispositivos de almacenamiento, etc.) [17].

Es la tecnología más importante utilizada en el proyecto debido a que está desarrollado sobre esta plataforma.

8. LVM

LVM es un gestor de volúmenes lógicos para plataformas Linux que permite gestionar las unidades de disco y dispositivo del sistema. Se puede crear espacios de almacenamientos abstractos y particiones virtuales para modificar fácilmente las particiones sin necesidad de preocuparse acerca del espacio disponible. Con LVM se construyen, siguiendo el orden, los bloques básicos: volúmenes físicos, grupo de volúmenes y volumen lógico [18]. Este último volumen hace referencia a la partición lógica común de los sistemas que se puede formatear con un sistema de archivos para luego ser utilizada.

Esta herramienta se ha utilizado para construir un LVM sobre la unidad de disco disponible en la máquina **iSCSI-almacenamiento**, que proporciona el espacio de almacenamiento compartido y distribuido de los nodos del clúster.

9. MariaDB

MariaDB es un sistema de administración de base de datos de código abierto y *software* libre. MariaDB procede de MySQL, por lo tanto, MariaDB tiene una gran compatibilidad (mismas órdenes, interfaces, etc.) [19]. MariaDB también es un completo del servidor LAMP.

MariaDB se ha utilizado como base de datos para el desarrollo del proyecto y para poder ejecutar correctamente el servidor LAMP.

10. MariaDB Galera Cluster

MariaDB Galera Cluster es una tecnología que permite crear un clúster de base de datos de MariaDB que puede estar formado por múltiples nodos y que se basa en la replicación de las bases de datos de manera síncrona [20].

Esta tecnología se ha empleado para la replicación de la base de datos de *WordPress* de manera síncrona entre los dos nodos en los que se encuentra (**Nodo3-almacenamiento**

y **Nodo4-almacenamiento**) y para que, posteriormente, se pueda realizar el balanceo de carga de esta.

11. Mozilla Firefox

Mozilla Firefox es un navegador web gratuito y de código abierto creado y proporcionado por Mozilla como una alternativa a *Internet Explorer* y *Chrome*, más rápida y segura. Está disponible para múltiples plataformas (Linux, Windows, macOS, etc.) y es coordinado por la Corporación Mozilla y la Fundación Mozilla [21].

Se ha utilizado el navegador de Firefox para efectuar las pruebas del servidor Apache, la instalación de *WordPress*, el uso de la página de administración de *WordPress*, las pruebas del blog ofrecido como servicio del clúster y las del balanceo de carga.

12. NetworkManager

NetworkManager es una utilidad que permite gestionar la configuración de la red y las conexiones. Su uso está disponible únicamente en plataformas Linux. Este servicio posee un demonio que ofrece información sobre el estado del servicio y la red. Además, existen varias utilidades y aplicaciones para administrar las configuraciones de la red [22].

En este caso, se ha utilizado la utilidad de línea de órdenes **nmcli** para crear nuevas conexiones, editarlas, verlas y eliminarlas.

13. PHP

PHP es un lenguaje de programación de código abierto utilizado para el desarrollo web. Este lenguaje puede ser incrustado en HTML (*HyperText Markup Language*)¹ [23].

PHP es otro de los componentes por los que está formado el **servidor LAMP** y en este proyecto, se utiliza para poder hacer uso de dicho servidor.

14. WordPress

WordPress es un sistema de gestión de contenidos que está centrado en crear y administrar cualquier tipo de sitio web de manera sencilla y visual, sin necesidad de programar. Esta plataforma se caracteriza por permitir diferentes tipos de roles predeterminados (administrador, editor, cliente, proveedor, etc.) para conceder a los administradores la posibilidad de controlar qué puede hacer cada usuario dentro del sitio web diseñado.

En este caso, se ha instalado y configurado para ofrecer un blog (a modo de prueba) en el clúster desarrollado.

¹Lenguaje de marcado que define el contenido y la estructura de dicho contenido de un sitio web.

Capítulo 7

Desarrollo

7.1. Análisis

Esta sección está dedicada a describir el estudio que se ha llevado a cabo sobre los diferentes requerimientos que debe cumplir el sistema y los servicios a ofrecer en el clúster a desarrollar. Se divide en cuatro apartados: requerimientos del sistema para la virtualización, del servicio web basado en Apache, del servicio de la base de datos y de la distribución Fedora Workstation 35 para la computación en clúster.

El sistema operativo que se ha instalado en la máquina anfitriona es **Fedora Workstation 35**, una edición de Fedora orientada a escritorio y que cuenta con el apoyo y patrocinio de [Red Hat](#).

Red Hat Enterprise Linux (RHEL) es la distribución y versión comercial de Fedora, por lo tanto, la documentación oficial de *Red Hat Enterprise Linux 8* sirve para tomarla como referencia y apoyo a la de documentación oficial de Fedora.

7.1.1. Requerimientos del sistema para la virtualización

Inicialmente, se analizan los requerimientos propuestos en la documentación oficial de Fedora [24] que debe poseer el sistema anfitrión para ejecutar de manera satisfactoria máquinas virtuales mediante la plataforma KVM (*Kernel based Virtual Machine*) - una infraestructura de virtualización en Linux que convierte el sistema operativo en un *hypervisor*¹, permitiendo que el host anfitrión ejecute entornos virtuales múltiples y aislados (máquinas virtuales).

¹Software que permite realizar la virtualización de hardware, crea y ejecuta máquinas virtuales, permitiendo utilizar diferentes sistemas operativos.

Así pues, los **requerimientos mínimos** que debe poseer el sistema anfitrión para instalar los componentes principales de la plataforma de virtualización en Fedora son [25]:

- 6 GB de espacio libre en disco para el anfitrión.
- 2 GB de RAM para el anfitrión.

Al mismo tiempo, el hipervisor KVM para Fedora Workstation 35 solo está disponible para equipos con procesadores AMD o Intel, con las extensiones Intel VT e Intel 64 y, AMD-V y AMD64, respectivamente [26].

También se han considerado los siguientes **requerimientos recomendados** para el sistema para que el entorno de virtualización funcione correctamente [25]:

- 6 GB más de espacio libre en disco requerido por cada sistema operativo de cada máquina virtual.
- 2 GB de RAM más para cada máquina virtual.
- Un núcleo de procesador o hiper-hilo para cada CPU virtualizada y otro para el hipervisor.

Por otra parte, se analizan cuáles son los diferentes **métodos de almacenamiento** que soportan las máquinas virtuales [27]:

- Archivos almacenados en medios locales (directorios).
- Particiones de discos físicos y virtuales.
- Sistemas de archivos.
- Volúmenes en LVM.
- Sistemas de archivos de red (NFS).
- Almacenamiento basado en iSCSI.
- Sistemas de archivos de espacio de usuario (GlusterFS).
- Contenedores de almacenamiento basado en SCSI utilizando dispositivos vHBA.

Para verificar que el sistema anfitrión consta de todos los requerimientos nombrados anteriormente, se han ejecutado una serie de comandos que se exponen en el **Anexo I**.

7.1.2. Requerimientos del servicio WordPress

En este apartado se describen los requerimientos del servicio web basado en Apache. El servicio web que se ha decidido utilizar es *WordPress*. Por lo tanto, los requerimientos para poder ejecutar *WordPress* en dos de los nodos del clúster son [28]:

- Un sistema operativo actualizado (en este caso, se ha utilizado Fedora Workstation 35).

- Acceso a Internet y permisos de administrador (root).
- Tener instalado y configurado un servidor LAMP. Este está formado por un sistema operativo Linux (Fedora) y los servicios de Apache, PHP y MariaDB o MySQL. Los nodos del clúster que ofrecerán el servicio de *WordPress* deben tener instalado los servicios que sean compatibles con este CMS, los cuales deben ser:
 - PHP con la versión 7.4 o superior instalada.
 - MariaDB con la versión 10.3 o superior instalada.
 - Apache o Nginx instalado, puesto que son los servidores más robustos y flexibles. Sin embargo, cualquier otro servidor que sea compatible con PHP y MySQL también funcionará.
- Soporte para HTTPS.
- 1 GB de espacio libre en disco.
- Al menos 512 MB de RAM.

7.1.3. Requerimientos del servicio de base de datos *MariaDB*

Los requerimientos para instalar y configurar el servicio de la base de datos elegida (MariaDB), son los básicos y necesarios para instalar cualquier otro servicio [29]:

- Un sistema operativo actualizado (se dispone de Fedora Workstation 35).
- Acceso a Internet y permisos de administrador (root).
- Un procesador (CPU) de 2 núcleos.
- Una memoria de 1024 MB.
- Protocolo de red TCP/IP.
- 1 GB de espacio libre en disco.
- Al menos 512 MB de RAM.

7.1.4. Requerimientos de Fedora para la computación en clúster

En este epígrafe se ha realizado un estudio de los componentes a utilizar en Fedora para desarrollar la infraestructura de clúster que ofrece *WordPress* en alta disponibilidad y con balanceo de carga.

Para ello, en primer lugar se ha analizado un servicio que balancee la carga del servicio web basado en Apache. Este servicio es HAProxy y además de proporcionar el balanceo de carga, ofrece alta disponibilidad. Luego se ha estudiado un servicio que proporcione alta disponibilidad para la base de datos (MariaDB Galera Cluster). Tras una investigación adicional,

se ha averiguado que HAProxy es compatible con Galera Cluster y como este proporciona un equilibrador de carga, se puede agregar dicho servicio al HAProxy para que la base de datos, además de estar en alta disponibilidad, pueda proporcionar balanceo de carga.

En este momento, con los servicios analizados, ya se tiene completamente disponible el clúster a desarrollar. Por lo tanto, lo único que falta es analizar los requerimientos de los servicios para poder ser instalados y configurados posteriormente:

- Los requerimientos *hardware* para el nodo (**Balanceador-de-carga**) que posee el servicio de HAProxy dependen de la carga de trabajo que se necesite administrar. En este caso, se supone que la carga de trabajo que administrará el servicio es de nivel medio (hasta unas 4000 conexiones por segundo), por ende, los requerimientos son [30]:
 - Un procesador (CPU) de 2 núcleos.
 - 1 GB de RAM.
- Los requerimientos *hardware* para cada nodo que posee el servicio de Galera Cluster (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**) son [31]:
 - Un procesador (CPU) de un solo núcleo.
 - 1 GB de RAM.
 - Conectividad de red de al menos 100 Mbps.
- Los requerimientos *software* para cada nodo que posee el servicio de Galera Cluster (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**) son [31]:
 - Sistema operativo linux.
 - Servidor de MariaDB o MySQL con la API wrsep.
 - Paquete de *Galera Replication* instalado.

7.2. Acondicionamiento del sistema

En esta sección, se explica todo el proceso que se ha llevado a cabo para la preparación del sistema anfitrión, detallando paso a paso las diferentes configuraciones y funcionalidades que se pueden realizar con respecto a la virtualización en la distribución de Fedora 35.

7.2.1. Preparación de la interfaz gráfica

Una de las características que posee esta distribución es que, a diferencia de la interfaz gráfica clásica a la que se está acostumbrado, no existen los botones de maximizar y minimizar en las ventanas de cualquier aplicación. Si se desea maximizar o minimizar la ventana de una aplicación, se debe hacer **doble clic** en la barra superior de la ventana.

Sin embargo, existe una posibilidad para hacer uso de las funciones de maximizar y minimizar una ventana (de cualquier aplicación) en esta distribución de manera usual. Para poder utilizarlas, se ha seguido una serie de pasos que se indican en el [Anexo II](#), donde se instala el paquete “*gnome-tweaks*”, indispensable para disponer de una interfaz común donde sí se puede utilizar las funciones usuales de dichos botones.

7.2.2. Configuraciones y operaciones

7.2.2.1. Actualización del sistema

La seguridad es un aspecto primordial a tener en cuenta para mantener el correcto funcionamiento de los sistemas. A pesar de disponer de herramientas de seguridad tales como un *software* de antivirus o un cortafuegos, es muy importante tener actualizado cualquier tipo de sistema operativo o *software* que posea el sistema, pues las actualizaciones instalan mejoras en el funcionamiento y en la seguridad del *software*, solucionando de esta manera errores y/o vulnerabilidades e instalando nuevas funcionalidades. Siendo así, se ha procedido a realizar una actualización del sistema antes de comenzar a efectuar cualquier tipo de configuración u operación en el sistema, ejecutando el siguiente comando:

```
[root@pc1-tft ~]# dnf upgrade
```

En este caso, se ha utilizado el comando *upgrade* ya que, a diferencia del *update*, este instala la versión más reciente del *software*, el cual viene con mejoras importantes: nuevas características, mejor seguridad, etc.

Tan pronto como finaliza la actualización, se comprueba con el comando posterior si está instalada la versión más reciente del repositorio, dando como resultado la última fecha de actualización del sistema.

```
[root@pc1-tft ~]# dnf check-update
```

7.2.2.2. Estado de la utilidad *NetworkManager*

NetworkManager es una utilidad de software que permite administrar la configuración y las conexiones de red [32]. *NetworkManager* dispone de un demonio que ofrece información sobre el estado de la red y algunas herramientas para configurar dicha red; creando, verificación, editando y eliminando conexiones o interfaces de red.

Con esta utilidad se puede configurar conexiones de tres tipos: *hardware*, virtual y VPN. Con respecto al *hardware*, las conexiones más comunes que se pueden crear son: Bluetooth, Ethernet, InfiniBand y Wi-Fi. En cuanto a las redes **virtuales**, las conexiones más utilizadas son: Bond, Puente, Túnel IP, MACsec, Team y VLAN. Las **conexiones VPN** más utilizadas son mediante SSH y OpenVPN, aunque existe una gran variedad de estas [33].

Para gestionar las conexiones de red de *NetworkManager* existen diferentes utilidades y aplicaciones, las cuales son:

1. **nmcli**: utilidad de línea de órdenes.
2. **nmtui**: aplicación con interfaz de usuario de texto.
3. **nm-connection-editor**: aplicación con interfaz gráfica de usuario.
4. **control-center**: aplicación GNOME con interfaz gráfica de usuario (En ajustes → *Network*).
5. **network connection icon**: icono de red de GNOME Shell en la zona superior derecha del escritorio (o zona inferior derecha del escritorio si se ha instalado “*gnome-tweaks*”).

Sin embargo, existe otra manera de configurar las conexiones de red manualmente, la cual es la que se suele utilizar en la mayoría de las distribuciones de Linux. Esta manera es mediante los scripts de red heredados, es decir, aquellos ficheros **ifcfg** que se encuentran en el directorio `/etc/sysconfig/network-scripts` y que están manejados por el servicio **network**. En el caso de Fedora Workstation 35 y RHEL 8, el servicio **network** y los scripts de red heredados, están obsoletos y, en versiones posteriores de RHEL, serán eliminados.

Por esta razón, el demonio de *NetworkManager* está configurado para iniciarse al arrancar el sistema y los ficheros *ifcfg* no se encuentran en el directorio mencionado, ya que *NetworkManager* utiliza un tipo de fichero de configuración diferente y más optimizado conocido como “**keyfile**”. Si se desea manejar la red mediante el servicio **network** habría que instalar el paquete “*network-scripts*”, lo cual es desaconsejable porque es un servicio que está desfasado y por ende, dejará de ser compatible en próximas distribuciones.

En consecuencia, se utilizará la utilidad *NetworkManager* para gestionar todas las conexiones necesarias de red. Como el demonio de *NetworkManager* se encuentra activo y habilitado por defecto para iniciarse cada vez que se inicia el sistema, solo se tendrá que comprobar que el estado de dicho servicio está ejecutándose correctamente y que no haya ningún error, tal y como se comprueba en la ilustración 7.1:

```
[root@pc1-tft ~]# systemctl status NetworkManager
● NetworkManager.service - Network Manager
   Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; enabled; vendor<
   Active: active (running) since Mon 2022-06-27 09:06:59 WEST; 5min ago
     Docs: man:NetworkManager(8)
    Main PID: 916 (NetworkManager)
      Tasks: 3 (limit: 19128)
     Memory: 9.8M
        CPU: 624ms
    CGroup: /system.slice/NetworkManager.service
           └─ 916 /usr/sbin/NetworkManager --no-daemon

jun 27 09:11:15 pc1-tft.com NetworkManager[916]: <info> [1656317475.3392] device (v>
jun 27 09:11:15 pc1-tft.com NetworkManager[916]: <info> [1656317475.3393] device (v>
jun 27 09:11:15 pc1-tft.com NetworkManager[916]: <info> [1656317475.3393] device (v>
jun 27 09:11:15 pc1-tft.com NetworkManager[916]: <info> [1656317475.3395] device (v>
```

Ilustración 7.1: Estado del NetworkManager

7.2.2.3. Instalación de la plataforma de virtualización

Una vez actualizado el sistema y observado el previo análisis desarrollado sobre los requerimientos del sistema necesarios para poder realizar una correcta virtualización en esta distribución, se ha llevado a cabo la instalación de la plataforma de virtualización **KVM** (*Kernel based Virtual Machine*). Para poder obtener esta plataforma, se necesita instalar los diferentes grupos de paquetes requeridos para configurar el sistema anfitrión de virtualización en el sistema operativo Fedora 35. La instalación de este grupo de paquetes se puede observar en el [Anexo III](#), así como su puesta en marcha y su correcto funcionamiento en el [Anexo IV](#), con la creación de una máquina virtual de prueba y sus correspondientes comprobaciones.

7.2.2.4. Operaciones con máquinas virtuales

Tras la instalación y comprobación de la herramienta de virtualización KVM con la creación de una máquina virtual de prueba mediante la utilidad gráfica “*virt-manager*”, se realizan diferentes operaciones para verificar que se efectúan de manera adecuada para su posterior uso en el desarrollo del clúster. Las operaciones llevadas a cabo son las que se mencionan a continuación y se pueden distinguir en el [Anexo V](#).

- Creación de una máquina virtual con *virsh*.
- Clonación de una máquina virtual con *virt-manager* y *virt-clone*.
- Creación de una máquina virtual con *virt-install*.

En esta distribución también se puede realizar la operación de migrar máquinas virtuales, en este caso, no se ha podido llevar a cabo la prueba de migración debido a que para esto se necesitan como mínimo dos equipos exactamente iguales, con las mismas condiciones de hardware y el mismo sistema operativo para poder efectuar dicha operación de manera correcta.

7.2.2.5. Utilización de recursos de almacenamiento virtual

Según lo indicado en los requerimientos del sistema para la virtualización, existen diferentes tipos de almacenamiento para las máquinas virtuales. La manera de proporcionar almacenamiento a las máquinas virtuales es mediante los *storage pools* y *storage volumes*, en español, **contenedores** y **volúmenes de almacenamiento**.

Un contenedor es una abstracción de un espacio de almacenamiento y es manejado por *libvirt* para gestionar el espacio. Existen dos tipos de contenedor: locales y de red. Los **locales** están conectados de manera directa al anfitrión y los de **red** son aquellos que están conectados mediante una red y protocolos estándar a un dispositivo de almacenamiento. Estos contenedores pueden estar basados en distintos tipos de almacenamiento (indicados en [requerimientos del sistema para la virtualización](#)). Los contenedores que se han utilizado hasta ahora son los que vienen por defecto cuando se instala KVM: los locales y

los basados en directorios. Los contenedores o *pools*, a su vez, se dividen en volúmenes de almacenamiento, y son esos volúmenes los que se enganchan como discos a las máquinas virtuales [27].

En este epígrafe se integrarán algunos de los recursos de almacenamiento mencionados, en los sistemas invitados, tanto recursos físicos integrados en el sistema anfitrión como recursos que provienen de equipos remotos dedicados a proporcionar este tipo de servicios. Primero se realizará con la utilidad gráfica “*virt-manager*” y después, mediante la línea de órdenes “*virsh*”.

Por un lado, se crearán recursos de almacenamiento basados en contenedores locales; creando volúmenes en los contenedores, creando una partición en el anfitrión y asociarla a la máquina virtual y, creando un contenedor en una partición lógica del disco del anfitrión.

Por otro lado, se crearán recursos basados en contenedores en red, creando dos contenedores de tipo NFS para alojar, en uno, imágenes ISO y en otro, volúmenes de máquinas virtuales. En este caso, como no se dispone de un servidor dedicado a ofrecer un servicio NFS, se simula uno con una máquina virtual, instalando el servicio NFS como servidor en dicha máquina y en la máquina donde se asociará el contenedor, como cliente.

En el [Anexo VI](#) se desglosan todos los pasos realizados para lograr integrar los recursos de almacenamiento indicados en las máquinas virtuales.

7.2.2.6. Infraestructura de red virtual

En última instancia, se determinan las diferentes configuraciones de redes que se pueden llevar a cabo para que las máquinas virtuales se puedan conectar al sistema anfitrión, a otras máquinas virtuales en el anfitrión y a otros dispositivos en redes externas. Las máquinas virtuales recién creadas tienen una configuración de red predeterminada que puede ser eliminada o editada de diferentes maneras [34]. Los diferentes tipos de configuraciones de red que existen para las máquinas virtuales son:

- **Modo enrutado mediante servicio NAT:** las máquinas virtuales en este modo están en una subred separada del anfitrión y se enrutan mediante un conmutador virtual que existe en el anfitrión y que permite las conexiones entrantes y salientes sin usar NAT.
- **Modo puente (bridge):** la máquina virtual está dentro de la misma subred que el equipo anfitrión, pues esta se conecta a un puente existente en el host que permite que sea visible en la red física, posibilitando la conexión de los otros dispositivos físicos de la red a conectarse y acceder a dicha máquina virtual.
- **Modo aislado:** las máquinas virtuales con este modo de red pueden comunicarse entre sí y con el sistema anfitrión, pero el tráfico de ellas no sale fuera del anfitrión ni puede recibir tráfico desde fuera del anfitrión.
- **Modo enrutado mediante la configuración manual de las reglas de IPTABLES:** con esta configuración, la máquina virtual es capaz de proporcionar servicios accesibles desde máquinas con IP no pertenecientes a la red de la organización.

A tal efecto, se demuestra en el [Anexo VII](#) la configuración de red en modo puente, dado que las configuraciones de las redes aisladas y NAT se detallan en la siguiente sección.

7.3. Diseño e implementación del clúster

Esta sección proporciona todos los pasos para llevar a cabo el diseño y despliegue de un clúster que ofrece como servicio el CMS (Sistema de gestión de contenidos) *WordPress*, el cual está basado en Apache y hace uso de una base de datos. Tanto el servicio web como la base de datos puede prestar alta disponibilidad y balanceo de carga a la hora de atender las peticiones de los clientes. Además, se ha creado un espacio de almacenamiento compartido y distribuido basado en la tecnología **iSCSI** (Sistema de Interfaz para computadoras pequeñas en Internet) y en el sistema de archivos distribuido **GFS2** (*Global File System 2*).

iSCSI es un protocolo de red de área de almacenamiento, en otras palabras, una tecnología de almacenamiento distribuido sobre IP que permite transferir datos procedentes de dispositivos de almacenamiento a sistemas *hosts* o viceversa, haciendo uso de Internet y el protocolo TCP/IP [35].

El **GFS2** (*Red Hat Global File System 2*), en español, el Sistema de archivos global 2 de Red Hat, brinda un sistema de archivos distribuido idóneo para clústeres de servidores en sistemas GNU/Linux, debido a que permite que todos los nodos tengan acceso simultáneo al dispositivo de bloques de almacenamiento que comparten, gestionando además la coherencia entre los nodos [14].

7.3.1. Diseño del clúster

En primer lugar, se ha diseñado la infraestructura del clúster. La simulación de este clúster se desplegará mediante la creación de varias máquinas virtuales en la plataforma de virtualización KVM (que simularán los nodos del clúster), la instalación del sistema operativo de Fedora Workstation 35 en los nodos (máquinas virtuales), la generación de la infraestructura de red, la instalación y puesta en marcha de los módulos software necesarios para el soporte de computación en clúster, etc.

Teniendo en cuenta estas consideraciones, el clúster consta de cinco nodos (**Ver ilustración 7.2**), denominados: **Nodo1-servicio**, **Nodo2-servicio**, **Nodo3-almacenamiento**, **Nodo4-almacenamiento** y **Balanceador-de-carga**. Los dos primeros nodos se encargan de **proporcionar el servicio** *Wordpress* basado en Apache. Los dos nodos siguientes **almacenarán la base de datos** de dicho servicio en alta disponibilidad y con balanceo de carga. El quinto nodo será el que **distribuirá el servicio** en alta disponibilidad y balanceando la carga de las peticiones de los usuarios a través de los nodos 1 y 2, ofreciendo finalmente el servicio en un único punto de acceso.

Como también se ha creado un área de almacenamiento compartido y distribuido basada en iSCSI y en el sistema de archivos distribuido GFS2, se ha elaborado otra máquina virtual

llamada **iSCSI-almacenamiento**, que simula un **servidor iSCSI** que aporta dicho espacio de almacenamiento compartido y distribuido (con el sistema de archivos GFS2) a todos los nodos del clúster para almacenar los ficheros del servicio web del nodo 1 y 2 (ya que al ofrecer un único servicio deberá tener un único directorio para los archivos compartidos) y los ficheros de configuración de los servicios de Apache y de la base de datos. Esta máquina no forma parte del clúster.

La **infraestructura de red** necesaria para todos los nodos del clúster está formada por tres redes diferentes: una red NAT, una red aislada de control y otra red aislada de almacenamiento. Por el contrario, el servidor iSCSI tendrá asociada únicamente la red NAT y la red aislada de almacenamiento del clúster.

Una vez determinado el diseño del clúster, se procede a desplegar toda la infraestructura básica del clúster: efectuar la creación de las distintas máquinas virtuales (nodos), la instalación del sistema operativo en los nodos, la creación y asignación de las redes a los diferentes nodos, etc.

En la ilustración 7.2 se muestra un diagrama de la **infraestructura final del clúster**. La red NAT mencionada corresponde con la dirección de red 192.168.122.0/24 del diagrama (color rojo); la red aislada de control refleja la dirección de red 10.22.132.0/24 del diagrama (color verde) y la red aislada de almacenamiento responde a la dirección de red 10.22.122.0/24 (color amarillo).

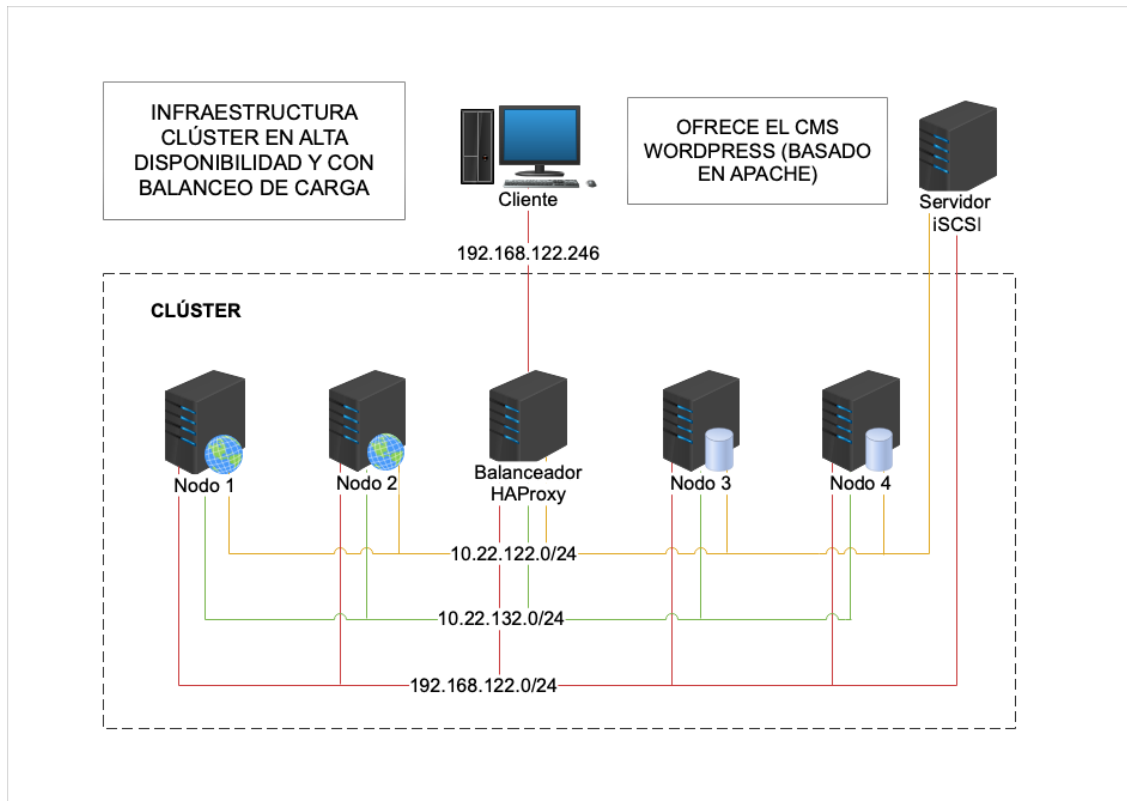


Ilustración 7.2: Infraestructura del clúster

Una vez se haya diseñado y desplegado la infraestructura del clúster, se continuará con la instalación y puesta en funcionamiento de los módulos software para el soporte de computación en clúster, para la instalación de los módulos necesarios para la explotación de *WordPress*, para el servicio del balanceo de carga y alta disponibilidad, para el servicio de almacenamiento compartido y distribuido, etc.

ADECUADO: En un entorno de producción, el diseño presentado no sería completamente bueno, ya que si el nodo **Balanceador-de-carga** se cae por algún tipo de fallo o problema, el servicio quedaría completamente inaccesible porque es el que mantiene el software para proporcionar alta disponibilidad y balanceo de carga al servicio web basado en Apache. Por lo tanto, sería idóneo tener otro nodo de respaldo para que, si el nodo **Balanceador-de-carga** por algún motivo deja de estar disponible, el servicio no se quede inalcanzable y pase a ofrecerlo el nodo de respaldo.

7.3.2. Implementación del clúster

En esta sección se detallan todos los pasos necesarios a seguir para desplegar la infraestructura del clúster mencionada previamente y se realizan las pruebas respectivas para verificar que todos los pasos se han realizado correctamente.

7.3.2.1. Creación de la infraestructura básica del clúster

Tal y como se ha comentado en la sección anterior, se prestará el servicio *WordPress* en alta disponibilidad y con balanceo de carga en el que intervienen seis máquinas virtuales (nodos), de las cuales cinco, forman parte del clúster. La **infraestructura de red** de todos los nodos que forman parte del clúster está dividida en tres redes: una red NAT para el tráfico generado por el servicio de *WordPress* y para tener acceso a los repositorios de *software* externos; y dos redes aisladas, una destinada al tráfico de control del clúster y otra para el tráfico de acceso al almacenamiento compartido y distribuido.

A continuación, se describe detalladamente el **rol** y las **características principales** de **cada nodo**:

- Los nodos (máquinas virtuales) llamados **Nodo1-servicio** y **Nodo2-servicio** proporcionarán el servicio de *WordPress* basado en **Apache**. Los nombres de dominio completamente cualificados serán **nodo1-tft.com** y **nodo2-tft.com**.
- Los nodos denominados **Nodo3-almacenamiento** y **Nodo4-almacenamiento** proveerán el servicio de la base de datos (**MariaDB**) en alta disponibilidad y con balanceo de carga mediante el software **MariaDB Galera Cluster**². Los nombres de dominio completamente cualificados serán **nodo3-tft.com** y **nodo4-tft.com**.

²Software que permite la creación y gestión de clústeres de bases de datos de manera síncrona entre los nodos del clúster.

- El nodo nombrado como **Balancedor-de-carga** ofrece el servicio de los nodos 1 y 2 en alta disponibilidad y con balanceo de carga a través del software HAProxy³.
- Los cinco nodos que forman el clúster comparten las siguientes características:
 - Los recursos que deberá tener cada máquina serán: 2 GB de RAM, 2 CPUs y 1 disco de 15 GB.
 - Las tres redes que deberá tener cada máquina será: la red NAT por defecto que se crea al instalar KVM (default - 192.168.122.0), la red aislada de control (10.22.132.0) y la red aislada de almacenamiento (10.22.122.0). Las dos redes aisladas no proporcionarán el servicio DHCP, ya que se asignarán las direcciones IP de manera estática.

El rol y las **características principales** de la máquina virtual que simula el **servidor iSCSI** son:

- La máquina virtual que simulará el servicio iSCSI para proveer el espacio de almacenamiento compartido y distribuido se define como **iSCSI-almacenamiento**. El nombre de dominio completamente cualificado será **iscsi-tft.com**.
 - Los recursos que deberá tener esta máquina son los mismos que los nodos del clúster: 2 GB de RAM, 2 CPUs y 1 disco de 15 GB.
 - Las redes que deberá tener esta máquina serán dos: la red NAT por defecto que se crea al instalar KVM (default - 192.168.122.0) y la red aislada de almacenamiento (10.22.122.0) para el tráfico de acceso al almacenamiento compartido de los nodos. La red aislada no proporcionará el servicio DHCP, ya que se asignará la dirección IP de forma estática.

Creación de las redes aisladas

Ahora, se lleva a cabo la creación de las dos redes aisladas mediante la utilidad gráfica “*virt-manager*”.

Desde la pantalla inicial de la utilidad, se selecciona en el menú superior la pestaña “Editar”, se accede a “Detalles de conexión” y se abre una ventana emergente. En ella, se selecciona la pestaña “Redes virtuales” en el menú superior y se hace clic en el icono “+” que se encuentra en la zona inferior izquierda. Tras esto, se abre otra ventana emergente donde se debe indicar las características necesarias para crear las dos redes aisladas.

Por un lado, se crea la red aislada de control y, como se puede apreciar en la ilustración 7.3, se denomina a la red como “**Control**”. Además, se selecciona el modo “*isolated*” (**aislada**) como tipo de red virtual y se habilita la dirección **IPv4**, indicando la dirección de red **10.22.132.0/24** y **desactivando** el servicio **DHCPv4**. Para concluir, si la **dirección**

³Software que proporciona un balanceo de carga y de alta disponibilidad para servicios basados en HTTP y TCP.

IPv6 no está desactivada, se **inhabilita** y se finaliza el proceso de creación de la red aislada de control.

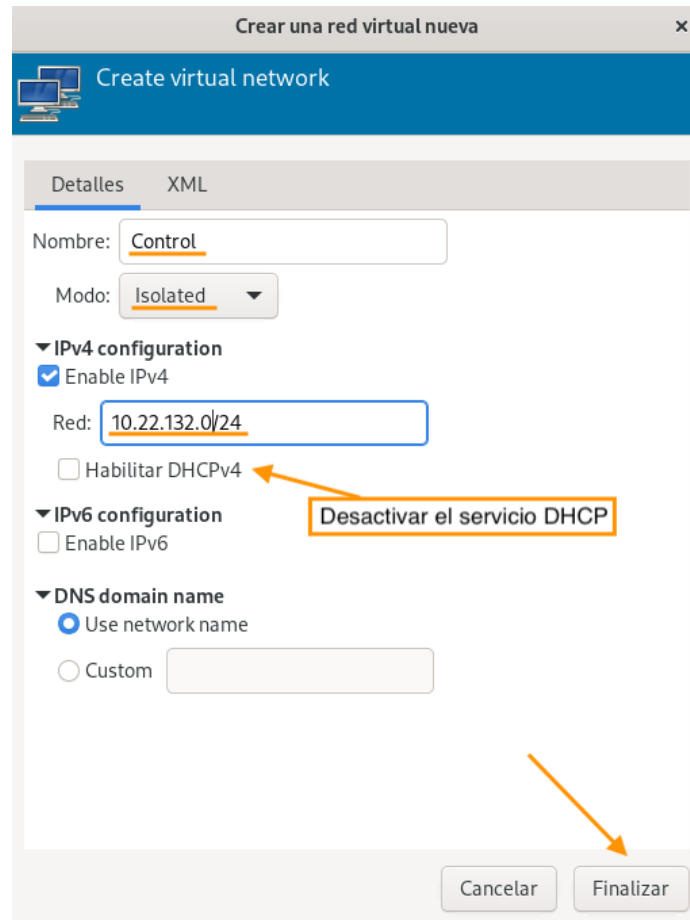


Ilustración 7.3: Creación de la red aislada de control del clúster

Por otro lado, la red aislada de almacenamiento se crea de manera similar, designando el nombre de la red como “**Almacenamiento**”, seleccionando el modo “**isolated**” (**aislada**) como tipo de red virtual y habilitando la dirección **IPv4** (indicando a su vez la dirección de red correspondiente, **10.22.122.0/24** y **desactivando** el servicio **DHCPv4**). También se debe **desactivar** la **dirección IPv6** si no lo estuviera y se finaliza el proceso de creación de la segunda red aislada (**Ver ilustración 7.4**).

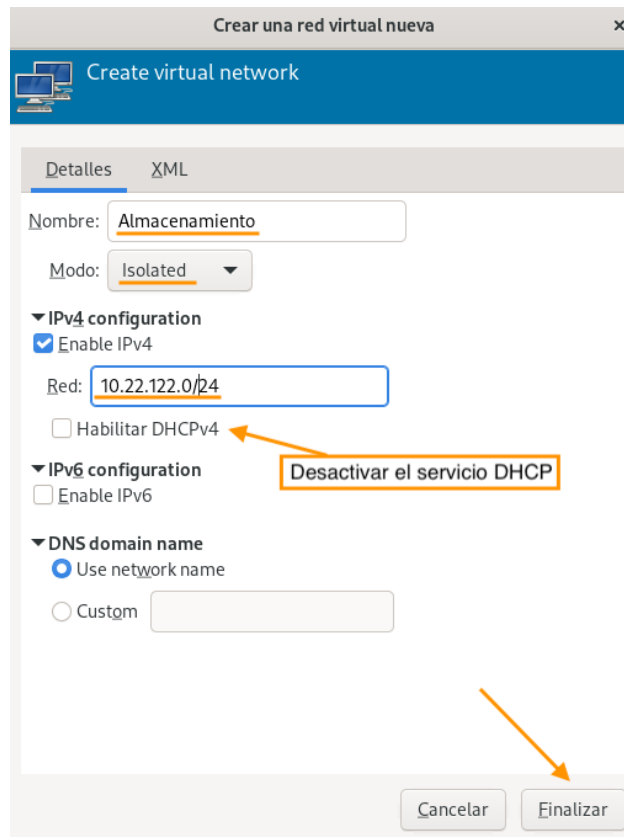


Ilustración 7.4: Creación de la red aislada de almacenamiento del clúster

A modo de resumen, en la ilustración 7.5 se muestra la ventana de “Detalles de conexión” y se manifiestan todas las redes que están disponibles en la plataforma de virtualización para ser asignadas a los nodos del clúster. Se puede ver que, aunque se han creado dos redes aisladas (Control y Almacenamiento), se presenta otra más denominada *default* y que, como se indicó antes, hace referencia a la red NAT creada por defecto cuando se instala KVM.

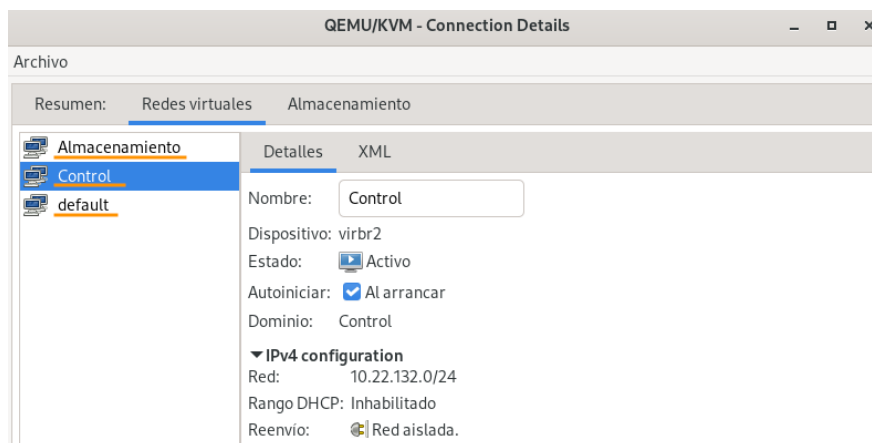


Ilustración 7.5: Resumen de todas las redes

Creación de las máquinas virtuales

Tras la generación de las redes, se procederá a crear las seis máquinas virtuales, de las cuales, cinco, representarán los distintos nodos del clúster. Para acelerar el proceso y agilizar el trabajo, se creará una máquina virtual base mediante la herramienta gráfica “*virt-manager*”. Se le asignará en principio una sola interfaz de red (enp1s0) conectada a la red NAT por defecto (default) y los recursos indicados con anterioridad (cantidad de CPU, tamaño de RAM y de imagen de disco). Para ello, se siguen los pasos para la creación de la máquina base del [Anexo IV](#) y se ajustan algunos parámetros en ciertos pasos para adaptar los recursos requeridos para los nodos.

En primer lugar, desde la página inicial del “*virt-manager*”, se hace clic en el icono “Crear una nueva máquina virtual” que se encuentra en la parte superior izquierda, debajo del campo “Archivo”, tal y como se especifica en el Anexo indicado. Y, seguidamente, se abre una ventana emergente donde se debe seguir un total de cinco pasos para finalizar el proceso de creación de la nueva máquina virtual.

El primer paso es la elección del método de instalación del sistema operativo, seleccionando, en este caso, la primera opción que se presenta para instalar Fedora Workstation 35 mediante la imagen ISO. El segundo paso consiste en indicar la ruta correspondiente a donde se encuentra dicha imagen ISO.

A partir del tercer paso varía la configuración de los recursos de la máquina virtual, específicamente en la selección de los ajustes de la memoria RAM, la CPU y la imagen de disco (en el cuarto paso). En este caso, se requiere como mínimo 2 GB de memoria RAM y 2 CPUs para trabajar cómodamente con las máquinas virtuales, por lo tanto, se configura este paso justo como se evidencia en la ilustración 7.6.

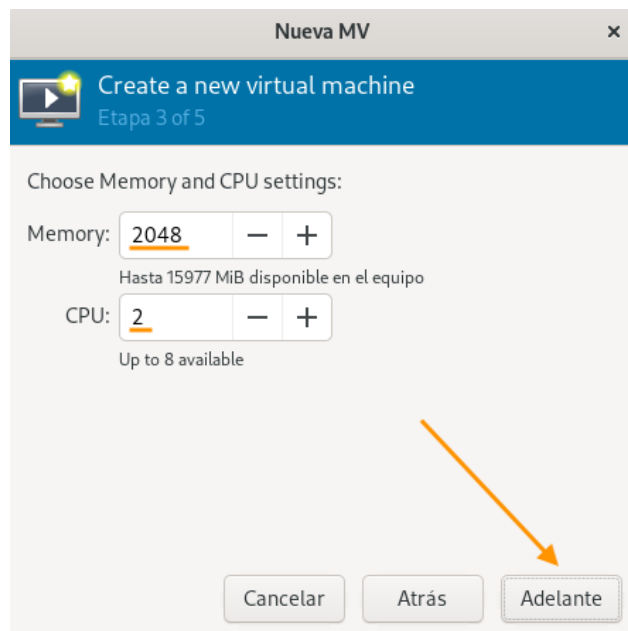


Ilustración 7.6: Paso 3 - Ajustes de la memoria RAM y la CPU

En la cuarta etapa, a diferencia del paso correspondiente en el Anexo mencionado, se crea una imagen de disco con 15,0 GB (**Ver ilustración 7.7**).

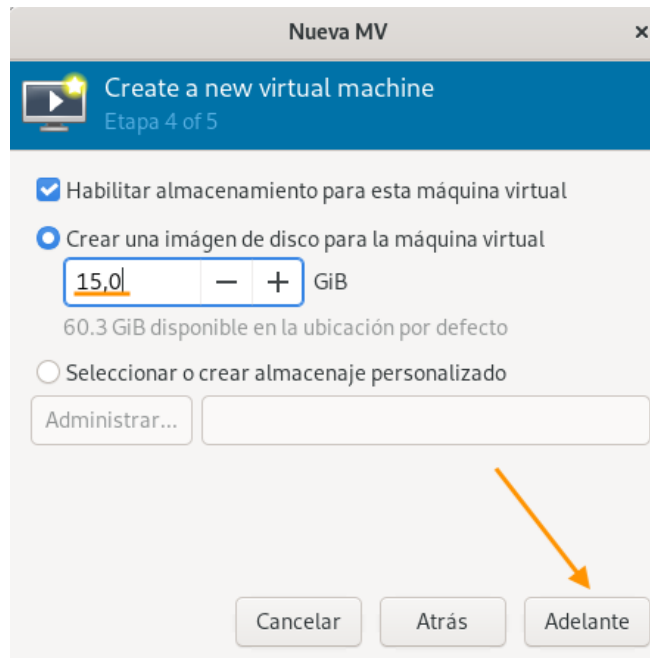


Ilustración 7.7: Paso 4 - Ajustes de la imagen de disco

Para concluir el proceso de creación de la máquina virtual base, en el último paso se indica el nombre que tendrá dicha máquina (**maquinabase**), se selecciona la red NAT (ya que todas las máquinas tendrán esta red) y se finaliza el proceso.

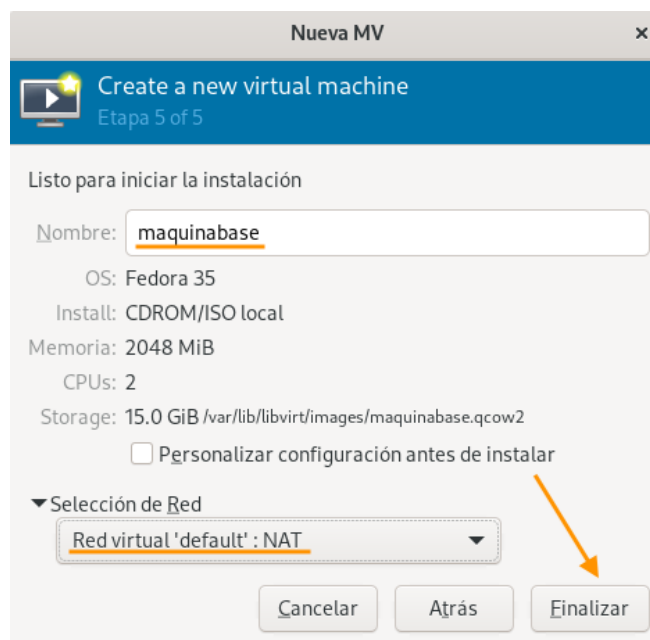


Ilustración 7.8: Paso 5 - Agregar nombre y seleccionar la red

Una vez creada la máquina virtual base, se inicia y se instala el sistema operativo siguiendo los pasos realizados en el [Anexo IV](#) a partir de [este punto](#). Luego, se lleva a cabo una actualización del sistema para dejar la máquina totalmente operativa para su posterior uso con el siguiente comando:

```
[root@fedora ~]# dnf upgrade
```

Después de la preparación inicial de la máquina base, se procede a crear las seis máquinas virtuales necesarias para llevar a cabo dicha infraestructura mediante la clonación de la máquina base. Para clonar una máquina virtual en la plataforma de virtualización de KVM, se puede seguir el apartado “[Clonación de una máquina virtual a través de virt-manager](#)” del [Anexo V](#). No obstante, es un proceso sencillo que basta con seleccionar la máquina virtual que se desea clonar (maquinabase), hacer clic en el botón derecho del ratón y seleccionar la opción de “**Clonar**”. Tras esta acción, se inicia una ventana emergente donde hay que modificar el nombre de la nueva máquina virtual para comenzar con el proceso de clonación.

En la ilustración 7.9 se muestra un ejemplo de la clonación del primer nodo (**Nodo1-servicio**) donde se modifica el nombre de la nueva máquina virtual y se le asigna la red NAT. Del mismo modo que se lleva a cabo esta acción, se procede a crear el resto de máquinas virtuales.

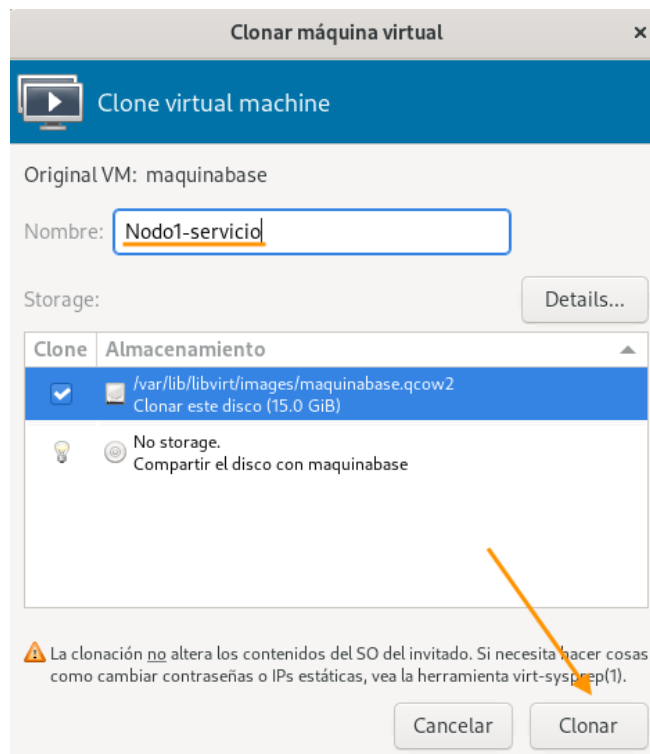


Ilustración 7.9: Clonación de la máquina base

Cabe destacar que las clonaciones se deben realizar **en frío**, es decir, con la máquina virtual a clonar **apagada completamente**.

Asignación y configuración de las interfaces de red

Una vez se hayan creado las cinco máquinas virtuales que hacen referencia a los diferentes nodos del clúster y la máquina virtual que hace referencia al **servidor iSCSI**, se tendrá que asignar las tarjetas de red a cada máquina y realizar los ajustes de configuración pertinentes para cada una de las interfaces de red.

Para asignar las interfaces de red se utiliza también la herramienta “*virt-manager*”. Por lo tanto, se comienza seleccionando la máquina a la que se va a agregar la red y desde la pestaña “Editar” del menú superior, se accede a “Detalles de la máquina virtual” y se abre una ventana emergente. En esta ventana, se hace clic en “Agregar hardware”, se selecciona el tipo de red y se elige el modelo de dispositivo “virtio” si no está indicado.

Para que esta operación no sea muy extensa, se mostrará las asignaciones de cada red aislada únicamente en el “**Nodo1-servicio**”. Para el resto de máquinas virtuales se debe seguir el mismo proceso, dado que todas tienen las mismas redes, exceptuando a la máquina “**iSCSI-almacenamiento**”, que solo hay que asignarle la red de Almacenamiento.

Partiendo de que cada máquina ya tiene asignada la red NAT por defecto, en la ilustración 7.10 se manifiesta como se asigna la red de Almacenamiento al “**Nodo1-servicio**”. De igual manera, se realiza lo mismo para la asignación de la red de Control.

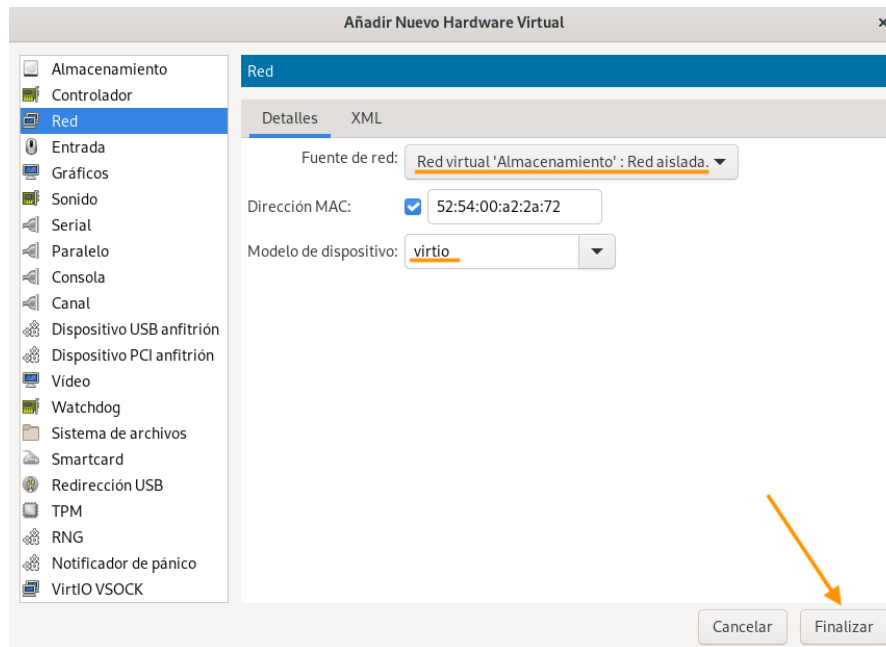


Ilustración 7.10: Asignación de la red de Almacenamiento al **Nodo1-servicio**

En resumidas cuentas, las tarjetas de redes que deberían tener asignadas todos los nodos son tres. Esto se puede verificar en el lateral izquierdo de la ventana “Detalles de la máquina virtual” tras seleccionar la máquina virtual y la pestaña “Editar” en el menú superior de la aplicación “*virt-manager*” (Ver ilustración 7.11).

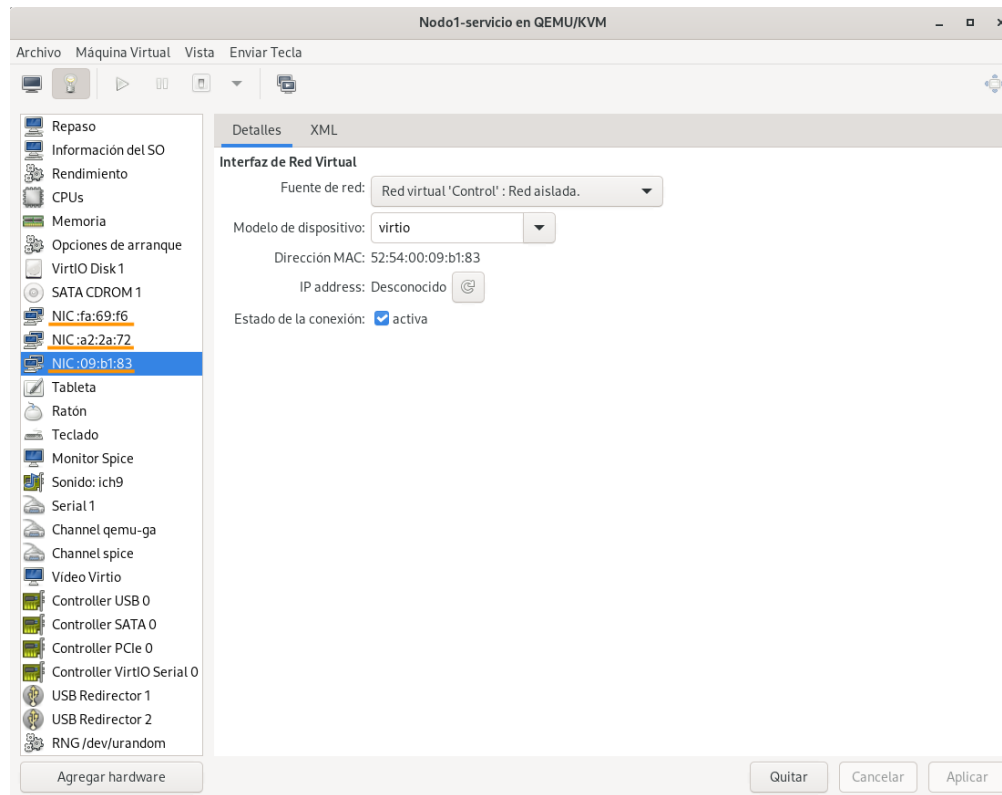


Ilustración 7.11: Comprobación de las redes asignadas al **Nodo1-servicio**

De la misma manera, se puede verificar las interfaces de red que debería tener la máquina “**iSCSI-almacenamiento**”: la red NAT y la red de Almacenamiento.

Como cada máquina virtual tiene asignada diferentes interfaces y por ende, distintas direcciones de red, lo siguiente que hay que realizar son las **configuraciones** correspondientes de cada una de las **interfaces de red** de cada máquina virtual creada. Para llevar a cabo esto, se ha utilizado la utilidad de línea de órdenes **nmcli** de *NetworkManager*.

Al igual que se viene haciendo en demostraciones anteriores, se muestran las gestiones pertinentes de las conexiones de red en el “**Nodo1-servicio**” para simplificar el proceso completo en todas las máquinas. De la misma manera, hay que realizar todos estos pasos en el resto de máquinas virtuales necesarias para dicho proyecto. Entonces, a partir de este punto, se procede a gestionar **nuevas conexiones** de **interfaces de red**.

Cuando se asignan las tarjetas de red a la máquina virtual, se crean las interfaces de red y en los casos en los que el servicio DHCP está habilitado, se configura automáticamente la dirección IP de la red, la puerta de enlace, la máscara de red, etc. En aquellos casos en los que el servicio DHCP no se habilita en la creación de una red, se debe configurar las interfaces manualmente para poder ser utilizadas. Como en este proyecto se ha optado por realizar una configuración estática, se hace uso de la utilidad **nmcli** para realizar dicha configuración en cada máquina virtual.

El nombre de las interfaces de red creadas para **cada nodo del clúster** son:

- Red NAT: **enp1s0**
- Red de Almacenamiento: **enp7s0**
- Red de Control: **enp8s0**

El nombre de las interfaces de red creadas para la máquina “**iSCSI-almacenamiento**” son:

- Red NAT: **enp1s0**
- Red de Almacenamiento: **enp9s0**

Nodo 1 - servicio

En la ilustración 7.12 se muestran todos los pasos necesarios y realizados para configurar la interfaz de red **enp7s0**, correspondiente a la **red de Almacenamiento**.

```
[root@fedora ~]# nmcli device status
DEVICE  TYPE      STATE      CONNECTION
enp1s0  ethernet  conectado  Conexión cableada 1
enp7s0  ethernet  desconectado --
enp8s0  ethernet  desconectado --
lo      loopback  sin gestión --
[root@fedora ~]# nmcli connection add con-name almacenamiento-enp7s0 ifname enp7s0 type ethernet
Conexión «almacenamiento-enp7s0» (b30b59fe-4602-4228-83aa-4b5bb7a117f3) añadida con éxito.
[root@fedora ~]# nmcli device status
DEVICE  TYPE      STATE      CONNECTION
enp1s0  ethernet  conectado  Conexión cableada 1
enp7s0  ethernet  conectando (obteniendo configuración IP) almacenamiento-enp7s0
enp8s0  ethernet  desconectado --
lo      loopback  sin gestión --
[root@fedora ~]# nmcli connection modify almacenamiento-enp7s0 ipv4.addresses '10.22.122.128/24' c
onnection.autoconnect yes ipv4.method manual
```

Ilustración 7.12: Configuración de la interfaz enp7s0 en el **Nodo1-servicio**

Para empezar, se comprueba el estado de las interfaces de red que tiene la máquina virtual asociada con el siguiente comando:

```
[root@fedora ~]# nmcli device status
```

Como se puede apreciar en la salida del comando de la ilustración 7.12, se muestra el listado de los dispositivos de red de la máquina virtual, su estado y los perfiles de conexión que usa cada dispositivo [36]. Entrando en detalle sobre los dispositivos de red, se diferencia que las interfaces de red **enp7s0** y **enp8s0** están resaltadas de color rojo debido a que están desconectadas y no tienen ningún perfil de conexión, ya que aún no se ha realizado ninguna configuración. Por lo tanto, el segundo paso que se manifiesta es para crear un perfil de conexión para la interfaz **enp7s0** con el siguiente comando:

```
[root@fedora ~]# nmcli connection add con-name
→ almacenamiento-enp7s0 ifname enp7s0 type ethernet
```

En el comando se indica el nombre de la conexión a crear (**almacenamiento-enp7s0**), el nombre de la interfaz a la que se asocia dicha conexión (**enp7s0**) y el tipo de dispositivo/interfaz (**ethernet**). En este caso el tipo de interfaz es Ethernet debido a que el sistema anfitrión está conectado a la red mediante un cable de red Ethernet. Sin embargo, si el equipo físico estuviera conectado a la red a través de una conexión Wi-Fi, el procedimiento que se está realizando para gestionar las interfaces de red sería completamente diferente.

En la ilustración 7.12 se puede percibir que, tras volver a ejecutar el comando que muestra el estado de los dispositivos de red, la línea de la interfaz **enp7s0** se encuentra ahora en color amarillo/naranja y el estado a pasado a estar en “conectando (obteniendo configuración IP)”. Además, se puede diferenciar que se ha añadido el nombre correspondiente del perfil de conexión recién creado.

Para completar el procedimiento de configuración de la interfaz de red **enp7s0** (red de Almacenamiento) que se distingue en la ilustración 7.12, se configura los ajustes IP de la red indicando la dirección IP y su máscara correspondiente, el método de IPv4 (automático/dinámico o manual/estático) y la conexión automática de la interfaz, tal y como se realiza en el comando posterior:

```
[root@fedora ~]# nmcli connection modify almacenamiento-enp7s0
→ ipv4.addresses '10.22.122.128/24' connection.autoconnect yes
→ ipv4.method manual
```

Acto seguido, en la ilustración 7.13 se reflejan los mismos pasos efectuados anteriormente para configurar ahora, la interfaz de red **enp8s0**, la cual corresponde a la **red de Control**.

```
[root@fedora ~]# nmcli device status
DEVICE  TYPE      STATE      CONNECTION
enp1s0  ethernet  conectado  Conexión cableada 1
enp7s0  ethernet  conectado  almacenamiento-enp7s0
enp8s0  ethernet  desconectado --
lo      loopback  sin gestión --
[root@fedora ~]# nmcli connection add con-name control-enp8s0 ifname enp8s0 type ethernet
Conexión «control-enp8s0» (340fe5a5-32c6-45dc-b3ba-5f1cb7dbc64c) añadida con éxito.
[root@fedora ~]# nmcli device status
DEVICE  TYPE      STATE      CONNECTION
enp1s0  ethernet  conectado  Conexión cableada 1
enp7s0  ethernet  conectado  almacenamiento-enp7s0
enp8s0  ethernet  conectando (obteniendo configuración IP) control-enp8s0
lo      loopback  sin gestión --
[root@fedora ~]# nmcli connection modify control-enp8s0 ipv4.addresses '10.22.132.128/24' connection.autoconnect yes ipv4.method manual
```

Ilustración 7.13: Configuración de la interfaz enp8s0 en el **Nodo1-servicio**

Una vez se hayan configurado las dos interfaces de red (enp7s0 y enp8s0), se comprueba que el estado de ambas sea “conectado”, que estén resaltadas en verde, indicando de la misma manera que están conectadas y que aparezca el nombre del perfil de la conexión:

```
[root@fedora ~]# nmcli device status
DEVICE  TYPE      STATE      CONNECTION
enp1s0  ethernet  conectado  Conexión cableada 1
enp7s0  ethernet  conectado  almacenamiento-enp7s0
enp8s0  ethernet  conectado  control-enp8s0
lo       loopback  sin gestión  --
```

Ilustración 7.14: Verificación de las interfaces enp7s0 y enp8s0 en el **Nodo1-servicio**

A continuación, en la ilustración 7.15 se verifica con el comando *ifconfig*⁴ que las direcciones IP de cada interfaz se hayan asignado correctamente y al mismo tiempo, se revisa que a la interfaz **enp1s0** (red NAT) se le haya asignado automáticamente la dirección IP.

```
[root@fedora ~]# ifconfig
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.122.113 netmask 255.255.255.0  broadcast 192.168.122.255
    inet6 fe80::c538:4165:bbb:211f prefixlen 64  scopeid 0x20<link>
    ether 52:54:00:fa:69:f6 txqueuelen 1000  (Ethernet)
    RX packets 1065 bytes 82686 (80.7 KiB)
    RX errors 0 dropped 898 overruns 0 frame 0
    TX packets 351 bytes 31504 (30.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp7s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.22.122.128 netmask 255.255.255.0  broadcast 10.22.122.255
    inet6 fe80::715f:bdd6:e914:aaa3 prefixlen 64  scopeid 0x20<link>
    ether 52:54:00:a2:2a:72 txqueuelen 1000  (Ethernet)
    RX packets 900 bytes 47036 (45.9 KiB)
    RX errors 0 dropped 898 overruns 0 frame 0
    TX packets 589 bytes 88715 (86.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp8s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.22.132.128 netmask 255.255.255.0  broadcast 10.22.132.255
    inet6 fe80::5caa:a568:4ef5:add4 prefixlen 64  scopeid 0x20<link>
    ether 52:54:00:09:b1:83 txqueuelen 1000  (Ethernet)
    RX packets 901 bytes 47106 (46.0 KiB)
    RX errors 0 dropped 898 overruns 0 frame 0
    TX packets 481 bytes 79821 (77.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 36 bytes 4226 (4.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 36 bytes 4226 (4.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Ilustración 7.15: Verificación de las direcciones IP del **Nodo1-servicio**

Para cerciorarse de que las redes se han creado y configurado correctamente, se llevan a cabo algunas comprobaciones. Primero, se verifica que la máquina “**Nodo1-servicio**” tiene **conectividad al exterior** mediante la interfaz **enp1s0** (red NAT) con la siguiente orden:

```
[root@fedora ~]# ping -I enp1s0 8.8.8.8
```

⁴Herramienta para gestionar las interfaces de red en los sistemas operativos Linux.

En el comando anterior se ha añadido la opción **-I** para indicar la interfaz por la que se quiere realizar el *ping*. También se puede comprobar esta operación suprimiendo la opción indicada y el *ping* se realiza correctamente de igual manera porque buscaría la interfaz por la que puede salir al exterior. El *ping* también puede realizarse por el nombre de dominio de algún sitio web, por ejemplo: `www.google.es`.

La siguiente prueba que se realizará es para asegurarse de que las **redes aisladas no tienen conectividad hacia el exterior**, por lo tanto, se ejecutan las siguientes órdenes:

```
[root@fedora ~]# ping -I enp7s0 8.8.8.8
[root@fedora ~]# ping -I enp8s0 8.8.8.8
```

Estas órdenes no deberían de responder el *ping* ya que una red aislada no está conectada a ninguna otra red. En la ilustración 7.16 se puede ver que se han efectuado dos *ping* con las interfaces de las redes aisladas y ninguno de los dos han respondido, con lo que se puede concluir que están correctamente configuradas.

```
[root@fedora ~]# ping -I enp8s0 www.xataka.com
PING d2t8dj4tr3q9od.cloudfront.net (108.157.96.112) from 10.22.132.128 enp8s0: 5
6(84) bytes of data.
^C
--- d2t8dj4tr3q9od.cloudfront.net ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1017ms

[root@fedora ~]# ping -I enp7s0 www.xataka.com
PING d2t8dj4tr3q9od.cloudfront.net (108.157.96.116) from 10.22.122.128 enp7s0: 5
6(84) bytes of data.
^C
--- d2t8dj4tr3q9od.cloudfront.net ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

Ilustración 7.16: Demostración de la conexión de las redes aisladas del **Nodo1-servicio**

Otra prueba que se efectuará es para asegurarse de la **conectividad** de las **redes aisladas** hacia la **puerta de enlace** o a una de las **máquinas de su red** (se realizará cuando se hayan configurado las interfaces de las otras máquinas). En las ilustraciones 7.17 y 7.18 se ejecuta un *ping* desde las interfaces de las redes aisladas a su correspondiente puerta de enlace, verificando que responden correctamente.

```
[root@fedora ~]# ping -I enp7s0 10.22.122.1
PING 10.22.122.1 (10.22.122.1) from 10.22.122.128 enp7s0: 56(84) bytes of data.
64 bytes from 10.22.122.1: icmp_seq=1 ttl=64 time=0.247 ms
64 bytes from 10.22.122.1: icmp_seq=2 ttl=64 time=0.376 ms
64 bytes from 10.22.122.1: icmp_seq=3 ttl=64 time=0.254 ms
64 bytes from 10.22.122.1: icmp_seq=4 ttl=64 time=0.263 ms
^C
--- 10.22.122.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3072ms
rtt min/avg/max/mdev = 0.247/0.285/0.376/0.052 ms
```

Ilustración 7.17: Demostración de la conexión de la interfaz `enp7s0` del **Nodo1-servicio**

```
[root@fedora ~]# ping -I enp8s0 10.22.132.1
PING 10.22.132.1 (10.22.132.1) from 10.22.132.128 enp8s0: 56(84) bytes of data.
64 bytes from 10.22.132.1: icmp_seq=1 ttl=64 time=0.232 ms
64 bytes from 10.22.132.1: icmp_seq=2 ttl=64 time=0.275 ms
64 bytes from 10.22.132.1: icmp_seq=3 ttl=64 time=0.253 ms
^C
--- 10.22.132.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2069ms
rtt min/avg/max/mdev = 0.232/0.253/0.275/0.017 ms
```

Ilustración 7.18: Demostración de la conexión de la interfaz enp8s0 del **Nodo1-servicio**

Y para finalizar con las pruebas respecto al nodo “**Nodo1-servicio**”, se comprueba que el **sistema anfitrión** tiene **conectividad** con las **redes aisladas**, tal y como se figura en las ilustraciones 7.19 y 7.20.

```
[root@pcl-tft ~]# ping 10.22.122.128
PING 10.22.122.128 (10.22.122.128) 56(84) bytes of data.
64 bytes from 10.22.122.128: icmp_seq=1 ttl=64 time=0.358 ms
64 bytes from 10.22.122.128: icmp_seq=2 ttl=64 time=0.295 ms
^C
--- 10.22.122.128 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1062ms
rtt min/avg/max/mdev = 0.295/0.326/0.358/0.031 ms
```

Ilustración 7.19: Demostración de la conectividad del anfitrión mediante la interfaz enp7s0 al **Nodo1-servicio**

```
[root@pcl-tft ~]# ping 10.22.132.128
PING 10.22.132.128 (10.22.132.128) 56(84) bytes of data.
64 bytes from 10.22.132.128: icmp_seq=1 ttl=64 time=0.390 ms
64 bytes from 10.22.132.128: icmp_seq=2 ttl=64 time=0.391 ms
64 bytes from 10.22.132.128: icmp_seq=3 ttl=64 time=0.266 ms
^C
--- 10.22.132.128 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
rtt min/avg/max/mdev = 0.266/0.349/0.391/0.058 ms
```

Ilustración 7.20: Demostración de la conectividad del anfitrión mediante la interfaz enp8s0 al **Nodo1-servicio**

Resto de nodos del clúster

En el resto de nodos, se realizan los mismos pasos que se han llevado a cabo en el “**Nodo1-servicio**” para configurar las dos interfaces de red (**enp7s0** y **enp8s0**), pero asignando direcciones IP diferentes.

Del mismo modo que se han realizado las comprobaciones en el “**Nodo1-servicio**”, se efectúan en el resto de nodos para verificar que haya conectividad hacia el exterior por la interfaz **enp1s0**, que no haya conectividad al exterior por las interfaces **enp7s0** y **enp8s0**, que haya conectividad hacia la puerta de enlace o cualquier otra máquina de las interfaces de las redes aisladas y que el anfitrión se pueda comunicar con las máquinas mediante las redes aisladas.

iSCSI almacenamiento

Para concluir con las configuraciones pertinentes de las interfaces de red de cada máquina virtual, se configura la interfaz **enp9s0** de la máquina “**iSCSI-almacenamiento**” y se comprueba que se haya asignado la dirección IP correspondiente. Cabe mencionar que esta máquina virtual, como se ha mencionado previamente, solo tiene asignada dos interfaces de red (red NAT - **enp1s0** y red Almacenamiento - **enp9s0**).

Después de configurar todas las interfaces de red de todas las máquinas, se efectúa la última prueba con el comando *ping* para verificar que la conexión entre las máquinas de cada red aislada es correcta. En la ilustración 7.21 se ha ejecutado correctamente un ping desde el “Nodo1-servicio” (**enp7s0 - 10.22.122.128**) a través de la interfaz **enp7s0** (**red de almacenamiento - 10.22.122.0/24**) a la dirección **10.22.122.131** (Nodo4-almacenamiento).

```
[root@fedora ~]# ping -I enp7s0 10.22.122.131
PING 10.22.122.131 (10.22.122.131) from 10.22.122.128 enp7s0: 56(84) bytes of data.
64 bytes from 10.22.122.131: icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from 10.22.122.131: icmp_seq=2 ttl=64 time=0.508 ms
64 bytes from 10.22.122.131: icmp_seq=3 ttl=64 time=0.478 ms
^C
--- 10.22.122.131 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2064ms
rtt min/avg/max/mdev = 0.263/0.416/0.508/0.109 ms
```

Ilustración 7.21: Verificación de la conectividad entre las máquinas de la red de Almacenamiento

En la ilustración 7.22 se ha ejecutado exactamente lo mismo que en la ilustración 7.21, desde el “Nodo1-servicio” (**enp8s0 - 10.22.132.128**) pero a través de la interfaz **enp8s0** (**red de control - 10.22.132.0/24**) a la dirección IP **10.22.132.131** (Nodo4-almacenamiento).

```
[root@fedora ~]# ping -I enp8s0 10.22.132.131
PING 10.22.132.131 (10.22.132.131) from 10.22.132.128 enp8s0: 56(84) bytes of data.
64 bytes from 10.22.132.131: icmp_seq=1 ttl=64 time=0.959 ms
64 bytes from 10.22.132.131: icmp_seq=2 ttl=64 time=0.470 ms
64 bytes from 10.22.132.131: icmp_seq=3 ttl=64 time=0.502 ms
^C
--- 10.22.132.131 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2012ms
rtt min/avg/max/mdev = 0.470/0.643/0.959/0.223 ms
```

Ilustración 7.22: Verificación de la conectividad entre las máquinas de la red de Control

Con todo esto, se puede concluir que las redes aisladas han sido creadas y configuradas correctamente y que la red NAT también funciona adecuadamente, ofreciendo conectividad al exterior.

A modo de resumen, en la tabla 7.1 se presenta una **recapitulación de todas las interfaces** de red que tiene asignada cada máquina virtual:

Máquinas virtuales	Interfaces de red		
	Red NAT	Red de Almacenamiento	Red de Control
	enp1s0	enp7s0/enp9s0	enps80
Nodo1-servicio	192.168.122.113	10.22.122.128	10.22.132.128
Nodo2-servicio	192.168.122.7	10.22.122.129	10.22.132.129
Nodo3-almacenamiento	192.168.122.234	10.22.122.130	10.22.132.130
Nodo4-almacenamiento	192.168.122.17	10.22.122.131	10.22.132.131
Balanceador-de-carga	192.168.122.246	10.22.122.132	10.22.132.132
iSCSI-almacenamiento	192.168.122.135	10.22.122.133	10.22.132.133

Cuadro 7.1: Resumen de las interfaces de red de cada máquina virtual.

Establecimiento del nombre de dominio de cada máquina

Tras la asignación de las diferentes redes a las seis máquinas virtuales y las distintas comprobaciones, realizadas para comprobar la conectividad al exterior y la conectividad entre las máquinas virtuales de las redes aisladas, se procede a emprender la última fase de la creación de la infraestructura básica del clúster.

La última fase consiste en establecer el nombre de dominio completamente cualificado para cada una de las máquinas virtuales a utilizar, tanto para los nodos del clúster como para la máquina “**iSCSI-almacenamiento**”. Esto se consigue cambiando el *hostname* (nombre del host) de cada máquina virtual y modificando el fichero “**/etc/hosts**”, donde se indica una relación entre un nombre de dominio y una dirección IP (de una de las interfaces de red asignadas previamente de las máquinas virtuales).

Así pues, para añadir un nuevo nombre de dominio a cada máquina virtual se utiliza la orden “`hostnamectl`” de la siguiente manera:

- * Para el nodo “**Nodo1-servicio**” se establecerá el nombre de host de la siguiente manera:

```
[root@fedora ~]# hostnamectl set-hostname nodo1-tft.com
```

Este comando se ejecuta en todos los nodos del clúster para modificar de la misma manera el *hostname* con sus respectivos nombres.

Una vez modificados los nombres de dominio, se reinician todas las máquinas para que se apliquen los cambios correctamente. Para corroborar que dicho nombre se haya cambiado

en cada máquina, se puede ejecutar el comando *hostname* o ver el contenido del fichero “/etc/hostname”.

Para poder resolver los nombres de dominio de cada máquina en cada una de las máquinas o nodos, se debe añadir en el fichero “/etc/hosts” una relación entre los nombres de dominio y las direcciones IPs de las máquinas que intervienen en el clúster.

En la ilustración 7.2 se muestra un esquema de las conexiones de red que tiene cada una de las máquinas (nodos). Como se puede observar, cada nodo se puede comunicar con cualquier nodo del clúster y por cualquiera de las tres interfaces de red que tiene asignadas. En los ficheros “/etc/hosts” se puede añadir un mismo nombre de dominio con diferentes direcciones IP, pero siempre que se realice una petición (por ejemplo, cuando se realice un *ping*) a ese nombre de dominio, responderá la primera línea (dirección IP) que esté escrita en dicho fichero. Por lo tanto, lo que interesa en tales circunstancias es que los nodos del clúster se comuniquen por la red de **Control**.

Con lo cual, las relaciones que se agregarán al fichero “/etc/hosts/” de todos los nodos serán el nombre de dominio de cada nodo y la dirección IP de la red aislada de Control (10.22.132.0/24). Además, como la máquina virtual “**iSCSI-almacenamiento**” simula un servidor iSCSI que permite un espacio de almacenamiento compartido y distribuido para los nodos del clúster, el fichero “/etc/hosts/” de todos los nodos del clúster también debe indicar el nombre de dominio de la máquina “**iSCSI-almacenamiento**” y la dirección IP de la red aislada de Almacenamiento (10.22.122.0/24) para permitir el tráfico de acceso al almacenamiento compartido.

Por lo tanto, el fichero “/etc/hosts/” de la máquina “**iSCSI-almacenamiento**” debe indicar los nombres de dominio de todos los nodos del clúster con su dirección IP correspondiente de la red aislada de Almacenamiento.

El contenido del fichero “/etc/hosts” de todos los nodos del clúster queda de la manera en la que se indica en el “**Nodo1-servicio**”, tal y como se expone en la ilustración 7.23.

```
[root@nodo1-tft ~]# cat /etc/hosts
# Loopback entries; do not change.
# For historical reasons, localhost precedes localhost.localdomain:
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1        localhost localhost.localdomain localhost6 localhost6.localdomain6
# See hosts(5) for proper format and other examples:
# 192.168.1.10 foo.mydomain.org foo
# 192.168.1.13 bar.mydomain.org bar
10.22.132.128 nodo1-tft.com
10.22.132.129 nodo2-tft.com
10.22.132.130 nodo3-tft.com
10.22.132.131 nodo4-tft.com
10.22.132.132 balanceador-tft.com
10.22.122.133 iscsi-tft.com
```

Ilustración 7.23: Modificación del fichero /etc/hosts del “**Nodo1-servicio**”

El contenido del fichero “/etc/hosts” de la máquina “**iSCSI-almacenamiento**” queda de la manera en la que se indica en la ilustración 7.24.


```
[root@iscsi-tft ~]# cat /etc/hosts
# Loopback entries; do not change.
# For historical reasons, localhost precedes localhost.localdomain:
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1        localhost localhost.localdomain localhost6 localhost6.localdomain6
# See hosts(5) for proper format and other examples:
# 192.168.1.10 foo.mydomain.org foo
# 192.168.1.13 bar.mydomain.org bar
10.22.122.128 nodo1-tft.com
10.22.122.129 nodo2-tft.com
10.22.122.130 nodo3-tft.com
10.22.122.131 nodo4-tft.com
10.22.122.132 balanceador-tft.com
10.22.122.133 iscsi-tft.com
```

Ilustración 7.24: Modificación del fichero `/etc/hosts` de la máquina **iSCSI-almacenamiento**

Se puede observar en las ilustraciones anteriores que tras el reinicio de las máquinas, el *prompt*⁵ de la terminal de cada una de las máquinas, muestra el nombre de dominio correspondiente (**nodo1-tft** y **iscsi-tft**, respectivamente).

A modo de prueba, para ultimar esta fase, se comprueba la conectividad en las seis máquinas usando los nombres de dominio completamente cualificados y recién modificados de todas las máquinas. Como se ha venido haciendo en pasos anteriores, para comprobar la conectividad entre todas las máquinas, se utiliza el comando *ping* y se ejecuta la siguiente orden en cada una de las máquinas para todos los *hostname* (nodo1-tft.com, nodo2-tft.com, nodo3-tft.com, nodo4-tft.com y balanceador-tft.com):

```
[root@nodo1-tft ~]# ping nodo1-tft.com
```

Hay que fijarse en que los resultados de cada una de las órdenes ejecutadas se resuelvan los nombres de dominio de cada máquina satisfactoriamente, respondiendo además con las direcciones IP correspondientes a cada máquina.

7.3.2.2. Instalación y configuración del servidor Apache

En esta sección se llevará a cabo la instalación y puesta en marcha del servicio Apache en los nodos “**Nodo1-servicio**” y “**Nodo2-servicio**”. Para ello, primeramente hay que instalar el servicio Apache en los dos nodos mencionados con el siguiente comando:

```
[root@nodo1-tft ~]# dnf -y install httpd
```

Una vez instalado, el siguiente paso es iniciar y habilitar el demonio **httpd** para que siempre que se arranque el sistema, este demonio se inicie automáticamente. Este paso también se realiza en ambos nodos. Por lo tanto, se ejecutan los dos comandos siguientes:

```
[root@nodo1-tft ~]# systemctl start httpd
[root@nodo1-tft ~]# systemctl enable httpd
[root@nodo1-tft ~]# systemctl status httpd
```

⁵Conjunto de caracteres que aparecen en la terminal del sistema operativo para señalar que está esperando a recibir cualquier tipo de comando.

El primero supone iniciar el **httpd**, el segundo conlleva habilitarlo para que se quede activado siempre y el tercero muestra el estado del demonio después de haberlo iniciado y habilitado. Con este último, se comprueba que se haya iniciado correctamente (sin errores) para poder verificar que el servicio está funcionando adecuadamente [37].

El próximo paso es añadir el servicio Apache al *firewall*. En Fedora Workstation 35 se utiliza el demonio del servicio *firewall* (**firewalld**) para modificar el cortafuegos de los equipos debido a la capacidad que tiene de ser dinámico y personalizable. Al ser dinámico, se puede gestionar las reglas sin tener que reiniciar el demonio del *firewall* o reiniciar el equipo.

Por consiguiente, se añade el servicio Apache al *firewall* para permitir que dicho servicio proporcione solicitudes a través de **HTTP** y **HTTPS** mediante los puertos 80 y 443. Antes de comenzar a modificar el *firewall*, se comprueba el listado del *firewall* que viene configurado por defecto cuando se instala el sistema operativo. Como se puede ver en la ilustración 7.25 resaltado de color amarillo, se listan las interfaces que tiene asignadas la máquina virtual, los servicios que están añadidos y todos los puertos que se encuentran abiertos por defecto.

```
[root@nodo1-tft ~]# firewall-cmd --list-all
FedoraWorkstation (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp1s0 enp7s0 enp8s0
  sources:
  services: dhcpv6-client mdns samba-client ssh
  ports: 1025-65535/udp 1025-65535/tcp
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Ilustración 7.25: Listado del *firewall* por defecto en el “Nodo1-servicio”

Dado que el *firewall* usualmente se configura de manera restringida, es decir, bloqueando todo (reglas, puertos, servicios, etc.) menos lo que se permite específicamente, se procede a eliminar todo el rango de puertos que está abierto, debido a que no se utilizarán todos estos.

Se puede verificar en la ilustración 7.26, con los dos primeros comandos, que se han eliminado de manera permanente los puertos con rango 1025-65535 y con el protocolo TCP y UDP. Seguidamente, se muestra un tercer comando ejecutado para recargar la información recién modificada y un cuarto comando para presentar el estado final del *firewall* después del cambio.

```
[root@nod01-tft ~]# firewall-cmd --permanent --remove-port=1025-65535/udp
success
[root@nod01-tft ~]# firewall-cmd --permanent --remove-port=1025-65535/tcp
success
[root@nod01-tft ~]# firewall-cmd --reload
success
[root@nod01-tft ~]# firewall-cmd --list-all
FedoraWorkstation (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp1s0 enp7s0 enp8s0
  sources:
  services: dhcpv6-client mdns samba-client ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Ilustración 7.26: Eliminación del rango de puertos por defecto del *firewall* en el **Nodo1-servicio**

Una vez eliminado el rango de puertos, se procede a agregar el servicio de Apache tanto en el “**Nodo1-servicio**” como en el “**Nodo2-servicio**”.

El servicio Apache (HTTP y HTTPS) se ha añadido de forma permanente en el “**Nodo1-servicio**” y se ha ejecutado el tercer comando para que se recargue la información modificada y las nuevas reglas entren en vigor de manera automática sin tener que reiniciar el demonio **firewalld** o la máquina. Se puede ver que se ha añadido correctamente los servicios HTTP y HTTPS tras la ejecución del último comando que aparece en la ilustración 7.27.

```
[root@nod01-tft ~]# firewall-cmd --permanent --add-service http
success
[root@nod01-tft ~]# firewall-cmd --permanent --add-service https
success
[root@nod01-tft ~]# firewall-cmd --reload
success
[root@nod01-tft ~]# firewall-cmd --list-all
FedoraWorkstation (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp1s0 enp7s0 enp8s0
  sources:
  services: dhcpv6-client http https mdns samba-client ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Ilustración 7.27: Añadiendo el servicio Apache al *firewall* en el **Nodo1-servicio**

Del mismo modo que se han eliminado los puertos y se ha añadido el servicio, se ejecutan los pasos para el “**Nodo2-servicio**”. En este momento, ya se puede comprobar que el servicio Apache funciona correctamente en el “**Nodo1-servicio**” y en el “**Nodo2-servicio**”. Esto se puede corroborar desde el sistema anfitrión, accediendo al navegador *Firefox* con la dirección IP de la red aislada de Almacenamiento de ambos nodos. En la ilustración 7.28 se puede ver

que funciona correctamente accediendo con la dirección IP del **Nodo2-servicio**.



Ilustración 7.28: Verificación del funcionamiento de Apache en el **anfitrión**

7.3.2.3. Instalación y configuración del HAProxy

En este epígrafe se continúa con la instalación y configuración del **HAProxy**, un *software* que ofrece un servicio de equilibrador de carga, alta disponibilidad y *proxy* para servicios HTTP y TCP, por ejemplo, para servicios conectados a Internet y servicios webs con un tráfico muy alto para que se distribuyan las multitudes de solicitudes entre diferentes servidores [15].

De manera predefinida, HAProxy utiliza el puerto 80. HAProxy permite definir diferentes servicios de *proxy* y desarrollar servicios de equilibrador de carga para los servidores de *proxy*. Estos *proxies* se configuran con un método llamado *frontend* y con uno o varios métodos denominados *backend*. El **frontend** representa el *proxy*, por lo tanto, en dicha configuración se indica la dirección IP, el puerto por el que escucha el *proxy* y el protocolo (HTTP o TCP). Sin embargo, el **backend** es un conjunto de servidores o *proxies* "reales" donde se establece el algoritmo del equilibrador de carga y se indica los servidores que repartirán la carga. Existen diversos algoritmos de equilibrador de carga: *Round-Robin* (roundrobin), *Static Round-Robin* (static-rr), *Least-Connection* (leastconn), *Source* (source), *URI* (uri), *URL Parameter* (url_param), *Header Name* (hdr) y *RDP Cookie* (rdp-cookie) [15].

En lo relativo a este proyecto, el *frontend* constituye la configuración del servicio **HAProxy** (dirección IP, puerto, modo, etc.) que será instalado en un nodo del clúster que ha sido creado para este fin ("**Balanceador-de-carga**"). Y, el *backend* representa los servidores reales, que en este caso son los nodos que darán el servicio web basado en Apache. En esta configuración hay que indicar los servidores en los que se repartirá la carga del servicio web y el algoritmo de programación que se utiliza para el equilibrio de carga.

Una vez precisada la información necesaria para continuar con la configuración del HAProxy, se procede a instalar dicho servicio con el siguiente comando en el nodo “**Balancedor-de-carga**”:

```
[root@balanceador-tft ~]# dnf -y install haproxy
```

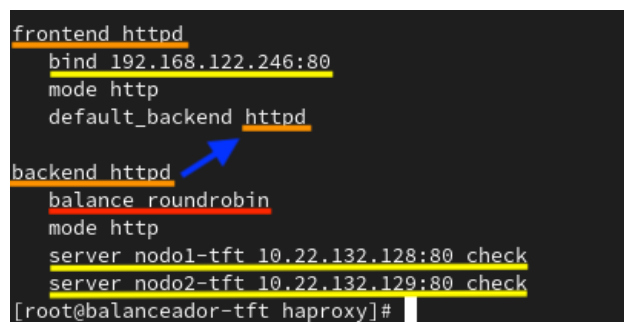
El siguiente paso es iniciar y activar el servicio de HAProxy para poder utilizarlo y para que siempre se inicie al arrancar el nodo que lo posee (“**Balancedor-de-carga**”). Para realizar esto, se ejecutan los comandos que se muestran a continuación, iniciando el servicio con el primero y habilitándolo con el segundo:

```
[root@balanceador-tft ~]# systemctl start haproxy
[root@balanceador-tft ~]# systemctl enable haproxy
```

Si se quiere comprobar que el servicio se ha iniciado correctamente, sin ningún tipo de errores, se puede ejecutar el siguiente comando para verificar su estado:

```
[root@balanceador-tft ~]# systemctl status haproxy
```

Una vez iniciado el servicio, hay que comenzar a realizar la configuración correspondiente para que el HAProxy pueda llevar a cabo su cometido, equilibrar la carga del tráfico entrante a uno de los servidores (máquinas o nodos) web indicados. Como se ha mencionado anteriormente, la configuración pertinente para el HAProxy es la generación de un sistema *frontend* para identificar al *proxy* y un sistema *backend* para representar, en este caso, las máquinas virtuales que van a ofrecer el servicio de manera balanceada. Esta configuración se indica en el fichero “**/etc/haproxy/haproxy.cfg**”, con lo cual, se accede a dicho fichero en la máquina “**Balancedor-de-carga**” y se añade las directivas *frontend* y *backend* con su respectiva configuración, tal y como se muestra señalado con el color naranja en la ilustración 7.29.



```
frontend httpd
  bind 192.168.122.246:80
  mode http
  default_backend httpd

backend httpd
  balance roundrobin
  mode http
  server nodo1-tft 10.22.132.128:80 check
  server nodo2-tft 10.22.132.129:80 check
[root@balanceador-tft haproxy]#
```

Ilustración 7.29: Modificación del fichero haproxy.cfg para agregar **HTTP**

El *frontend* llamado **httpd** está configurado con la dirección IP **192.168.122.246** y escucha en el puerto **80** usando el parámetro “*bind*” (resaltado en color amarillo). Si se vuelve a contemplar la ilustración 7.2, la dirección IP especificada en la directiva del frontend corresponde a la dirección IP de la red NAT del nodo “**Balancedor-de-carga**”, la cual es la dirección que sale al exterior y por ende, por la que los usuarios podrán acceder al servidor web.

La configuración del *backend*, también llamado **httpd**, especifica con el parámetro “*server*” (resaltado en color amarillo) las direcciones IP reales de los servidores que contienen el servicio de Apache: para el “**Nodo1-servicio**”, la dirección IP **10.22.132.128** y para el “**Nodo2-servicio**”, la dirección IP **10.22.132.129**. A ambos nodos se le indica también el puerto por el que escuchan (80). Además, se precisa con el parámetro “*balance*” (señalado en color rojo) el algoritmo **Round-Robin**, el cual será el encargado de gestionar el balanceo de carga del servidor web basado en Apache.

La elección de este algoritmo es debido a su uso común. El algoritmo **Round-Robin** reparte cada solicitud entre los servidores reales (máquinas virtuales, en este caso) respecto a los pesos asignados. Si no se indica el parámetro “*weight*”, el peso que se les asigna por defecto a todas las máquinas es de 1. Con este algoritmo todas las máquinas virtuales se tratan por igual, sin tener en cuenta la carga de conexiones existentes o la capacidad de procesamiento de cada una [15].

En ambas directivas se indican con el parámetro “*mode*” el protocolo **HTTP** para la nueva instancia del HAProxy. Una vez creadas ambas directivas, en el *frontend* se añade el parámetro “*default.backend*” para proporcionarle al *frontend* el nombre del *backend*.

De igual forma que se ha eliminado el rango de puertos del *firewall* en el “**Nodo1-servicio**” y “**Nodo2-servicio**”, se procederá a eliminar el mismo rango de puertos en el nodo “**Balanceador-de-carga**”. Cuando se haya eliminado, el siguiente paso es abrir los puertos de escucha del HAProxy. Como se ha indicado anteriormente, el puerto por el que escucha es el 80, el cual corresponde al protocolo HTTP, por lo tanto, se procederá a agregar dicho servicio al *firewall* del nodo “**Balanceador-de-carga**” (Ver ilustración 7.30).

```
[root@balanceador-tft ~]# firewall-cmd --permanent --add-service http
success
[root@balanceador-tft ~]# firewall-cmd --reload
success
[root@balanceador-tft ~]# firewall-cmd --list-all
FedoraWorkstation (active)
target: default
icmp-block-inversion: no
interfaces: enp1s0 enp7s0 enp8s0
sources:
services: dhcpv6-client http mdns samba-client ssh
ports:
protocols:
forward: no
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
```

Ilustración 7.30: Abriendo el puerto del HAProxy al *firewall* en el **Balanceador-de-carga**

En la ilustración 7.30 se puede verificar que, aparte de haberse agregado el servicio HTTP, se ha eliminado el rango de puertos.

Para que las configuraciones realizadas en el fichero de configuración de HAProxy pasen a estar operativas, se debe reiniciar dicho servicio:

```
[root@balanceador-tft ~]# systemctl restart haproxy
```

En este momento, si tras el reinicio del HAProxy no ha ocurrido ningún tipo de problema, ya está disponible el balanceo de carga del servicio web basado en Apache. Entonces, para probarlo solo hay que acceder desde el sistema anfitrión al navegador por defecto de Fedora (*Firefox*) y realizar una búsqueda de la dirección IP del nodo “**Balancedor-de-carga**” para poder comprobar que se muestra la página de prueba de Apache (**Ver ilustración 7.31**).

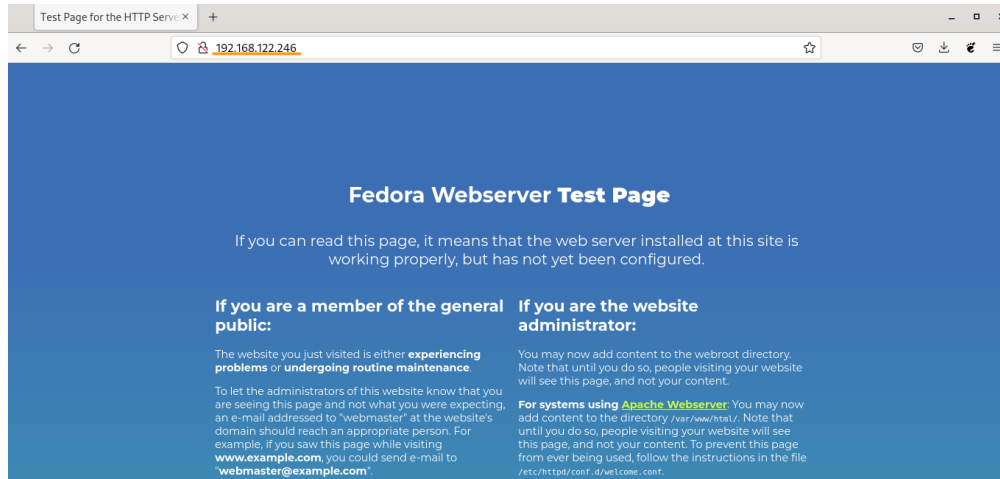


Ilustración 7.31: Verificación del correcto funcionamiento de Apache

Para verificar que el balanceo de carga funciona adecuadamente hay que determinar qué nodo es el que está ofreciendo el servicio en cada momento. Esto se puede resolver creando o modificando (si ya existe) el fichero “**index.html**” en el directorio “**/var/www/html**” de cada uno de los nodos que ofrece el servicio para que Apache pueda servirlo al navegador que lo solicite (con la dirección IP). En las ilustraciones 7.32 y 7.33 se revisa que el directorio en el que se encuentra situado sea el correspondiente a “**/var/www/html**” y se crea el fichero “**index.html**” añadiendo una línea en HTML que indica el nodo que está dando el servicio dependiendo de la máquina en la que se esté ejecutando.

```
[root@nodo1-tft html]# pwd
/var/www/html
[root@nodo1-tft html]# vi index.html
[root@nodo1-tft html]# cat index.html
<h1>Está dando el servicio el nodo1-tft.com</h1>
[root@nodo1-tft html]#
```

Ilustración 7.32: Creación del fichero index.html en el **Nodo1-servicio**

```
[root@nodo2-tft html]# pwd
/var/www/html
[root@nodo2-tft html]# vi index.html
[root@nodo2-tft html]# cat index.html
<h1>Está dando el servicio el nodo2-tft.com</h2>
[root@nodo2-tft html]#
```

Ilustración 7.33: Creación del fichero index.html en el **Nodo2-servicio**

A continuación, se vuelve a probar el servicio accediendo con la dirección IP del nodo

“**Balancedor-de-carga**” desde el anfitrión para ver qué nodo es el que está dando el servicio. En primera instancia lo da el “**Nodo1-servicio**” (Ver ilustración 7.34).



Ilustración 7.34: Verificación del balanceo de carga con el **Nodo1-servicio**

Luego se recarga la página haciendo clic en el icono “↻” para comprobar que el balanceo de carga funciona adecuadamente y que el servicio está siendo ofrecido por ambos nodos de manera alternada. Y, en segunda instancia, lo da el ‘**Nodo2-servicio**’ (Ver ilustración 7.35).



Ilustración 7.35: Verificación del balanceo de carga con el **Nodo2-servicio**

Como última comprobación, se agrega el nombre de dominio del nodo “**Balancedor-de-carga**” para corroborar que, accediendo desde el nombre de dominio, el servicio trabaja exactamente igual que con la dirección IP. En la ilustración 7.36 se puede ver que proporciona el servicio de manera correcta.



Ilustración 7.36: Verificación del balanceo de carga con el nombre de dominio

7.3.2.4. Instalación y configuración del software *High Availability Add-On*

En esta sección se comenzará a darle forma al concepto del clúster, a configurarlo de manera que conforme una totalidad de cinco nodos, tal y como se ha mencionado anteriormente.

Para ello, se necesita instalar los paquetes de *software Red Hat High Availability Add-On* en todos los nodos que formarán el clúster. Los paquetes de software de *Red Hat High Availability Add-On* son un tipo de software que está destinado a proporcionar herramientas y procesos para crear, ver, editar y administrar un clúster sin necesidad de tener que configurar un clúster real [38].

Por tanto, para **instalar los paquetes de software** de *Red Hat High Availability Add-On* desde canal de Alta Disponibilidad se ejecuta el siguiente comando en todos los nodos del clúster (Nodo1-servicio, Nodo2-servicio, Nodo3-almacenamiento, Nodo4-almacenamiento y Balanceador-de-carga):

```
[root@nodo1-tft ~]# dnf -y install pcs pacemaker fence-agents-all
```

Como se está ejecutando el demonio **firewalld**, hay que habilitar los puertos necesarios para el *High Availability* de Red Hat en todos los nodos del clúster de la siguiente manera:

```
[root@nodo1-tft ~]# firewall-cmd --permanent --add-service
→ high-availability
[root@nodo1-tft ~]# firewall-cmd --reload
```

Hay que recordar que, previamente, en los nodos “**Nodo1-servicio**”, “**Nodo2-servicio**” y “**Balanceador-de-carga**”, se ha eliminado el rango de puertos que estaba abierto en el *firewall*. De la misma manera, se realiza en los nodos restantes del clúster: “**Nodo3-almacenamiento**” y “**Nodo4-almacenamiento**”.

Una vez añadido el componente de Alta Disponibilidad y eliminado el rango de puertos activados por defecto en el “**Nodo3-almacenamiento**” y en el “**Nodo4-almacenamiento**”, el listado final del *firewall* de todos los nodos quedaría justo como se señala en la ilustración 7.37.

```
[root@nodo1-tft ~]# firewall-cmd --list-all
FedoraWorkstation (active)
target: default
icmp-block-inversion: no
interfaces: enp1s0 enp7s0 enp8s0
sources:
services: dhcpv6-client high-availability http https mdns samba-client ssh
ports:
protocols:
forward: no
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
```

Ilustración 7.37: Listado del *firewall* de todos los nodos

Para poder utilizar el comando **pcs** en cada nodo, y en consecuencia, poder configurar el clúster, es imprescindible **establecer una contraseña** para el usuario **hacluster** en cada nodo del clúster y autenticar al usuario para cada nodo del clúster. El nombre de usuario **hacluster** configurado es para administrar la utilidad **pcs**. Tanto el nombre como la contraseña de este usuario debe ser la misma para cada nodo.

En la ilustración 7.38 se demuestra el comando para establecer la contraseña al usuario `hacluster` y su ejecución, donde se indica la contraseña dos veces como método de seguridad. Este paso se aplica en todos los nodos del clúster, aunque solo se muestre en el “**Nodo1-servicio**”.

```
[root@nodo1-tft ~]# passwd hacluster
Cambiando la contraseña del usuario hacluster.
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: todos los tokens de autenticación se actualizaron exitosamente.
```

Ilustración 7.38: Establecimiento de la contraseña del usuario `hacluster`

Antes de comenzar a efectuar la configuración del clúster, es necesario arrancar el demonio `pcsd` y habilitarlo para que se inicie en el arranque de cada nodo del clúster. Las órdenes que se muestran a continuación se ejecutarán en **todos los nodos** del clúster:

```
[root@nodo1-tft ~]# systemctl start pcsd
[root@nodo1-tft ~]# systemctl enable pcsd
```

Finalmente, hay que **autenticar al usuario** `hacluster` en todos los nodos del clúster con la utilidad `pcs`. Este comando se puede ejecutar en cualquiera de los cinco nodos del clúster, aunque no es necesario ejecutarla en todos nodos. En este caso se ejecuta en el “**Nodo1-servicio**”, ya que es donde se llevará a cabo toda la configuración del clúster. Al ejecutar la orden siguiente en el “**Nodo1-servicio**”, se autenticará al usuario “`hacluster`” en el `nodo1-tft.com`, en el `nodo2-tft.com`, en el `nodo3-tft.com`, en el `nodo4-tft.com` y en el `balanceador-tft.com` (Ver ilustración 7.39):

```
[root@nodo1-tft ~]# pcs host auth nodo1-tft.com addr=10.22.132.128 nodo2-tft.com addr=10.22.132.129  
nodo3-tft.com addr=10.22.132.130 nodo4-tft.com addr=10.22.132.131 balanceador-tft.com addr=10.22.132  
.132 -u hacluster
Password:
nodo1-tft.com: Authorized
balanceador-tft.com: Authorized
nodo2-tft.com: Authorized
nodo4-tft.com: Authorized
nodo3-tft.com: Authorized
```

Ilustración 7.39: Autenticación del usuario “`hacluster`” en todos los nodos del clúster

7.3.2.5. Creación y configuración del clúster

En este apartado se iniciará la creación del clúster formado por los cinco nodos previamente mencionados. Los comandos relativos a la administración del clúster se pueden ejecutar desde cualquier nodo del clúster, pero como se comentó anteriormente, se ejecutarán todos en el “**Nodo1-servicio**”.

Para la creación del clúster se ejecuta el siguiente comando donde se indica el nombre del clúster y los nodos que lo van a formar. En este caso, el nombre del clúster será “**cluster-tft**”.

```
[root@nodo1-tft ~]# pcs cluster setup cluster-tft --start
→ nodo1-tft.com addr=10.22.132.128 nodo2-tft.com
→ addr=10.22.132.129 nodo3-tft.com addr=10.22.132.130
→ nodo4-tft.com addr=10.22.132.131 balanceador-tft.com
→ addr=10.22.132.132
```

La opción “*-start*” que se ha agregado al comando, se encarga de iniciar los servicios del clúster en todos los nodos. Las direcciones IP que se han añadido para configurar el clúster son las de la red aislada de **Control**, pues las otras dos redes no están dedicadas a este fin.

Después de crear el clúster, se configura los servicios habilitándolos para que se arranquen cada vez que se inicien los nodos del clúster sin tener que iniciarlos manualmente (**Ver ilustración 7.40**):

```
[root@nodo1-tft ~]# pcs cluster enable --all
nodo1-tft.com: Cluster Enabled
nodo2-tft.com: Cluster Enabled
nodo3-tft.com: Cluster Enabled
nodo4-tft.com: Cluster Enabled
balanceador-tft.com: Cluster Enabled
```

Ilustración 7.40: Habilitando los servicios del clúster

Para concluir esta fase, se verifica que el clúster está en correcto funcionamiento utilizando el comando ***pcs status***. En la ilustración 7.41 se presenta el resultado de la ejecución de dicho comando, en la que se puede apreciar, entre otras cosas, que no hay ningún tipo de error, el nombre del clúster (*cluster-tft*) y un listado de los cinco nodos que están configurados y que están disponibles (*Online*).

```
[root@nodo1-tft ~]# pcs status
Cluster name: cluster-tft

WARNINGS:
No stonith devices and stonith-enabled is not false

Cluster Summary:
* Stack: corosync
* Current DC: nodo4-tft.com (version 2.1.3-0.2.rc2.fc35-dff7c3a726) - partition with quorum
* Last updated: Tue Jun 14 12:53:54 2022
* Last change: Tue Jun 14 12:52:33 2022 by hacluster via crmd on nodo4-tft.com
* 5 nodes configured
* 0 resource instances configured

Node List:
* Online: [ balanceador-tft.com nodo1-tft.com nodo2-tft.com nodo3-tft.com nodo4-tft.com ]

Full List of Resources:
* No resources

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

Ilustración 7.41: Estado del clúster

7.3.2.6. Establecimiento de un mecanismo de aislamiento de nodos

En este momento, el clúster ya está construido y está funcionando correctamente. Pero como puede ocurrir en muchos casos, alguno de sus nodos puede experimentar algún fallo de *software* o de *hardware* que pueda ocasionar un funcionamiento deficiente del mismo. En ocasiones puede que algún nodo pueda efectuar accesos no autorizados de manera concurrente al espacio de almacenamiento compartido, forzando a corromper los datos, o pueda dejar de tener información vigente del estado del clúster. En cualquier caso, una forma de impedir que estos inconvenientes ocurran es aislando el nodo del clúster que haya experimentado algún tipo de fallo para que no pueda comprometer los datos y los servicios que pueda ofrecer el clúster [39].

A este método se le suele llamar **mecanismo de aislamiento** o de *fencing*. Existen diversos tipos, pero el que se usará es el *STONITH* (*Shoot The Other Node In The Head*), lo que significa en español: “Dispárale al otro nodo en la cabeza”. Este mecanismo se basa en aislar un nodo que tiene algún tipo de fallo para que no cause ninguna perturbación en el clúster, dejándolo inactivo (apagándolo) o reiniciándolo [40].

Existen diferentes mecanismos para realizar el aislamiento, de los cuales, en este caso, se utilizará el mecanismo de aislamiento basado en el host anfitrión. En este mecanismo, todos los nodos tienen acceso al anfitrión, por lo tanto, este es el que va a determinar y a parar el nodo afectado. Para configurar este aislamiento, se hace uso de la utilidad “**fence_virt**” en el host anfitrión para permitir la comunicación del anfitrión con el grupo de nodos mediante una dirección IP *multicast* [39].

La configuración de este mecanismo requiere instalación y configuración del mecanismo en el sistema anfitrión y en todos los nodos del clúster. Por lo tanto, en primer lugar, se procederá a configurar el sistema anfitrión y después, se seguirá con la configuración de todos los nodos del clúster.

Sistema anfitrión

Primeramente, se procede a instalar los paquetes necesarios para aplicar el mecanismo de aislamiento en el anfitrión:

```
[root@pc1-tft ~]# dnf -y install fence-virt fence-virt  
→ fence-virt-multicast fence-virt-libvirt
```

Cuando finalice la instalación, se configura el *firewall* para permitir el tráfico de comunicación a través de la dirección IP *multicast* abriendo el puerto 1229 en la zona *libvirt* de la siguiente manera:

```
[root@pc1-tft ~]# firewall-cmd --zone=libvirt --permanent  
→ --add-port=1229/udp  
[root@pc1-tft ~]# firewall-cmd --zone=libvirt --permanent  
→ --add-port=1229/tcp
```

```
[root@pc1-tft ~]# firewall-cmd --reload
```

En este punto, solo queda realizar una serie de configuraciones para que el mecanismo de aislamiento pueda funcionar correctamente.

En primer lugar, se debe generar un fichero de configuración del servicio **fence-virt** denominado “/etc/fence_virt.conf”, que tras su configuración contempla diferentes parámetros necesarios para el mecanismo: dirección IP *multicast*, el puerto de configuración, un fichero con una clave compartida para que los nodos puedan realizar peticiones de *fencing*, etc. [39]. Se utiliza el comando *fence_virt* para generar dicho fichero en el *host* anfitrión y se especifican las opciones que se muestran resaltadas en las ilustraciones 7.42 y 7.43. En las opciones que no se ha señalado nada, se ha presionado la tecla “Enter” para utilizar el valor proporcionado por defecto.

```
[root@pc1-tft ~]# fence_virt -c
Module search path [/usr/lib64/fence-virt/]:

Available backends:
  libvirt 0.3
Available listeners:
  vsock 0.1
  multicast 1.2

Listener modules are responsible for accepting requests
from fencing clients.

Listener module [multicast]:

The multicast listener module is designed for use environments
where the guests and hosts may communicate over a network using
multicast.

The multicast address is the address that a client will use to
send fencing requests to fence_virt.

Multicast IP Address [225.0.0.12]:

Using ipv4 as family.

Multicast IP Port [1229]:

Setting a preferred interface causes fence_virt to listen only
on that interface. Normally, it listens on all interfaces.
In environments where the virtual machines are using the host
machine as a gateway, this *must* be set (typically to virbr0).
Set to 'none' for no interface.

Interface [virbr0]: virbr2

The key file is the shared key information which is used to
authenticate fencing requests. The contents of this file must
be distributed to each physical host and virtual machine within
a cluster.

Key File [/etc/cluster/fence_xvm.key]:
```

Ilustración 7.42: Primera parte de la generación del fichero fence_virt.conf

Tal y como se ve en las ilustraciones 7.42 y 7.43, casi todas las opciones se han fijado para que se apliquen por defecto, excepto la que solicita la interfaz, que se ha indicado la **virbr2** (correspondiente a la interfaz de la red de Control) y la que solicita reemplazar el fichero existente, que se ha señalado una “y” para informar que si se quiere reemplazar.

```
Key File [/etc/cluster/fence_xvm.key]:

Backend modules are responsible for routing requests to
the appropriate hypervisor or management layer.

Backend module [libvirt]:

The libvirt backend module is designed for single desktops or
servers. Do not use in environments where virtual machines
may be migrated between hosts.

Libvirt URI [qemu:///system]:

Configuration complete.

=== Begin Configuration ===
backends {
    libvirt {
        uri = "qemu:///system";
    }
}

listeners {
    multicast {
        port = "1229";
        family = "ipv4";
        interface = "virbr2";
        address = "225.0.0.12";
        key_file = "/etc/cluster/fence_xvm.key";
    }
}

fence_virt {
    module_path = "/usr/lib64/fence-virt/";
    backend = "libvirt";
    listener = "multicast";
}

=== End Configuration ===
Replace /etc/fence_virt.conf with the above [y/N]? y
```

Ilustración 7.43: Segunda parte de la generación del fichero fence_virt.conf

Antes de solicitar el reemplazo del fichero en la ilustración 7.43, se puede observar un cuadro de color rojo señalando la información generada tras la ejecución del comando, esta información es la que se añadirá en el fichero “/etc/fence_virt.conf”. Si se comprueba dicho fichero (*cat /etc/fence_virt.conf*) se podrá ver que esa información está plasmada en él.

En segundo lugar, se procede a crear el fichero que contiene la clave compartida. Para ello, primero se crea un nuevo directorio (**/etc/cluster**) para generar dicho fichero en él y luego, se crea el fichero con los permisos adecuados y se genera el contenido (la clave compartida) del fichero. En la ilustración 7.44 se exponen todos los pasos señalados a seguir.

```
[root@pc1-tft ~]# mkdir -p /etc/cluster
[root@pc1-tft ~]# cd /etc/cluster/
[root@pc1-tft cluster]# touch /etc/cluster/fence_xvm.key
[root@pc1-tft cluster]# ls -l
total 0
-rw-r--r--. 1 root root 0 jun 13 10:16 fence_xvm.key
[root@pc1-tft cluster]# chmod 0600 /etc/cluster/fence_xvm.key
[root@pc1-tft cluster]# ls -l
[root@pc1-tft cluster]# dd if=/dev/urandom bs=512 count=1 of=/etc/cluster/fence_xvm.key
1+0 registros leídos
1+0 registros escritos
512 bytes copied, 0,000431653 s, 1,2 MB/s
```

Ilustración 7.44: Creación del fichero que contiene la clave compartida

Finalmente, se inicia y habilita el servicio **fence_virt** para que se inicie cada vez que se arranca el sistema anfitrión:

```
[root@pc1-tft ~]# systemctl start fence_virt
[root@pc1-tft ~]# systemctl enable fence_virt
```

Para comprobar la conectividad *multicast* se ejecuta la siguiente orden en el anfitrión:

```
[root@pc1-tft ~]# fence_xvm -o list
```

El resultado del comando debe imprimir un listado de todas las máquinas virtuales donde se muestra el nombre de cada una, el ID y la palabra “on” u “off” dependiendo de si tiene disponible este mecanismo o no. En todos los nodos del clúster debe salir la palabra “on”. Si el comando no responde, se debe reiniciar el **sistema anfitrión**.

Nodos del clúster

Al igual que en el sistema anfitrión, el primer paso a llevar a cabo en **todos los nodos** es la instalación de los paquetes software necesarios:

```
[root@nodo1-tft ~]# dnf -y install fence-virt fence-virt
```

Tras finalizar la instalación, se configura el *firewall* en **todos los nodos** del clúster para permitir el tráfico de comunicación con el *host* anfitrión a través de la red de control (10.22.132.1) y del puerto 1229 en la zona *public* (por defecto) de la siguiente manera:

```
[root@nodo1-tft ~]# firewall-cmd --permanent --add-rich-rule='rule
→ family="ipv4" source address="10.22.132.1" port port="1229"
→ protocol="tcp" accept'
[root@nodo1-tft ~]# firewall-cmd --reload
```

El siguiente paso consiste en crear un directorio administrativo (**/etc/cluster**) en **todos los nodos** para copiar el fichero que contiene la clave compartida que utilizarán los nodos

para realizar las peticiones de aislamiento. Pero para ello, primero se debe iniciar y habilitar el servicio **sshd** en **todos los nodos** y en el **host anfitrión** para poder llevar a cabo el proceso de la copia del fichero:

```
[root@nodo1-tft ~]# systemctl start sshd
[root@nodo1-tft ~]# systemctl enable sshd
```

Después, hay que permitir al servicio **ssh** conectarse con el usuario **root** y habilitar el poder conectarse con contraseña. Esto se consigue editando el fichero “**/etc/ssh/sshd_config**”, poniendo a “**yes**” los parámetros “**PasswordAuthetication**” y “**PermitRootLogin**”. Una vez modificado, se reinicia el servicio **sshd** y ya estaría listo para crear el directorio:

```
[root@nodo1-tft ~]# systemctl restart sshd
```

Ahora se procede a crear la carpeta administrativa y a copiar el fichero de la clave compartida que se encuentra en el **host anfitrión**. Este proceso también se realiza en **todos los nodos** (Ver ilustración 7.45).

```
[root@nodo1-tft ~]# mkdir -p /etc/cluster
[root@nodo1-tft ~]# scp root@10.22.132.1:/etc/cluster/fence_xvm.key /etc/cluster
The authenticity of host '10.22.132.1 (10.22.132.1)' can't be established.
ED25519 key fingerprint is SHA256:+c1LWgP6ob6h+56Dj0fKqD4Kzci2TXPn0UevHsEm/Ck.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.22.132.1' (ED25519) to the list of known hosts.
root@10.22.132.1's password:
fence_xvm.key                               100% 512   780.2KB/s   00:00
```

Ilustración 7.45: Copia del fichero que contiene la clave compartida en los nodos

En este momento, el *fencing* ya está disponible para su uso. Para comprobar la conectividad *multicast* se emplea la orden *fence_xvm* en todos los nodos del clúster:

```
[root@nodo1-tft ~]# fence_xvm -o list
balanceador-de-carga      ba83a335-a749-431b-aa40-c563c40a81d7 on
maquinabase              09ca42ff-ac33-46ba-898c-587f217c4261 off
Nodo1-servicio           29661eed-46cb-4e26-a43a-10c05eb446ed on
Nodo2-servicio           d89c61e5-18d7-452a-88b4-237680331344 on
Nodo3-almacenamiento    33dab936-5c6b-44cf-b91a-0132a279413b on
Nodo4-almacenamiento    7d590662-f2fd-4740-9429-e9ebe95f6e95 on
prueba_bridge            c65070c2-5555-4390-82e7-a3b799eedce3 off
prueba_conbackup         ee20e0b3-358d-415f-926b-144dfd52f31e off
prueba_simulacion_nfs   03dde571-dc9d-4b12-88f5-fa9f1977d39f off
prueba_virt-install     212d25fe-7693-4c75-8851-99cd35babf66 off
prueba_virt-manager     e5f96b0d-bbc3-4f8f-bff7-d62d9b6950ab off
```

Ilustración 7.46: Comprobación de la conectividad *multicast*

En el caso de que el comando no responda, se debe reiniciar todos los nodos.

El último paso de la configuración del *fencing* en los nodos, consiste en **crear el mecanismo de aislamiento** de tipo *fence_xvm* que se acaba de configurar en el clúster a través de la siguiente orden:


```
[root@nodo1-tft ~]# pcs stonith create xvmfence fence_xvm
→ key_file=/etc/cluster/fence_xvm.key
```

La ejecución de este comando solo se debe realizar en un único nodo, y como se ha mencionado previamente, todas las operaciones administrativas del clúster se llevan a cabo en el “**Nodo1-servicio**”.

Una vez finalizado el proceso de configuración en todos los nodos y en el sistema anfitrión, se reinician **todos los nodos** para verificar que el mecanismo de *fencing* se haya creado correctamente y que el estado global del clúster también sea correcto.

En la ilustración 7.47 se muestra el resultado de la ejecución de las órdenes *pcs stonith* y *pcs status*. Estos comandos no hacen falta ejecutarlos en todos los nodos, ya que al ser un clúster, al ejecutar la orden en uno solo, muestra el estado de todos los nodos.

```
[root@nodo1-tft ~]# pcs stonith
* xvmfence (stonith:fence_xvm): Started balanceador-tft.com
[root@nodo1-tft ~]# pcs status
Cluster name: cluster-tft
Cluster Summary:
* Stack: corosync
* Current DC: nodo1-tft.com (version 2.1.3-0.2.rc2.fc35-dff7c3a726) - partition with quorum
* Last updated: Tue Jun 14 13:09:01 2022
* Last change: Tue Jun 14 13:03:27 2022 by root via cibadmin on nodo1-tft.com
* 5 nodes configured
* 1 resource instance configured

Node List:
* Online: [ balanceador-tft.com nodo1-tft.com nodo2-tft.com nodo3-tft.com nodo4-tft.com ]

Full List of Resources:
* xvmfence (stonith:fence_xvm): Started balanceador-tft.com

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

Ilustración 7.47: Estado del fencing y del clúster

7.3.2.7. Creación del recurso HAProxy

En este apartado, se definirá el recurso que será ofrecido y manejado por el clúster. El recurso a ofrecer será el HAProxy. Para poder efectuar esto, se ejecuta el siguiente comando en el “**Nodo1-servicio**”, indicando el nombre del recurso y el grupo al que pertenece.

```
[root@nodo1-tft ~]# pcs resource create HAProxy service:haproxy
```

Para que este servicio sea manejado por el clúster correctamente, se debe apagar y deshabilitar el servicio en el nodo que se ofrece, es decir, en el “**Balanceador-de-carga**”.

```
[root@balanceador-tft ~]# systemctl stop haproxy
[root@balanceador-tft ~]# systemctl disable haproxy
```

Una vez creado el recurso, se debe comprobar el estado global del clúster (Ver [ilustración 7.48](#)).

```
[root@balanceador-tft ~]# pcs status
Cluster name: cluster-tft
Cluster Summary:
 * Stack: corosync
 * Current DC: nodo1-tft.com (version 2.1.3-0.2.rc2.fc35-dff7c3a726) - partition with quorum
 * Last updated: Tue Jun 14 13:40:37 2022
 * Last change: Tue Jun 14 13:40:27 2022 by root via cibadmin on balanceador-tft.com
 * 5 nodes configured
 * 2 resource instances configured

Node List:
 * Online: [ balanceador-tft.com nodo1-tft.com nodo2-tft.com nodo3-tft.com nodo4-tft.com ]

Full List of Resources:
 * xvmfence (stonith:fence_xvm): Started nodo1-tft.com
 * HAProxy (service:haproxy): Started balanceador-tft.com

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

Ilustración 7.48: Estado del clúster y su recurso

El nodo “**Balanceador-de-carga**” es el único que posee el servicio HAProxy, el cual ofrece alta disponibilidad y balanceo de carga del servicio web basado en Apache. Como se menciona en el apartado “[Diseño del clúster](#)”, en un entorno de producción esto no se podría diseñar de esta manera, ya que si este nodo tiene algún problema, el servicio entero dejaría de ofrecerse.

A raíz de esto, se procede a configurar una restricción de ubicación para determinar que el recurso HAProxy evite todos los nodos del clúster menos el nodo “**Balanceador-de-carga**”. Para configurar esta restricción se ejecuta la siguiente orden, indicando con el parámetro “*avoids*” que se quiere evitar iniciar el recurso con los nodos especificados [41]:

```
[root@nodo1-tft ~]# pcs constraint location HAProxy avoids
→ nodo1-tft.com nodo2-tft.com nodo3-tft.com nodo4-tft.com
```

Para verificar que la restricción funciona adecuadamente, se parará el nodo que ofrece el servicio, “**Balanceador-de-carga**”, con el comando `pcs node standby` y se comprueba el estado global del cluster (Ver [ilustración 7.49](#)). Como se puede observar, al pararse el nodo “**Balanceador-de-carga**” y al no intentar iniciarse el recurso por cualquier otro nodo debido a la restricción configurada, el recurso de HAProxy se encuentra parado.

```

[root@nodo1-tft ~]# pcs node standby balanceador-tft.com
[root@nodo1-tft ~]# pcs status
Cluster name: cluster-tft
Cluster Summary:
 * Stack: corosync
 * Current DC: nodo3-tft.com (version 2.1.3-0.2.rc2.fc35-dff7c3a726) - partition with
quorum
 * Last updated: Tue Jun 21 09:30:45 2022
 * Last change: Tue Jun 21 09:30:42 2022 by root via cibadmin on nodo1-tft.com
 * 5 nodes configured
 * 2 resource instances configured

Node List:
 * Node balanceador-tft.com: standby
 * Online: [ nodo1-tft.com nodo2-tft.com nodo3-tft.com nodo4-tft.com ]

Full List of Resources:
 * xvmfence (stonith:fence_xvm): Started nodo1-tft.com
 * HAProxy (service:haproxy): Stopped (not installed)

Daemon Status:
 corosync: active/enabled
 pacemaker: active/enabled
 pcsd: active/enabled

```

Ilustración 7.49: Estado del clúster tras parar el nodo **Balanceador-de-carga**

7.3.2.8. Instalación y configuración de MariaDB Galera Cluster

El desarrollo de un clúster en alta disponibilidad requiere tener otra infraestructura de clúster, pero en este caso de base de datos, para que cada operación que se realice en un nodo de los dos nodos que ofrece el servicio, se vea reflejada instantáneamente en el resto de nodos. Además, el acceso a los datos de la base de datos debe estar sincronizado de forma que se pueda leer o escribir en la base de datos desde cualquier nodo del clúster [42].

MariaDB Galera Cluster es un *software* que permite la creación y gestión de clústeres de bases de datos de manera síncrona entre los nodos del clúster. Gracias a este *software* se podrá crear un clúster de base de datos formado por los nodos que están destinados a almacenar la base de datos del servicio que ofrece el clúster (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**).

Como se indica en la documentación de Red Hat, el paquete **mariadb-server-galera** incluye los paquetes “*mariadb-server*” y “*galera*”, necesarios para construir el clúster de base de datos. Por lo tanto, se instala el primer paquete en el “**Nodo3-almacenamiento**” y “**Nodo4-almacenamiento**” de la siguiente manera:

```
[root@nodo3-tft ~]# dnf -y install mariadb-server-galera
```

Como el *firewall* está activado, se abren los siguientes puertos para que el clúster pueda funcionar correctamente en el “**Nodo3-almacenamiento**” y “**Nodo4-almacenamiento**” [43]:

```

[root@nodo3-tft ~]# firewall-cmd --permanent --add-port 3306/tcp
[root@nodo3-tft ~]# firewall-cmd --permanent --add-port 4567/tcp
[root@nodo3-tft ~]# firewall-cmd --permanent --add-port 4568/tcp
[root@nodo3-tft ~]# firewall-cmd --permanent --add-port 4444/tcp

```

```
[root@nodo3-tft ~]# firewall-cmd --permanent --add-port 4567/udp
[root@nodo3-tft ~]# firewall-cmd --reload
```

Si el servicio de **mariadb** no se encuentra parado, se ejecuta el siguiente comando para apagarlo en los nodos “**Nodo3-almacenamiento**” y “**Nodo4-almacenamiento**”:

```
[root@nodo3-tft ~]# systemctl stop mariadb
```

Con el servicio de **mariadb** parado, se accede al fichero “**/etc/my.cnf.d/galera.cnf**” y se configura los parámetros que se encuentran resaltados en las ilustraciones 7.50 y 7.51. Estos pasos tienen que ser configurados en ambos nodos (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**).

```
# Enable wsrep
wsrep_on=1

# Full path to wsrep provider library or 'none'
wsrep_provider=/usr/lib64/galera/libgalera_smm.so

# Provider specific configuration options
#wsrep_provider_options=

# Logical cluster name. Should be the same for all nodes.
wsrep_cluster_name="galera_cluster"

# Group communication system handle
wsrep_cluster_address="gcomm://10.22.122.130,10.22.122.131"

# Human-readable node name (non-unique). Hostname by default.
wsrep_node_name="nodo3-tft"

# Base replication <address|hostname>[:port] of the node.
# The values supplied will be used as defaults for state transfer receiving,
# listening ports and so on. Default: address of the first network interface.
wsrep_node_address="10.22.122.130"

# Address for incoming client connections. Autodetect by default.
#wsrep_node_incoming_address=

# How many threads will process writesets from other nodes
wsrep_slave_threads=1
```

Ilustración 7.50: Primera parte del fichero galera.cnf

El primer parámetro que está señalado en la ilustración 7.50, “*wsrep_on*”, se iguala a 1 para permitir replicar las actualizaciones realizadas en la sesión en curso. En el parámetro “*wsrep_cluster_name*” se indica el nombre del clúster de base de datos. El tercer parámetro define las direcciones IP de los nodos que forman el clúster de base de datos. Los últimos dos parámetros están relacionados con la información individual de cada nodo, donde primero se le indica el nombre del nodo y después la dirección IP [44].

El último parámetro modificado en el fichero es el “*wsrep_sst_method*”, que permite definir el script que utiliza el nodo para realizar las transferencias. En este caso se ha usado, **rsync**, el parámetro por defecto que realiza las transferencias mucho más rápidas que el resto de opciones [44].

```

wsrep_drupal_282555_workaround=0

# enable "strictly synchronous" semantics for read operations
wsrep_causal_reads=0

# Command to call when node status or cluster membership changes.
# Will be passed all or some of the following options:
# --status - new status of this node
# --uuid   - UUID of the cluster
# --primary - whether the component is primary or not ("yes"/"no")
# --members - comma-separated list of members
# --index  - index of this node in the list
wsrep_notify_cmd=

##
## WSREP State Transfer options
##

# State Snapshot Transfer method
wsrep_sst_method=rsync

# Address which donor should send State Snapshot to.
# Should be the address of THIS node. DON'T SET IT TO DONOR ADDRESS!!!
# (SST method dependent. Defaults to the first IP of the first interface)
#wsrep_sst_receive_address=

# SST authentication string. This will be used to send SST to joining nodes.
# Depends on SST method. For mysqldump method it is root:<root password>
wsrep_sst_auth=root:

```

Ilustración 7.51: Segunda parte del fichero galera.cnf

Tras la configuración de los ficheros de Galera, **se inicia el clúster** desde uno de los dos nodos y luego se van agregando el resto de nodos (Nodo4-almacenamiento). En este caso, se ha realizado en el “**Nodo3-almacenamiento**” de la siguiente manera:

```
[root@nodo3-tft ~]# galera_new_cluster
```

La ejecución de este comando no muestra ninguna salida si tuvo éxito, es decir, si se inició correctamente el servicio de **mariadb** en el nodo ejecutado.

En el “**Nodo4-almacenamiento**”, se inicia el servicio de **mariadb** como se suele hacer normalmente para arrancar dicho nodo del clúster de base de datos:

```
[root@nodo3-tft ~]# systemctl start mariadb
```

IMPORTANTE: Cada vez que se reinicien los nodos del clúster, se debe realizar estos pasos: ejecutar el comando *galera_new_cluster* en uno de los dos nodos para iniciar el clúster y agregar el otro nodo con el comando *systemctl start mariadb*. Para saber en qué nodo se debe ejecutar cada comando, se accede al fichero “*/var/lib/mysql/grastate.dat*” en cada uno de ellos y se analiza el parámetro **seqno**. Aquel nodo que tenga un valor de **seqno** mayor, es el nodo donde se deberá ejecutar el primer comando. Por lo tanto, en el otro nodo, se ejecutará el segundo comando [45].

Si se quiere comprobar que se ha creado bien el clúster, se ejecuta la orden que se ha aplicado en la ilustración 7.52 y el resultado muestra el tamaño del clúster o el número de nodos por el que está formado el clúster.

```
[root@nodo3-tft ~]# mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
Enter password:
+-----+
| Variable_name | Value |
+-----+
| wsrep_cluster_size | 2 |
+-----+
```

Ilustración 7.52: Verificación del tamaño del clúster

Para **probar la replicación** de la base de datos en los nodos, es decir, para verificar que cualquier cambio que se realice en un nodo se vea reflejado en el otro, hay que crear una base de datos de prueba en uno de los nodos. Esto se realizará en el “**Nodo3-almacenamiento**” con el comando posterior:

```
[root@nodo3-tft ~]# mysql -u root -p
```

Tras ejecutar el comando anterior, se solicita la contraseña del root. Como normalmente no hay ninguna contraseña configurada por defecto, para poder realizar correctamente la ejecución del comando, se deja la contraseña vacía, es decir, se hace clic en la tecla “Enter” y rápidamente se inicia el monitor de MySQL [46]. En él, se ejecuta la siguiente orden de MySQL para crear la base de datos y se sale del *shell* de MySQL:

```
mysql> CREATE DATABASE prueba_backup;
mysql> EXIT;
```

Ahora se ejecuta el mismo comando para conectarse al monitor de MySQL en el “**Nodo4-almacenamiento**” y consultar todas las bases de datos:

```
[root@nodo4-tft ~]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.5.5-10.5.16-MariaDB MariaDB Server

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| prueba_backup |
+-----+
4 rows in set (0,00 sec)

mysql> exit
Bye
```

Ilustración 7.53: Probando la replicación de la base de datos

Se puede observar en la ilustración 7.53 que la base de datos **prueba_backup** creada en el “**Nodo3-almacenamiento**”, ha sido replicada al “**Nodo4-almacenamiento**”.

7.3.2.9. Inclusión de MariaDB Galera Cluster y Estadísticas del HAProxy en el HAProxy

Esta sección consiste en configurar de nuevo el HAProxy, pero en este caso, para agregar **MariaDB Galera Cluster** al **HAProxy** para balancear la carga entre los dos nodos del clúster de base de datos (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**).

Aparte de las directivas *frontend* y *backend*, existe la directiva *listen*, que hace exactamente lo mismo que las anteriores, ya que es la combinación de las funcionalidades de ambas (*frontend* y *backend*). Seguido de esta directiva se indica el nombre del servicio que escuchará las solicitudes [47].

En este caso se crearán dos directivas *listen*, la primera para la configuración del servicio **MariaDB Galera Cluster** y la segunda, para la configuración del servicio de **estadísticas de HAProxy**. Este servicio permite analizar en detalle información sobre las conexiones, los estados de los nodos del clúster que ofrecen el servicio de HTTP y el de MariaDB Galera Cluster, la transferencia de datos, etc.

Por consiguiente, se realizan las configuraciones correspondientes igual que se ponen de manifiesto en las ilustraciones 7.54 y 7.55.

```
# Balanceo de carga para Galera Cluster
listen galera
  bind 192.168.122.246:3306
  balance roundrobin
  mode tcp
  option tcpka
  server nodo3-tft 10.22.122.130:3306 check weight 1
  server nodo4-tft 10.22.122.131:3306 check weight 1
```

Ilustración 7.54: Modificación del fichero haproxy.cfg para agregar **Galera Cluster**

Como se puede observar en la primera ilustración, se ha agregado la dirección IP (de la red NAT) del nodo “**Balanceador-de-carga**” y el puerto 3306, correspondiente al puerto por defecto del protocolo de MySQL. Se indica el algoritmo balanceador de carga y finalmente, se describe la información de los servidores de base de datos que repartirán la carga (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**).

```
# Estadísticas de HAProxy
listen stats
  bind 192.168.122.246:9000
  mode http
  stats enable
  stats realm HAProxy\ Statistics
  stats uri /haproxy?stats
  stats auth haproxy:haproxy
  stats admin if TRUE
```

Ilustración 7.55: Modificación del fichero haproxy.cfg para agregar las **estadísticas de HAProxy**

De igual forma, para escuchar por el servicio de estadísticas de HAProxy, se indica la dirección IP (de la red NAT) del nodo “**Balanceador-de-carga**” y el puerto 9000, utilizado

para acceder a las estadísticas del HAProxy. Además, se añaden una serie de parámetros necesarios para la configuración, tales como la *url* del sitio web, la autenticación al sitio web, la habilitación de las estadísticas, la autenticación como administrador, etc.

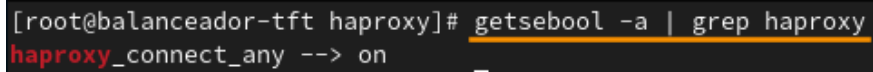
Como en la configuración del HAProxy se ha añadido los puertos 3306 y 9000, también hay que agregarlos al *firewall* para que pueda permitir la conexión. Los comandos a ejecutar para abrir dichos puertos son:

```
[root@balanceador-tft ~]# firewall-cmd --permanent --add-port
→ 3306/tcp
[root@balanceador-tft ~]# firewall-cmd --permanent --add-port
→ 9000/tcp
[root@balanceador-tft ~]# firewall-cmd --reload
```

Además, se deben abrir todos los puertos HAProxy del **SELinux** justo como se muestra a continuación, activando el booleano “haproxy_connect_any”:

```
[root@balanceador-tft ~]# setsebool -P haproxy_connect_any 1
```

Se puede verificar que el booleano de SELinux se ha activado correctamente ejecutando el comando que se muestra en la ilustración 7.56. Si está en modo “*on*” indica que está activado.



```
[root@balanceador-tft haproxy]# getsebool -a | grep haproxy
haproxy_connect_any --> on
```

Ilustración 7.56: Comprobación del booleano de SELinux

Concluyendo con la configuración del fichero HAProxy, para que se apliquen los cambios, se reinicia el servicio HAProxy:

```
[root@balanceador-tft ~]# systemctl restart haproxy
```

Para probar que ambos servicios operan adecuadamente, se para el servicio de HAProxy en el nodo “**Balanceador-de-carga**”, ya que anteriormente se reinició para actualizar los cambios en el fichero de configuración y al reiniciarse, el servicio se inicia.

```
[root@balanceador-tft ~]# systemctl stop haproxy
```

Con el servicio parado y manejado por el clúster, se comprueba en primer lugar el balanceo de carga de la base de datos y en segundo lugar, se verifica la página de estadísticas del HAProxy.

Comprobación del balanceo de carga de la base de datos

Para poder realizar la prueba del balanceo de carga de la base de datos se debe realizar una pequeña configuración previa en el monitor de MySQL. Como anteriormente se ha creado una base de datos para probar la replicación de la base de datos entre nodos, ahora se debe crear un usuario asociado a esta base de datos para poder comprobar el balanceo de carga.

Para crear el usuario se ejecuta en primera instancia el siguiente comando en uno de los dos nodos de almacenamiento (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**). En este caso, se utiliza el “**Nodo3-almacenamiento**” para realizar las correspondientes comprobaciones:

```
[root@nodo3-tft ~]# mysql -u root -p
```

Una vez ejecutado el comando anterior, se indica la contraseña del *root* y se ejecuta las siguientes órdenes de MySQL para crear el usuario, para otorgarle los privilegios necesarios sobre la base de datos existente (*prueba.backup*) y para recargar/limpiar los privilegios [48]. Finalmente, se sale del *shell* de MySQL:

```
mysql> CREATE USER prueba@%' IDENTIFIED BY 'pruebapass';
mysql> GRANT ALL PRIVILEGES ON prueba_backup.* to prueba@%'
  ↪ IDENTIFIED BY 'pruebapass';
mysql> FLUSH PRIVILEGES;
mysql> EXIT;
```

En las dos primeras consultas SQL se indica el “%” para indicar que el usuario **prueba** se puede conectar desde cualquier nombre de dominio y con la contraseña “**pruebapass**”. Además, en la segunda consulta se agrega el símbolo “*” para indicar que todos los permisos serán asignados a la base de datos **prueba_backup** y se aplicarán a todas las tablas [48].

Tras las consultas MySQL realizadas, se procede a verificar el balanceo de carga de la base de datos. Para hacer esto solo basta con ejecutar la siguiente orden varias veces seguidas para ver el nombre de quien está ofreciendo la base de datos en cada instante:

```
[root@nodo3-tft ~]# mysql -u prueba -p -h 192.168.122.246 -e "show
  ↪ variables like 'wsrep_node_name'"
```

En la ilustración 7.57 se puede distinguir que, las tres veces que se ejecutó este comando, el nombre de los nodos que ofrecía la base de datos iba cambiando.

```
[root@nodo3-tft ~]# mysql -u prueba -p -h 192.168.122.246 -e "show variables like 'wsrep_node_name'"
Enter password:
+-----+
| Variable_name | Value |
+-----+
| wsrep_node_name | nodo3-tft |
+-----+
[root@nodo3-tft ~]# mysql -u prueba -p -h 192.168.122.246 -e "show variables like 'wsrep_node_name'"
Enter password:
+-----+
| Variable_name | Value |
+-----+
| wsrep_node_name | nodo4-tft |
+-----+
[root@nodo3-tft ~]# mysql -u prueba -p -h 192.168.122.246 -e "show variables like 'wsrep_node_name'"
Enter password:
+-----+
| Variable_name | Value |
+-----+
| wsrep_node_name | nodo3-tft |
+-----+
```

Ilustración 7.57: Prueba de balanceo de carga de la base de datos

Comprobación de las estadísticas del HAProxy

Las estadísticas de HAProxy permiten visualizar la información sobre las conexiones, los estados de cada nodo del clúster que ofrecen servicio, etc., detalladamente. Para corroborar que se puede acceder a esta página, se accede desde *Firefox* a la siguiente dirección: **192.168.122.246:9000/haproxy?stats**. Como se puede ver, se indica la dirección IP del nodo “**Balanceador-de-carga**”, el puerto 9000 y la adición de la *url* “/haproxy?stats”, tal y como se configuró en el fichero de configuración de HAProxy.

En la ilustración 7.58 se muestra una captura de pantalla de la ventana emergente que se inició tras añadir la dirección mencionada en el buscador de *Firefox*. En esta ventana se solicita autenticación, ya que como se indicó en el fichero de configuración de HAProxy que se necesita autenticación con el usuario y la contraseña de haproxy, hay que autenticarse para poder acceder a la página de estadísticas.

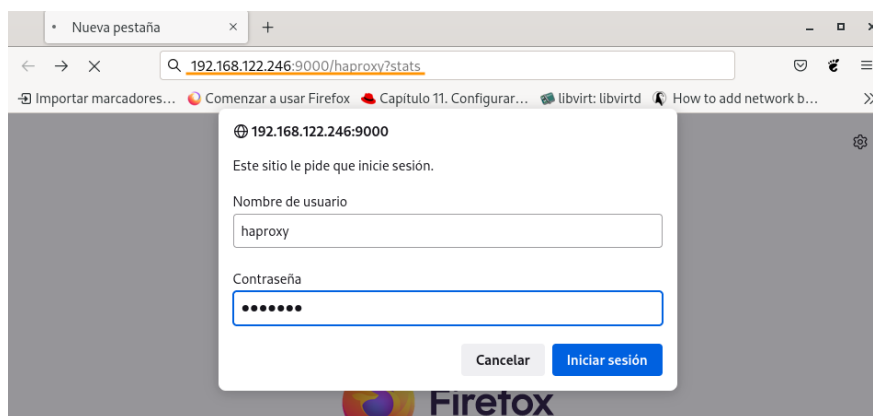


Ilustración 7.58: Autenticación en la página de estadísticas

Una vez autenticados, se muestra la página principal de estadísticas de HAProxy, en la que se ha querido resaltar la parte que se muestra en la ilustración 7.59. Como se puede

observar, se presentan diferentes tablas en las que cada una muestra información en detalle de cada uno de los servicios configurados en el HAProxy y al indicar en el fichero de configuración que se podrá acceder a la página de estadísticas como administrador, se podrá llevar a cabo cualquier acción que se desee en la página.

En las **dos primeras tablas** se muestra la información por defecto que viene configurada en el fichero de configuración de HAProxy.

En las **dos tablas siguientes** se representa la información configurada del servicio HTTP en el fichero de configuración HAProxy, donde se ha creado el *frontend* y el *backend*. Cada tabla representa a estas dos directivas. En la tabla del *backend* se puede manifestar los dos nodos que ofrecen el servicio de Apache y que se encuentran resaltados de color verde, precisando que están funcionando correctamente. Si por ejemplo se apaga el **nodo1-tft**, se resaltará de color rojo para indicar que no está disponible.

En las **dos últimas tablas** se muestran los servicios añadidos en esta sección, *Stats* y *Galera*, respectivamente. Del mismo modo que los nodos del servicio de Apache, los nodos del clúster de MariaDB Galera también se destacan de color verde para indicar que están funcionando correctamente y si se apagara uno, el otro se mostrará en color rojo para señalar que no está en uso.

The screenshot displays the HAProxy statistics page with five tables. Each table has a header with columns: Queue, Session rate, Sessions, Bytes, Denied, Errors, Warnings, and Server. The 'static' table shows a 'Backend' with a status of 'DOWN'. The 'app' table shows four 'app' servers in 'DOWN' state. The 'httpd' table shows a 'Frontend' with a status of 'OPEN'. The 'stats' table shows a 'Frontend' with a status of 'OPEN' and a 'Backend' with a status of 'UP'. The 'galera' table shows a 'Frontend' with a status of 'OPEN' and two 'nodo' servers in 'UP' state.

Ilustración 7.59: Prueba de la página de estadísticas de HAProxy

7.3.2.10. Instalación y activación del cliente de MariaDB

En este apartado se instalará y activará el cliente de MariaDB en los nodos que ofrecerán el servicio web basado en Apache (**Nodo1-servicio** y **Nodo2-servicio**) para poder gestionar el servicio de *WordPress* próximamente. Por consiguiente, los siguientes pasos se realizarán en ambos nodos (**Nodo1-servicio** y **Nodo2-servicio**).

Primero, para instalar el paquete de MariaDB se ejecuta el siguiente comando:

```
[root@nodo1-tft ~]# dnf -y install mariadb-server
```

Y para iniciar y habilitar el servicio para que cada vez que los nodos se arranquen se inicien, se ejecuta las siguientes órdenes:

```
[root@nodo1-tft ~]# systemctl start mariadb  
[root@nodo1-tft ~]# systemctl enable mariadb
```

Finalmente, para que la instalación de MariaDB sea más segura, se modifican los valores por defecto, ya que no son aconsejables para montar un servidor en producción. Por ello, se ejecuta la siguiente orden en los nodos del clúster que utilizarán MariaDB (**Nodo1-servicio**, **Nodo2-servicio**, **Nodo3-almacenamiento** y **Nodo4-almacenamiento**):

```
[root@nodo1-tft ~]# mysql_secure_installation
```

En la ejecución de este comando, se tendrá que responder una serie de preguntas, las cuales son las siguientes y se contestarán a todas afirmativamente (y):

1. Cambiar a autenticación `unix_socket`.
2. Cambiar la contraseña de `root`.
3. Eliminar los usuarios anónimos.
4. No permitir el inicio de sesión de `root` de forma remota.
5. Eliminar la base de datos de prueba y el acceso a ella.
6. Recargar tablas de privilegios ahora.

Una vez finalizada las cuestiones, si se desea ejecutar el comando de MySQL para acceder al *shell* de MySQL, habrá que autenticarse con la nueva contraseña indicada.

7.3.2.11. Configuración del espacio de almacenamiento compartido y distribuido

Tal y como se ha mencionado previamente, se ha diseñado un área de almacenamiento compartido y distribuido basada en la tecnología **iSCSI** y en el sistema de archivos distribuido **GFS2** para almacenar todos los datos comunes que tienen los nodos del clúster. Para desarrollarlo, se ha creado otra máquina virtual denominada “**iSCSI-almacenamiento**” que

pretende simular un servicio iSCSI para proporcionar el espacio de almacenamiento compartido y distribuido con el sistema de archivos GFS2. Los ficheros que se van a almacenar en el espacio de almacenamiento son: los ficheros del servicio web del “**Nodo1-servicio**” y del “**Nodo2-servicio**”, los ficheros de configuración de los servicios de Apache y los de la base de datos. Para esta demostración, se configurará un clúster con almacenamiento iSCSI con fines **demostrativos**.

Como la máquina virtual “**iSCSI-almacenamiento**” se encargará de ofrecer un espacio de almacenamiento, hay que crear y asociar un nuevo volumen a la máquina. Para ello, se selecciona en el menú superior de la pantalla inicial la pestaña “Editar”, se accede a “Detalles de la máquina virtual” y se abre una ventana emergente donde se hace clic en “Agregar hardware” y se selecciona el recurso de “Almacenamiento” para agregar el nuevo volumen indicando el tamaño (10 GiB), justo como se observa en la ilustración 7.60.

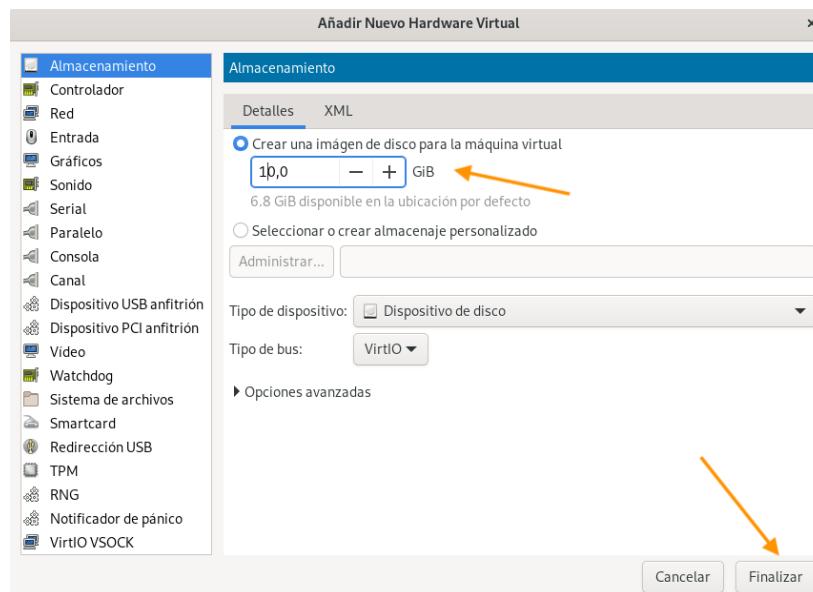


Ilustración 7.60: Creación y asociación de un nuevo volumen en **iSCSI-almacenamiento**

Se puede verificar que dicho volumen se ha agregado correctamente iniciando la máquina virtual y ejecutando el comando `lsblk` para corroborar que el nuevo disco (vdb) se encuentra entre el resto (**Ver ilustración 7.61**).

```
[prueba@fedora ~]$ lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
sr0 11:0 1 1024M 0 rom
zram0 251:0 0 1,9G 0 disk [SWAP]
vda 252:0 0 15G 0 disk
├─vda1 252:1 0 1G 0 part /boot
└─vda2 252:2 0 14G 0 part /home
vdb 252:16 0 10G 0 disk
```

Ilustración 7.61: Verificación del nuevo volumen

Con el nuevo volumen de almacenamiento disponible, se procede a la instalación de los paquetes *software* necesarios tanto en la máquina “**iSCSI-almacenamiento**” como en **todos los nodos** del clúster.

Tal y como se ha comentado, la tecnología iSCSI es un protocolo que permite configurar un área de almacenamiento compartido en un dispositivo de almacenamiento y realizar transferencias entre equipos y el dispositivo de almacenamiento. El dispositivo se conoce como *target* y los equipos que se conectan al dispositivo se denominan *initiator* [35].

iSCSI es un protocolo basado en red, por esto, para los equipos que desean utilizar el almacenamiento iSCSI, primero deben encontrar el dispositivo en la red y luego realizar una conexión hacia él, lo que se conoce como **descubrimiento** y **conexión u inicio de sesión** de los nodos.

Instalación de los paquetes de software

Así pues, se procede a instalar los paquetes necesarios en todas las máquinas virtuales. Para el dispositivo de almacenamiento se descargará la herramienta **targetcli** debido a que permite gestionar interconexiones de iSCSI y para los equipos que se conectarán a dicho dispositivo, el paquete **iscsi-initiator-utils** [?].

Por lo tanto, para la máquina virtual “**iSCSI-almacenamiento**” (dispositivo de almacenamiento) se instala:

```
[root@balanceador-tft ~]# dnf -y install targetcli
```

Y en todos los nodos del clúster, que actuarán como clientes, se instala:

```
[root@nodo1-tft ~]# dnf -y install iscsi-initiator-utils
```

En el caso de Fedora Workstation 35, al ejecutar este último comando, no se ha realizado ninguna instalación debido a que el paquete ya se encontraba instalado.

Configuración del disco compartido

El siguiente paso es configurar el disco/volumen que será el almacenamiento compartido para que, posteriormente, pueda ser utilizado. Para ello, se crea un **LVM (Logical Volumen Manager)**⁶ desde el servidor iSCSI, es decir, se crea un volumen lógico LVM que consta del disco **/dev/vdb** [49].

En primer lugar, se crea un volumen físico con el disco **/dev/vdb** para poder utilizarlo posteriormente en la creación del grupo de volúmenes. Y, después de crear el grupo de volúmenes, se crea el volumen lógico a partir de este.

⁶Gestor de volúmenes lógicos de Linux que permite crear volúmenes físicos, lógicos y un grupo de volúmenes y administrarlos como se desee.

Los comandos ejecutados en el nodo “iSCSI-almacenamiento” se pueden ver en la ilustración 7.62:

```
[root@iscsi-tft ~]# pvcreate /dev/vdb
Physical volume "/dev/vdb" successfully created.
[root@iscsi-tft ~]# vgcreate vg_iscsi /dev/vdb
Volume group "vg_iscsi" successfully created
[root@iscsi-tft ~]# lvcreate -l 100%FREE -n lv_iscsi vg_iscsi
Logical volume "lv_iscsi" created.
```

Ilustración 7.62: Creación de un LVM en iSCSI-almacenamiento

En la ilustración 7.63 se puede corroborar que se ha creado correctamente el volumen lógico.

```
[root@iscsi-tft ~]# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sr0                  11:0    1 1024M  0 rom
zram0                251:0    0   1,9G  0 disk [SWAP]
vda                  252:0    0    15G  0 disk
├─vda1               252:1    0     1G  0 part /boot
└─vda2               252:2    0    14G  0 part /home
vdb                  252:16   0    10G  0 disk /
└─vg_iscsi-lv_iscsi 253:0    0    10G  0 lvm
```

Ilustración 7.63: Verificación del nuevo volumen con LVM en iSCSI-almacenamiento

Creando el almacenamiento compartido

Para empezar, se inicia y habilita el demonio **iscsid** en **todos los nodos** del clúster. Como siempre, se realiza la demostración en un nodo del clúster para evitar hacer el proceso tedioso. Los comandos a ejecutar son:

```
[root@iscsi-tft ~]# systemctl start iscsid
[root@iscsi-tft ~]# systemctl enable iscsid
```

Una vez iniciado el demonio, **se analiza** el nombre del *initiator* de cada uno de los nodos y **se modifican** para evitar la complejidad del nombre y trabajar cómodamente con el nombre identificativo de cada uno, el cual es más fácil de recordar. El fichero `/etc/iscsi-d/initiatorname.iscsi` indica el nombre del *initiator* en el nodo en el que se compruebe. En la ilustración 7.64 se muestran los pasos realizados para diferenciar el cambio del antiguo nombre al nuevo. Los pasos se ejecutan en **todos los nodos** del clúster.

```
[root@nod01-tft ~]# cd /etc/iscsi
[root@nod01-tft iscsi]# cat initiatorname.iscsi
InitiatorName=iqn.1994-05.com.redhat:47353f27534
[root@nod01-tft iscsi]# vi initiatorname.iscsi
[root@nod01-tft iscsi]# cat initiatorname.iscsi
InitiatorName=iqn.1994-05.com.redhat:nod01
```

Ilustración 7.64: Cambio del nombre del *initiator*

El cambio realizado se puede apreciar subrayado con el color amarillo, donde primero se identificaba al nodo con un identificador y después de modificarlo, con el nombre del nodo correspondiente.

A continuación, se inicia y habilita la herramienta **targetcli** en la máquina del servidor iSCSI (**iSCSI-almacenamiento**) para que se inicie en el momento del arranque con las siguientes órdenes:

```
[root@nodo1-tft ~]# systemctl start targetcli
[root@nodo1-tft ~]# systemctl enable targetcli
```

Además, se abre el puerto 3260 por el protocolo TCP en el *firewall* de la máquina “**iSCSI-almacenamiento**” y se recarga la configuración:

```
[root@iscsi-tft ~]# firewall-cmd --zone=libvirt --permanent
↪ --add-port=3260/tcp
[root@iscsi-tft ~]# firewall-cmd --reload
```

Acto seguido, se ejecuta el comando **targetcli** con el fin de obtener una CLI (*Command-line interface*)⁷, en español, interfaz de línea de comandos, de iSCSI y poder crear el almacenamiento compartido. Para esto, se siguen una serie de pasos indicados en la ilustración 7.65.

```
[root@iscsi-tft ~]# targetcli
targetcli shell version 2.1.54
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.

/> cd /backstores/block
/backstores/block> create iscsi_almacenamiento /dev/vg_iscsi/lv_iscsi
Created block storage object iscsi_almacenamiento using /dev/vg_iscsi/lv_iscsi.
/backstores/block> cd /iscsi
/iscsi> create
Created target iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664:sn.b115191a1fa2.
Created TPG 1.
/iscsi> cd iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664:sn.b115191a1fa2/tpg1/acls
/iscsi/iqn.20...fa2/tpg1/acls> create iqn.1994-05.com.redhat:nodo1
Created Node ACL for iqn.1994-05.com.redhat:nodo1
/iscsi/iqn.20...fa2/tpg1/acls> create iqn.1994-05.com.redhat:nodo2
Created Node ACL for iqn.1994-05.com.redhat:nodo2
/iscsi/iqn.20...fa2/tpg1/acls> create iqn.1994-05.com.redhat:nodo3
Created Node ACL for iqn.1994-05.com.redhat:nodo3
/iscsi/iqn.20...fa2/tpg1/acls> create iqn.1994-05.com.redhat:nodo4
Created Node ACL for iqn.1994-05.com.redhat:nodo4
/iscsi/iqn.20...fa2/tpg1/acls> create iqn.1994-05.com.redhat:balanceador
Created Node ACL for iqn.1994-05.com.redhat:balanceador
/iscsi/iqn.20...fa2/tpg1/acls> cd /iscsi/iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664:sn.b115191a1fa2/tpg1/luns
/iscsi/iqn.20...fa2/tpg1/luns> create /backstores/block/iscsi_almacenamiento
Created LUN 0.
Created LUN 0->0 mapping in node ACL iqn.1994-05.com.redhat:balanceador
Created LUN 0->0 mapping in node ACL iqn.1994-05.com.redhat:nodo4
Created LUN 0->0 mapping in node ACL iqn.1994-05.com.redhat:nodo3
Created LUN 0->0 mapping in node ACL iqn.1994-05.com.redhat:nodo2
Created LUN 0->0 mapping in node ACL iqn.1994-05.com.redhat:nodo1
/iscsi/iqn.20...fa2/tpg1/luns> cd /
```

Ilustración 7.65: Pasos para crear el almacenamiento compartido

⁷Interfaz de usuario basada en texto que permite administrar programas [?].

En primer lugar, se ha accedido al directorio `/backstores/block` y desde ahí se ha creado un **objeto de almacenamiento en bloque** porque el dispositivo de almacenamiento que se posee es un volumen LVM y, este tipo de almacenamiento, permite dispositivos físicos y lógicos. En segundo lugar, se ha posicionado en el directorio `/iscsi` y se ha creado un **target iSCSI** para el dispositivo de almacenamiento con un nombre por defecto, ya que al ejecutar el comando `create` sin indicar un nombre, se asigna uno por defecto. Al crear un **target iSCSI** se crea un **Grupo de Portales de Destino** asociado que se denomina “`tpg1`”.

En tercer lugar, para permitir el acceso de los *initiators* al dispositivo *target*, se hace uso de las listas de control de acceso (ACL). Se navega hasta el directorio `/iscsi/identificador_target/tpg1/acls` y se crean las **ACL**. Como previamente se ha modificado el nombre de identificación de los *initiators*, se utiliza el comando `create` seguido del nombre de cada uno de los nodos del clúster para configurar las ACL.

Para concluir, se navega a `/iscsi/identificador_target/tpg1/luns` y se crea un **LUN** (*Logical unit number*)⁸ iSCSI del *target* de almacenamiento en bloque.

A continuación, se navega hasta la raíz (`/`) y se verifica que el *target* iSCSI de almacenamiento en bloque, el *target* iSCSI, las listas de control de acceso y el LUN, se han creado correctamente. Tras dicha verificación, se guarda la configuración y se sale del CLI de iSCSI (Ver ilustración 7.66).

```

/iscsi/iqn.20...fa2/tpg1/luns> cd /
/> ls
o- / ..... [ ... ]
  o- backstores ..... [ ... ]
    o- block ..... [Storage Objects: 1]
      | o- iscsi_almacenamiento [/dev/vg_iscsi/lv_iscsi (10.0GiB) write-thru activated]
        | o- alua ..... [ALUA Groups: 1]
          | o- default_tg_pt_gp ..... [ALUA state: Active/optimized]
        o- fileio ..... [Storage Objects: 0]
        o- pscsi ..... [Storage Objects: 0]
        o- ramdisk ..... [Storage Objects: 0]
      o- iscsi ..... [Targets: 1]
        o- iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664:sn.b115191a1fa2 ..... [TPGs: 1]
          o- tpg1 ..... [no-gen-acls, no-auth]
            o- acls ..... [ACLs: 5]
              | o- iqn.1994-05.com.redhat:balanceador ..... [Mapped LUNs: 1]
                | o- mapped_lun0 ..... [lun0 block/iscsi_almacenamiento (rw)]
              | o- iqn.1994-05.com.redhat:nodo1 ..... [Mapped LUNs: 1]
                | o- mapped_lun0 ..... [lun0 block/iscsi_almacenamiento (rw)]
              | o- iqn.1994-05.com.redhat:nodo2 ..... [Mapped LUNs: 1]
                | o- mapped_lun0 ..... [lun0 block/iscsi_almacenamiento (rw)]
              | o- iqn.1994-05.com.redhat:nodo3 ..... [Mapped LUNs: 1]
                | o- mapped_lun0 ..... [lun0 block/iscsi_almacenamiento (rw)]
              | o- iqn.1994-05.com.redhat:nodo4 ..... [Mapped LUNs: 1]
                | o- mapped_lun0 ..... [lun0 block/iscsi_almacenamiento (rw)]
            o- luns ..... [LUNs: 1]
              | o- lun0 [block/iscsi_almacenamiento (/dev/vg_iscsi/lv_iscsi) (default_tg_pt_gp)]
                | o- portals ..... [Portals: 1]
                  | o- 0.0.0.0:3260 ..... [OK]
            o- loopback ..... [Targets: 0]
            o- vhost ..... [Targets: 0]
/> saveconfig
configuration saved to /etc/target/saveconfig.json
/> exit
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/target/backup/.
Configuration saved to /etc/target/saveconfig.json

```

Ilustración 7.66: Verificación del almacenamiento compartido recién creado

⁸Dirección que se asigna a una unidad de disco duro para diferenciarla dentro de un bus SCSI [?].

Una vez creado el almacenamiento compartido, lo siguiente que se debe hacer es descubrir los nodos que utilizarán dicho almacenamiento e iniciar sesión para que el disco dedicado a este propósito aparezca visible para todos los nodos.

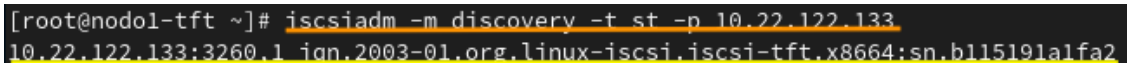
Descubriendo el almacenamiento compartido

El proceso de descubrimiento e inicio de sesión se realiza con la herramienta **iscsiadm**, el cual forma parte del paquete **iscsi-initiator-utils** instalado previamente.

Para descubrir el *target* se ejecuta el siguiente comando en **todos los nodos**:

```
[root@nodo1-tft ~]# iscsiadm -m discovery -t st -p 10.22.122.133
```

Para comprobar que se ha ejecutado correctamente, se debería mostrar como resultado exactamente lo mismo que se aprecia en la ilustración 7.67, en **todos los nodos**.



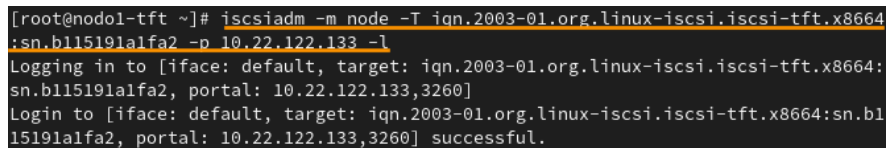
```
[root@nodo1-tft ~]# iscsiadm -m discovery -t st -p 10.22.122.133
10.22.122.133:3260.1 iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664:sn.b115191alfa2
```

Ilustración 7.67: Descubriendo el *target* en todos los nodos

Seguidamente, se reinicia el demonio **iscsid** en **todos los nodos** para asegurar que el descubrimiento del *target* se ha efectuado adecuadamente:

```
[root@nodo1-tft ~]# systemctl restart iscsid
```

Una vez descubierto el dispositivo *target*, se inicia sesión en **cada uno de los nodos** con la dirección IP de la máquina “**iSCSI-almacenamiento**” (por la red de almacenamiento) y con el IQN (iSCSI qualified name) del *target*, es decir, con la ristra de caracteres utilizada para identificar de manera única a dicho dispositivo. El comando a ejecutar es el que se expresa a continuación:



```
[root@nodo1-tft ~]# iscsiadm -m node -T iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664
:sn.b115191alfa2 -p 10.22.122.133 -l
Logging in to [iface: default, target: iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664:
sn.b115191alfa2, portal: 10.22.122.133,3260]
Login to [iface: default, target: iqn.2003-01.org.linux-iscsi.iscsi-tft.x8664:sn.bl
15191alfa2, portal: 10.22.122.133,3260] successful.
```

Ilustración 7.68: Iniciando sesión en todos los nodos

En la ilustración 7.68 se muestra la ejecución del comando en el “**Nodo1-servicio**” y el resultado tras dicha ejecución, donde se puede apreciar que se ha iniciado sesión al *target* especificado por la dirección IP y puerto detallados. Este comando se debe ejecutar en **todos los nodos** del clúster, ya que se han añadido a la ACL previamente.

Se vuelve a reiniciar el demonio de **iscsid** en **cada uno de los nodos** para corroborar que los inicios de sesión se han llevado a cabo correctamente.

En este momento, ya se puede comprobar que en cada nodo, al ejecutar el comando `lsblk` se muestra un nuevo disco llamado `/dev/sda`, actuando como si fuera un disco físico. Este disco es el que será utilizado como almacenamiento compartido para todos los nodos que han sido identificados o conectados.

```
[root@nodo1-tft ~]# lsblk
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda   8:0    0   10G  0 disk
sr0   11:0   1 1024M  0 rom
zram0 251:0   0   1,9G  0 disk [SWAP]
vda   252:0   0    15G  0 disk
├─vda1 252:1   0     1G  0 part /boot
└─vda2 252:2   0    14G  0 part /home
/

[root@nodo1-tft ~]# fdisk -l | grep -i sd
Disco /dev/sda: 10 GiB, 10733223936 bytes, 20963328 sectores
```

Ilustración 7.69: Comprobación del disco en todos los nodos

Como se puede ver en la ilustración 7.69, se ha hecho visible en todos los nodos el disco de la máquina **iSCSI-almacenamiento**. Se puede corroborar esto comparando el tamaño del disco con el que se ha mostrado en la ilustración 7.61, ya que ambos son de 10 GB.

Instalando y configurando el sistema de archivos GFS2

En este apartado, se instalará y configurará el sistema de archivos que tendrá el área de almacenamiento compartido de los nodos del clúster. Como también se pretende tener un espacio de almacenamiento distribuido, es decir, que todo lo que se modifique en el área de almacenamiento de un nodo, se vea reflejado automáticamente en el resto de nodos, se debe formatear el disco con el sistema de archivos GFS2, ya que brinda un sistema de archivos distribuido específicamente para clúster de servidores en Linux, permitiendo que todos los nodos tengan acceso simultáneo al dispositivo de almacenamiento que están compartiendo [14].

Para permitir realizar dicha instalación y configuración, se debe crear previamente un volumen físico y un grupo de volúmenes sobre el disco `/dev/sda`. En la ilustración 7.70, se muestra la ejecución de los comandos correspondientes para crear el volumen físico y el grupo de volúmenes en el “**Nodo1-servicio**”.

```
[root@nodo1-tft ~]# pvcreate /dev/sda
Physical volume "/dev/sda" successfully created.
[root@nodo1-tft ~]# vgcreate vg_almacenamiento /dev/sda
Volume group "vg_almacenamiento" successfully created
```

Ilustración 7.70: Creación del volumen físico y el grupo de volúmenes

En este caso, estos comandos se ejecutan solamente en el nodo mencionado y se reinician todas las máquinas para actualizar la información de los discos en todas. En la ilustración 7.71 se verifica la correcta creación del volumen físico y del grupo de volúmenes.

```
[root@nodo1-tft ~]# pvscan
PV /dev/sda   VG vg_almacenamiento   lvm2 [9,99 GiB / 5,99 GiB free]
Total: 1 [9,99 GiB] / in use: 1 [9,99 GiB] / in no VG: 0 [0  ]
[root@nodo1-tft ~]# vgscan
Found volume group "vg_almacenamiento" using metadata type lvm2
```

Ilustración 7.71: Comprobación del volumen físico y el grupo de volúmenes creados

Con el grupo de volúmenes creado y visible en todas las máquinas, se necesita instalar los paquetes de *software* necesarios en **todos los nodos** [50], para ello se ejecuta:

```
[root@nodo1-tft ~]# dnf -y install lvm2-lockd gfs2-utils dlm
```

Seguidamente, se inician y habilitan los servicios **lvmlockd** y **dlm** en **todos los nodos** del clúster:

```
[root@nodo1-tft ~]# systemctl start lvmlockd
[root@nodo1-tft ~]# systemctl start dlm
[root@nodo1-tft ~]# systemctl enable lvmlockd
[root@nodo1-tft ~]# systemctl enable dlm
```

El siguiente paso es modificar el parámetro “*use.lvmlockd*” del fichero `/etc/lvm/lmv.conf`, igualándolo a 1 para que el protocolo de cerrojo se configure correctamente para el acceso al volumen compartido [50]. Este parámetro se debe editar en **todos los nodos** del clúster. En la ilustración 7.72 se muestra la porción del fichero donde se ha modificado el parámetro.

```
# Configuration option global/use_lvmlockd.
# Use lvmlockd for locking among hosts using LVM on shared storage.
# Applicable only if LVM is compiled with lockd support in which
# case there is also lvmlockd(8) man page available for more
# information.
use_lvmlockd = 1
```

Ilustración 7.72: Modificación del parámetro `use.lvmlockd` del fichero `lvm.conf`

Una vez modificado el parámetro en todos los nodos, se reinicia el dominio **lvmlockd** en **todos los nodos** del clúster para actualizar los cambios.

```
[root@nodo1-tft ~]# systemctl restart lvmlockd
```

A continuación, se ejecuta el siguiente comando para iniciar el gestor de bloqueos para el grupo de volúmenes compartido [50]. En este caso, se ejecuta en **todos los nodos** del clúster, **excepto** en el “**Nodo1-servicio**”, ya que fue donde se creó el grupo de volúmenes.

```
[root@nodo2-tft ~]# vgchange --lock-start vg_almacenamiento
```

En este punto, ya estaría casi preparado el almacenamiento compartido y distribuido para todos los nodos, lo único que faltaría es crear un volumen lógico para los ficheros del servicio Apache y otro para los del servicio de MariaDB y realizar las correspondientes configuraciones.

Lo siguiente que se procederá a ejecutar es la instalación del servidor LAMP, necesario para poder instalar WordPress y por último, **se crearán y configurarán los volúmenes lógicos** mencionados.

7.3.2.12. Instalación y configuración del servidor LAMP

Hasta el momento, el clúster está completamente configurado, ofreciendo el servicio Apache y la base de datos de MariaDB en alta disponibilidad y con balanceo de carga. También está disponible el espacio de almacenamiento compartido y distribuido para todos los nodos del clúster. Para completar el proyecto, hay que instalar y configurar el CMS de *WordPress* para finalmente mover los directorios comunes de los nodos (Apache y MariaDB) al área de almacenamiento compartido.

Por lo tanto, lo siguiente que se llevará a cabo es la instalación de *WordPress* para ofrecer este servicio web en la infraestructura de clúster diseñada y desarrollada. Sin embargo, para ello se debe **instalar y configurar** previamente el **software necesario** para que *WordPress* pueda ser ejecutado. Este es el **servidor LAMP**, el cual es un entorno formado por los servicios de “Linux”, “Apache”, “MySQL” y “PHP” que permite alojar y ejecutar servicios webs.

Para realizar la descarga de *WordPress*, hay que disponer de una serie de requisitos previos descritos en la sección “**Requerimientos del servicio web basado en Apache**”. En general, uno de los principales es tener un sistema Linux ejecutando un servidor LAMP, es decir, un servicio web disponiendo de los servicios PHP, Apache, MySQL o MariaDB. En este caso, se utilizará MariaDB, ya que se ha seleccionado e instalado previamente. La base de datos de MariaDB, como se ha mencionado anteriormente, está alojada en los nodos “**Nodo3-almacenamiento**” y “**Nodo4-almacenamiento**”.

En el **Anexo VIII** se puede observar todos los pasos llevados a cabo para realizar la instalación y configuración de la pila LAMP en los nodos que ofrecen el servicio web (**Nodo1-servicio** y **Nodo2-servicio**).

7.3.2.13. Instalación y configuración de *WordPress*

Tras la instalación y configuración necesaria para formar el servidor LAMP completamente, se lleva a cabo la **instalación de *WordPress*** para Fedora Workstation 35 en **ambos nodos**. Como se ha comentado previamente, la carpeta del contenido web de Apache y *WordPress* y los ficheros de configuración de Apache estarán almacenados en el espacio de almacenamiento compartido y distribuido.

En este caso se trabajará sobre HTTP y se integrará *WordPress* en Fedora Workstation 35 como la **página principal** del servicio web que ofrece [51].

Antes de comenzar con la descarga del CMS, se ejecuta el siguiente comando para instalar dos herramientas del sistema necesarias en el “**Nodo1-servicio**” y “**Nodo2-servicio**”:

```
[root@nodo1-tft ~]# dnf -y tar wget
```

En Fedora Workstation 35 no hace falta instalar estas herramientas, ya que cuando se va a instalarlas, se muestra un mensaje indicando que ya vienen instaladas cuando se instala y configura el sistema operativo.

Descargando WordPress

Ahora bien, se procede a descargar el paquete que posee *WordPress* desde la web oficial con la herramienta *wget* de la siguiente manera y en **ambos nodos**:

```
[root@nodo1-tft ~]# wget  
↪ https://es.wordpress.org/latest-es_ES.tar.gz
```

El comando se ha efectuado correctamente si informa que la carpeta “latest-es_ES.tar.gz” se ha guardado y además, si se identifica en el directorio donde se encontraba cuando se ejecutó dicho comando.

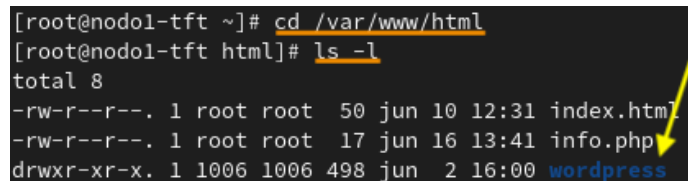
Configurando archivos y permisos de WordPress

Antes de proceder a la instalación de *WordPress*, se deben realizar una serie de **ajustes necesarios** para dejar configurado el sistema para ejecutar *WordPress* adecuadamente [51].

Primero, se descomprime la carpeta recién descargada de *WordPress* en la ubicación deseada. En este caso, se descomprime en el directorio `/var/www/html` ya que, como se ha indicado, se integrará como la página principal del servicio web. Para descomprimir la carpeta se ejecuta en **ambos nodos** lo siguiente:

```
[root@nodo1-tft ~]# tar xf latest-es_ES.tar.gz -C /var/www/html
```

Se puede verificar que se ha descomprimido la carpeta accediendo al directorio `/var/www/html` y comprobando que se haya creado una nueva carpeta llamada “*wordpress*” (**Ver ilustración 7.73**).



```
[root@nodo1-tft ~]# cd /var/www/html  
[root@nodo1-tft html]# ls -l  
total 8  
-rw-r--r--. 1 root root 50 jun 10 12:31 index.html  
-rw-r--r--. 1 root root 17 jun 16 13:41 info.php  
drwxr-xr-x. 1 1006 1006 498 jun 2 16:00 wordpress
```

Ilustración 7.73: Verificación de la carpeta descomprimida de WordPress

Luego, se cambian los permisos de usuario y grupo al directorio “*wordpress*” y todo su contenido para que *WordPress* pueda escribir en dicho directorio. El usuario al que se le da dichos permisos es a “*apache*”, ya que es el encargado de ejecutar el servicio web [51]. En la

ilustración 7.74 se puede ver el comando ejecutado y el cambio de los permisos en la carpeta “wordpress”.

```
[root@nodo1-tft ~]# chown -R apache:apache /var/www/html/wordpress/
[root@nodo1-tft ~]# ls -l /var/www/html
total 8
-rw-r--r--. 1 root root 50 jun 10 12:31 index.html
-rw-r--r--. 1 root root 17 jun 16 13:41 info.php
drwxr-xr-x. 1 apache apache 498 jun 2 16:00 wordpress
```

Ilustración 7.74: Cambio de permisos de la carpeta wordpress

Para poder navegar a la página oficial de *WordPress*, se edita el fichero `/etc/httpd/conf.d/wordpress.conf` añadiendo las directivas señaladas (flecha amarilla) en la ilustración 7.75, permitiendo de este modo, el uso de los archivos `.htaccess` en el directorio (`/var/www/html/wordpress`) [51].

```
[root@nodo1-tft ~]# vi /etc/httpd/conf.d/wordpress.conf
[root@nodo1-tft ~]# cat /etc/httpd/conf.d/wordpress.conf
<Directory /var/www/html/wordpress>
    AllowOverride All
</Directory>
```

Ilustración 7.75: Modificación del fichero wordpress.conf

A continuación, se creará un *virtual host* sencillo para establecer que *WordPress* será la página principal del servicio web de Fedora 35, evitando de esta manera que el fichero `index.html` previamente creado para la verificación del correcto funcionamiento de Apache, se muestre inicialmente. Para crear este *virtual host*, se accede al fichero `/etc/httpd/conf/httpd.conf` y se agrega la directiva con el parámetro “*DocumentRoot*” para especificar el directorio “wordpress” (Ver ilustración 7.76).

```
<VirtualHost *:80>
    DocumentRoot /var/www/html/wordpress
</VirtualHost>
```

Ilustración 7.76: Modificación del fichero wordpress.conf

Una vez modificados y guardados los ficheros, se recarga el demonio `httpd` para actualizar dicha configuración en los **dos nodos**:

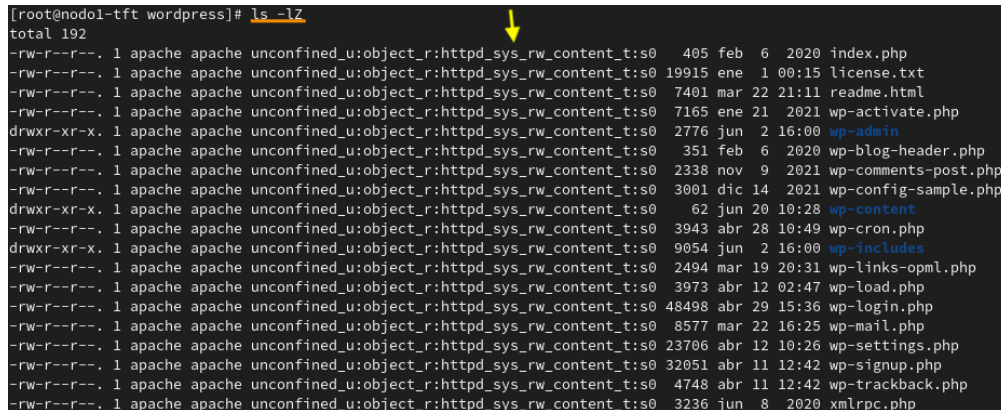
```
[root@nodo1-tft ~]# systemctl reload httpd
```

Configurando SELinux para WordPress

SELinux permanece en modo *enforcing* durante todo el desarrollo del proyecto. Debido a esto, hay que añadir el contexto que se presenta a continuación para darle permisos a *WordPress* para que pueda manipular el directorio “wordpress” y su contenido, ya que no basta solo con los permisos de usuario y grupo. Los comandos a ejecutar en **ambos nodos** son:


```
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_rw_content_t
↳ "/var/www/html/wordpress(/.*)?"
[root@nodo1-tft ~]# restorecon -R /var/www/html/wordpress
```

En la ilustración 7.77 se puede verificar el cambio de los contextos SELinux en todo el contenido de la carpeta “wordpress”:



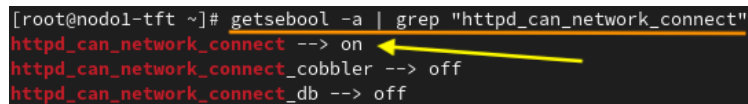
```
[root@nodo1-tft wordpress]# ls -lZ
total 192
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 405 feb 6 2020 index.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 19915 ene 1 00:15 license.txt
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 7401 mar 22 21:11 readme.html
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 7165 ene 21 2021 wp-activate.php
drwxr-xr-x. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 2776 jun 2 16:00 wp-admin
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 351 feb 6 2020 wp-blog-header.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 2338 nov 9 2021 wp-comments-post.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 3001 dic 14 2021 wp-config-sample.php
drwxr-xr-x. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 62 jun 20 10:28 wp-content
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 3943 abr 28 10:49 wp-cron.php
drwxr-xr-x. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 9054 jun 2 16:00 wp-includes
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 2494 mar 19 20:31 wp-links-opml.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 3973 abr 12 02:47 wp-load.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 48498 abr 29 15:36 wp-login.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 8577 mar 22 16:25 wp-mail.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 23706 abr 12 10:26 wp-settings.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 32051 abr 11 12:42 wp-signup.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 4748 abr 11 12:42 wp-trackback.php
-rw-r--r--. 1 apache apache unconfined_u:object_r:httpd_sys_rw_content_t:s0 3236 jun 8 2020 xmlrpc.php
```

Ilustración 7.77: Contextos de SELinux en la carpeta wordpress

También hay que activar el booleano “*httpd_can_network_connect*” en los nodos “**Nodo1-servicio**” y “**Nodo2-servicio**” para permitir el acceso a la red del servicio web a ofrecer:

```
[root@nodo1-tft ~]# setsebool -P httpd_can_network_connect on
```

Se puede comprobar que el estado de este booleano está en modo “on” ejecutando el comando que se pone de manifiesto en la ilustración 7.78:



```
[root@nodo1-tft ~]# getsebool -a | grep "httpd_can_network_connect"
httpd_can_network_connect --> on
httpd_can_network_connect_cobbler --> off
httpd_can_network_connect_db --> off
```

Ilustración 7.78: Booleano de SELinux activado

Creando una base de datos y un usuario para WordPress

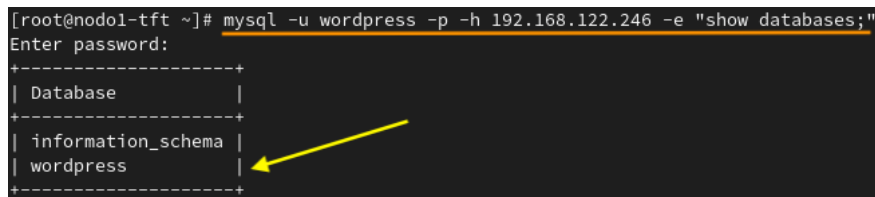
WordPress necesita una base de datos para poder ser utilizado, ya sea MySQL o MariaDB. En este caso, como se ha explicado antes, se utiliza MariaDB para gestionar la base de datos. Por lo tanto, se procede a crear una nueva base de datos y un usuario para manejarla en el CLI de MySQL ejecutando el comando “*mysql -u root -p*” y siguiendo las sentencias SQL indicados a continuación:

```
mysql> CREATE DATABASE wordpress;
mysql> CREATE USER wordpress@'%' IDENTIFIED BY 'passwordpress';
mysql> GRANT ALL PRIVILEGES ON wordpress.* to wordpress@'%'
↳ IDENTIFIED BY 'passwordpress';
```



```
mysql> FLUSH PRIVILEGES;
mysql> EXIT;
```

Esto se ejecuta en uno de los nodos dedicados al almacenamiento de la base de datos (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**). En este caso se ha realizado en el “**Nodo4-almacenamiento**”, por lo tanto, en el resto de los nodos, se debería de haber replicado la base de datos y el usuario. En la ilustración 7.79 se puede verificar que, ejecutando el comando subrayado en naranja, que trata de ejecutar una consulta SQL para mostrar las bases de datos del usuario “*wordpress*” en el “**Nodo1-servicio**”, se observa la base de datos “*wordpress*” creada anteriormente.



```
[root@nodo1-tft ~]# mysql -u wordpress -p -h 192.168.122.246 -e "show databases;"
Enter password:
+-----+
| Database |
+-----+
| information_schema |
| wordpress        |
+-----+
```

Ilustración 7.79: Verificación de la réplica de la base de datos

Este comando puede ser ejecutado en cualquiera de los otros nodos del clúster para comprobar que se haya replicado correctamente la base de datos. En este preciso momento, *WordPress* ya se encuentra totalmente configurado para ser instalado.

Instalando *WordPress*

Como se ha instalado Apache en dos nodos, es necesario parar el servicio **httpd** en un nodo de ellos para poder saber qué nodo está dando el servicio y poder instalar *WordPress* y, volver a realizar lo mismo, pero a la inversa, para que los ficheros que se generan y configuran en la instalación, se encuentren en los directorios de ambos nodos. Por ello, se para primero el demonio **httpd** en el “**Nodo1-servicio**” para comenzar a configurar *WordPress* en el “**Nodo2-servicio**”:

```
[root@nodo1-tft ~]# systemctl stop httpd
```

En este punto, se sabe que el nodo que está dando el servicio es el “**Nodo2-servicio**”, por lo tanto, se ejecutan los pasos descritos en el [Anexo IX](#) para realizar la instalación y configuración restante de *WordPress*.

Cuando se hayan realizado todos los pasos del [Anexo IX](#) y se tenga instalado *WordPress* en el “**Nodo2-servicio**”, se siguen los mismos pasos para instalarlo en el “**Nodo1-servicio**”, pero primero, se inicia el servicio **httpd** en este y luego se para en el “**Nodo2-servicio**” para que dicha instalación se efectúe en el nodo correcto.

Una vez instalado *WordPress* en ambos nodos, se realiza una última comprobación para verificar que el sitio web que ofrece *WordPress* y, el balanceo de carga del servicio Apache,

siguen trabajando correctamente. Para probar el balanceo de carga, y por ende, el funcionamiento de *WordPress*, se edita el código HTML del tema predeterminado de *WordPress* en cada uno de los nodos, añadiendo la frase “El nodo 1 está dando el servicio” en el “**Nodo1-servicio**” y “El nodo 2 está dando el servicio” en el “**Nodo2-servicio**” justo como se aprecia en la ilustración 7.80.

```

Twenty Twenty-Two: header.html (parts/header.html)
Contenido del archivo seleccionado:
1 <!-- wp:group {"layout":{"inherit":true}} -->
2 <div class="wp-block-group"><!-- wp:group {"align":"wide","style":{"spacing":{"top":"var(--wp--custom--spacing--small, 1.25rem)"},"layout":{"typ
3 <div class="wp-block-group alignwide" style="padding-top:var(--wp--custom-
4 <div class="wp-block-group"><!-- wp:site-logo {"width":64} /-->
5
6 <!-- wp:site-title /--></div>
7 <!-- /wp:group -->
8
9 <!-- wp:navigation {"layout":{"type":"flex","setCascadingProperties":true,
10 <!-- wp:page-list {"isNavigationChild":true,"showSubMenuIcon":true,"openSu
11 <!-- /wp:navigation --></div>
12 <!-- /wp:group --></div>
13 <!-- /wp:group -->
14 <h4 style="text-align:center;">El nodo 1 está dando el servicio</h4>

```

Ilustración 7.80: Modificación del tema de *WordPress*

Si se navega a la dirección 192.168.122.246 desde el navegador de *Firefox*, se carga la página principal del sitio web que ofrece *WordPress*, deduciendo de esta manera que el servicio se ofrece correctamente. Además, se puede observar en el encabezado de la página principal del sitio web, la frase que se ha añadido en el tema, verificando que el servicio lo está ofreciendo ahora mismo el “**Nodo1-servicio**” (Ver ilustración 7.81).

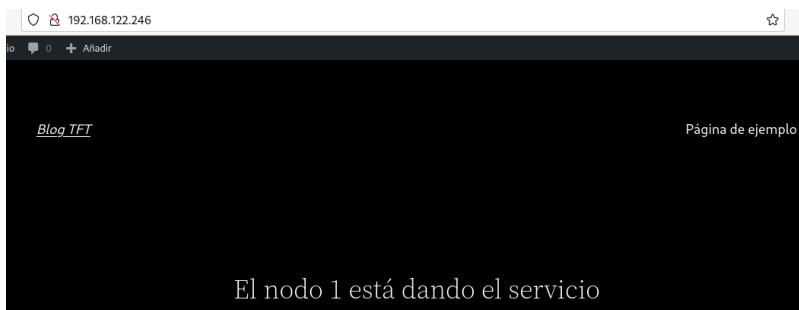


Ilustración 7.81: Verificación del Blog TFT en el **Nodo1-servicio**

Tras recargar el navegador haciendo clic en el icono “↻”, se puede ver en el encabezado de la página principal que la frase ha cambiado, apareciendo ahora que el “**Nodo2-servicio**” es el que ofrece el servicio.

Con esto, se puede concluir que tanto el sitio web que ofrece *WordPress* como el balanceo de carga está trabajando adecuadamente.

Ilustración 7.82: Verificación del Blog TFT en el **Nodo2-servicio**

7.3.2.14. Creación y configuración de volúmenes lógicos

Creando y configurando un volumen lógico para ficheros de Apache

En este apartado, se crea un volumen lógico en un **único nodo** del clúster para utilizarlo como almacenamiento para todos los ficheros que posee el servicio **Apache**. No obstante, para que se puedan llevar a cabo las operaciones de LVM es necesario **desactivar el protocolo de cerrojo** activado anteriormente, el cual es el encargado de controlar los accesos al volumen compartido. Para desactivar esto, hay que volver a cambiar el parámetro “*use_lvmlockd*” en el fichero de configuración `/etc/lvm/lvm.conf`, asignando el valor por defecto (0).

En este caso, se ha realizado en el “**Nodo1-servicio**” y, en la ilustración 7.83, se verifica que el comando para crear dicho volumen lógico ha sido finalizado y que se ha creado correctamente el volumen con **4 GB** de tamaño y con el nombre “**lv_apache**” [49].

```
[root@nodol-tft ~]# lvcreate --activate sv -L4G -n lv_apache vg_almacenamiento
WARNING: gfs2 signature detected on /dev/vg_almacenamiento/lv_apache at offset 6
5536. Wipe it? [y/n]: y
Wiping gfs2 signature on /dev/vg_almacenamiento/lv_apache.
Logical volume "lv_apache" created.
```

Ilustración 7.83: Creación del volumen lógico para el contenido Apache

Luego, se reinician todas las máquinas para actualizar la información de los discos en cada una de ellas. Cuando se hayan iniciado todas las máquinas, se comprueba que los servicios `iscsid`, `lvmlockd` y `dlm` estén **activos** y sin ningún tipo de errores y, se verifica que en **cada una** de las máquinas, el nuevo volumen lógico ha sido actualizado tal y como figura en la ilustración 7.84.

```
[root@nodol-tft ~]# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda                                  8:0    0   10G  0 disk
├─vg_almacenamiento-lv_apache       253:0    0    4G  0 lvm
sr0                                  11:0    1 1024M  0 rom
zram0                                251:0    0   1,9G  0 disk [SWAP]
vda                                  252:0    0    15G  0 disk
├─vda1                               252:1    0     1G  0 part /boot
└─vda2                               252:2    0   14G  0 part /home
/
```

Ilustración 7.84: Comprobación del volumen lógico para el contenido Apache

Si por cualquier razón, alguno de los nodos del clúster arranca antes de la máquina “**iSCSI-almacenamiento**”, el disco no aparecerá conectado. Para solucionar esto, cuando se haya iniciado la máquina del servidor iSCSI, se reinicia el servicio **iscsid** en aquellos nodos que se hayan arrancado antes:

```
[root@nodo1-tft ~]# systemctl restart iscsid
```

Para poder hacer uso de un volumen lógico, hay que formatearlo con un sistema de archivos. Como en este caso se ha indicado que se utilizará el GFS2, se ejecuta el comando que se muestra en la ilustración 7.85 en el “**Nodo1-servicio**” para formatearlo. Esto se realiza solamente en **un nodo**.

```
[root@nodo1-tft ~]# mkfs.gfs2 -j5 -J 10 -p lock_dlm -t cluster-tft:gfs2 /dev/mapper/vg_almacenamiento-lv_apache
/dev/mapper/vg_almacenamiento-lv_apache is a symbolic link to /dev/dm-0
This will destroy any data on /dev/dm-0
Are you sure you want to proceed? [y/n] y
Discarding device contents (may take a while on large devices): Done
Adding journals: Done
Building resource groups: Done
Creating quota file: Done
Writing superblock and syncing: Done
Device:          /dev/mapper/vg_almacenamiento-lv_apache
Block size:      4096
Device size:     4,00 GB (1048576 blocks)
Filesystem size: 4,00 GB (1048575 blocks)
Journals:        5
Journal size:    10MB
Resource groups: 21
Locking protocol: "lock_dlm"
Lock table:      "cluster-tft:gfs2"
UUID:           32c9d3dd-46bf-4930-9ba2-27b414331957
```

Ilustración 7.85: Formateando el volumen lógico para el contenido Apache

Las opciones indicadas en el comando indican el protocolo “*lock_dlm*” con la opción “-p”, el nombre del clúster con la opción “-t”, el número de “*journals*” que se utilizarán con la opción “-j” y el tamaño mínimo del “*journal*” a usar con la opción “-J”. El valor especificado para el número de *journals* es igual que el número de nodos que pueden acceder al clúster, en este caso, el número de nodos del clúster (5), ya que todos podrán acceder al dispositivo compartido. Además, se ha indicado un tamaño de 10 MB para el “*journal*”, puesto que no será un sistema muy grande. Finalmente, se indica la ruta del volumen lógico donde se va a formatear el sistema de archivos [52].

Una vez formateado, se reinician los servicios **iscsid**, **lvmlockd** y **dml** en **cada nodo**:

```
[root@nodo1-tft ~]# systemctl restart iscsid
[root@nodo1-tft ~]# systemctl restart lvmlockd
[root@nodo1-tft ~]# systemctl restart dlm
```

Con todo esto, el área de almacenamiento compartido y distribuido ya está listo para ser utilizado. Para llevar a cabo una pequeña comprobación antes de mover el contenido de Apache al directorio compartido, se puede montar el dispositivo en una carpeta temporal y verificar que todo lo que se haga en un nodo, se pueda ver instantáneamente en el otro. Para

hacer esto, hay que ejecutar el siguiente comando en dos nodos como mínimo para ver dicha función:

```
[root@nodo1-tft ~]# mount -t gfs2
→ /dev/mapper/vg_almacenamiento-lv_apache /mnt
```

Tras esta pequeña demostración, se comienza a realizar los pasos restantes para configurar un directorio compartido para todo el contenido de Apache y del servicio web. En primer lugar, se crean los directorios correspondientes en el “**Nodo1-servicio**” y en el “**Nodo2-servicio**” y se revisan los contextos de SELinux de las carpetas que van a pasar a estar en el espacio compartido.

Para crear el directorio, se ejecuta el siguiente comando en **ambos nodos**:

```
[root@nodo1-tft ~]# mkdir /apache_compartido
```

Los contextos a examinar son los de todos los ficheros de los directorios `/etc/httpd` y `/var/www`. Para ver los contextos de cada directorio y contenido, se utiliza el comando “`ls -lZ`”, seguido del directorio o fichero a inspeccionar. Los contextos que están aplicados a la carpeta `/etc/httpd`, a su **contenido** y a las carpetas `/etc/httpd/conf` y `/etc/httpd/conf.d` son exactamente los mismos en la tercera columna: “**httpd_config_t**”.

Sin embargo, los contextos de la tercera columna del directorio `/var/www`, son más variados. El contenido de dicho directorio, tiene configurado el contexto “**httpd_sys_content_t**” para la carpeta `html` y el contexto “**httpd_sys_script_exec_t**” para la carpeta `cgi-bin`. A su vez, el contenido de la carpeta `html` tiene el mismo contexto, a excepción de la carpeta “`wordpress`” y su **contenido**, que poseen el contexto “**httpd_sys_rw_content_t**”.

Dejando esto claro, se procede a montar el dispositivo de almacenamiento en la carpeta compartida recién creada para que se monte automáticamente en el arranque de las máquinas, tal y como se presenta a continuación:

```
/dev/mapper/vg_almacenamiento-lv_apache /apache_compartido gfs2
→ _netdev 0 0
```

Este fichero se debe modificar en **ambos nodos**, añadiendo exactamente lo mismo. Se ha agregado el dispositivo de almacenamiento, el punto de montaje (carpeta compartida en los dos nodos), el sistema de archivos utilizado (gfs2), la opción para montar el dispositivo (`_netdev`: montaje a través de red) y las opciones donde se indica si se harán respaldos del sistema de archivos o revisiones de la partición en busca de errores [53].

Para no reiniciar los nodos, se ejecuta el comando “`mount -a`” para que todos los sistemas de archivos detallados en el fichero `/etc/fstab` se monten tal y como está indicado. En la ilustración 7.86 se monta los sistemas de archivos del fichero `/etc/fstab` y se verifica que se haya montado.

```
[root@nodo1-tft ~]# mount -a
[root@nodo1-tft ~]# mount | grep /dev/mapper/vg_almacenamiento-lv_apache
/dev/mapper/vg_almacenamiento-lv_apache on /apache_compartido type gfs2 (rw,relatime,
seclabel,rgrplvb,_netdev)
```

Ilustración 7.86: Montaje del dispositivo de almacenamiento

Una vez montado, se crean los directorios que tendrán los ficheros del contenido de Apache y *WordPress* (`www`) y los de configuración (`httpd`). Estos comandos solo se ejecutarán en un **nodo (Nodo1-servicio)**, ya que al tener montado el directorio compartido, la información se actualiza instantáneamente en los nodos que tienen montado dicho directorio en el dispositivo de almacenamiento.

```
[root@nodo1-tft ~]# mkdir /apache_compartido/www
[root@nodo1-tft ~]# mkdir /apache_compartido/httpd
```

Ahora, se copia todo el contenido de las carpetas `/var/www` y `/etc/httpd` en `/apache_compartido/www` y `/apache_compartido/httpd` respectivamente. Los comandos que se ejecutarán en el “**Nodo1-servicio**” son:

```
[root@nodo1-tft ~]# cp -r /var/www/* /apache_compartido/www/
[root@nodo1-tft ~]# cp -r /etc/httpd/*
→ /apache_compartido/httpd/
```

Se comprueba que se haya copiado correctamente el contenido de las carpetas y a la vez, se verifica desde el “**Nodo2-servicio**” que la sincronización funciona adecuadamente (Ver **ilustración 7.87**).

```
[root@nodo2-tft ~]# ls -l /apache_compartido/www/
total 16
drwxr-xr-x. 2 root root 3864 jun 30 10:15 cgi-bin
drwxr-xr-x. 3 root root 3864 jun 30 10:15 html
[root@nodo2-tft ~]# ls -l /apache_compartido/httpd/
total 56
drwxr-xr-x. 2 root root 3864 jun 30 10:16 conf
drwxr-xr-x. 2 root root 3864 jun 30 10:16 conf.d
drwxr-xr-x. 2 root root 3864 jun 30 10:16 conf.modules.d
lrwxrwxrwx. 1 root root 19 jun 30 10:16 logs -> ../../var/log/httpd
lrwxrwxrwx. 1 root root 29 jun 30 10:16 modules -> ../../usr/lib64/httpd/modules
lrwxrwxrwx. 1 root root 10 jun 30 10:16 run -> /run/httpd
lrwxrwxrwx. 1 root root 19 jun 30 10:16 state -> ../../var/lib/httpd
```

Ilustración 7.87: Verificación de la replicación del contenido

Una vez copiados, se agregan los contextos de SELinux necesarios e indicados previamente. Primero, se ejecuta el siguiente comando para hacer que el usuario “`system_u`” sea el usuario de contexto SELinux del directorio compartido:

```
[root@nodo1-tft ~]# cd /apache_compartido
[root@nodo1-tft ~]# runcon -u system_u /bin/bash
```

Este comando se ejecuta únicamente en el “**Nodo1-servicio**”, ya que al estar haciendo modificaciones sobre el directorio compartido, los cambios se ven reflejado en el otro nodo

automáticamente.

Después, se ejecutan los comandos presentados en la ilustración 7.88 para **cambiar** todos los **contextos de SELinux** de las carpetas **/www** y **/httpd** del directorio compartido. Además, se ejecutan dos comandos para cambiar los permisos de usuario y grupo de la carpeta “wordpress” y contenido y para cambiar el usuario de contexto SELinux en el fichero **wp-config.php** de la carpeta “wordpress”. Finalmente, con el último comando, se aplican los contextos.

```
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_config_t "/apache_compartido/httpd"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_config_t "/apache_compartido/httpd(/.*)?"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_content_t "/apache_compartido/www"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_content_t "/apache_compartido/www/html"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_script_exec_t "/apache_compartido/www/cgi-bin"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_content_t "/apache_compartido/www/html(/.*)?"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_script_exec_t "/apache_compartido/www/cgi-bin(/.*)?"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_rw_content_t "/apache_compartido/www/html/wordpress"
[root@nodo1-tft ~]# semanage fcontext -a -t httpd_sys_rw_content_t "/apache_compartido/www/html/wordpress(/.*)?"
[root@nodo1-tft ~]# chown -R apache:apache /apache_compartido/www/html/wordpress/
[root@nodo1-tft ~]# chcon -u system_u /apache_compartido/www/html/wordpress/wp-config.php
[root@nodo1-tft ~]# restorecon -R /apache_compartido/
[root@nodo1-tft ~]#
```

Ilustración 7.88: Cambio de contextos

Lo siguiente que se debe hacer es **renombrar los directorios** que han sido copiados (**/var/www** y **/etc/httpd**), en **ambos nodos**. Se modifica el nombre de los directorios “conf”, “conf.d” y “www” desde la ruta original.

```
[root@nodo1-tft var]# mv www www_backup
[root@nodo1-tft httpd]# mv conf conf_backup
[root@nodo1-tft httpd]# mv conf.d conf.d_backup
```

Una vez renombrados, **se crean los enlaces simbólicos** desde los directorios originales a los directorios compartidos correspondientes. En la ilustración 7.89 se muestran las órdenes ejecutadas para crear los enlaces simbólicos de los tres directorios. También se puede observar los enlaces simbólicos recién creados señalados con flechas amarillas y las carpetas renombradas, a modo de seguridad, señaladas con flechas rojas.

```
[root@nodo1-tft ~]# ln -s /apache_compartido/www/ /var/www
[root@nodo1-tft ~]# ln -s /apache_compartido/httpd/conf /etc/httpd/conf
[root@nodo1-tft ~]# ln -s /apache_compartido/httpd/conf.d /etc/httpd/conf.d
[root@nodo1-tft ~]# ls -l /var | grep www
lrwxrwxrwx. 1 root root 23 jun 30 11:05 www -> /apache_compartido/www/ ←
drwxr-xr-x. 1 root root 22 jun 30 09:12 www_backup ←
[root@nodo1-tft ~]# ls -l /etc/httpd | grep conf
lrwxrwxrwx. 1 root root 29 jun 30 11:05 conf -> /apache_compartido/httpd/conf ←
drwxr-xr-x. 1 root root 30 jun 29 14:23 conf_backup ←
lrwxrwxrwx. 1 root root 31 jun 30 11:06 conf.d -> /apache_compartido/httpd/conf.d ←
drwxr-xr-x. 1 root root 172 jun 29 14:23 conf.d_backup ←
drwxr-xr-x. 1 root root 298 jun 8 10:31 conf.modules.d
```

Ilustración 7.89: Enlaces simbólicos

Creando y configurando un volumen lógico para ficheros de MariaDB

En este último apartado, se crea otro volumen lógico en un nodo del clúster para usarlo como almacenamiento para ciertos ficheros de configuración de **MariaDB**. Como se ha indicado también en el apartado anterior, para poder crear el volumen lógico hay que desactivar el protocolo de cerrojo temporalmente cambiando el valor del parámetro “*use_lvmlckd*” (del fichero `/etc/lvm/lvm.conf`) a 0.

Al igual que en el apartado anterior, se crea el volumen lógico desde el “**Nodo3-servicio**” con 2 GB de tamaño y con el nombre “**lv_mariadb**”. Seguidamente, se formatea el volumen lógico y se reinician todos los servicios (iscsid, lvmlckd y dlm) del mismo modo que se realizó para el volumen lógico de Apache.

A continuación, se crean los directorios compartidos llamados de igual manera en los nodos “**Nodo3-almacenamiento**” y “**Nodo4-almacenamiento**”:

```
[root@nodo3-tft ~]# mkdir /mariadb_compartido
```

Entonces, se procede a montar el dispositivo de almacenamiento en la carpeta compartida recién creada para que se monte automáticamente en el arranque de las máquinas, tal y como se presenta a continuación:

```
/dev/mapper/vg_almacenamiento-lv_mariadb    /mariadb_compartido    gfs2
→ _netdev    0    0
```

Ahora, tras analizar los contextos de SELinux del fichero `/etc/my.cnf` y del directorio `/etc/my.cnf.d` y su contenido, se concluye que el contexto a agregar en todos es “**mysqld_etc_t**”.

Del mismo modo que en el apartado anterior, se ejecuta el comando “*mount -a*” para montar todos los sistemas de archivos que se encuentren en el fichero `/etc/fstab` sin necesidad de reiniciar los nodos. En la ilustración 7.90 se muestra el montaje y que se ha efectuado correctamente.

```
[root@nodo3-tft ~]# mount -a
[root@nodo3-tft ~]# mount | grep /dev/mapper/vg_almacenamiento-lv_mariadb
/dev/mapper/vg_almacenamiento-lv_mariadb on /mariadb_compartido type gfs2 (rw,relatime,seclabel,rgrplvb,_netdev)
```

Ilustración 7.90: Montaje del dispositivo de almacenamiento 2

Cuando esté montado, en el “**Nodo3-almacenamiento**”, se crea el directorio **my.cnf.d** y se copia el contenido del fichero **my.cnf** y de dicho directorio en el espacio compartido:

```
[root@nodo3-tft ~]# mkdir /mariadb_compartido/my.cnf.d
```

Y se copia el fichero **my.cnf** y todo el contenido de la carpeta `/etc/my.cnf.d` excepto el fichero **galera.cnf**. Los comandos se ejecutarán solamente en el “**Nodo3-almacenamiento**”, ya que la información dentro de este espacio compartido, al estar montada, se sincroniza:


```
[root@nodo3-tft ~]# cp /etc/my.cnf /mariadb_compartido
[root@nodo3-tft ~]# rsync -a --exclude 'gale*' /etc/my.cnf.d/
↪ /mariadb_compartido/my.cnf.d/
```

El segundo comando establece que se copie todo el contenido del directorio `/etc/my.cnf.d` menos aquellos ficheros que comiencen por “gale”. Como en este caso solo existe un fichero que empiece así, no lo copia, ya que este fichero debe ser único en los dos nodos de almacenamiento.

Una vez se hayan copiado, se agregan los contextos de SELinux indicados previamente desde el “**Nodo3-almacenamiento**”. En primer lugar, se ejecuta el comando para hacer que el usuario “system_u” sea el usuario de contexto de SELinux del directorio compartido:

```
[root@nodo3-tft ~]# cd /mariadb_compartido
[root@nodo3-tft ~]# runcon -u system_u /bin/bash
```

Seguidamente, en **uno** de los dos nodos, se ejecutan los comandos plasmados en la ilustración 7.91 para cambiar los contextos de SELinux del fichero `/my.cnf` y la carpeta `/my.cnf.d` del directorio compartido de mariadb y para cambiar el usuario de contexto SELinux. Y para concluir, se ejecuta el último comando para aplicar los contextos.

```
[root@nodo3-tft ~]# semanage fcontext -a -t mysqld_etc_t "/mariadb_compartido/my.cnf"
[root@nodo3-tft ~]# semanage fcontext -a -t mysqld_etc_t "/mariadb_compartido/my.cnf.d"
[root@nodo3-tft ~]# semanage fcontext -a -t mysqld_etc_t "/mariadb_compartido/my.cnf.d(/.*)?"
"
[root@nodo3-tft ~]# chcon -u system_u /mariadb_compartido/my.cnf
[root@nodo3-tft ~]# chcon -u system_u /mariadb_compartido/my.cnf.d/
[root@nodo3-tft ~]# chcon -u system_u /mariadb_compartido/my.cnf.d/*
[root@nodo3-tft ~]# restorecon -R /mariadb_compartido/
```

Ilustración 7.91: Aplicando contextos

El siguiente paso es renombrar el fichero `/etc/my.cnf` y los ficheros del directorio `/etc/my.cnf.d` que han sido copiados al espacio compartido. Esto se debe realizar en los dos nodos (**Nodo3-almacenamiento** y **Nodo4-almacenamiento**) de la siguiente manera:

```
[root@nodo3-tft ~]# mv /etc/my.cnf /etc/my.cnf_backup
[root@nodo3-tft ~]# mv /etc/my.cnf.d/auth_gssapi.cnf
↪ /etc/my.cnf.d/auth_gssapi.cnf_backup
[root@nodo3-tft ~]# mv /etc/my.cnf.d/client.cnf
↪ /etc/my.cnf.d/client.cnf_backup
[root@nodo3-tft ~]# mv /etc/my.cnf.d/cracklib_password_check.cnf
↪ /etc/my.cnf.d/cracklib_password_check.cnf_backup
[root@nodo3-tft ~]# mv /etc/my.cnf.d/enable_encryption.preset
↪ /etc/my.cnf.d/enable_encryption.preset_backup
[root@nodo3-tft ~]# mv /etc/my.cnf.d/mariadb-server.cnf
↪ /etc/my.cnf.d/mariadb-server.cnf_backup
[root@nodo3-tft ~]# mv /etc/my.cnf.d/spider.cnf
↪ /etc/my.cnf.d/spider.cnf_backup
```

Cuando se hayan renombrado, se crean los enlaces simbólicos de la misma manera que se

realizó para los ficheros de Apache (desde los ficheros originales a los ficheros compartidos correspondientes) y se comprueban que se hayan creado correctamente (**Ver ilustración 7.92**).

```
[root@nodo4-tft ~]# ls -l /etc/my.cnf
lrwxrwxrwx. 1 root root 26 jun 30 13:48 /etc/my.cnf -> /mariadb_compartido/my.cnf
[root@nodo4-tft ~]# ls -l /etc/my.cnf.d/
total 52
lrwxrwxrwx. 1 root root 44 jun 30 13:48 auth_gssapi.cnf -> /mariadb_compartido/my.cnf.d/auth_gssapi.cnf
-rw-r--r--. 1 root root 42 may 23 22:21 auth_gssapi.cnf_backup
lrwxrwxrwx. 1 root root 39 jun 30 13:48 client.cnf -> /mariadb_compartido/my.cnf.d/client.cnf
-rw-r--r--. 1 root root 295 jul 22 2021 client.cnf_backup
lrwxrwxrwx. 1 root root 56 jun 30 13:48 cracklib_password_check.cnf -> /mariadb_compartido/my.cnf.d/cracklib_password_check.cnf
-rw-r--r--. 1 root root 54 may 23 22:21 cracklib_password_check.cnf_backup
lrwxrwxrwx. 1 root root 53 jun 30 13:48 enable_encryption.preset -> /mariadb_compartido/my.cnf.d/enable_encryption.preset
-rw-r--r--. 1 root root 763 may 18 08:55 enable_encryption.preset_backup
-rw-r--r--. 1 root root 3552 jun 15 10:38 galera.cnf
lrwxrwxrwx. 1 root root 47 jun 30 13:48 mariadb-server.cnf -> /mariadb_compartido/my.cnf.d/mariadb-server.cnf
-rw-r--r--. 1 root root 1458 may 23 21:46 mariadb-server.cnf_backup
lrwxrwxrwx. 1 root root 39 jun 30 13:49 spider.cnf -> /mariadb_compartido/my.cnf.d/spider.cnf
-rw-r--r--. 1 root root 120 may 18 08:55 spider.cnf_backup
```

Ilustración 7.92: Verificación de los enlaces simbólicos

Para concluir, en la siguiente sección se verifica que los cambios realizados no hayan interferido en el servicio de *WordPress*.

7.4. Evaluación

Tras finalizar el desarrollo de la infraestructura de clúster que ofrece el servicio *WordPress* basado en Apache en alta disponibilidad y con balanceo de carga en la distribución de Fedora Workstation 35, se realizarán distintas pruebas para verificar el funcionamiento final.

El procedimiento posterior para la evaluación consta de tres fases: la evaluación de escritura en la base de datos, la evaluación del servicio en alta disponibilidad y con balanceo de carga y por último, la evaluación del balanceo de carga en la base de datos.

Evaluación de escritura en la base de datos

En este epígrafe, se evaluará que se puede efectuar cualquier operación en la base de datos, que esta permanecerá guardada y sincronizada en todos los nodos de manera instantánea. Para probar esto, se realizan dos escrituras: se crea una nueva entrada en el *blog* que ofrece *WordPress* como sitio web y luego, se modificará el tema.

Para poder crear una nueva entrada, desde el escritorio o panel de administración de *WordPress*, se dirige al menú izquierdo, se elige la opción “Entradas” → “Añadir nueva”. Luego, se abre una página similar a la que se muestra en la ilustración 7.93 y se añade cualquier tipo de información para verificar que si se recarga la página o si se sale del sitio y se vuele a entrar, esta información sigue apareciendo.



Ilustración 7.93: Primera prueba de escritura en la base de datos

Tras salir del sitio, limpiar la caché del navegador y volver a acceder al *blog*, la nueva entrada añadida sigue presentándose en el servicio *WordPress*, lo cual, quiere decir que la base de datos, se ha escrito, guardado y replicado al resto de nodos de almacenamiento correctamente.

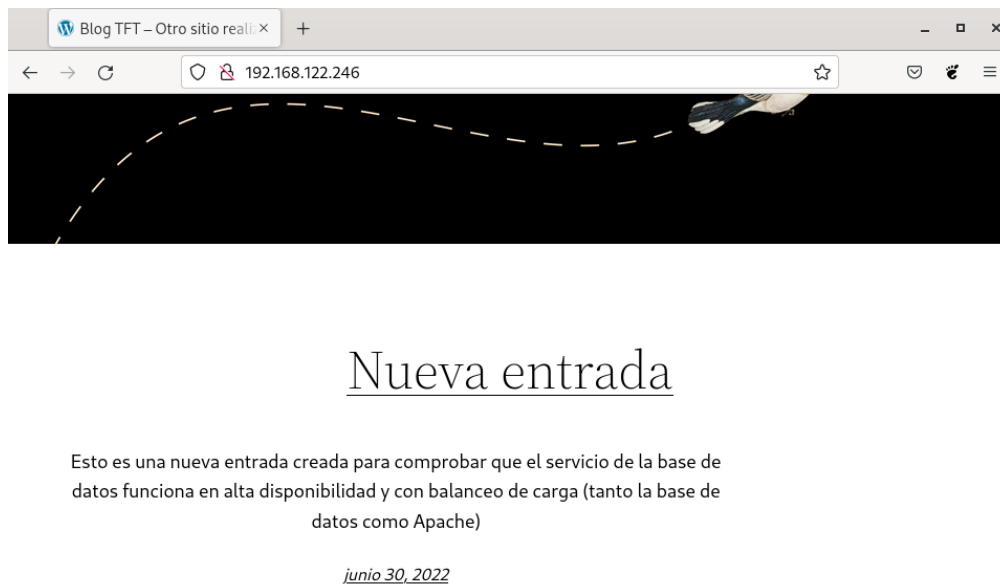


Ilustración 7.94: Verificación de la primera prueba de escritura en la base de datos

Ahora, se procede a modificar el tema de *WordPress* predeterminado para volver a comprobar que la base de datos se está guardando y replicando correctamente. Para editar el tema, desde el escritorio o panel de administración, se dirige al menú izquierdo, se elige la opción “Apariencia” → “Editor de temas”. Después, se inicia una página que contiene un panel a la izquierda con diferentes directorios y ficheros, estos son los ficheros del propio tema. Entonces, se elige el fichero **parts/header.html** y se añade lo que se muestra resaltado en la ilustración 7.95.

```

1 <!-- wp:group {"layout":{"inherit":true}} -->
2 <div class="wp-block-group"><!-- wp:group {"align":"wide","style":
3 1.25rem)"}}, {"layout":{"type":"flex","justifyContent":"space-between"}
4 <div class="wp-block-group alignwide" style="padding-top:var(--wp-
5 {"type":"flex"}) -->
6 <!-- wp:site-logo {"width":64} /-->
7 <!-- /wp:group -->
8
9 <!-- wp:navigation {"layout":{"type":"flex","setCascadingProperties":true}
10 <!-- wp:page-list {"isNavigationChild":true,"showSubMenuIcon":true}
11 <!-- /wp:navigation --></div>
12 <!-- /wp:group --></div>
13 <!-- /wp:group -->
14 <h4 style="text-align:center;">Prueba de escritura</h4>

```

Ilustración 7.95: Segunda prueba de escritura en la base de datos

Para verificar de nuevo que la base de datos se está replicando correctamente, se reinicia varias veces la página. En la ilustración 7.96 se puede corroborar que se ha modificado el *header* del tema con la frase “Prueba de escritura” y que tras reiniciar varias veces la página, sigue apareciendo.



Ilustración 7.96: Verificación de la segunda prueba de escritura en la base de datos

Con esta prueba se concluye que la base de datos replica adecuadamente el contenido y de manera totalmente síncrona, sin tener ningún tipo de problema el estar balanceando el servicio HTTP y la propia base de datos.

Evaluación del servicio en alta disponibilidad y con balanceo de carga

En esta evaluación, se evaluará la alta disponibilidad y el balanceo de carga del servicio que ofrece el clúster. Para esto, hay que verificar qué nodo está dando el servicio en cada momento, añadiendo la porción de código que se muestra a continuación en el fichero

`/var/www/html/wordpress/index.php`. Este código permite mostrar una ristra de caracteres con el *hostname* del nodo que ofrece el servicio de Apache para poder identificar qué nodo es el que está dando el servicio en este momento [54].

```
$salida = shell_exec(hostname);  
echo "<center><h4>Soy el $salida</h4>";
```

Para corroborar que esta frase va cambiando, y que por ende, se efectúa correctamente el balanceo de carga, se elimina la caché del navegador y se accede mediante el nombre de dominio o dirección IP del nodo balanceador (quien balancea la carga del nodo que ofrece el servicio) al *blog* de *WordPress* y se recarga la página para observar cómo va alternándose el *hostname* de la frase agregada. En las ilustraciones 7.97 y 7.98 se puede apreciar el cambio realizado en el fichero `index.php`.



Ilustración 7.97: Verificación del balanceo de carga del servicio



Ilustración 7.98: Verificación del balanceo de carga del servicio después de recargar

Evaluación del balanceo de carga en la base de datos

Para comprobar el balanceo de carga en la base de datos, se ejecuta varias veces un comando de MySQL desde cualquiera de los cuatro nodos del clúster (Nodo1-servicio, Nodo2-servicio, Nodo3-almacenamiento y Nodo4-almacenamiento) para ir viendo qué nodo de almacenamiento (Nodo3-almacenamiento y Nodo4-almacenamiento) es el que está ofreciendo la base de datos en este preciso momento.

```
[root@nodo1-tft ~]# mysql -u wordpress -p -h 192.168.122.246 -e "show variables like 'wsrep_node_name'"
Enter password:
+-----+-----+
| Variable_name | Value |
+-----+-----+
| wsrep_node_name | nodo3-tft |
+-----+-----+
[root@nodo1-tft ~]# mysql -u wordpress -p -h 192.168.122.246 -e "show variables like 'wsrep_node_name'"
Enter password:
+-----+-----+
| Variable_name | Value |
+-----+-----+
| wsrep_node_name | nodo4-tft |
+-----+-----+
[root@nodo1-tft ~]# mysql -u wordpress -p -h 192.168.122.246 -e "show variables like 'wsrep_node_name'"
Enter password:
+-----+-----+
| Variable_name | Value |
+-----+-----+
| wsrep_node_name | nodo3-tft |
+-----+-----+
```

Ilustración 7.99: Verificación del balanceo de carga de la base de datos

En la ilustración 7.99 se puede apreciar el comando ejecutado que se encarga de mostrar el nombre del nodo que está ofreciendo la base de datos en el momento en el que se ejecuta el comando. Las opciones indicadas en el comando son para conectarse a MySQL con el usuario “wordpress” especificando la opción **-u**, para señalar la contraseña con la que se desea conectar mediante la opción **-p** y para conectarse al servidor MySQL del *host* ofrecido con la opción **-h** [55].

Se puede determinar que el balanceo de carga de la base de datos es adecuado, debido a que se ha ejecutado tres veces el comando y el nombre del nodo que ofrece la base de datos en el momento de la ejecución del comando, va variando entre el **nodo3-tft** y el **nodo4-tft**.

Capítulo 8

Conclusiones y trabajo futuro

El desarrollo de este proyecto conduce a la culminación de los estudios del Grado en Ingeniería Informática. El proceso que supuso diseñar y realizar una infraestructura de clúster, es algo que me ayudará a la hora de enfrentarme al mundo laboral, dado que construir este tipo de infraestructuras no es una tarea fácil.

La elaboración de este trabajo requiere tener en cuenta muchas consideraciones para que se pueda lograr que funcione todo el conjunto de servicios y configuraciones necesarias para formar el clúster correctamente. Además, conlleva mucho esfuerzo y tiempo, sobre todo aquel que se invierte en la solución de innumerables errores que se van ocasionando a lo largo del proyecto.

El enfrentarse a este tipo de proyectos ha sido un desafío, tanto en el ámbito profesional como en el personal, ya que abordar el trabajo con tecnologías que no se habían empleado previamente y en una distribución que se desconocía, a pesar de haber trabajado previamente con sistemas Linux, ha contribuido a mejorar mi formación como administradora de sistemas, ampliando mi conocimiento sobre las tecnologías de virtualización, de iSCSI, de sistemas de archivos, de alta disponibilidad, de balanceo de carga e incluso de los sistemas Linux.

Gracias al esfuerzo dedicado y a la gran cantidad de horas invertidas, se ha logrado alcanzar los objetivos acordados en el tiempo establecido. El resultado que se ha obtenido es mucho más de lo que se esperaba en un principio, puesto que, además de desarrollar con éxito la totalidad del trabajo, se pudo llevar a cabo un extra: crear un servidor iSCSI que proporcione un área de almacenamiento compartido y distribuido (fuera del clúster) a los nodos del clúster. Asimismo, también se consiguió aplicar la alta disponibilidad y el balanceo de carga no solo al servicio web, sino a la base de datos de dicho servicio.

Por todo esto, finalizo el trabajo, satisfecha y orgullosa del resultado obtenido. En este trabajo, se ha analizado todos los requerimientos necesarios para diseñar y desarrollar un clúster formado por cinco nodos que ofrece el CMS de *WordPress* (basado en Apache y con la base de datos en MariaDB) en alta disponibilidad y con balanceo de carga y que, además, poseen un espacio almacenamiento compartido y distribuido basado en la tecnología iSCSI. Para lograr estos resultados, se ha analizado todos los requerimientos a los que debe ajustarse

el sistema anfitrión, los servicios a ofrecer en el clúster y la distribución de Fedora. Luego, se ha acondicionado el sistema para el desarrollo del clúster y se ha diseñado la topología del clúster y la infraestructura de redes que posee. Y para finalizar se ha creado, instalado y configurado cada uno de los elementos necesarios que forman el clúster: las máquinas virtuales, las redes, los servicios, los discos, la creación del clúster, el almacenamiento compartido, etc.

Como propuesta de mejora en un trabajo futuro, se invita a desarrollar un sistema que permita respaldar al nodo que ofrece el balanceo de carga en el clúster, pues si este nodo falla por algún tipo de problema, el servicio dejaría de ofrecerse. Esta mejora se tuvo en cuenta en el desarrollo del proyecto, pero debido a que la única posibilidad que se tenía con las condiciones que se disponían, era crear un único nodo equilibrador de cargas, no se pudo conseguir. De la misma manera ocurre con el servidor iSCSI montado, si este tuviera alguna caída, el servicio web dejaría de ofrecerse porque todos los ficheros del contenido y de configuración de Apache, así como algunos de la base de datos, se encuentran en él. También, se podría obtener un certificado SSL para configurar Apache, capacitándolo a utilizar el protocolo HTTPS para que las conexiones del servicio web sean más seguras.

Bibliografía

- [1] Bowen, R.: *CentOS Project shifts focus to CentOS Stream*. Disponible en: <https://blog.centos.org/2020/12/future-is-centos-stream/>, diciembre 2020. [Accedido en junio de 2022].
- [2] García, R. y Quesada A.: *Computación en Clúster*. Disponible en: https://aep22.ulpgc.es/pluginfile.php/786542/mod_resource/content/3/COMPUTACION%20EN%20%20CLUSTER.pdf, 2020. [Accedido en junio de 2022].
- [3] ULPGC: *Objetivos del Título*. Disponible en: https://www2.ulpgc.es/archivos/plan_estudios/4008_40/ObjetivosyCompetenciasdelGII.pdf, 2022. [Accedido en julio de 2022].
- [4] Wikipedia®: *Apache License*. Disponible en: https://es.wikipedia.org/wiki/Apache_License, junio 2022. [Accedido en julio de 2022].
- [5] Wikipedia®: *GNU General Public License*. Disponible en: https://es.wikipedia.org/wiki/GNU_General_Public_License, junio 2022. [Accedido en julio de 2022].
- [6] FSF®: *GNU General Public License*. Disponible en: <https://www.gnu.org/licenses/licenses.html#FDL>, abril 2022. [Accedido en julio de 2022].
- [7] Wikipedia®: *GNU Lesser General Public License*. Disponible en: https://es.wikipedia.org/wiki/GNU_Lesser_General_Public_License, marzo 2022. [Accedido en julio de 2022].
- [8] Wikipedia®: *Mozilla Public License*. Disponible en: https://es.wikipedia.org/wiki/Mozilla_Public_License, diciembre 2020. [Accedido en julio de 2022].
- [9] Wikipedia®: *Licencia PHP*. Disponible en: https://es.wikipedia.org/wiki/Licencia_PHP, septiembre 2019. [Accedido en julio de 2022].
- [10] Wikipedia®: *Servidor HTTP Apache*. Disponible en: https://es.wikipedia.org/wiki/Servidor_HTTP_Apache, marzo 2022. [Accedido en julio de 2022].
- [11] Wikipedia®: *Fedora (sistema operativo)*. Disponible en: [https://es.wikipedia.org/wiki/Fedora_\(sistema_operativo\)](https://es.wikipedia.org/wiki/Fedora_(sistema_operativo)), julio 2022. [Accedido en julio de 2022].
- [12] Red Hat, Inc.®: *Fedora*. Disponible en: <https://getfedora.org/es/>, julio 2022. [Accedido en julio de 2022].

- [13] Fedora_Project: *Using Firewalld. What is firewalld?* Disponible en: https://docs.fedoraproject.org/en-US/quick-docs/firewalld/#_what_is_firewalld, julio 2022. [Accedido en julio de 2022].
- [14] Red Hat, Inc.[®]: *Gestión de dispositivos de almacenamiento. Solución en clúster del sistema de archivos GFS2.* Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/managing_storage_devices/gfs2-file-system-clustered-solution-overview-of-available-storage-options, 2022. [Accedido en junio de 2022].
- [15] Levine, S. y Wadeley S.: *Load Balancer Administration. Haproxy.* Disponible en: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/load_balancer_administration/s1-lvs-haproxy-vsa, 2022. [Accedido en junio de 2022].
- [16] Red Hat, Inc.[®]: *Configuring and managing high availability clusters. High Availability Add-On overview.* Disponible en: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_high_availability_clusters/assembly-overview-of-high-availability-configuring-and-managing-high-availability-clusters, julio 2022. [Accedido en julio de 2022].
- [17] Wikipedia[®]: *Kernel-based Virtual Machine.* Disponible en: https://es.wikipedia.org/wiki/Kernel-based_Virtual_Machine, febrero 2021. [Accedido en julio de 2022].
- [18] ArchLinux, Wiki: *LVM.* Disponible en: [https://wiki.archlinux.org/title/LVM_\(Español\)](https://wiki.archlinux.org/title/LVM_(Español)), junio 2022. [Accedido en julio de 2022].
- [19] Wikipedia[®]: *MariaDB.* Disponible en: <https://es.wikipedia.org/wiki/MariaDB>, mayo 2022. [Accedido en julio de 2022].
- [20] Red Hat, Inc.[®]: *Despliegue de diferentes tipos de servidores. Replicar MariaDB con Galera.* Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/deploying_different_types_of_servers/replicating-mariadb-with-galera-using-mariadb, 2022. [Accedido en julio de 2022].
- [21] Wikipedia[®]: *Firefox Mozilla.* Disponible en: https://es.wikipedia.org/wiki/Mozilla_Firefox, julio 2022. [Accedido en julio de 2022].
- [22] Wikipedia[®]: *NetworkManager.* Disponible en: <https://es.wikipedia.org/wiki/NetworkManager>, noviembre 2021. [Accedido en julio de 2022].
- [23] The_PHP_Group: *¿Qué es PHP?* Disponible en: <https://www.php.net/manual/es/intro-what-is.php>, 2022. [Accedido en julio de 2022].
- [24] Fedora: *Getting started with virtualization.* Disponible en: <https://docs.fedoraproject.org/es/quick-docs/getting-started-with-virtualization/>, 2022. [Accedido en junio de 2022].
- [25] Red Hat, Inc.[®]: *Configuring and managing virtualization. Introducing virtualization in RHEL.* Disponible en: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_virtualization/introduc

- ing-virtualization-in-rhel_configuring-and-managing-virtualization, 2022. [Accedido en junio de 2022].
- [26] Red Hat, Inc.[®]: *Configuración y gestión de la virtualización. Compatibilidad y limitaciones de las funciones en la virtualización de RHEL 8*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_virtualization/feature-support-and-limitations-in-rhel8-virtualization_configuring-and-managing-virtualization, 2022. [Accedido en junio de 2022].
- [27] Red Hat, Inc.[®]: *Configuración y gestión de la virtualización. Gestión del almacenamiento de las máquinas virtuales*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_virtualization/managing-storage-for-virtual-machines_configuring-and-managing-virtualization, 2022. [Accedido en junio de 2022].
- [28] WordPress.org: *Requisitos*. Disponible en: <https://es.wordpress.org/about/requirements/>, 2022. [Accedido en julio de 2022].
- [29] Vivaubuntu: *Requisitos MariaDB*. Disponible en: https://vivaubuntu.com/mariadb-en-ubuntu-20-04-instalar/#REQUISITOS_MARIADB, 2022. [Accedido en julio de 2022].
- [30] HAProxy: *Hardware Recommendations*. Disponible en: <https://www.haproxy.com/documentation/hapee/latest/getting-started/hardware/>, 2022. [Accedido en julio de 2022].
- [31] Galera Cluster: *Installing Galera Cluster*. Disponible en: <https://galeracluster.com/library/training/tutorials/galera-installation.html>, 2022. [Accedido en julio de 2022].
- [32] Red Hat, Inc.[®]: *Configuración y gestión de redes. Introducción a NetworkManager*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/getting-started-with-networkmanager_configuring-and-managing-networking, 2022. [Accedido en junio de 2022].
- [33] Red Hat, Inc.[®]: *Configuración y gestión de redes. Cómo gestiona NetworkManager varias pasarelas por defecto*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/con-how-networkmanager-manages-multiple-default-gateways_managing-the-default-gateway-setting, 2022. [Accedido en junio de 2022].
- [34] Red Hat, Inc.[®]: *Configuración y gestión de la virtualización. Configuración de las conexiones de red de las máquinas virtuales*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_virtualization/configuring-virtual-machine-network-connections_configuring-and-managing-virtualization, 2022. [Accedido en junio de 2022].

- [35] Bokoč, P., Čapek T., Ančincová B., Ruseva Y. y Exelbierd B.: *Guía de Instalación. Apéndice B. Discos iSCSI*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/7/html/installation_guide/appe-iscsi-disks, 2022. [Accedido en junio de 2022].
- [36] Red Hat, Inc.[®]: *Configuración y gestión de redes. Introducción a nmcli*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/getting-started-with-nmcli-configuring-and-managing-networking, 2022. [Accedido en junio de 2022].
- [37] Fedora.Project: *System Administrator's Guide. Servers. Web Servers*. Disponible en: <https://docs.fedoraproject.org/en-US/fedora/f35>, julio 2022. [Accedido en junio de 2022].
- [38] Red Hat, Inc.[®]: *Configuración y gestión de clusters de alta disponibilidad. Cómo empezar con Pacemaker*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_high_availability_clusters/assembly-getting-started-with-pacemaker-configuring-and-managing-high-availability-clusters, 2022. [Accedido en junio de 2022].
- [39] García, R. y Quesada A.: *Práctica 7.2: Diseño y despliegue de un clúster básico*. Disponible en: https://aep22.ulpgc.es/pluginfile.php/786759/mod_resource/content/15/Ficha_P7.2.Cluster_Basico.pdf, 2021. [Accedido en junio de 2022].
- [40] Red Hat, Inc.[®]: *Configuración y gestión de clusters de alta disponibilidad. Configuración del cercado en un clúster de alta disponibilidad de Red Hat*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html-single/configuring_and_managing_high_availability_clusters/index#assembly_configuring-fencing-configuring-and-managing-high-availability-clusters, 2022. [Accedido en junio de 2022].
- [41] Red Hat, Inc.[®]: *Configuring and managing high availability clusters. Determining which nodes a resource can run on*. Disponible en: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_high_availability_clusters/assembly-determining-which-node-a-resource-runs-on-configuring-and-managing-high-availability-clusters, 2022. [Accedido en junio de 2022].
- [42] Red Hat, Inc.[®]: *Despliegue de diferentes tipos de servidores. Replicar MariaDB con Galera*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/deploying_different_types_of_servers/replicating-mariadb-with-galera-using-mariadb, 2022. [Accedido en junio de 2022].
- [43] Server_World[®]: *MariaDB 10.5: MariaDB Galera Cluster*. Disponible en: https://www.server-world.info/en/note?os=Fedora_34&p=mariadb&f=4, 2022. [Accedido en junio de 2022].

-
- [44] Galera_Cluster: *MySQL wsrep Options*. Disponible en: <https://galeracluster.com/library/documentation/mysql-wsrep-options.html>, 2022. [Accedido en junio de 2022].
- [45] RomSolutions: *MySQL Galera*. Disponible en: https://notas.romsolutions.es/index.php/MySQL_Galera, septiembre 2020. [Accedido en junio de 2022].
- [46] MySQL_5.0: *Puesta en marcha y comprobación después de la instalación. Hacer seguras las cuentas iniciales de MySQL*. Disponible en: <https://documentation.help/MySQL-5.0-es/ch03s03.html>, 2022. [Accedido en junio de 2022].
- [47] Red Hat, Inc.[®]: *Understanding Red Hat Openstack Platform High Availability. Using HAProxy*. Disponible en: https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/14/html/understanding_red_hat_openstack_platform_high_availability/haproxy, 2022. [Accedido en junio de 2022].
- [48] MySQL_5.0: *Sentencias de administración de base de datos. Sentencias de administración de base de datos*. Disponible en: <https://documentation.help/MySQL-5.0-es/ch03s03.html>, 2022. [Accedido en junio de 2022].
- [49] Red Hat, Inc.[®]: *Gestión del Administrador de volumen lógico. Ejemplos de configuración de LVM*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/6/html/logical_volume_manager_administration/lvm_examples, 2022. [Accedido en junio de 2022].
- [50] Red Hat, Inc.[®]: *Configuring and managing high availability clusters. GFS2 file systems in a cluster*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_high_availability_clusters/assembly_configuring-gfs2-in-a-cluster-configuring-and-managing-high-availability-clusters, 2022. [Accedido en junio de 2022].
- [51] ElInstalador: *Cómo instalar WordPress en Fedora 35*. Disponible en: <https://comoinstalar.me/como-instalar-wordpress-en-fedora-32/>, junio 2021. [Accedido en junio de 2022].
- [52] García, R. y Quesada A.: *Práctica 8.2: Instalación de un servicio de almacenamiento de alta disponibilidad basado en la tecnología iSCSI y el sistema de archivos GFS2*. [Accedido en junio de 2022].
- [53] García, R. y Quesada A.: *Gestión de sistemas de archivos. Montaje persistente de sistemas de archivos*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/, 2022. [Accedido en junio de 2022].
- [54] PHP: *shell exec*. Disponible en: <https://www.php.net/manual/es/function.shell-exec.php>, 2022. [Accedido en junio de 2022].
- [55] Oracle[®]: *MySQL - Linux man page*. Disponible en: <https://linux.die.net/man/1/mysql>, 2012. [Accedido en junio de 2022].

- [56] Fedora_Project: *Getting started with virtualization (libvirt). Enabling hardware virtualization support*. Disponible en: <https://docs.fedoraproject.org/en-US/quick-docs/getting-started-with-virtualization/>, julio 2022. [Accedido en julio de 2022].
- [57] Wikipedia[®]: *WordPress*. Disponible en: <https://es.wikipedia.org/wiki/WordPress>, junio 2022. [Accedido en julio de 2022].
- [58] The_GNOME_Project: *Getting GNOME*. Disponible en: <http://www-cs-faculty.stanford.edu/~uno/abcde.html>, 2022. [Accedido en mayo de 2022].
- [59] Red Hat, Inc.[®]: *Despliegue de diferentes tipos de servidores. Configuración del servidor NFS*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/deploying_different_types_of_servers/nfs-server-configuration-exporting-nfs-shares#doc-wrapper, 2022. [Accedido en mayo de 2022].
- [60] Red Hat, Inc.[®]: *Configuración y gestión de redes. Configurar un puente de red*. Disponible en: https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/configuring-a-network-bridge-configuring-and-managing-networking, 2022. [Accedido en mayo de 2022].

Anexo

ANEXO I: Comprobación de los requerimientos del sistema anfitrión

En primer lugar, se verifica que se cumplen los requisitos mínimos para instalar la plataforma de virtualización, corroborando el espacio libre en disco y la memoria RAM de la que dispone el sistema anfitrión con los comandos que se aprecian a continuación:

- Comprobar el espacio libre en disco:

```
[root@pc1-tft ~]# free -h
```

- Comprobar el tamaño de la RAM:

```
[root@pc1-tft ~]# dmidecode | grep Size
```

En la ilustración 1 se muestra el resultado obtenido al ejecutar el primer comando para verificar el espacio libre en disco y, en la ilustración 2 se observa el resultado devuelto tras ejecutar el segundo comando y comprobar el tamaño de RAM disponible.

```
[aridpd@pc1-tft ~]$ free -h
              total        used         free       shared  buff/cache   available
Mem:           15Gi        2,5Gi         10Gi         67Mi        2,8Gi        12Gi
Swap:          8,0Gi           0B         8,0Gi
```

Ilustración 1: Verificación del espacio libre en disco del sistema anfitrión

```
[root@pc1-tft ~]# dmidecode | grep Size
Installed Size: 32 kB
Maximum Size: 32 kB
Installed Size: 32 kB
Maximum Size: 32 kB
Installed Size: 256 kB
Maximum Size: 256 kB
Installed Size: 8 MB
Maximum Size: 8 MB
Runtime Size: 64 kB
ROM Size: 4 MB
Range Size: 16 GB
Size: 4 GB
Size: 4 GB
Size: 4 GB
Size: 4 GB
```

Ilustración 2: Verificación de la RAM del sistema anfitrión

Es conveniente recordar que el espacio libre mínimo en disco que debe tener el sistema anfitrión para instalar KVM es de 6 GB, por lo que, si se analiza la ilustración 1, se puede

concluir que el espacio libre en disco cumple con este requerimiento, ya que es de 10 GB. Con respecto a la memoria RAM mínima, si se observa la ilustración 2, se puede verificar que el equipo tiene 4 módulos de 4 GB cada uno, por lo tanto, tiene un total de 16 GB y por esto, cumple también con el requerimiento mínimo de 2 GB.

De la misma manera, es interesante revisar la arquitectura y toda la información de la CPU del sistema anfitrión. Esto se puede realizar con el siguiente comando:

```
[root@pci-tft ~]# lscpu
```

Como se aprecia en la ilustración 3, el procesador es un Intel de 64 bits, por lo tanto, cumple también con uno de los requerimientos necesarios a cumplir del hipervisor KVM para esta distribución.

```
[root@pci-tft ~]# lscpu
Arquitectura:                x86_64
modo(s) de operación de las CPUs: 32-bit, 64-bit
Tamaños de las direcciones: 36 bits physical, 48 bits virtual
Orden de los bytes:         Little Endian
CPU(s):                      8
Lista de la(s) CPU(s) en línea: 0-7
ID de fabricante:           GenuineIntel
ID de fabricante de BIOS:   Intel(R) Corporation
Nombre del modelo:          Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz
Nombre del modelo de BIOS:  Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz
Familia de CPU:              6
Modelo:                      42
Hilo(s) de procesamiento por núcleo: 2
Núcleo(s) por «socket»:    4
«Socket(s)»:                 1
Revisión:                    7
CPU MHz máx.:                3800,0000
CPU MHz mín.:                1600,0000
BogoMIPS:                    6784.54
Indicadores:                 fpu vme de pse tsc msr pae mce cx8 apic sep mt
                             rr pge mca cmov pat pse36 clflush dts acpi mmx
                             fxsr sse sse2 ht tm pbe syscall nx rdtscp lm
                             constant_tsc arch_perfmon pebs bts rep_good no
                             pl xtopology nonstop_tsc cpuid aperfmperf pni
                             pclmulqdq dtes64 monitor ds_cpl vmx smx est tm
                             2 ssse3 cx16 xtpr pdcm pcid sse4_1 sse4_2 x2ap
                             ic popcnt tsc_deadline_timer aes xsave avx lah
                             f_lm epb pti ssbd ibrs ibpb stibp tpr_shadow v
                             nmi flexpriority ept vpid xsaveopt dtherm ida
                             arat pln pts md_clear flush_lld
```

Ilustración 3: Verificación de la información de la CPU del sistema anfitrión

Para concluir con el estudio de los requerimientos del sistema, si se tiene en cuenta que, como se mencionó en el análisis previo, KVM requiere de una CPU con extensiones de virtualización, hay que verificar que esta tenga soporte de las extensiones Intel VT o AMD-V requeridas, aunque normalmente alguna de ellas se suelen encontrar en la mayoría de las CPU de consumo. Para corroborar que esto se cumple, se ejecuta el siguiente comando:

```
[root@pci-tft ~]# egrep '^flags.*(vmx|svm)' /proc/cpuinfo
```

Si este comando no imprime nada como resultado, el sistema no es compatible con las extensiones de virtualización indicadas.

En la ilustración 4, se muestra la ejecución del comando en el sistema anfitrión, donde ha devuelto como resultado diferentes tipos de *flags* resaltados en color rojo. El flag “vmx”

que aparece resaltado demuestra que el equipo sí es compatible con este tipo de extensiones.

```
[root@pc1-tft ~]# egrep '^flags.*(vmx|svm)' /proc/cpuinfo
flags      : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ht tm pbe syscall nx rdtscp lm cons
tant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmp
erf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm pci
d sse4_1 sse4_2 x2apic popcnt tsc_deadline_timer aes xsave avx lahf_lm epb pti s
sbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid xsaveopt dtherm ida ar
at pln pts md_clear flush_lld
flags      : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ht tm pbe syscall nx rdtscp lm cons
tant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmp
erf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm pci
d sse4_1 sse4_2 x2apic popcnt tsc_deadline_timer aes xsave avx lahf_lm epb pti s
sbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid xsaveopt dtherm ida ar
at pln pts md_clear flush_lld
flags      : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ht tm pbe syscall nx rdtscp lm cons
tant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmp
erf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm pci
d sse4_1 sse4_2 x2apic popcnt tsc_deadline_timer aes xsave avx lahf_lm epb pti s
sbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid xsaveopt dtherm ida ar
at pln pts md_clear flush_lld
flags      : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ht tm pbe syscall nx rdtscp lm cons
tant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmp
erf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm pci
d sse4_1 sse4_2 x2apic popcnt tsc_deadline_timer aes xsave avx lahf_lm epb pti s
sbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid xsaveopt dtherm ida ar
at pln pts md_clear flush_lld
```

Ilustración 4: Verificación de las extensiones de la CPU del sistema anfitrión

ANEXO II: Instalación de la herramienta *Gnome Tweaks*

GNOME es un entorno de escritorio utilizado en la mayoría de las distribuciones de GNU/Linux. En Fedora Workstation 35, que es la distribución con la que se está trabajando, ya viene instalado **GNOME 41** [58]. Mientras que, *Gnome Shell*, es la interfaz de usuario básica de este entorno que viene con un modelo de escritorio diferente al utilizado en versiones anteriores. Una característica a distinguir de las anteriores es que las ventanas de las aplicaciones, programas o herramientas no poseen en primera instancia los botones para maximizar y minimizar.

Es por esto que, la herramienta *Gnome Tweaks* se ha instalado, ya que permite controlar opciones avanzadas de *Gnome Shell*, como puede ser la personalización del escritorio, de las fuentes del sistema, de cursores, de temas, etc. Esta herramienta es una extensión del *Gnome Shell* que, aparte de proporcionar los botones de maximizar y minimizar, facilita a su vez una barra de menú en la parte inferior del escritorio.

De esta manera, para instalar dicha extensión hay que seguir una serie de pasos. En primer lugar, se instala el paquete “*gnome-tweaks*”, tal y como se muestra a continuación:

```
[root@pc1-tft ~]# dnf install gnome-tweaks
```

En segundo lugar, se accede a la [página oficial](#) de las extensiones de **Gnome** y en el buscador principal, se filtra por el nombre de la extensión “*Dash to Panel*”. Cuando se haya localizado, se selecciona y seguidamente, se presenta una página similar a la que plasma la ilustración 5. En este punto, se hace clic en el botón que aparece en la parte superior derecha para habilitar la extensión (modo *ON*).

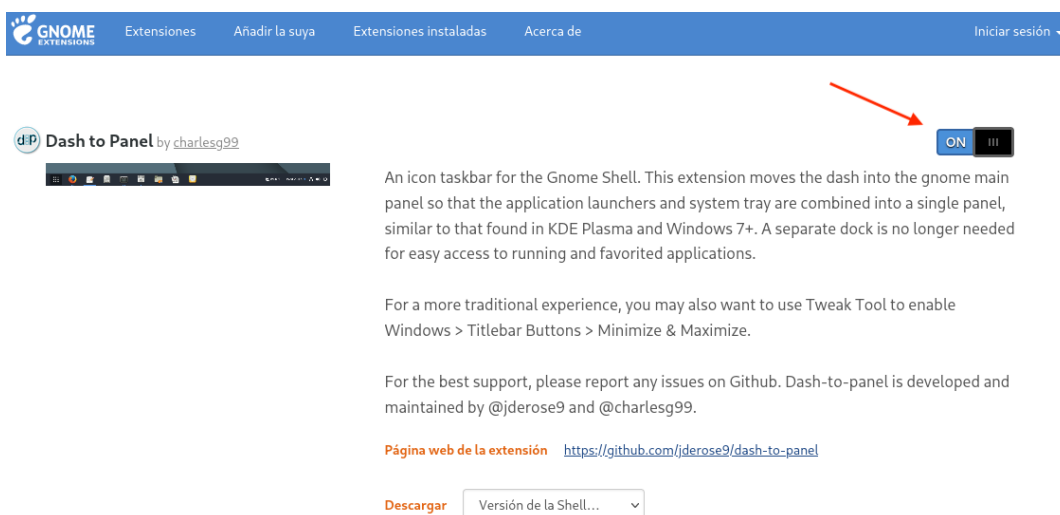


Ilustración 5: Activación de la extensión “*Dash to Panel*”

En tercer lugar, se ejecuta el siguiente comando para abrir el panel de control de la herramienta *Gnome Tweaks*.

```
[root@pc1-tft ~]# gnome-tweaks
```

Seguidamente, se despliega la ventana principal de la herramienta y se selecciona en la barra lateral izquierda el campo “Barras de título de las ventanas”. Del mismo modo, se activa en la sección “Botones de la barra de título” las opciones de “Maximizar” y “Minimizar”, así como se muestra en la ilustración 6.

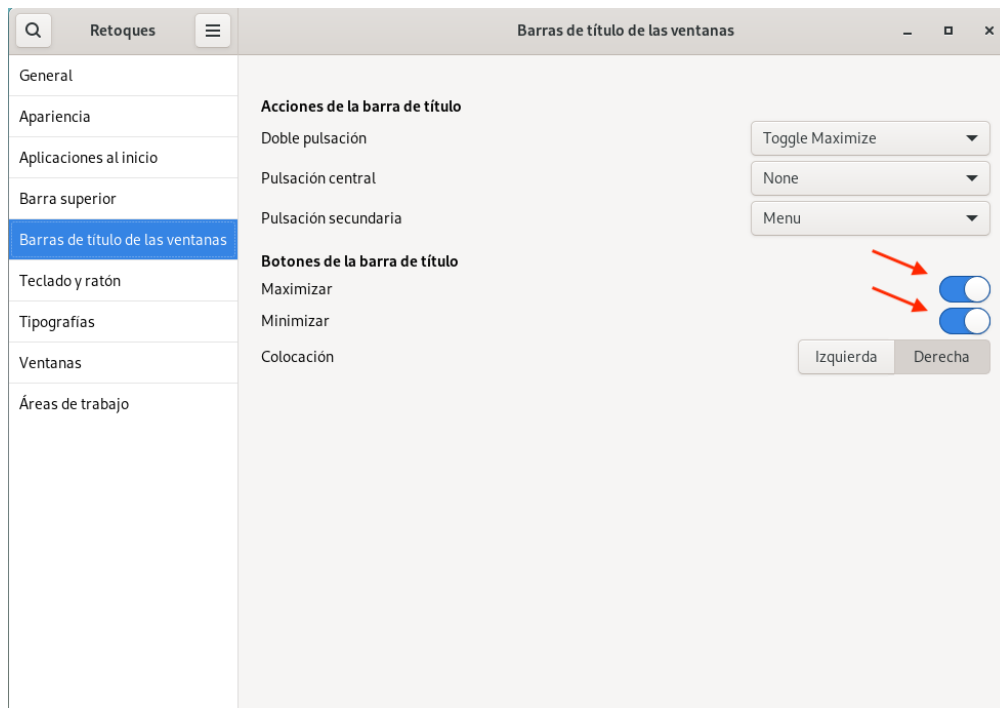


Ilustración 6: Activación de los botones de la barra de título

Instantáneamente, aparecen los botones de maximizar y minimizar en las barras de título de todas las ventanas, como se puede observar en la ilustración 6.

ANEXO III: Instalación del paquete de virtualización KVM

Tal y como se ha explicado en el análisis de los [requerimientos del sistema para la virtualización](#), es necesario disponer de unos requerimientos mínimos del sistema y del hipervisor KVM. También es recomendable poseer otros requerimientos superiores a los mínimos para que el entorno de virtualización funcione de la manera más óptima. Como estos requerimientos ya han sido corroborados previamente, se puede proceder a instalar la plataforma de virtualización.

Siendo así, se instala KVM a través de la línea de comandos utilizando el grupo de paquetes de virtualización proporcionado por la distribución. Los paquetes se dividen en tres secciones principales: obligatorios, opcionales y por defecto.

En este caso, se instalan los paquetes obligatorios, los predeterminados y los opcionales con el comando posterior:

```
[root@pc1-tft ~]# dnf group install --with-optional virtualization
```

Los paquetes que se han instalado se pueden ver tras su ejecución ejecutando:

```
[root@pc1-tft ~]# dnf groupinfo virtualization
```

En la ilustración 7, la ejecución de este comando da como resultado todos los paquetes que se han instalado en el sistema.

```
[aridpd@pc1-tft ~]$ dnf groupinfo virtualization
Última comprobación de caducidad de metadatos hecha hace 1:11:03, el vie 08 abr 2022 13:25:50.
Grupo: Virtualización
Descripción: Paquetes para un entorno de virtualización gráfico
Paquetes obligatorios:
  virt-install
Paquetes predeterminados:
  libvirt-daemon-config-network
  libvirt-daemon-kvm
  qemu-kvm
  virt-manager
  virt-viewer
Paquetes opcionales:
  libguestfs-tools
  python3-libguestfs
  virt-top
```

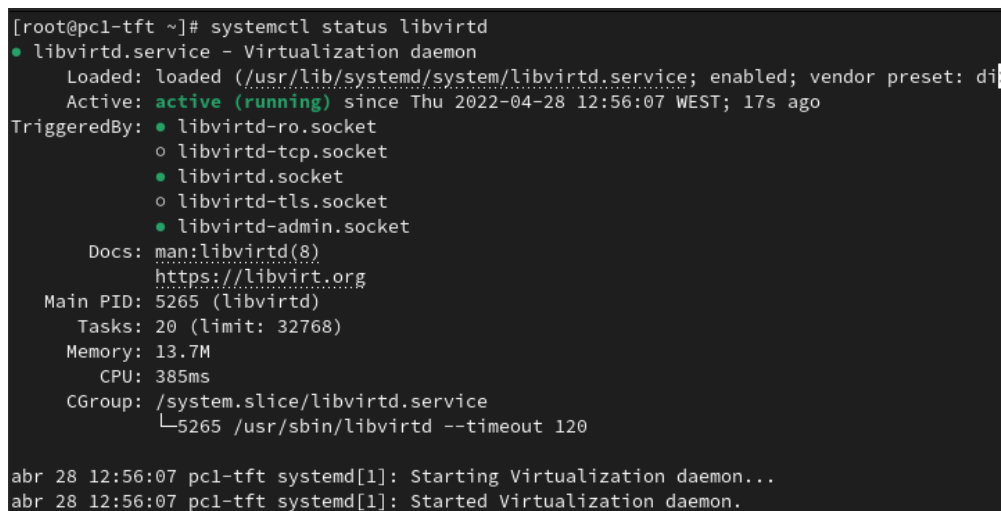
Ilustración 7: Paquetes de virtualización instalados en el sistema anfitrión

Después de instalar los paquetes, se inicia y se habilita el servicio *libvirtd*¹ para que cada vez que se inicie el sistema, este también lo haga, ya que, para poder utilizar las herramientas gráficas o las de línea de órdenes que permiten gestionar las máquinas virtuales, debe estar activo:

```
[root@pc1-tft ~]# systemctl start libvirtd
[root@pc1-tft ~]# systemctl enable libvirtd
```

Tras habilitar e iniciar este servicio, se comprueba el estado para que verificar que no haya ningún tipo de errores de la siguiente manera:

```
[root@pc1-tft ~]# systemctl status libvirtd
```



```
[root@pc1-tft ~]# systemctl status libvirtd
● libvirtd.service - Virtualization daemon
   Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2022-04-28 12:56:07 WEST; 17s ago
     TriggeredBy: ● libvirtd-ro.socket
                  ○ libvirtd-tcp.socket
                  ● libvirtd.socket
                  ○ libvirtd-tls.socket
                  ● libvirtd-admin.socket
     Docs: man:libvirtd(8)
           https://libvirt.org
   Main PID: 5265 (libvirtd)
     Tasks: 20 (limit: 32768)
    Memory: 13.7M
       CPU: 385ms
    CGroup: /system.slice/libvirtd.service
            └─5265 /usr/sbin/libvirtd --timeout 120

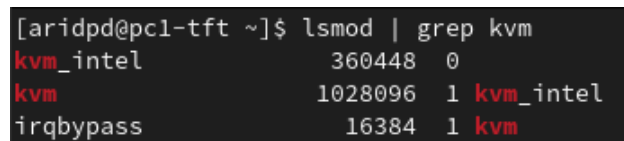
abr 28 12:56:07 pc1-tft systemd[1]: Starting Virtualization daemon...
abr 28 12:56:07 pc1-tft systemd[1]: Started Virtualization daemon.
```

Ilustración 8: Estado del servicio libvirtd

Una vez instalada la plataforma e iniciado el demonio *libvirtd*, hay que verificar que los módulos del kernel KVM están correctamente cargados:

```
[root@pc1-tft ~]# lsmod | grep kvm
```

Lo que debe imprimir la orden anterior es lo que se señala en la ilustración 9.



```
[aridpd@pc1-tft ~]$ lsmod | grep kvm
kvm_intel          360448 0
kvm                1028096 1 kvm_intel
irqbypass         16384 1 kvm
```

Ilustración 9: Comprobación de los módulos del kernel KVM

Finalmente, se constata que las herramientas para la gestión de la virtualización se pueden iniciar correctamente con las siguientes órdenes:

¹Demonio responsable del manejo de las máquinas virtuales y las invocaciones a las librerías que permiten la interacción con el hipervisor y sistemas anfitriones. También controla al hipervisor.

- Orden para ejecutar la utilidad de línea de comandos de virtualización:

```
[root@pc1-tft ~]# virsh
```

- Orden para ejecutar la aplicación *Virtual Machine Manager* que proporciona una interfaz gráfica de virtualización:

```
[root@pc1-tft ~]# virt-manager
```

Tras ejecutar el último comando, se inicia la interfaz gráfica de KVM, tal y como se presenta en la ilustración 10.

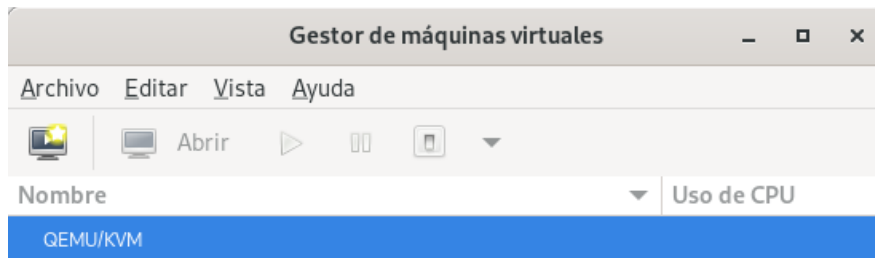


Ilustración 10: Interfaz gráfica de KVM (*virt-manager*)

También se puede acceder a la interfaz gráfica “*virt-manager*” en el menú de las aplicaciones del sistema.

ANEXO IV: Creación de una máquina virtual mediante la utilidad *virt-manager*

Crear una máquina virtual con la plataforma de virtualización KVM se puede realizar de varias maneras, con *virsh*, *virt-manager*, *virt-install*, etc. En este caso, se creará con *virt-manager* siguiendo una serie de pasos.

Tras descargar la imagen ISO de Fedora Workstation 35 en la [página oficial](#) e iniciar la interfaz gráfica de KVM, se procede a crear la nueva máquina haciendo clic en el icono “Crear una nueva máquina virtual”(Ver **ilustración 11**) en la esquina superior izquierda, debajo del campo “Archivo” en el menú.

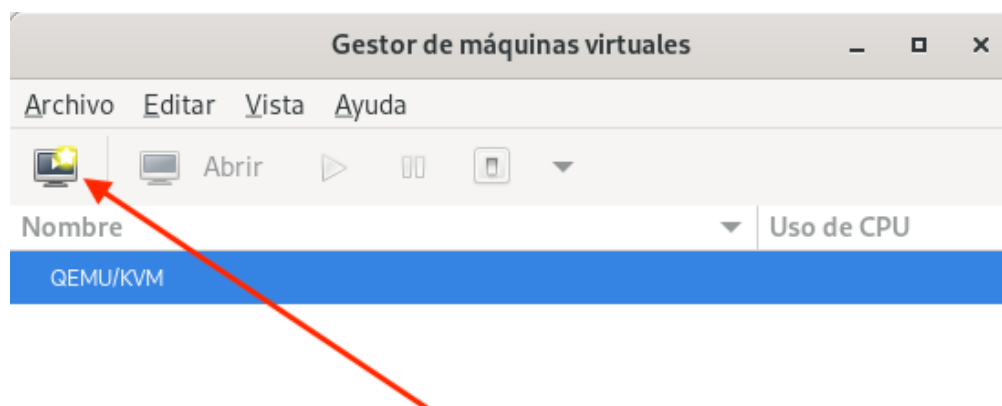


Ilustración 11: Crear nueva máquina virtual con (*virt-manager*)

Una vez hecho clic en el botón, se muestra una ventana emergente para comenzar el proceso de creación de la nueva máquina virtual. El primer paso es elegir el método de instalación del sistema operativo. Como en este caso ya se ha descargado la imagen ISO, se selecciona la primera opción que aparece y se hace clic en “Adelante” para continuar con el proceso (Ver **ilustración 12**).

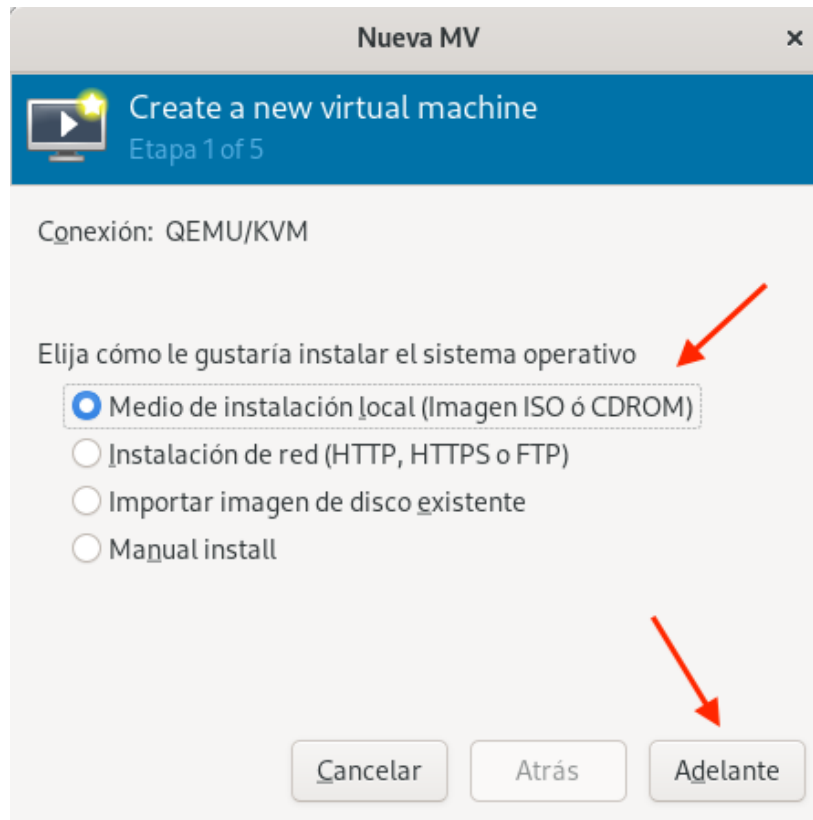


Ilustración 12: Paso 1 - Selección del medio de instalación

En el segundo paso, se selecciona la imagen ISO del sistema operativo a ejecutar tras hacer clic en “Explorar” (**Ver ilustración 13**) y se pasa a la siguiente fase.

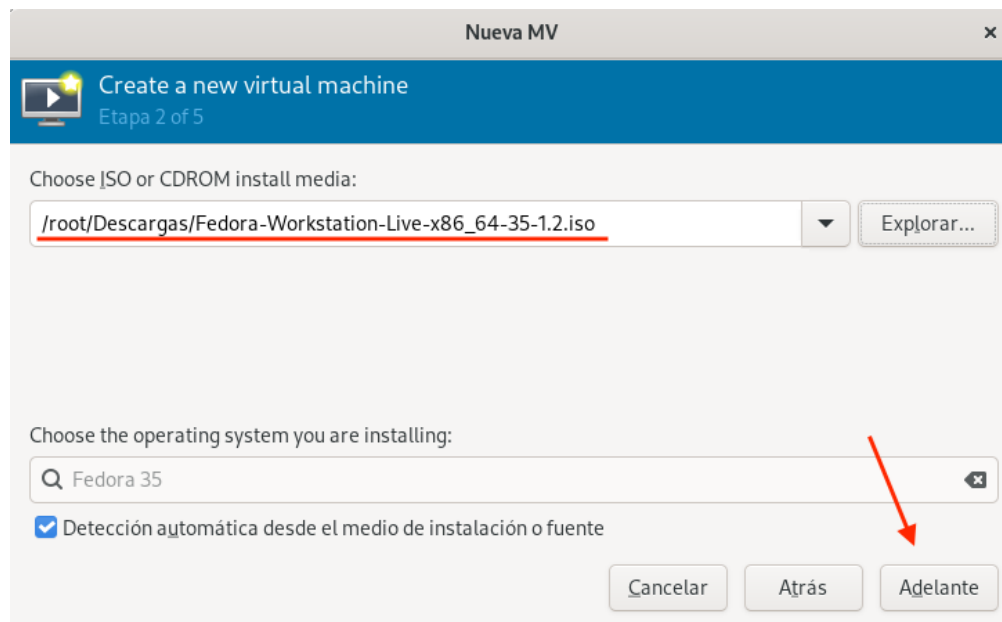


Ilustración 13: Paso 2 - Selección de la imagen ISO

En la tercera etapa, se añade el tamaño de la memoria y el de la CPU, justo como indica la ilustración 14.

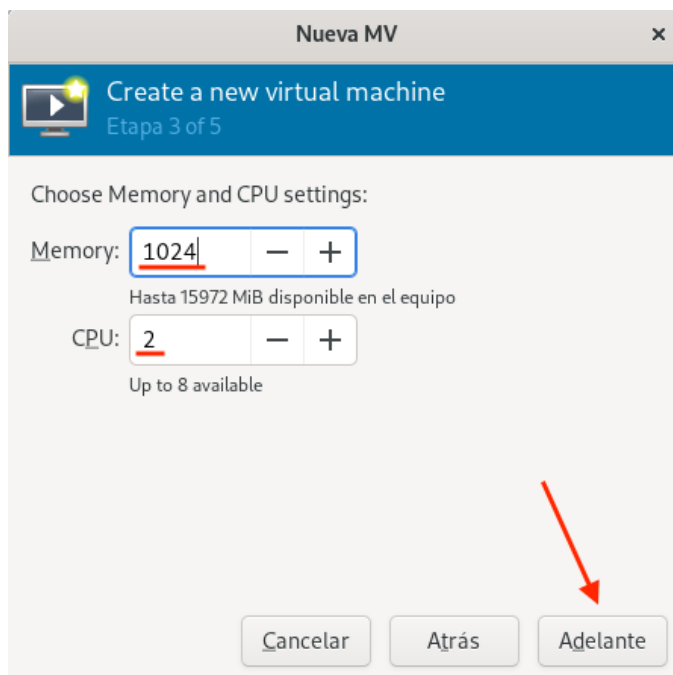


Ilustración 14: Paso 3 - Ajustes de la memoria y CPU

En la siguiente etapa se determina el tamaño que tendrá la imagen de disco para la máquina virtual (**Ver ilustración 15**) y se continúa haciendo clic en “Adelante”.

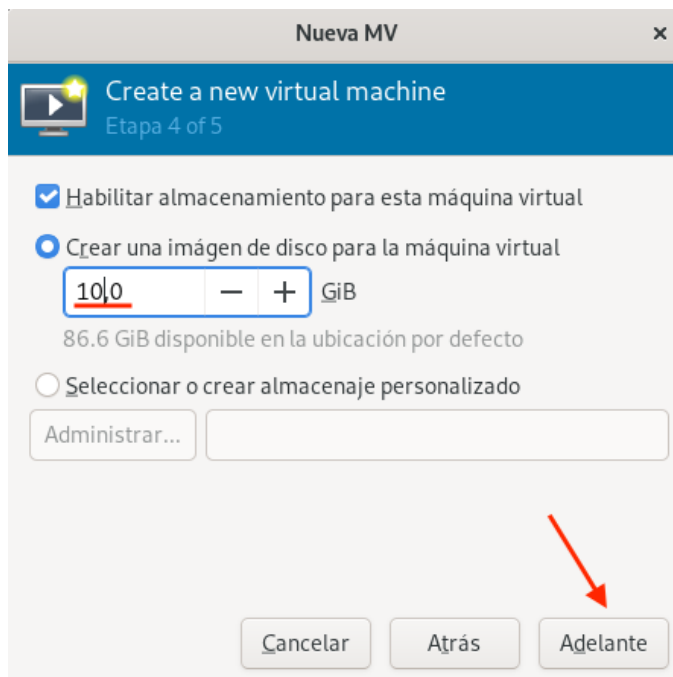


Ilustración 15: Paso 4 - Ajustes de la imagen de disco

Por último, en la quinta fase se establece un nombre para la máquina virtual (**Ver ilustración 16**) y se finaliza el proceso de creación, lo cual lanza la máquina virtual y comienza a iniciarse el sistema operativo. Respecto a la selección de red, se mantiene la red que se encuentra por defecto, que es una red de tipo NAT (*Network Address Translation*)².

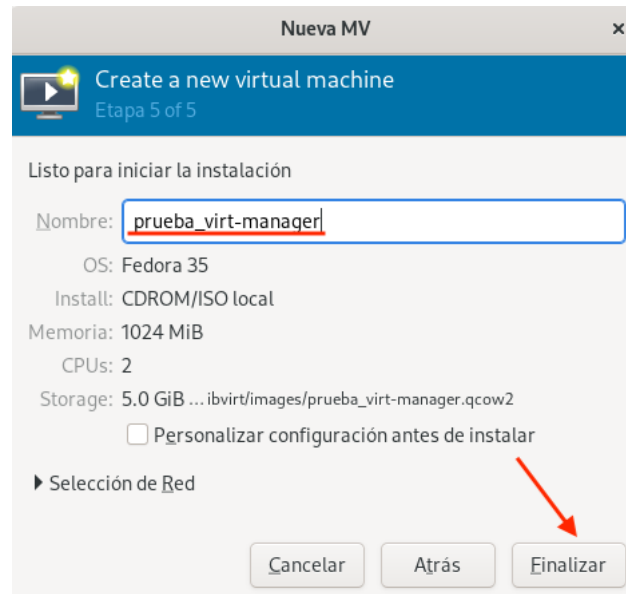


Ilustración 16: Paso 5 - Establecimiento del nombre de la máquina

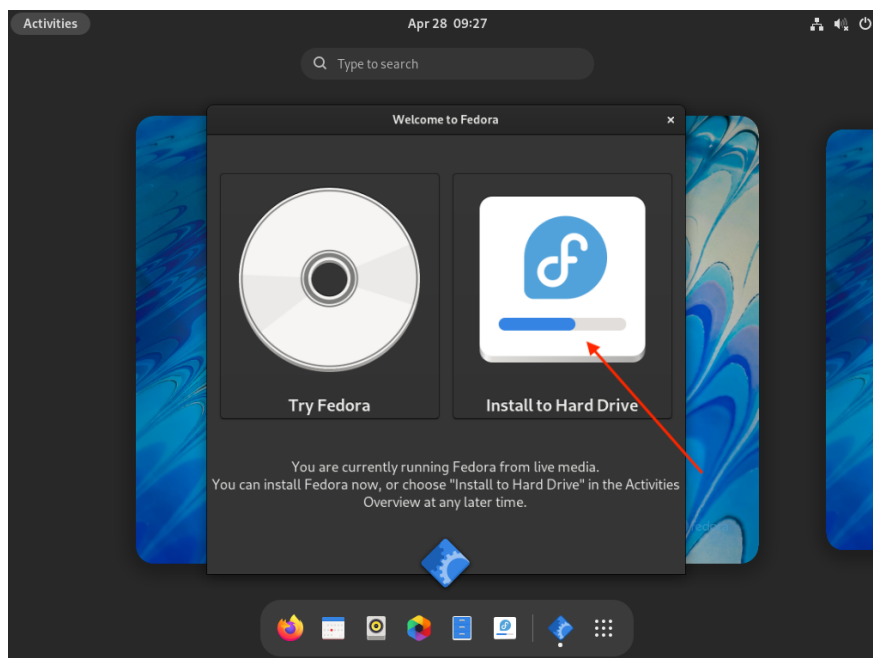


Ilustración 17: Elección del método de uso del sistema operativo

²Mecanismo para intercambiar paquetes entre dos redes con direcciones incompatibles.

Una vez iniciada la máquina, se presentan dos posibles opciones a elegir: utilizar Fedora como un sistema operativo *live*³ o instalar Fedora en el disco (**Ver ilustración 17**). En este aspecto, se selecciona la segunda opción para tener instalado el sistema operativo en el disco virtual de la máquina virtual, ya que si se elige la primera opción, cualquier cambio, configuración o instalación que se realice no se guardará cuando se apague o reinicie el sistema.

En el primer paso de instalación del sistema operativo, se selecciona el idioma como se muestra en la ilustración 18 y se hace clic en “Comenzar” para continuar con el proceso de instalación.

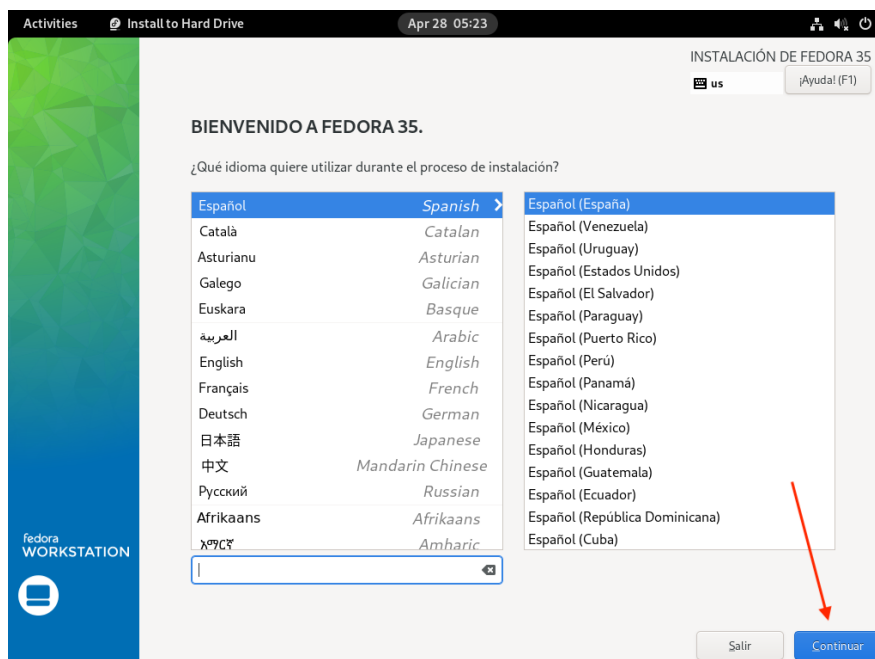


Ilustración 18: Paso 1 de la instalación - Selección de idioma

En el segundo paso, se realizan los ajustes generales y necesarios para la configuración e instalación de la distribución (**Ver ilustración 19**), los cuales se resumen posteriormente:

- * Se corrobora que la distribución del teclado preestablecida corresponda con el idioma que se ha seleccionado en el paso anterior, en este caso, el español.
- * Se determina la fecha y hora del sistema eligiendo el huso horario “Océano Atlántico/-Canarias” en el mapa que se muestra al seleccionar el campo “Fecha y hora”.
- * Se escoge el único disco que se manifiesta al acceder al campo “Destino de la instalación” para instalar Fedora y se selecciona el tipo de particionado automático para que se asignen las particiones necesarias automáticamente.

Cuando se hayan determinado estos ajustes, ya se puede comenzar con la instalación haciendo clic en el botón “Empezar instalación”.

³Sistema operativo que se carga completamente en la memoria RAM para ejecutarse, sin necesidad de instalar nada en el disco duro del sistema.

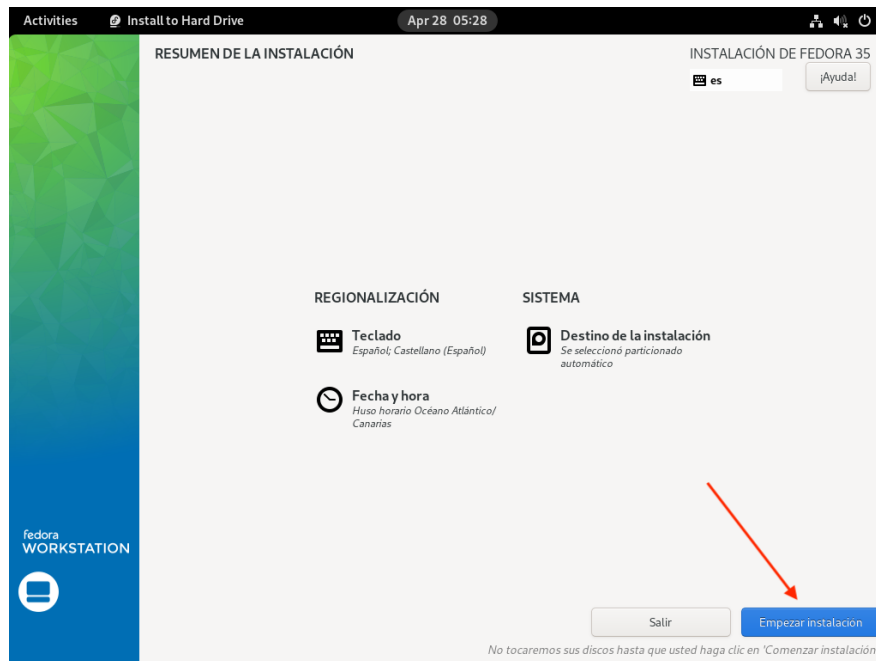


Ilustración 19: Paso 2 de la instalación - Ajustes generales

Así pues, las particiones principales que se crean automáticamente al instalar la distribución de Fedora son las siguientes:

- Una partición `/`: representa la raíz del árbol del sistema de archivos y por ende, contiene todos los archivos del sistema.
- Una partición `/boot`: contiene todos los archivos necesarios para el proceso de arranque.
- Una partición `/home`: contiene todos los directorios de *home* - en español, inicio (Escritorio, Documentos, Descargas, etc.), de todos los usuarios del sistema.
- Una partición `[swap]`: utilizada para apoyar a la memoria virtual extendiendo la memoria del sistema.

En el momento en el que la instalación finalice, se hace clic en “Terminar la instalación” y se reinicia la máquina virtual para aplicar los cambios.

Tras reiniciar la máquina virtual, se inicia un asistente de configuración que solicitará establecer o configurar ciertos parámetros necesarios para poder realizar un buen uso del sistema. Estos parámetros pueden ser: permitir o no la ubicación y los informes de errores, crear un usuario, habilitar o no repositorios de terceros, etc.

Concluida la instalación, la máquina virtual ya estará completamente lista para el uso de todas las funcionalidades que ofrece la distribución de Fedora 35.

Comprobación de la conectividad de la máquina

En este apartado se comprueba que la máquina virtual tiene conectividad hacia al exterior para su posterior funcionamiento. Antes de nada, se consulta la dirección IP que se le ha asignado automáticamente a la máquina, ya que, el DHCP (*Dynamic Host Configuration Protocol*)⁴ está activado en la red NAT que se le ha asignado.

```
[prueba@localhost-live ~]$ ifconfig
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.122.152 netmask 255.255.255.0 broadcast 192.168.122.255
    inet6 fe80::8698:9858:3520:65f3 prefixlen 64 scopeid 0x20<link>
    ether 52:54:00:fe:7e:77 txqueuelen 1000 (Ethernet)
    RX packets 50998 bytes 122417221 (116.7 MiB)
    RX errors 0 dropped 60 overruns 0 frame 0
    TX packets 34562 bytes 2337465 (2.2 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 26 bytes 2766 (2.7 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 26 bytes 2766 (2.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Ilustración 20: Examinar la dirección IP asignada a la máquina

Para poder llevar a cabo la verificación de que la máquina virtual tiene conectividad con otros dispositivos y hacia internet, se ejecuta el comando “ping” seguido de la dirección IP o el nombre de dominio que se quiera comprobar. Este comando es una utilidad de diagnóstico que permite verificar el estado de la conexión de la máquina donde se ejecuta hacia otros dispositivos o hacia internet.

Para verificar que el comando ejecutado con la dirección IP o nombre de dominio indicado responde afirmativamente, es decir, que el sistema donde se esté ejecutando tiene conectividad con el dispositivo u *host* destino, el resultado tiene que mostrar el nombre de dominio al que se intenta conectar y su dirección IP entre paréntesis enviando paquetes de 64 bytes normalmente, al destino. También se muestra ciertos parámetros que indican que se están comunicando correctamente (el *icmp_seq*, el *time to live* y el *time*).

Tan pronto como se sepa la dirección IP, se puede verificar que la máquina tiene conectividad tanto a la puerta de enlace predeterminada como a la dirección IP del host anfitrión (10.13.15.47). Del mismo modo, se puede realizar la comprobación hacia internet indicando cualquiera de las direcciones IP o resoluciones de nombres que se encuentren fuera de la red a la que se está conectado. Por ejemplo, a la dirección IP de google 8.8.8.8 o 8.8.4.4. También se puede realizar con el nombre de dominio, que es www.google.es.

⁴Es un protocolo de red cliente/servidor que se encarga de proporcionar automáticamente direcciones IP, máscaras de subred, puertas de enlaces predeterminadas, etc.

Comprobación de conectividad desde la máquina virtual a la puerta de enlace y al host anfitrión:

```
[prueba@localhost-live ~]$ ping 192.168.122.1
PING 192.168.122.1 (192.168.122.1) 56(84) bytes of data.
64 bytes from 192.168.122.1: icmp_seq=1 ttl=64 time=0.321 ms
64 bytes from 192.168.122.1: icmp_seq=2 ttl=64 time=0.270 ms
64 bytes from 192.168.122.1: icmp_seq=3 ttl=64 time=0.243 ms
^C
--- 192.168.122.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2081ms
rtt min/avg/max/mdev = 0.243/0.278/0.321/0.032 ms
[prueba@localhost-live ~]$ ping 10.13.15.47
PING 10.13.15.47 (10.13.15.47) 56(84) bytes of data.
64 bytes from 10.13.15.47: icmp_seq=1 ttl=64 time=0.281 ms
64 bytes from 10.13.15.47: icmp_seq=2 ttl=64 time=0.269 ms
64 bytes from 10.13.15.47: icmp_seq=3 ttl=64 time=0.258 ms
^C
--- 10.13.15.47 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2050ms
rtt min/avg/max/mdev = 0.258/0.269/0.281/0.009 ms
[prueba@localhost-live ~]$
```

Ilustración 21: Ping desde la máquina virtual

Asimismo, el host anfitrión también debería de poder conectarse con la máquina virtual:

```
[root@pc1-tft ~]# ping 192.168.122.152
PING 192.168.122.152 (192.168.122.152) 56(84) bytes of data.
64 bytes from 192.168.122.152: icmp_seq=1 ttl=64 time=0.414 ms
64 bytes from 192.168.122.152: icmp_seq=2 ttl=64 time=0.278 ms
64 bytes from 192.168.122.152: icmp_seq=3 ttl=64 time=0.307 ms
64 bytes from 192.168.122.152: icmp_seq=4 ttl=64 time=0.365 ms
64 bytes from 192.168.122.152: icmp_seq=5 ttl=64 time=0.311 ms
64 bytes from 192.168.122.152: icmp_seq=6 ttl=64 time=0.238 ms
^C
--- 192.168.122.152 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5125ms
rtt min/avg/max/mdev = 0.238/0.318/0.414/0.057 ms
```

Ilustración 22: Ping desde el host anfitrión hacia la máquina virtual

ANEXO V: Creación de máquinas virtuales mediante distintas utilidades

En esta sección se desarrollan los pasos de las diferentes maneras que existen de crear máquinas virtuales según las utilidades que se pueden emplear y que proporciona KVM. Estas utilidades se encargan de gestionar la virtualización a través de la interfaz de línea de comandos o mediante la interfaz gráfica de usuario.

Las herramientas que se utilizarán son: “*virsh*”, “*virt-install*” y “*virt-clone*” como interfaz de línea de comandos y “*virt-manager*” como interfaz gráfica.

1. Creación de una máquina virtual mediante una copia de seguridad

Este apartado consiste en crear una máquina virtual por medio de una clonación manual de la máquina virtual creada anteriormente mediante la utilidad “*virt-manager*”. Esta máquina posee una instalación del sistema operativo Fedora Workstation 35 (Ver en el [Anexo IV](#)).

Una clonación manual es llevar a cabo una copia a mano de los ficheros esenciales para la creación de una máquina virtual en KVM, o lo que también se conoce como una copia de seguridad. En el caso de KVM, se generan dos archivos al crear una máquina virtual: el fichero de configuración XML y el archivo de la imagen del disco. El fichero de configuración XML especifica los ajustes y componentes de las máquinas virtuales creadas, por lo tanto, será este el que se utilizará para realizar la clonación o creación de la nueva máquina virtual.

Tras esta copia, se modifican las características principales que definen a cada máquina virtual. Estas características únicas son: el nombre del fichero de configuración XML copiado, el UUID (*Universally Unique Identifier*)⁵, el nombre del disco y la dirección MAC (*Media Access Control*)⁶. A continuación, se distinguen los pasos a tomar para este proceso.

Buscar y copiar los ficheros

La forma más eficaz y rápida de buscar los ficheros necesarios para la clonación de la máquina virtual es aplicando el siguiente comando en la terminal del host anfitrión:

```
[root@pc1-tft ~]# find / -name prueba_virt_manager*
```

⁵Identificador único de 16 bytes que posibilita distinguir objetos en el sistema o en distintos entornos.

⁶Identificador único de 48 bits que se asigna a cada tarjeta de red de los dispositivos conectados a la red.

```
[root@pc1-tft ~]# find / -name prueba_virt-manager*
/var/lib/libvirt/images/prueba_virt-manager.qcow2
/var/log/libvirt/qemu/prueba_virt-manager.log
/etc/libvirt/qemu/prueba_virt-manager.xml
```

Ilustración 23: Búsqueda de ficheros de configuración XML y de disco de la máquina virtual

En la ilustración 23, se puede apreciar la salida de la ejecución del comando. Se ha realizado una búsqueda por el nombre “prueba_virt_manager”, ya que el nombre de la máquina creada anteriormente se llama de esta manera y los ficheros se generan con dicho nombre.

Tras finalizar la búsqueda en el sistema de archivos del host anfitrión, se han encontrado tres ficheros, de los cuales, aquellos que tienen la extensión .xml y .qcow2 son los que se necesitarán, pues el primero (.xml), es el archivo de especificación de la máquina y el segundo (.qcow2), es el disco de la máquina virtual.

Una vez encontrados los dos archivos, se procede a realizar la copia de ellos mediante la línea de comandos en el mismo directorio en el que se encuentran pero con diferentes nombres:

```
[root@pc1-tft ~]# cp /etc/libvirt/qemu/prueba_virt-manager.xml
→ /etc/libvirt/qemu/prueba_conbackup.xml
[root@pc1-tft ~]# cp /var/lib/libvirt/images/prueba_virt-manager.qcow2
→ /var/lib/libvirt/images/prueba_conbackup.qcow2
```

Si la ejecución de ambos comandos han finalizado correctamente, en la terminal no se imprime ningún mensaje como resultado (**Ver ilustración 24**).

```
[root@pc1-tft ~]# cp /etc/libvirt/qemu/prueba_virt-manager.xml /etc/libvirt/qemu
/prueba_conbackup.xml
[root@pc1-tft ~]# cp /var/lib/libvirt/images/prueba_virt-manager.qcow2 /var/lib/
libvirt/images/prueba_conbackup.qcow2
```

Ilustración 24: Copia de los ficheros principales de la máquina virtual

Modificación del fichero de configuración XML

Como se ha mencionado, hay que modificar el fichero de especificación de la máquina para ajustar los parámetros a la nueva máquina a crear (nombre de la máquina, UUID, nombre del disco y MAC).

Primeramente, se accede al fichero XML y en él se modifica el nombre de la máquina virtual y se elimina el UUID, debido a que, al iniciar la máquina con este fichero, se genera uno automáticamente. El UUID también puede ser generado con el comando *uuidgen* si se prefiere añadir al fichero previamente (**Ver ilustración 25**).


```

<domain type='kvm'>
  <name>prueba_conbackup</name>
  <uuid></uuid>
  <metadata>
    <libosinfo:libosinfo xmlns:libosinfo="http://libosinfo.org/xmlns/libvirt/domain/1.0">
      <libosinfo:os id="http://fedoraproject.org/fedora/35"/>
    </libosinfo:libosinfo>
  </metadata>
  <memory unit='KiB'>1048576</memory>
  <currentMemory unit='KiB'>1048576</currentMemory>
  <vcpu placement='static'>2</vcpu>
  <os>
    <type arch='x86_64' machine='pc-q35-6.1'>hvm</type>
    <boot dev='hd' />
  </os>

```

Ilustración 25: Modificación del nombre de la máquina virtual en el fichero XML

En segundo lugar, se cambia la ruta donde se encuentra el disco de la nueva máquina, reemplazando el nombre de la máquina anterior por el que se desee, pues al iniciar la máquina virtual este se creará con dicho nombre (**Ver ilustración 26**).

```

<devices>
  <emulator>/usr/bin/qemu-system-x86_64</emulator>
  <disk type='file' device='disk'>
    <driver name='qemu' type='qcow2' />
    <source file='/var/lib/libvirt/images/prueba_conbackup.qcow2' />
    <target dev='vda' bus='virtio' />
    <address type='pci' domain='0x0000' bus='0x04' slot='0x00' function='0x0' />
  </disk>
  <disk type='file' device='cdrom'>
    <driver name='qemu' type='raw' />
    <target dev='sda' bus='sata' />
    <readonly />
    <address type='drive' controller='0' bus='0' target='0' unit='0' />
  </disk>
  <controller type='usb' index='0' model='qemu-xhci' ports='15'>
    <address type='pci' domain='0x0000' bus='0x02' slot='0x00' function='0x0' />
  </controller>
  <controller type='sata' index='0'>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x1f' function='0x2' />
  </controller>

```

Ilustración 26: Modificación de la ruta del disco de la máquina virtual en el fichero XML

Por último, se elimina la dirección MAC que ha sido asignada a la máquina virtual cuando esta se ha creado, pues no pueden existir, al igual que el UUID, dos MAC idénticas para distintas máquinas virtuales. Se puede diferenciar en la ilustración 27 que en la sección de las interfaces de red no aparece ninguna directiva con la MAC de la máquina.

```

<controller type='pci' index='7' model='pcie-root-port'>
  <model name='pcie-root-port' />
  <target chassis='7' port='0x16' />
  <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x6' />
</controller>
<interface type='network'>
  <source network='default' />
  <model type='virtio' />
  <address type='pci' domain='0x0000' bus='0x01' slot='0x00' function='0x0' />
</interface>

```

Ilustración 27: Eliminación de la MAC de la máquina virtual en el fichero XML

Crear e iniciar la máquina virtual

Una vez los ficheros están listos y en su directorio correspondiente, se utiliza “*virsh*”, la utilidad de línea de comandos, para crear la máquina virtual a través del fichero de configuración XML con el comando:

```
[root@pcl-tft ~]# virsh-define /etc/libvirt/qemu/prueba_conbackup.xml
```

En la ilustración 28, se muestra el resultado que ha mostrado por pantalla la terminal al ejecutar el comando para crear la máquina virtual. Para comprobar que se ha registrado la máquina correctamente, se ejecuta la segunda orden que aparece en la ilustración, pues esta, lista todas las máquinas virtuales que han sido creadas, pudiendo corroborar de esta manera que en el listado aparece el nombre de la nueva máquina virtual.

```
[root@pcl-tft ~]# virsh define /etc/libvirt/qemu/prueba_conbackup.xml
Domain 'prueba_conbackup' defined from /etc/libvirt/qemu/prueba_conbackup.xml

[root@pcl-tft ~]# virsh list --all
Id     Nombre                Estado
-----
-      prueba_conbackup     apagado
-      prueba_virt-manager  apagado
```

Ilustración 28: Creación de la máquina virtual a partir del fichero XML

En este punto, ya se puede iniciar la máquina virtual para verificar que se ha creado correctamente con el comando posterior:

```
[root@pcl-tft ~]# virsh start prueba_conbackup
```

Si este comando ha iniciado adecuadamente la máquina, deberá imprimir un mensaje indicando que la máquina se ha iniciado. De la misma manera que se ha comprobado que se ha creado la máquina anteriormente, se puede verificar también que la máquina está iniciada con el segundo comando que se muestra en la ilustración 29, el cual muestra un listado de todas las máquinas encendidas.

```
[root@pcl-tft ~]# virsh start prueba_conbackup
Domain 'prueba_conbackup' started

[root@pcl-tft ~]# virsh list
Id     Nombre                Estado
-----
6      prueba_conbackup     ejecutando
```

Ilustración 29: Iniciar la máquina virtual

Para verificar que realmente la máquina está encendida, se podría utilizar la utilidad con entorno gráfico (*virt-manager*) o ejecutar el siguiente comando para que abra la ventana de la máquina rápidamente:

```
[root@pc1-tft ~]# virt-viewer prueba_conbackup
```

Comprobación de la generación del UUID y de la MAC

Tras el inicio de la máquina virtual, se comprueba en el fichero de configuración XML que se han generado automáticamente el UUID (**Ver ilustración 30**) y la MAC (**Ver ilustración 31**) en la nueva máquina virtual.

```
<domain type='kvm'>
  <name>prueba_conbackup</name>
  <uuid>ee20e0b3-358d-415f-926b-144dfd52f31e</uuid>
  <metadata>
    <libosinfo:libosinfo xmlns:libosinfo="http://libosinfo.org/xmlns/libvirt/domain/1.0">
      <libosinfo:os id="http://fedoraproject.org/fedora/35"/>
    </libosinfo:libosinfo>
  </metadata>
```

Ilustración 30: Comprobación de la generación automática del UUID

```
<interface type='network'>
  <mac address='52:54:00:12:af:51'>
  <source network='default'>
  <model type='virtio'>
  <address type='pci' domain='0x0000' bus='0x01' slot='0x00' function='0x0'>
</interface>
```

Ilustración 31: Comprobación de la generación automática de la MAC

Comprobación de la conectividad

De igual forma que se ha realizado la verificación de la conectividad de la máquina virtual en el **Anexo IV**, se lleva a cabo aquí, haciendo uso del comando *ping* para corroborar que la máquina puede conectarse a la puerta de enlace, al host anfitrión y a internet. Además de cerciorarse de que el host anfitrión tiene conexión con la máquina virtual.

2. Clonación de una máquina virtual a través de virt-manager

En este apartado se realizará una clonación de la máquina virtual original (la creada en el anexo anterior) con el comando “*virt-manager*”, mediante el entorno gráfico. Hay que tener en cuenta que para realizar cualquier clonación hay que tener la máquina virtual a clonar apagada.

Para iniciar la clonación con el “*virt-manager*”, hay que situarse en la máquina virtual a clonar, hacer clic derecho y seleccionar la opción de “Clonar” (**Ver ilustración 32**).

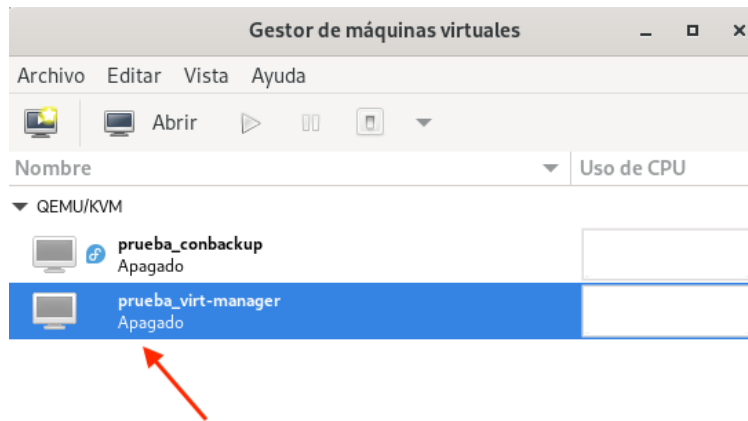


Ilustración 32: Clonar la máquina virtual con *virt-manager*

En la ilustración 33 se plasma la ventana que aparece tras elegir la opción de clonar anteriormente y donde hay que indicar el nombre de la nueva máquina virtual para iniciar el proceso de clonación.

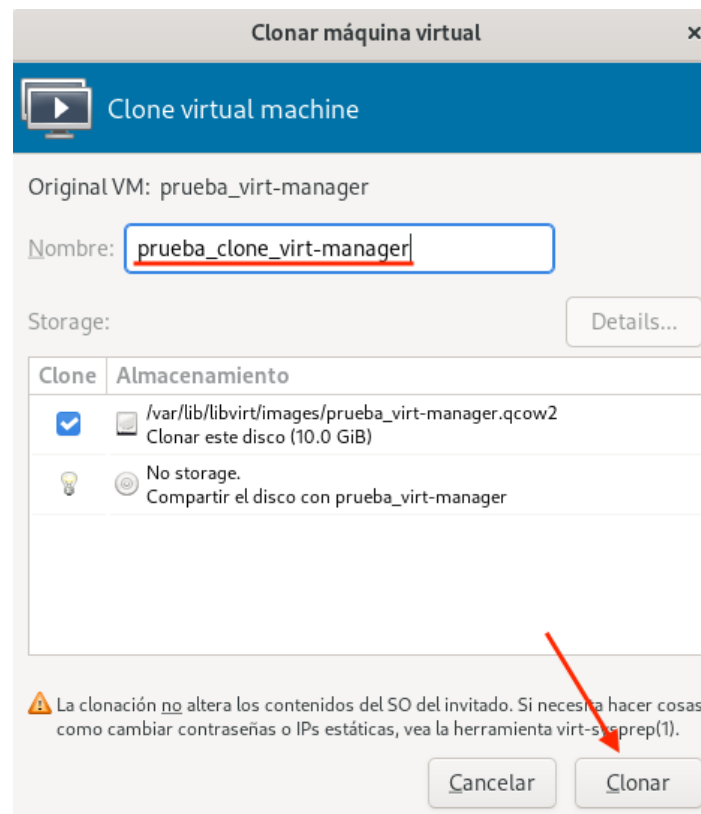


Ilustración 33: Cambiar nombre de la máquina virtual a clonar

Seguidamente, se inicia la máquina virtual mediante la interfaz gráfica de usuario (*virt-manager*) para comprobar la conectividad hacia el exterior, tal y como se realiza en el [Anexo IV](#).

3. Clonación de una máquina virtual por medio de `virt-clone`

En este otro apartado se realizará, al igual que en el apartado anterior, una clonación de la máquina virtual original (la creada en el anexo anterior), pero ahora con el comando “*virt-clone*”, desde la línea de comandos. Como se indicó anteriormente, hay que recordar que para realizar una clonación con cualquier tipo de herramienta, la máquina virtual que se va a clonar debe permanecer apagada.

El comando que se ejecuta para clonar la máquina virtual es el siguiente:

```
[root@pci-tft ~]# virt-clone --original prueba_virt-manager --name
→ prueba_clone_virt-clone --file
→ /var/lib/libvirt/images/prueba_clone_virt-clone.qcow2
```

En la ilustración 34 se muestra el resultado que se ha generado tras ejecutar el comando anterior, el cual indica que se ha asignado el disco y que la máquina se ha creado correctamente. Los parámetros que se le han indicado son los mínimos para poder realizar una clonación desde la línea de comandos: el nombre de la máquina que se va a clonar, el nombre de la nueva máquina que se va a generar y el fichero de donde se creará el nuevo disco asociado a la nueva máquina.

```
[root@pci-tft ~]# virt-clone --original prueba_virt-manager --name prueba_clone_
virt-clone --file /var/lib/libvirt/images/prueba_clone_virt-clone.qcow2
Asignando 'prueba_clone_virt-clone.qcow2' | 10 GB 00:16
El clon 'prueba_clone_virt-clone' ha sido creado exitosamente.
```

Ilustración 34: Clonar la máquina virtual con *virt-clone*

Acto seguido, para verificar que se ha registrado la nueva máquina adecuadamente, se ejecuta la primera orden que aparece en la ilustración 35, pues esta, lista todas las máquinas virtuales que han sido creadas. Se puede comprobar que en el listado aparece el nombre de todas las máquinas que han sido creadas anteriormente y de la recién creada.

```
[root@pci-tft ~]# virsh list --all
Id     Nombre                               Estado
-----
-      prueba_clone_virt-clone              apagado
-      prueba_clone_virt-manager            apagado
-      prueba_conbackup                     apagado
-      prueba_virt-manager                  apagado

[root@pci-tft ~]# virsh start prueba_clone_virt-clone
Domain 'prueba_clone_virt-clone' started
```

Ilustración 35: Comprobación de la máquina virtual creada con *virt-clone*

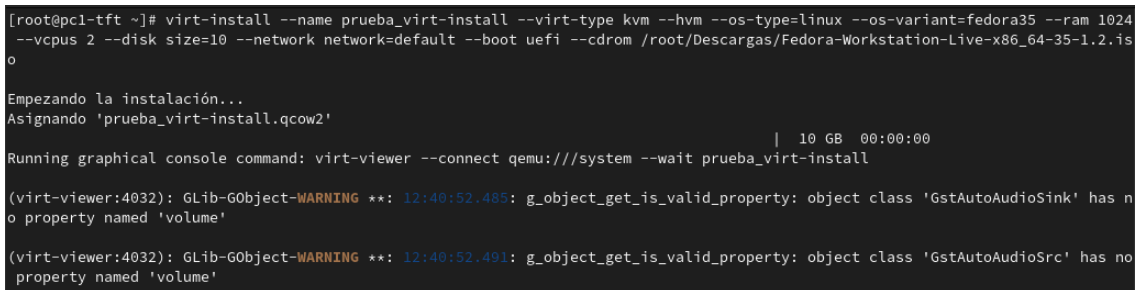
Para concluir, de la misma manera en la que se ha iniciado la máquina virtual creada en el apartado anterior, se inicia esta también para comprobar que la conectividad es correcta (Ver ilustración 35).

4. Creación de una máquina virtual con `virt-install`

Este último apartado trata de crear e instalar desde cero una máquina virtual con la utilidad de línea de órdenes “*virt-install*”. Por lo tanto, el comando a ejecutar es el que se muestra a continuación:

```
[root@pc1-tft ~]# virt-install --name prueba_virt-install --virt-type
↳ kvm --hvm --os-type=linux --os-variant=fedora35 --ram 1024 --vcpus
↳ 2 --disk size=10 --network network=default --boot uefi --cdrom
↳ /root/Descargas/Fedora-Workstation-Live-x86_64-35-1.2.iso
```

La salida de la ejecución del comando que se ha mencionado se puede comprobar en la ilustración 36, donde se puede apreciar que la instalación ha comenzado y que se ha asignado el disco adecuadamente a la máquina virtual.



```
[root@pc1-tft ~]# virt-install --name prueba_virt-install --virt-type kvm --hvm --os-type=linux --os-variant=fedora35 --ram 1024
--vcpus 2 --disk size=10 --network network=default --boot uefi --cdrom /root/Descargas/Fedora-Workstation-Live-x86_64-35-1.2.is
o
Empezando la instalación...
Asignando 'prueba_virt-install.qcow2'
| 10 GB 00:00:00
Running graphical console command: virt-viewer --connect qemu:///system --wait prueba_virt-install
(virt-viewer:4032): Glib-GObject-WARNING **: 12:40:52.485: g_object_get_is_valid_property: object class 'GstAutoAudioSink' has no
property named 'volume'
(virt-viewer:4032): Glib-GObject-WARNING **: 12:40:52.491: g_object_get_is_valid_property: object class 'GstAutoAudioSrc' has no
property named 'volume'
```

Ilustración 36: Creación de la máquina virtual con *virt-install*

Luego, se inicia la máquina virtual creada y se siguen los pasos que se han llevado a cabo en el [Anexo IV](#) para instalar el sistema operativo de Fedora Workstation 35.

ANEXO VI: Creación de recursos de almacenamiento para las máquinas virtuales

Según lo mencionado en el apartado “**Utilización de los recursos de almacenamiento virtual**”, se trabajará con los dos tipos de contenedores que existen para las máquinas virtuales: locales y de red. Se crearán nuevos contenedores, volúmenes y particiones lógicas. Por lo tanto, este epígrafe se dividirá en dos secciones; una dedicada a los contenedores locales y sus correspondientes actividades y otra para los contenedores en red.

1. Contenedores locales

1.1. Creación de un volumen en el contenedor por defecto

En esta primera parte, se procederá a crear un nuevo volumen de 1 GB con nombre “Vol1_p5” en el contenedor por defecto y se asociará a una máquina virtual llamada “prueba_virt_manager”.

Este apartado se realizará desde el panel de administración de hardware de la máquina virtual proporcionado por KVM. Se efectuarán los pasos en “*virt-manager*” y a través del *shell* con las órdenes: *virsh vol-create-as* y *virsh attach-device* o *virsh attach-disk*.

1.1.1. Creación y asociación con *virt-manager*

Para crear un nuevo volumen mediante la utilidad “*virt-manager*”, hay que seleccionar en el menú superior, la pestaña “Editar” y acceder a “Detalles de conexión”. Una vez aquí, se elige el contenedor donde se va a crear el nuevo volumen (*default*) y se hace clic en el icono “+” que está al lado del campo “Volúmenes”. Seguidamente, se abre una ventana que se deberá modificar como se muestra en la ilustración 37, añadiendo el nombre del volumen (Vol1_p5) y la capacidad máxima que tendrá (1 GB).

El siguiente paso es asociar el volumen a la máquina virtual, para ello se selecciona la máquina “prueba_virt_manager” y desde el menú superior, en la pestaña “Editar”, se accede a “Detalles de la máquina virtual” y se abre una ventana emergente, se hace clic en “Agregar hardware” y se añade el volumen recién creado.

Tras la asociación de dicho volumen a la máquina virtual, se comprueba que esta tenga dos volúmenes o discos asociados: el que viene designado por defecto y el que se acaba de crear (**Ver ilustración 38**).

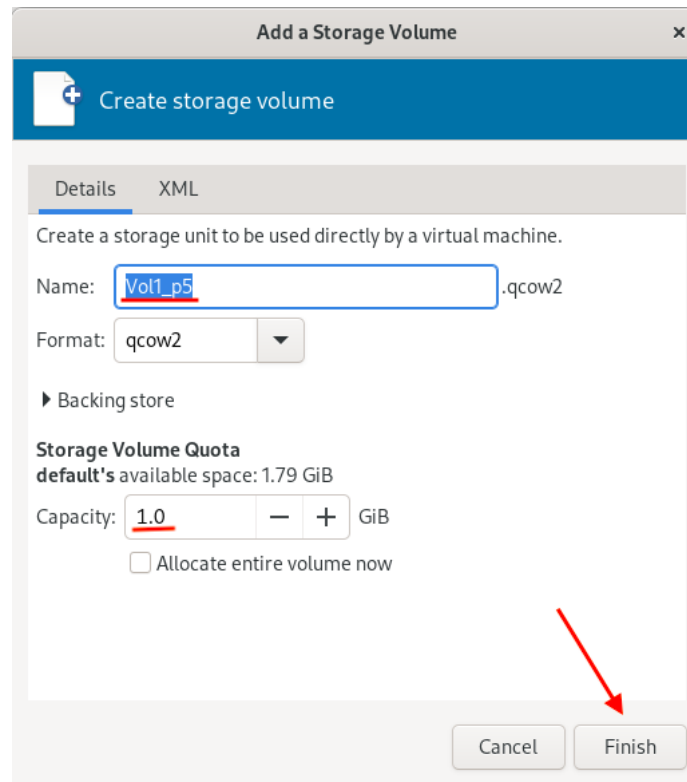


Ilustración 37: Creación del volumen en el contenedor por defecto

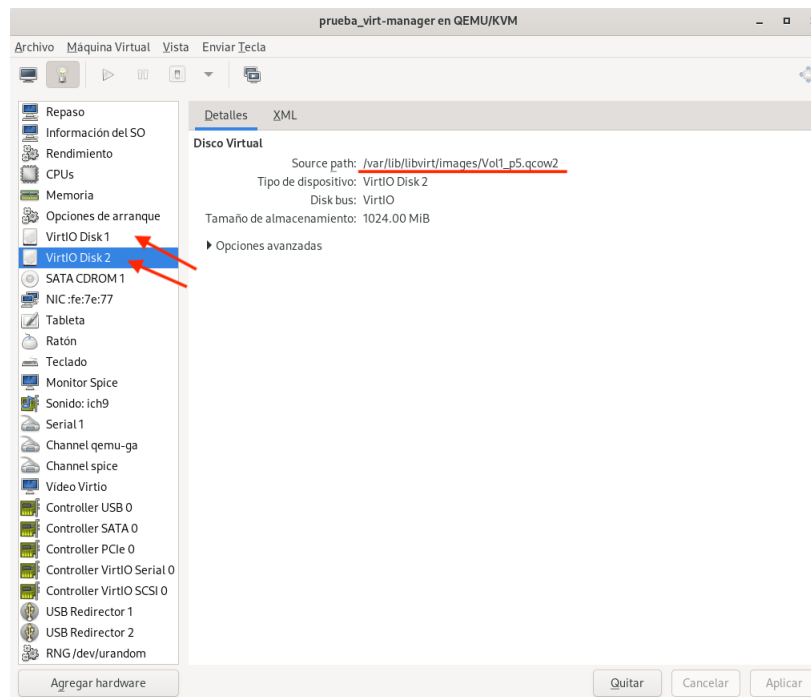


Ilustración 38: Comprobación de los volúmenes asignados a la máquina virtual

En este momento, ya se puede ejecutar y abrir la máquina virtual para crear un fichero

y corroborar que se pueda almacenar datos en el nuevo volumen. No obstante, antes de poder realizar esto, hay que preparar el volumen creando una partición, formateándola y montándola para verificar que se puede usar como almacenamiento, por lo tanto, se procede a desarrollar los pasos necesarios.

Pasos para preparar el volumen

En primer lugar, se consulta todos los dispositivos de bloques que dispone la máquina virtual, es decir, los discos, las particiones, los dispositivos usb, etc., mostrando también información más detallada de estos como puede ser: los sistemas de archivos existentes, los puntos de montaje y el tamaño total de cada partición, disco o bloque.

```
[root@fedora ~]# lsblk -f
NAME FSTYPE FSVER LABEL          UUID                                FSAVAIL FSUSE% MOUNTPOINTS
sr0
zram0
vda
├─vda1
│   ext4  1.0          fbf32932-2458-4b44-8fdf-cc974c34e701  735,1M   18% /boot
├─vda2
│   btrfs          fedora_localhost-live
│                               316261cc-dfb9-44cb-857c-9ada41034b51  5,7G    34% /home
└─vdb
```

Ilustración 39: Examinar los discos y particiones

Como se puede observar en la ilustración 39, el disco “vdb” no cuenta con ninguna partición, sistema de archivos o punto de montaje. Por lo tanto, se deduce que este es el nuevo volumen que se ha añadido a la máquina virtual y por ende, es donde se deben realizar todos los pasos.

Paso 1 - Crear una partición en el nuevo volumen

Se comienza entonces ejecutando el comando **fdisk** indicando el nuevo disco virtual (/dev/vdb) para dividir en forma lógica un disco duro, es decir, crear una partición lógica de dicho disco.

Se introduce la letra ‘n’ para crear una nueva partición. Luego, se introduce la letra ‘p’ para indicar el tipo de partición primaria y para concluir, se pulsa la tecla ‘Enter’ en las siguientes tres preguntas para elegir los valores que se ofrecen por defecto del número de partición, del tamaño del primer sector y del tamaño del último sector. En la última pregunta, se le asigna por defecto 952,7 MiB como tamaño total de la partición, ya que al crear el volumen, se le asignó solamente un 1 GB, lo que equivale aproximadamente a los 952.7 MiB si se tiene en cuenta que normalmente no se posee 1 GB completo. Para terminar, se escribe ‘w’ y ya estaría creada la nueva partición. El proceso completo se muestra en la ilustración 40.

Luego, se corrobora que se haya creado correctamente la nueva partición y, como se puede apreciar en la ilustración 41, se ha establecido con el nombre “vdb1”.

```
[root@fedora ~]# fdisk /dev/vdb
Bienvenido a fdisk (util-linux 2.37.2).
Los cambios solo permanecerán en la memoria, hasta que decida escribirlos.
Tenga cuidado antes de utilizar la orden de escritura.

El dispositivo no contiene una tabla de particiones reconocida.
Se ha creado una nueva etiqueta de disco DOS con el identificador de disco 0x84ae3f0c.

Orden (m para obtener ayuda): p
Disco /dev/vdb: 953,67 MiB, 1000000512 bytes, 1953126 sectores
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0x84ae3f0c

Orden (m para obtener ayuda): n
Tipo de partición
  p  primaria (0 primaria(s), 0 extendida(s), 4 libre(s))
  e  extendida (contenedor para particiones lógicas)
Seleccionar (valor predeterminado p): p
Número de partición (1-4, valor predeterminado 1):
Primer sector (2048-1953125, valor predeterminado 2048):
Último sector, +/-sectores o +/-tamaño{K,M,G,T,P} (2048-1953125, valor predeterminado 1953125):

Crea una nueva partición 1 de tipo 'Linux' y de tamaño 952,7 MiB.

Orden (m para obtener ayuda): w
Se ha modificado la tabla de particiones.
Llamando a ioctl() para volver a leer la tabla de particiones.
Se están sincronizando los discos.
```

Ilustración 40: Paso 1 - Crear partición en el nuevo volumen

```
[root@fedora ~]# lsblk -f
NAME FSTYPE FSVER LABEL        UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
sr0
zram0
vda
├─vda1
│   ext4  1.0          fb32932-2458-4b44-8fdf-cc974c34e701  735,1M   18% /boot
├─vda2
│   btrfs   fedora_localhost-live 316261cc-dfb9-44cb-857c-9ada41034b51  5,7G    34% /home
vdb
├─vdb1
```

Ilustración 41: Verificación de la partición creada en el nuevo volumen

Paso 2 - Formatear la partición del nuevo volumen

Una vez creada la partición, hay que formatearla para poder utilizarla, esto se efectúa creando un sistema de archivos de la misma manera que se realiza en la ilustración 42. Se le indica la partición a formatear (/dev/vdb1) y con la opción “-t”, el tipo de formato que se le va a asignar a la partición, el cual es el formato **ext4**.

```
[root@fedora ~]# mkfs -t ext4 /dev/vdb1
mke2fs 1.46.3 (27-Jul-2021)
Descartando los bloques del dispositivo: hecho
Se está creando un sistema de ficheros con 261888 bloques de 4k y 65536 nodos-i
UUID del sistema de ficheros: e2251eb4-5af1-4cdc-89ab-176047f7081f
Respaldos del superbloque guardados en los bloques:
    32768, 98304, 163840, 229376

Reservando las tablas de grupo: hecho
Escribiendo las tablas de nodos-i: hecho
Creando el fichero de transacciones (4096 bloques): hecho
Escribiendo superbloques y la información contable del sistema de ficheros: hecho
```

Ilustración 42: Paso 2 - Agregar formato a la partición del nuevo volumen

A pesar de haber formateado la partición, para poder hacer uso de ella y almacenar cualquier tipo de datos, hay que crear un punto de montaje en la máquina donde se almacenarán

los archivos. De modo que, primero se creará un directorio y se montará en dicha partición y luego, se creará un fichero para comprobar que se puede almacenar datos en el nuevo disco de la máquina virtual.

Paso 3 - Crear y montar un directorio en la partición del nuevo volumen

El nuevo directorio creado se llama “particion” y su ruta absoluta es “/root/particion”, la cual se usará para realizar el proceso de montaje sobre la partición (/dev/vdb1). En la ilustración 43 se puede ver la creación del directorio y el montaje realizado.

```
[root@fedora ~]# mkdir particion
[root@fedora ~]# ls -l
total 4
-rw-----. 1 root root 480 abr 28 10:35 anaconda-ks.cfg
drwxr-xr-x. 1 root root  0 abr 29 13:11 particion
[root@fedora ~]# mount /dev/vdb1 /root/particion
```

Ilustración 43: Paso 3 - Crear y montar nuevo directorio

Después de montar la carpeta, se comprueba que la partición se haya formateado según el formato indicado en el paso 2 y que el punto de montaje se haya realizado con éxito:

```
[root@fedora ~]# lsblk -f
NAME FSTYPE FSVER LABEL          UUID                                FSAVAIL FSUSE% MOUNTPOINTS
sr0
zram0
vda
├─vda1
│   ext4  1.0                fbf32932-2458-4b44-8fdf-cc974c34e701  735,1M   18% /boot
├─vda2
│   btrfs                fedora_localhost-live
│   316261cc-dfb9-44cb-857c-9ada41034b51  5,7G    34% /home
│   /
vdb
├─vdb1
│   ext4  1.0                e2251eb4-5af1-4cdc-89ab-176047f7081f  921,2M   0% /root/particion
```

Ilustración 44: Verificación de la partición creada, su formato y su punto de montaje

Creación del fichero en la partición del nuevo disco

Después de efectuar todos los pasos anteriores para preparar el volumen como almacenamiento, hay que cerciorarse de que se pueden almacenar datos en el nuevo disco de la máquina virtual. Esto se logra creando un fichero de prueba en el directorio que ha sido montado el disco, justo como se indica en la ilustración 45.

```
[root@fedora ~]# cd particion
[root@fedora particion]# vi fichero_prueba.txt
[root@fedora particion]# cat fichero_prueba.txt
Esto es una prueba
[root@fedora particion]#
```

Ilustración 45: Paso 4 - Creación del fichero en el nuevo volumen

En este momento, se puede concluir que, el nuevo volumen creado en el contenedor que proporciona KVM por defecto, se ha asignado correctamente a la máquina virtual y opera tal y como se esperaba.

1.1.2. Creación y asociación con virsh

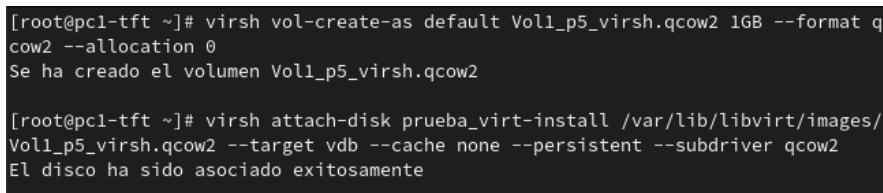
Este apartado consiste en desarrollar los mismos pasos que en el apartado anterior, pero ahora, desde la línea de comandos. Para crear un nuevo volumen con virsh, se ejecuta el siguiente comando:

```
[root@pc1-tft ~]# virsh vol-create-as default Vol1_p5_virsh.qcow2 1
→ GB --format qcow2 --allocation 0
```

El comando anterior, crea un nuevo volumen de 1 GB en el contenedor por defecto de KVM, pero no se asigna a ninguna máquina virtual, por lo tanto, aún no está disponible para su uso. De la misma manera que se realizó con la herramienta “*virt-manager*”, para que este nuevo volumen sea utilizado, hay que asociarlo a una máquina virtual ejecutando el siguiente comando:

```
[root@pc1-tft ~]# virsh attach-disk prueba_virt-install
→ /var/lib/libvirt/images/Vol1_p5_virsh.qcow2 --target vdb
→ --cache none --persistent --subdriver qcow2
```

La ejecución de la orden anterior, ha asociado el nuevo volumen creado con virsh a la máquina “prueba_virt-install”, creada únicamente para este propósito. En la ilustración 46 se muestra la ejecución de los comandos empleados y el mensaje de salida que han proporcionado como resultado, el cual es, en ambos casos, de éxito.



```
[root@pc1-tft ~]# virsh vol-create-as default Vol1_p5_virsh.qcow2 1GB --format qcow2 --allocation 0
Se ha creado el volumen Vol1_p5_virsh.qcow2

[root@pc1-tft ~]# virsh attach-disk prueba_virt-install /var/lib/libvirt/images/Vol1_p5_virsh.qcow2 --target vdb --cache none --persistent --subdriver qcow2
El disco ha sido asociado exitosamente
```

Ilustración 46: Creación y asignación de un nuevo volumen a una nueva máquina virtual

Se puede verificar que se haya creado y asociado correctamente el nuevo volumen a la máquina virtual desde la interfaz gráfica “*virt-manager*”, comprobando que la máquina virtual disponga de dos discos virtuales (o volúmenes) asociados: el que se asocia a la máquina virtual al crearse y el que se ha creado recientemente. Además, se realizan los mismos **Pasos para preparar el volumen** y verificar que se puede crear un fichero en dicho volumen.

1.2. Creación de una partición lógica en el anfitrión

En esta segunda parte, se creará una nueva partición lógica en el host anfitrión de 1 GB de capacidad en la partición extendida del disco. El objetivo de esta es ofrecer a un sistema

invitado (máquina virtual), la posibilidad de trabajar directamente sobre una partición del disco físico del sistema anfitrión.

Existen 3 tipos de particiones en un disco duro: particiones primarias, extendidas y lógicas. La **partición primaria** es reconocida como la partición de arranque y puede tener un sistema operativo que se encargue del arranque del sistema. La **partición extendida** actúa como una primaria y puede contener varias particiones lógicas en su interior.

En un disco duro físico, pueden existir 4 particiones primarias como máximo o 3 particiones primarias y 1 extendida. Como el sistema anfitrión utilizado no dispone de una partición extendida, antes de proceder a generar la partición lógica, hay que crear la partición extendida.

Por lo cual, se comienza ejecutando el comando **fdisk** indicando el disco que se quiere particionar (/dev/sda) y se responden a las preguntas planteadas de la misma manera que se puede comprobar en la ilustración 47.

```
[root@pcl-tft ~]# fdisk /dev/sda
Bienvenido a fdisk (util-linux 2.37.4).
Los cambios solo permanecerán en la memoria, hasta que decida escribirlos.
Tenga cuidado antes de utilizar la orden de escritura.

Orden (m para obtener ayuda): n
Tipo de partición
  p  primaria (0 primaria(s), 0 extendida(s), 4 libre(s))
  e  extendida (contenedor para particiones lógicas)
Seleccionar (valor predeterminado p): e
Número de partición (1-4, valor predeterminado 1):
Primer sector (2048-976773167, valor predeterminado 2048):
Último sector, +/-sectores o +/-tamaño{K,M,G,T,P} (2048-976773167, valor predeterminado 976773167): +20G

Crea una nueva partición 1 de tipo 'Extended' y de tamaño 20 GiB.
Partición #1: contiene un ntfs en la firma.

¿Desea eliminar la firma? [S]í/[N]o: s

La firma se borrará mediante una orden de escritura.

Orden (m para obtener ayuda): p
Disco /dev/sda: 465,76 GiB, 500107862016 bytes, 976773168 sectores
Modelo de disco: WDC WD5000AAKX-0
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0xb57b0d4e

Disposit.  Inicio Comienzo   Final Sectores Tamaño Id Tipo
/dev/sda1      2048 41945087 41943040    20G  5 Extendida

Se va a borrar la firma del sistema de ficheros/RAID en la partición 1.
```

Ilustración 47: Creación de la partición extendida en el sistema anfitrión

A resumidas cuentas, se introduce la letra ‘n’ para crear una nueva partición. A continuación, se introduce la letra ‘e’ para indicar el tipo de partición extendida y se pulsa la tecla ‘Enter’ en las dos preguntas siguientes para elegir los valores que se ofrecen por defecto del número de partición y del tamaño del primer sector. Para concluir, se añade el tamaño del último sector (20 GB).

El siguiente paso es el punto central de este apartado, crear la partición lógica en el sistema anfitrión a partir de una partición extendida. Para llevar a cabo esto, solo basta con

introducir de nuevo la letra ‘n’ para crear una nueva partición. Seguidamente, se introduce la letra ‘l’, ya que ahora, la partición que se va a crear en la partición extendida (recién creada) es lógica. En la siguiente pregunta se pulsa la tecla ‘Enter’ para elegir el valor por defecto del tamaño del primer sector y, en la última pregunta, se añade el tamaño del último sector (1 GB), es decir, el tamaño total que tendrá dicha partición. Para concluir, se escribe ‘w’ para confirmar la creación de la partición extendida y de la partición lógica.

Acto seguido, en la ilustración 48, se muestran todos los pasos mencionados. Cabe destacar que, en dicha ilustración se muestra toda la información de las particiones creadas: se identifica la partición extendida con el dispositivo `/dev/sda1` y la partición lógica (de tipo Linux) con el dispositivo `/dev/sda5`.

```
Orden (m para obtener ayuda): n
Tipo de partición
  p  primaria (0 primaria(s), 1 extendida(s), 3 libre(s))
  l  lógica (la numeración empieza por 5)
Seleccionar (valor predeterminado p): l

Se añade la partición lógica 5
Primer sector (4096-41945087, valor predeterminado 4096):
Último sector, +/-sectores o +/-tamaño{K,M,G,T,P} (4096-41945087, valor predeterminado 41945087): +1G

Crea una nueva partición 5 de tipo 'Linux' y de tamaño 1 GiB.

Orden (m para obtener ayuda): p
Disco /dev/sda: 465,76 GiB, 500107862016 bytes, 976773168 sectores
Modelo de disco: WDC WD5000AAKX-0
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0xb57b0d4e

Disposit.  Inicio Comienzo   Final Sectores Tamaño Id Tipo
/dev/sda1      2048 41945087 41943040    206  5 Extendida
/dev/sda5      4096 2101247 2097152     16 83 Linux

Se va a borrar la firma del sistema de ficheros/RAID en la partición 1.

Orden (m para obtener ayuda): w
Se ha modificado la tabla de particiones.
Llamando a ioctl() para volver a leer la tabla de particiones.
Se están sincronizando los discos.
```

Ilustración 48: Creación de la partición lógica en el sistema anfitrión

Tras finalizar el proceso de creación de ambas particiones, hay que reiniciar el sistema anfitrión para que se apliquen correctamente los cambios realizados.

Después de reiniciar el sistema, se debe crear el fichero de configuración del disco físico en formato XML con las características plasmadas en la ilustración 49. Como se mencionó anteriormente, la partición lógica que se ha creado es `/dev/sda5`, por lo tanto, es esta la que hay que indicar en el fichero. La directiva “*target*” será `vdc` ya que en la máquina virtual donde se va a asociar (“`prueba_virt-install`”) ya existen otros dos discos (`vda` y `vdb`).

```

1 <disk type='block' device='disk'>
2   <driver name='qemu' type='raw' cache='none' />
3   <source dev='/dev/sda5' />
4   <target dev='vdc' bus='virtio' />
5 </disk>

```

Ilustración 49: Creación del fichero de configuración del disco

Al igual que en los apartados anteriores, hay que asociar el nuevo disco creado a la máquina virtual llamada “prueba_virt-install”. Por lo tanto, se asocia el disco a la máquina virtual con la utilidad “virsh” mediante la terminal del anfitrión, indicando el nombre de la máquina virtual y el fichero del disco físico de la misma manera que se ve en la ilustración 50, desde la ruta donde está almacenado dicho fichero o también, con su ruta absoluta.

```

[root@pcl-tft ~]# cd /etc/libvirt/qemu
[root@pcl-tft qemu]# ls -l
total 56
-rw-r--r--. 1 root root 163 jun  1 10:43 disco_fisico.xml
drwx-----. 3 root root 4096 abr  8 14:04 networks
-rw-----. 1 root root 5930 abr 28 11:22 prueba_clone_virt-clone.xml
-rw-----. 1 root root 5954 abr 29 09:39 prueba_clone_virt-manager-clone.xml
-rw-----. 1 root root 5936 abr 28 11:13 prueba_clone_virt-manager.xml
-rw-----. 1 root root 5909 abr 28 11:06 prueba_conbackup.xml
-rw-----. 1 root root 6368 abr 29 13:29 prueba_virt-install.xml
-rw-----. 1 root root 6587 abr 29 12:58 prueba_virt-manager.xml
[root@pcl-tft qemu]# virsh attach-device --config prueba_virt-install disco_fisico.xml
El dispositivo fue asociado exitosamente

```

Ilustración 50: Asociación del disco a la máquina virtual

Para comprobar que se puede almacenar datos en este nuevo volumen, hay que realizar los “[Pasos para preparar el volumen](#)” que se han efectuado anteriormente en otros apartados.

1.3. Creación de un contenedor en una partición lógica del disco del anfitrión

En este tercer apartado, se comenzará creando otra partición lógica en el sistema anfitrión y se seguirá construyendo un contenedor de almacenamiento en dicha partición. El objetivo de esta demostración es crear un contenedor en una partición lógica del sistema anfitrión que se utilice para almacenar volúmenes (discos virtuales) para las máquinas virtuales.

Este apartado se efectuará con la utilidad “virt-manager” y a través del *shell* con las órdenes: *virsh pool-define* y *virsh pool-build* y *virsh pool-autostart*.

1.3.1. Creación con virt-manager

Con lo cual, se procede a crear una partición lógica, en este caso, de 2 GB y posteriormente, un sistema de archivos con un formato de tipo **ext4**. Cuando se haya efectuado esto,

se creará un contenedor llamado “Contenedor_Particion” y un nuevo volumen llamado “Vol1” de 1 GB que después se asociará a la máquina virtual.

Se utiliza de nuevo la utilidad `fdisk` para realizar una partición de `/dev/sda`. Al igual que en la partición construida en el apartado anterior, los pasos a seguir son:

1. Se introduce la letra ‘n’ para crear una nueva partición.
2. Se introduce la letra ‘l’ para crear la partición de tipo lógica.
3. Se pulsa la tecla ‘Enter’ para elegir el valor por defecto del tamaño del primer sector.
4. Se añade el tamaño del último sector (2 GB), lo que indica el tamaño total que tendrá la partición.
5. Se introduce la letra ‘w’ para confirmar la creación de la partición lógica.

A continuación, se corrobora que se ha creado correctamente la nueva partición `/dev/sda6` haciendo uso del comando `lsblk` como figura en la ilustración 51. Se puede comprobar las particiones que se han llevado a cabo anteriormente, la partición extendida que corresponde al nombre `/dev/sda1` y las particiones lógicas que cuelgan de esta: `/dev/sda5` y la recién creada, `/dev/sda6`.

```
[root@pcl-tft ~]# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
sda   8:0    0 465,8G 0 disk
├─sda1 8:1    0    1K 0 part
├─sda5 8:5    0    1G 0 part
└─sda6 8:6    0    2G 0 part
sdb   8:16   0 223,6G 0 disk
├─sdb1 8:17   0    46G 0 part
├─sdb2 8:18   0 600M 0 part /boot
├─sdb3 8:19   0  474M 0 part
├─sdb4 8:20   0    1K 0 part
├─sdb5 8:21   0   98G 0 part /
└─sdb6 8:22   0   10G 0 part /home
sr0   11:0    1 1024M 0 rom
zram0 252:0   0    8G 0 disk [SWAP]
```

Ilustración 51: Verificación de la nueva partición en el sistema anfitrión

Tras crear la partición, se formatea a un sistema de archivos con un formato de tipo `ext4` como figura en la ilustración 52.

```
[root@pcl-tft ~]# mkfs -t ext4 /dev/sda6
mke2fs 1.46.3 (27-Jul-2021)
Se está creando un sistema de ficheros con 524288 bloques de 4k y 131072 nodos-i
UUID del sistema de ficheros: 379eb5ac-212f-4fbc-ab2c-0fcbf806bdf
Respaldos del superbloque guardados en los bloques:
    32768, 98304, 163840, 229376, 294912

Reservando las tablas de grupo: hecho
Escribiendo las tablas de nodos-i: hecho
Creando el fichero de transacciones (16384 bloques): hecho
Escribiendo superbloques y la información contable del sistema de ficheros: 0/1
hecho
```

Ilustración 52: Agregar formato `ext4` a la partición en el sistema anfitrión

En este punto, ya se puede crear el contenedor y posteriormente, el volumen de prueba. En este caso, se realizará mediante la utilidad “*virt-manager*”.

Para crear el contenedor, desde el menú superior de la herramienta gráfica, se selecciona la pestaña “Editar” y se accede a “Detalles de conexión”. Una vez se haya abierto la ventana emergente de los “Detalles de conexión”, se selecciona en el menú superior, la pestaña “Almacenamiento” y se hace clic en el icono “+” que se encuentra en la parte inferior izquierda de dicha ventana. En consecuencia, se presenta otra ventana para llevar a cabo la creación del contenedor, donde se le indica el nombre que tendrá (Contenedor_Particion), se elige el tipo de contenedor que será, que en este caso se escoge el tipo “fs” y finalmente, se añade la ruta fuente desde donde se creará el contenedor, que en este caso es la partición /dev/sda6 y se finaliza el proceso. Respecto a la ruta absoluta de destino del nuevo contenedor y al formato, se mantienen los valores que se encuentran por defecto (**Ver ilustración 53**).

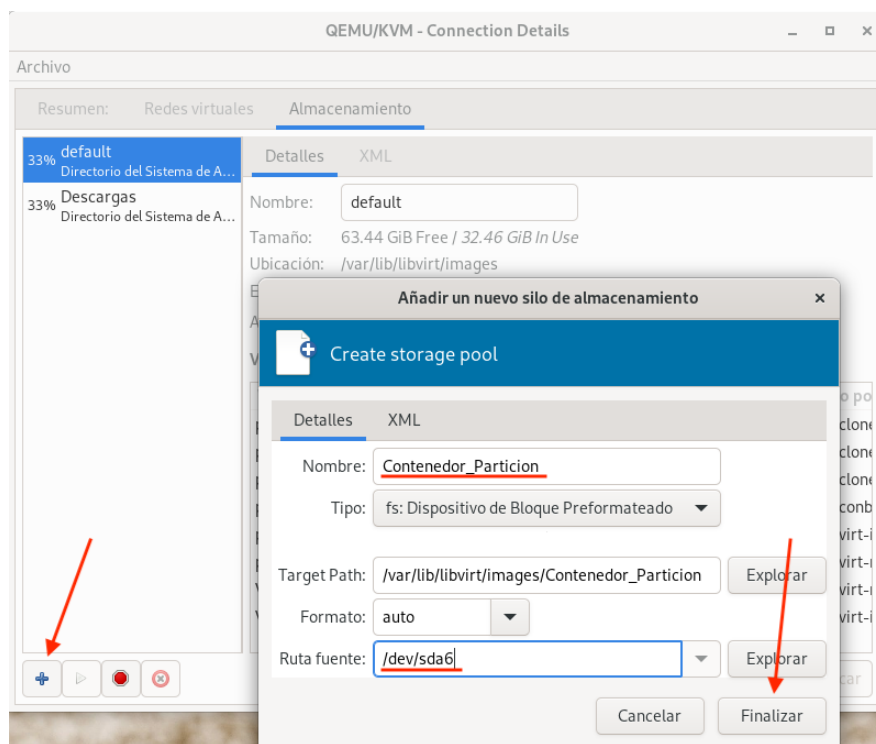


Ilustración 53: Crear un nuevo contenedor sobre una partición lógica

De manera similar y desde la ventana “Detalles de la conexión”, se selecciona el contenedor recién creado para crear un volumen en él y se hace clic en el icono “+” que se encuentra al lado del campo “Volúmenes” (**Ver ilustración 54**). Seguidamente, se abre una ventana donde se deberá indicar ciertos parámetros para la creación de dicho volumen.

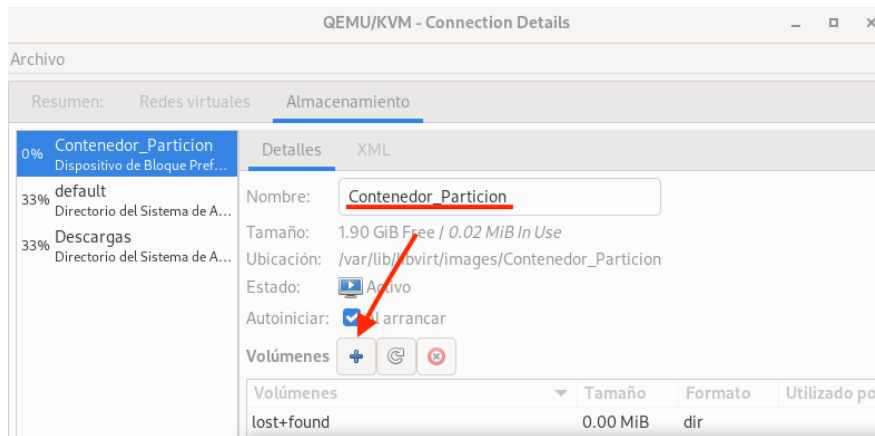


Ilustración 54: Creación de un nuevo volumen en el Contenedor_Particion

Del mismo modo que en apartados anteriores, se crea el volumen indicando el nombre (Vol1) y la capacidad máxima (1 GB) que tendrá (**Ver ilustración 55**).

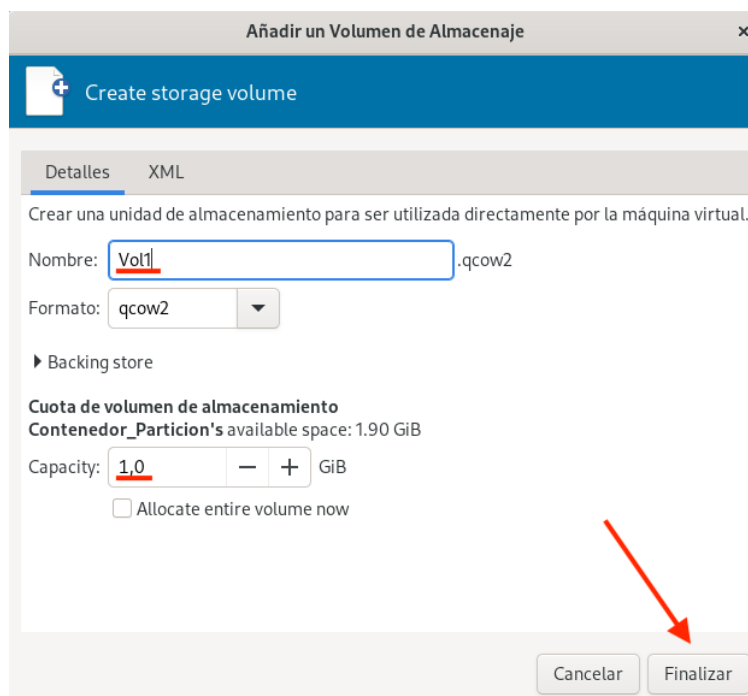


Ilustración 55: Añadir el nuevo volumen al contenedor

Ahora solo queda asociar este nuevo volumen a la máquina virtual, esto se hace desde el menú superior eligiendo la pestaña “Editar” y seleccionando “Detalles de la máquina virtual”. Una vez iniciada la ventana, se selecciona la opción “Agregar hardware” que se encuentra en la zona inferior izquierda para añadir o asociar el volumen a la máquina virtual de prueba. Desde aquí, se añade el nuevo volumen indicando la ruta absoluta de donde se encuentra, se comprueba que el tipo de bus sea *VirtIO* y se finaliza el proceso de asociación (**Ver ilustración 56**).

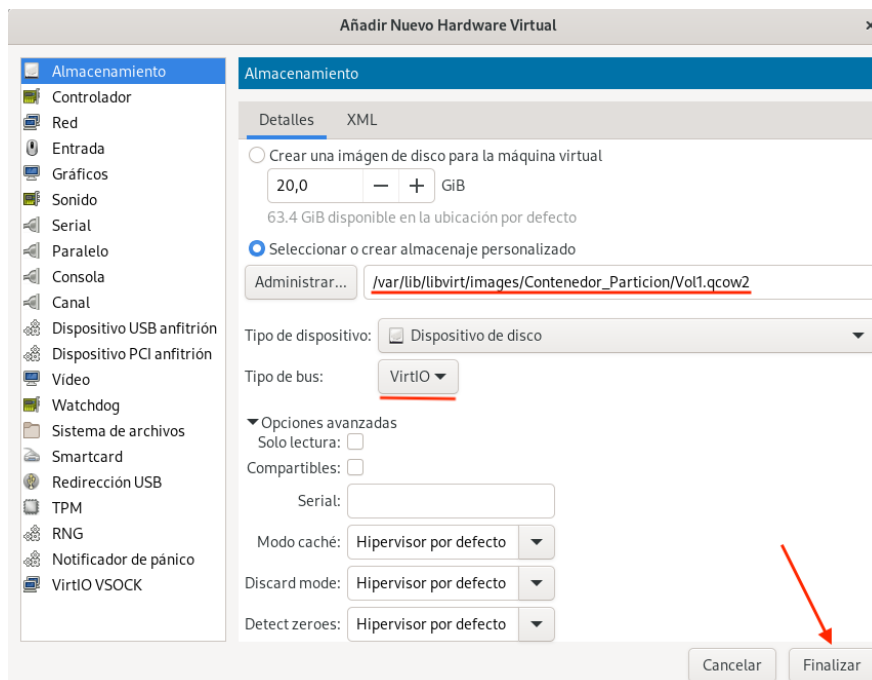


Ilustración 56: Asociar el volumen a la máquina virtual de prueba

Para demostrar ahora que se puede almacenar datos en este volumen, se lleva a cabo lo que se ha ido realizando en todos los apartados anteriores; crear una nueva partición en la máquina virtual, formatearla y montarla (**Pasos para preparar el volumen**).

Si se desea que el nuevo volumen (o disco virtual) asociado a la máquina, se monte automáticamente en el arranque de la máquina, hay que modificar el fichero **fstab** (que se encuentra en el directorio `/etc`), añadiendo el dispositivo (partición o recurso), el punto de montaje (carpeta), el sistema de archivos, las opciones donde se especifican los parámetros para montar el dispositivo y las opciones donde se indica si se harán respaldos del sistema de archivos o revisiones de la partición en busca de errores [53]. Por ejemplo, si se quiere montar el volumen recién creado en una carpeta para que cada vez que se inicie el sistema se pueda acceder a esta sin tener que realizar el montaje manualmente, el fichero se editaría como se presenta a continuación:

```
/dev/vdb1 /montaje_prueba ext4 defaults 0 0
```

En este caso, como la máquina virtual donde se ha asociado dicho volumen solo tenía el volumen principal (`/dev/vda`), el nombre del nuevo disco virtual es `/dev/vdb` y por ende, el nombre de la partición será `/dev/vdb1`.

El dispositivo se puede identificar por el nombre de la partición (`/dev/vdb1`), por la etiqueta o por el UUID del sistema de archivos de dicha partición. Para averiguar el UUID del sistema de archivos de la partición se puede ejecutar cualquiera de los dos comandos siguientes:

```
[root@fedora ~]# lsblk --fs /dev/vdb1
```

```
[root@fedora ~]# blkid /dev/vdb1
```

Asimismo, para identificar la etiqueta de la partición (si tiene) se puede ejecutar el segundo comando.

Existe una gran variedad de sistemas de archivos que puede utilizar el dispositivo para montar directorios, los más comunes son; ext4, nfs, ntfs, xfs, gfs2, smb, auto, etc. En cuyo caso, se ha seleccionado el de **ext4** porque es el sistema de archivos principal de cualquier sistema operativo que está basado en Linux.

Al igual que los sistemas de archivos, también existen varias opciones para realizar dicho montaje. Se ha seleccionado *defaults* porque asigna las opciones de montaje por defecto para el sistema de archivos escogido.

Para probar que el montaje automático funciona, se puede reiniciar el sistema (*reboot*) o ejecutar la orden **mount -a** para que se recargue el *fstab* y se monte todo lo que está en el fichero sin reiniciar el sistema.

1.3.2. Creación con virsh

Para la realización de este apartado, se ha creado otra partición lógica en el sistema anfitrión para crear el contenedor a partir de esta con la utilidad “*virsh*”. El nombre de esta nueva partición es **/dev/sda7**.

Con la partición creada en el anfitrión, el siguiente paso es crear el contenedor. Esta demostración se realizará mediante la línea de comandos, con la orden “*virsh*”, pero para que esto funcione, hay que crear un fichero XML con las características necesarias para crear dicho contenedor. Estas características varían según el tipo de contenedor, en este caso serán las mínimas necesarias para que el contenedor se cree correctamente: el tipo de contenedor (fs), el nombre, el dispositivo (partición), el formato del dispositivo y la ruta fuente del contenedor donde se va a montar (**Ver ilustración 57**).

Ilustración 57: Elaboración del fichero XML para la creación del contenedor

Una vez se haya creado el fichero con las especificaciones necesarias para crear el contenedor, se accede al directorio donde se encuentra dicho fichero y se ejecutan las tres órdenes que figuran en la ilustración 58. La primera se encarga de definir un nuevo contenedor de almacenamiento, la segunda crea el contenedor y la última, activa el inicio automático del contenedor.

```
[root@pc1-tft Documentos]# virsh pool-define Contenedor_Particion_Virsh.xml
El grupo Contenedor_Particion_Virsh ha sido definido desde Contenedor_Particion_Virsh.xml

[root@pc1-tft Documentos]# virsh pool-build Contenedor_Particion_Virsh
El pool Contenedor_Particion_Virsh ha sido compilado

[root@pc1-tft Documentos]# virsh pool-autostart Contenedor_Particion_Virsh
El grupo Contenedor_Particion_Virsh ha sido marcado como iniciable automáticamente
```

Ilustración 58: Definir, construir e iniciar el Contenedor_Particion_Virsh

Con esto, ya estaría creado el nuevo contenedor. El siguiente paso es crear un nuevo volumen y asociarlo a una máquina, y como ya se ha comentado anteriormente, para poder utilizar el volumen hay que realizar los [Pasos para preparar el volumen](#).

2. Contenedores en red

2.1. Creación de un contenedor NFS de imágenes ISO

En este apartado, se crea un contenedor de tipo NFS (desde el host anfitrión) para agregar imágenes de sistemas operativos para su instalación en máquinas virtuales. Esta demostración se realizará mediante la utilidad gráfica “*virt-manager*” y se creará una máquina virtual que simulará un servidor NFS.

2.1.1. Creación del simulador de un servidor NFS

1. Instalación del paquete `nfs-utils`

Entonces, se comienza clonando una máquina virtual de las ya existentes (creadas en anexos anteriores) e instalando en dicha máquina el paquete correspondiente para adquirir el servicio NFS. Por lo tanto, para instalar este paquete, hay que ejecutar la siguiente orden:

```
[root@fedora ~]# dnf install nfs-utils
```

En el caso de Fedora Workstation 35, este paquete ya se encuentra instalado cuando se instala el sistema operativo. Así pues, el siguiente paso es iniciar y habilitar los servicios `nfs-server` y `rpcbind` [?]:

```
[root@fedora ~]# systemctl enable --now nfs-server
```

```
[root@fedora ~]# systemctl enable --now rpcbind
```

Tras la ejecución de estas órdenes, hay que verificar que el estado de ambos servicios se encuentren tal y como se muestran en las ilustraciones 59 y 60, destacando el estado del servicio en *active* y el parámetro *enabled* activo.

```
[root@fedora ~]# systemctl status nfs-server
● nfs-server.service - NFS server and services
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; enabled; vendor preset: disabled)
   Active: active (exited) since Wed 2022-06-01 13:10:47 WEST; 15s ago
     Main PID: 2636 (code=exited, status=0/SUCCESS)
        CPU: 40ms

jun 01 13:10:47 fedora systemd[1]: Starting NFS server and services...
jun 01 13:10:47 fedora systemd[1]: Finished NFS server and services.
lines 1-8/8 (END)
```

Ilustración 59: Estado del servicio nfs-server

```
[root@fedora ~]# systemctl status rpcbind
● rpcbind.service - RPC Bind
   Loaded: loaded (/usr/lib/systemd/system/rpcbind.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2022-06-01 13:09:28 WEST; 12min ago
     TriggeredBy: ● rpcbind.socket
        Docs: man:rpcbind(8)
     Main PID: 2507 (rpcbind)
       Tasks: 1 (limit: 1103)
      Memory: 1.4M
         CPU: 26ms
       CGroup: /system.slice/rpcbind.service
              └─2507 /usr/bin/rpcbind -w -f

jun 01 13:09:28 fedora systemd[1]: Starting RPC Bind...
jun 01 13:09:28 fedora systemd[1]: Started RPC Bind.
```

Ilustración 60: Estado del servicio rpcbind

Lo siguiente que hay que hacer es añadir los servicios **nfs**, **rpc-bind** y **mountd** al *firewall* para permitir a los clientes acceder a los recursos compartidos de NFS:

```
[root@fedora ~]# firewall-cmd --permanent --add-service nfs
success
[root@fedora ~]# firewall-cmd --permanent --add-service rpc-bind
success
[root@fedora ~]# firewall-cmd --permanent --add-service mountd
success
[root@fedora ~]# firewall-cmd --reload
success
```

Ilustración 61: Añadir los servicios necesarios para utilizar NFS como cliente

2. Creación de una carpeta de prueba

A modo de prueba, para comprobar más adelante el objetivo de este apartado, se crea la carpeta **/images** y se le cambian los permisos de la siguiente manera para permitir un correcto montaje posteriormente:

```
[root@fedora ~]# mkdir /images
[root@fedora ~]# chmod 777 /images
```

Una vez creada, se añade una imagen ISO en dicho directorio para verificar cuando se cree el contenedor, que se exporta correctamente todo el contenido de la carpeta:

```
[root@fedora ~]# cd /images/
[root@fedora images]# ls -l
total 0
-rw-r--r--. 1 root root 2009333760 jun  1 13:42 Fedora-Workstation-Live-x86_64-35-1.2.iso
[root@fedora images]#
```

Ilustración 62: Verificación del contenido del directorio `/images` en el simulador NFS

3. Cambio del hostname del simulador

Antes de seguir con la configuración del servicio NFS, se cambia el *hostname* de la máquina virtual mediante la terminal aplicando el siguiente comando:

```
[root@fedora ~]# hostnamectl set-hostname simulador-nfs.com
```

Después, se reinicia la máquina virtual para que se apliquen los cambios y se añade la siguiente línea al fichero `/etc/hosts`:

```
192.168.122.92 simulador-nfs.com
```

4. Configuración del fichero de exportación

Luego, se configura la exportación del servidor NFS modificando el fichero `/etc/exports` para permitir los directorios que se exportan a los *hosts* remotos indicados y con qué opciones [59]. En este caso, la información que hay que añadir a dicho fichero es la siguiente:

```
/images 10.13.15.47(rw,sync,no_root_squash,no_all_squash)
```

En esta situación, el *host* con dirección IP: 10.13.15.47 (**host anfitrión**) puede montar el directorio `/images` desde el servidor NFS simulado. Los permisos que tiene este host sobre el directorio exportado son de lectura/escritura y las escrituras son síncronas. El `no_root_squash` y `no_all_squash` permite al usuario conectado remotamente utilizar los privilegios de dicho usuario remoto.

Ahora, se exporta el directorio recién añadido al fichero `/etc/exports` con el siguiente comando:

```
[root@simulador-nfs ~]# exportfs -arv
```

Y por último, se reinician los servicios `rpcbind` y `nfs-server`:

```
[root@simulador-nfs ~]# systemctl restart rpcbind
[root@simulador-nfs ~]# systemctl restart nfs-server
```

Con esto, ya se puede comprobar que la carpeta puede ser montada en el sistema anfitrión de la siguiente manera:

```
[root@simulador-nfs ~]# mount -t nfs -o rw 192.168.122.92:/images
↪ /var/lib/libvirt/images/
```

Si el comando se ejecuta correctamente, es decir, si la carpeta `/images` se monta en la carpeta `/var/lib/libvirt/images`, no debería de imprimir ningún mensaje por pantalla, directamente se monta sin mostrar ningún mensaje de éxito.

2.1.2. Creación del contenedor de tipo NFS con virt-manager

Para crear el contenedor se utilizará la utilidad gráfica “*virt-manager*” y se creará del mismo modo que se realizó en “1.3. Creación de un contenedor en una partición lógica del disco del anfitrión.” Una vez se haya abierto la ventana emergente de los “Detalles de conexión” y se encuentre en la pestaña “Almacenamiento”, se hace clic en el icono “+”, que se encuentra en la parte inferior izquierda de dicha ventana, para llevar a cabo la creación del contenedor. Aquí, se le indica el nombre que tendrá (Contenedor_NFS_1), se elige el tipo de contenedor que será (netfs), se añade el nombre del equipo que ofrece el servidor NFS (el que se ha creado anteriormente - **simulador-nfs.com**) y finalmente, se añade la ruta fuente que, en este caso, es la carpeta que va a ser exportada de dicho servidor NFS (`/images`) y se finaliza el proceso (**Ver ilustración 63**).

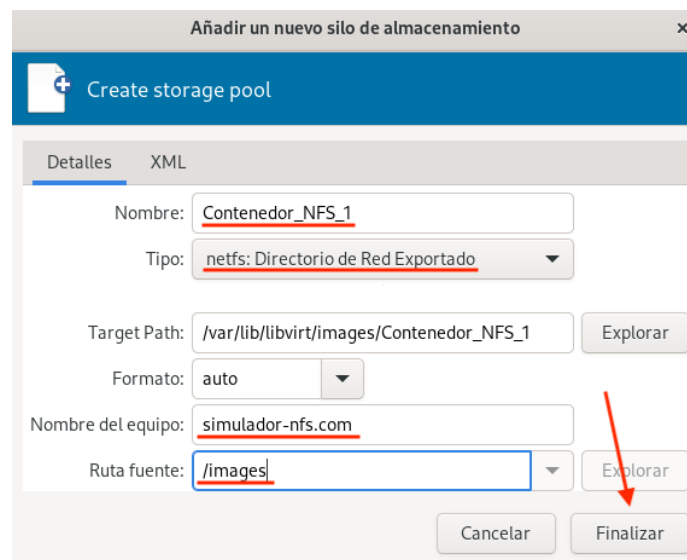


Ilustración 63: Creación del contenedor de tipo NFS

Una vez se finaliza el proceso de creación, el contenedor recién creado ya aparece disponible en la ventana “Detalles de la conexión” - “Almacenamiento” con la imagen ISO como parte del almacenamiento de este (**Ver ilustración 64**). Si se aprecia a la vez la ilustración 62, se puede determinar que las imágenes ISO de ambas ilustraciones coinciden.

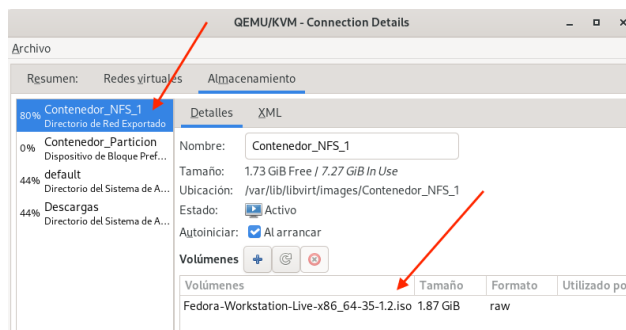


Ilustración 64: Verificación del contenido del contenedor

2.2. Creación de un contenedor NFS para volúmenes de máquinas virtuales

En esta última demostración, se crea otro contenedor de tipo NFS que funcione como contenedor de volúmenes de máquinas virtuales, ya que este es un escenario habitual de producción en entornos de virtualización, utilizar un servidor común de almacenamiento que proporciona espacio a las máquinas virtuales.

Esto se realizará mediante la máquina virtual creada anteriormente para simular un servidor NFS, simulador-nfs.com. Se utilizará de nuevo la herramienta virt-manager para la creación del contenedor.

2.2.1. Ajustes en el simulador NFS

Al igual que se hizo en el apartado anterior, se crea una nueva carpeta **/volumes** y se le cambian los permisos de la siguiente manera para permitir un correcto montaje posteriormente:

```
[root@simulador-nfs ~]# mkdir /volumes
[root@simulador-nfs ~]# chmod 777 /volumes
```

Seguidamente, se modifica de la misma manera el fichero **/etc/exports**, añadiendo otra línea igual pero para el otro directorio:

```
/images 10.13.15.47(rw,sync,no_root_squash,no_all_squash)
/volumes 10.13.15.47(rw,sync,no_root_squash,no_all_squash)
```

Acto seguido, se exporta el directorio recién añadido al fichero **/etc/exports**:

```
[root@simulador-nfs ~]# exportfs -arv
```

Y por último, se reinician los servicios **rpcbind** y **nfs-server**:

```
[root@simulador-nfs ~]# systemctl restart rpcbind
[root@simulador-nfs ~]# systemctl restart nfs-server
```

Si se quiere comprobar los directorios exportados o compartidos, se puede ejecutar el siguiente comando:

```
[root@psimulador-nfs ~]# showmount -e
```

2.2.2. Creación del contenedor de tipo NFS con virt-manager

Se crea el contenedor como se ha ido haciendo en los apartados anteriores, mediante la herramienta “*virt-manager*” y desde la ventana “Detalles de la conexión” - “Almacenamiento” que se accede haciendo clic en la pestaña “Editar”, que se encuentra en el menú superior.

Como se percibe en la ilustración 65, se le indica el nombre que tendrá (Contenedor_NFS_2), se selecciona el tipo de contenedor (netfs), se añade el nombre del equipo que ofrece el servidor NFS (simulador-nfs.com) y finalmente, se añade la ruta fuente que, en este caso, es la carpeta que va a ser exportada de dicho servidor NFS (/volumes) y se finaliza el proceso.

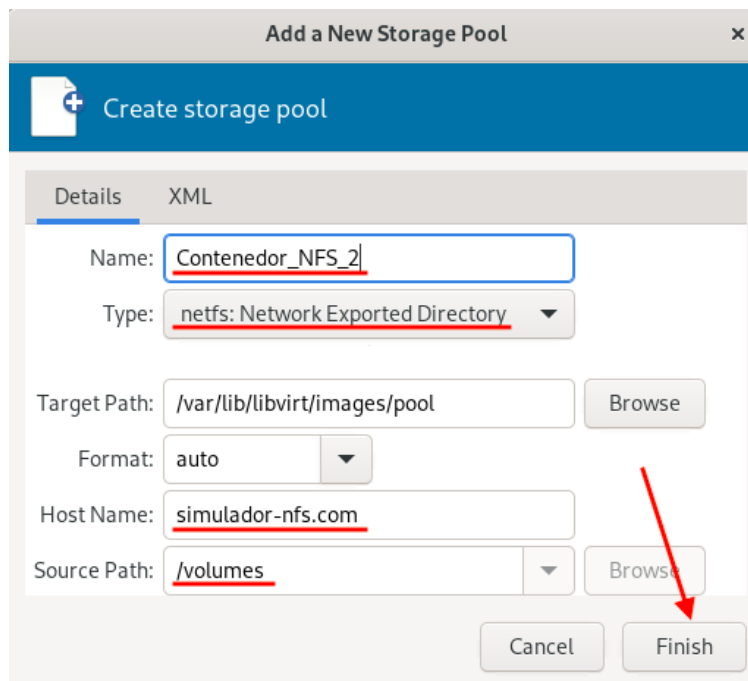


Ilustración 65: Creación del segundo contenedor de tipo NFS

Cuando se finaliza el proceso de creación, el segundo contenedor NFS creado ya aparece disponible en la ventana “Detalles de la conexión” - “Almacenamiento” (Ver ilustración 66).

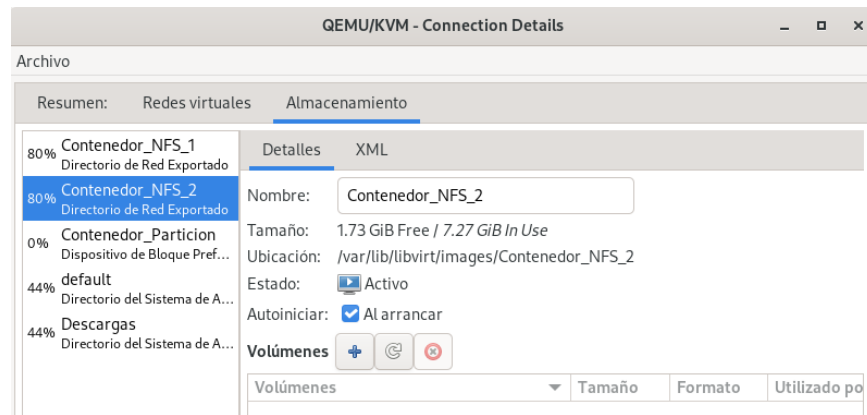


Ilustración 66: Verificación del segundo contenedor de tipo NFS

Ahora, se crea un nuevo volumen llamado “Vol_para_contenedor” y con una capacidad máxima de 1 GB, tal y como se indica en la ilustración 67:

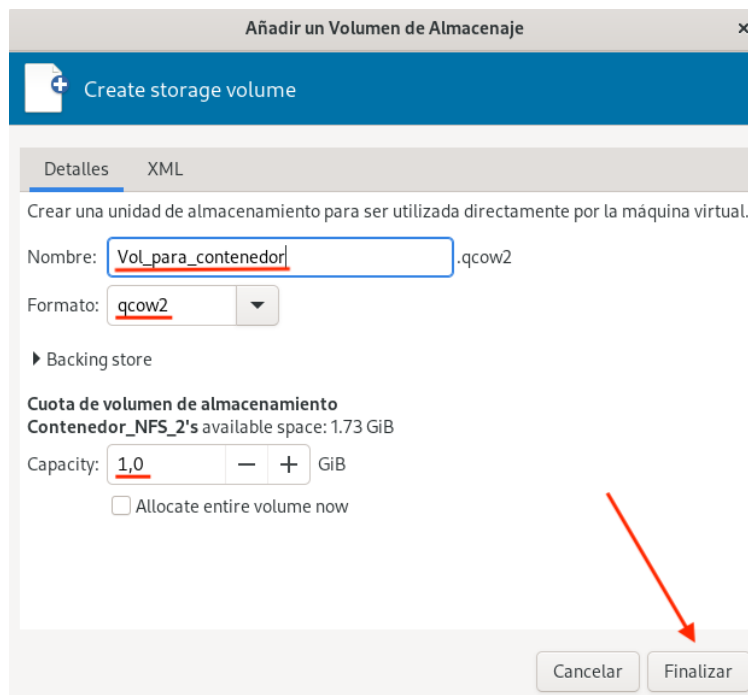


Ilustración 67: Creación de un volumen en el segundo contenedor NFS

Y por último, se verifica en la máquina virtual del simulador NFS si el volumen creado se ha almacenado en el directorio correspondiente (/volumes):

```
[root@simulador-nfs ~]# cd /volumes
[root@simulador-nfs volumes]# ls -l
total 324
-rw-----. 1 root root 1074135040 jun  2 11:49 Vol_para_contenedor.qcow2
[root@simulador-nfs volumes]#
```

Ilustración 68: Verificación del volumen en la máquina del simulador NFS

ANEXO VII: Creación de una red en modo *bridge*

Un *bridge* (o puente) es un dispositivo que opera en la capa de enlace (capa 2) reenviando el tráfico entre redes basándose en una tabla de direcciones MAC. Esta tabla la construye el puente escuchando el tráfico de la red y aprendiendo los hosts que están conectados a cada red. El puente interconecta dos segmentos de red pasando datos de una red hacia otra, con base en la dirección física de destino de cada paquete [60].

Por lo tanto, en este apartado, se creará un *bridge* en la interfaz física para integrar una máquina virtual a la misma red que el *host*. Para realizar esto, primero hay que crear el *bridge* (puente). Para configurar este tipo de configuración y cualquier otra, se utilizará la utilidad **nmcli** desde la línea de comandos mencionada en la sección “[Estado de la utilidad NetworkManager](#)”.

El primer paso es crear el puente en el host anfitrión, para ello, primero se crea una interfaz de red de tipo *bridge* y se visualiza todas las interfaces de red del equipo (**Ver ilustración 69**):

```
[root@pci-tft ~]# nmcli connection add type bridge con-name bridge0 ifname bridge0
Conexión «bridge0» (169aa0e7-eb7c-498c-a8b5-07db88444b20) añadida con éxito.
[root@pci-tft ~]# nmcli device status
```

DEVICE	TYPE	STATE	CONNECTION
eno0	ethernet	conectado	Conexión cableada 1
bridge0	bridge	conectando (obteniendo configuración IP)	bridge0
wlp0s29u1u1	wifi	no disponible	--
enp5s1	ethernet	sin gestión	--
lo	loopback	sin gestión	--

Ilustración 69: Creación de una interfaz de red tipo puente

Lo siguiente que hay que hacer es asignar la interfaz al puente, como esta interfaz aún no ha sido configurada se crea un nuevo perfil de conexión. Después de esto, tal y como se puede observa en la ilustración 70, se configuran las características IP del *bridge*, indicando la dirección IPv4 con la máscara de red, la puerta de enlace, los servidores de nombres de dominio (los de la ULPGC) y el dominio de búsqueda DNS de la conexión. Además, se le indica que el método de configuración es manual y se deshabilita el protocolo STP (Spanning Tree Protocol, en español, protocolo de árbol de expansión)⁷, ya que por defecto se encuentra activado.

```
[root@pci-tft ~]# nmcli connection add type ethernet slave-type bridge con-name bridge0-slave ifname eno0 master bridge0
Conexión «bridge0-slave» (3a65ee42-939d-4f2a-a11-8699d6034c7a) añadida con éxito.
[root@pci-tft ~]# nmcli connection modify bridge0 ipv4.addresses '10.13.15.47/24'
[root@pci-tft ~]# nmcli connection modify bridge0 ipv4.gateway '10.13.15.1'
[root@pci-tft ~]# nmcli connection modify bridge0 ipv4.dns '193.145.138.100 193.145.138.200'
[root@pci-tft ~]# nmcli connection modify bridge0 ipv4.dns-search 'ciber.ulpgc.es pci-tft.com'
[root@pci-tft ~]# nmcli connection modify bridge0 ipv4.method manual
[root@pci-tft ~]# nmcli connection modify bridge0 stp no
```

Ilustración 70: Configuración de la interfaz de red tipo puente

⁷Protocolo que garantiza la disponibilidad de las conexiones de red deshabilitando los enlaces redundantes.

A continuación, se activa la conexión y se revisa que el estado de la nueva interfaz *bridge* se encuentre “conectado” y que la columna “CONNECTION” aparezca el nombre de la conexión del puerto.

```
[root@pc1-tft ~]# nmcli connection up bridge0
La conexión se ha activado correctamente (master waiting for slaves) (ruta activa D-Bus: /org/freedesktop/NetworkManager/ActiveConnection/15)
[root@pc1-tft ~]# nmcli device
DEVICE    TYPE    STATE    CONNECTION
eno0      ethernet conectado Conexión cableada 1
bridge0   bridge  conectado bridge0
wlp0s29u1u1 wifi    no disponible --
enp5s1    ethernet sin gestión --
lo        loopback sin gestión --
```

Ilustración 71: Activación de la red y verificación del estado

En RHEL 8, cuando se inicia el sistema, se activa el controlador y los puertos. Como en este caso, solo se activa la conexión de puerto, hay que habilitar un parámetro de la conexión puente y reactivar el puente de la misma manera que como se aprecia en la ilustración 72:

```
[root@pc1-tft ~]# nmcli connection modify bridge0 connection.autoconnect-slaves 1
[root@pc1-tft ~]# nmcli connection up bridge0
La conexión se ha activado correctamente (master waiting for slaves) (ruta activa D-Bus: /org/freedesktop/NetworkManager/ActiveConnection/16)
```

Ilustración 72: Configuración de otros ajustes

Para verificar que la *bridge* se ha creado correctamente, se comprueba el estado del enlace del dispositivo *eno0* (Ethernet), el cual es el puerto del puente y se verifica que la interfaz *bridge* posee una dirección IP. Esta dirección IP, coincide con la dirección IP de la interfaz física (*eno0*) y ahora esta, no tiene ninguna asignada, por lo tanto, se puede decir que se han intercambiado la configuración (**Ver ilustración 73**).

```
[root@pc1-tft ~]# ip link show master bridge0
3: eno0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel master bridge0 state UP mode DEFAULT group default qlen 1000
    link/ether 00:22:4d:4d:e5:6c brd ff:ff:ff:ff:ff:ff
    altname enp0s25
[root@pc1-tft ~]# bridge link show
3: eno0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 master bridge0 state forwarding priority 32 cost 100
[root@pc1-tft ~]# ifconfig
bridge0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.13.15.47 netmask 255.255.255.0 broadcast 10.13.15.255
    inet6 fe80::9df4:c3e4:148b:a89f prefixlen 64 scopeid 0x20<link>
    ether 0e:57:13:a9:15:92 txqueuelen 1000 (Ethernet)
    RX packets 126 bytes 33818 (33.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 117 bytes 11990 (11.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eno0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    ether 00:22:4d:4d:e5:6c txqueuelen 1000 (Ethernet)
    RX packets 6504 bytes 3424856 (3.2 MiB)
    RX errors 0 dropped 12 overruns 0 frame 0
    TX packets 5209 bytes 905670 (884.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 20 memory 0xf3400000-f3420000
```

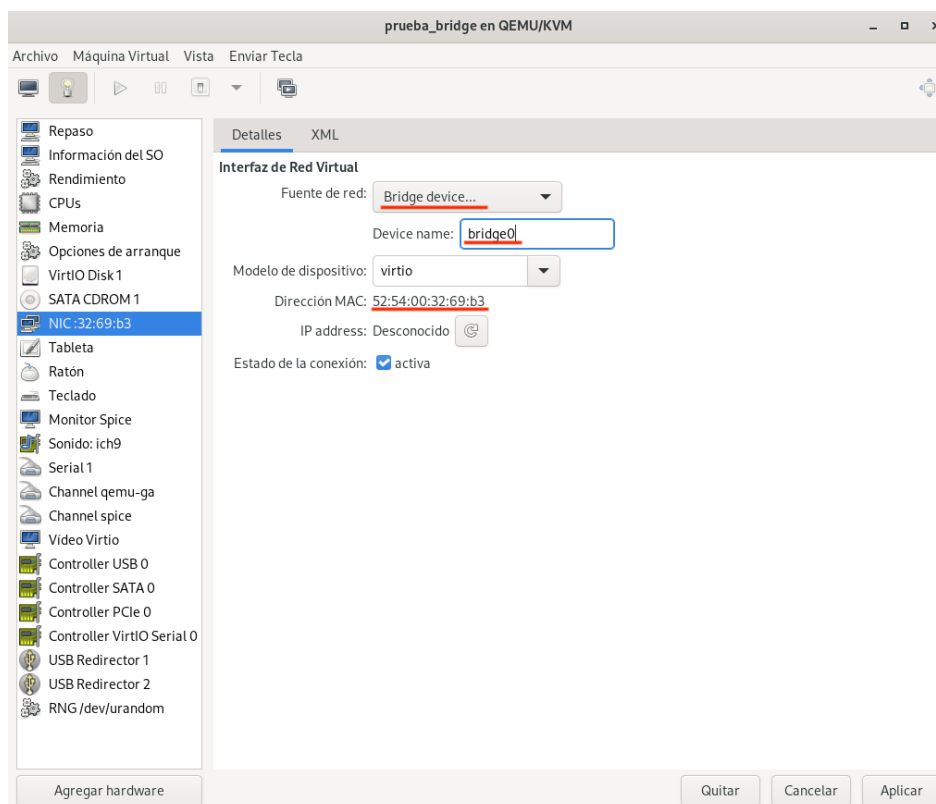
Ilustración 73: Verificación de la interfaz *bridge*

Para dar por terminada correctamente la configuración *bridge*, hay que verificar que la interfaz *bridge* tenga conexión a internet. Por lo cual, se verifica que el comando *ping* responda tanto por dirección IP como por nombres de dominio (**Ver ilustración 74**):

```
[root@pcl-tft ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=30.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=116 time=30.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=116 time=30.1 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=116 time=30.1 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 30.000/30.049/30.101/0.035 ms
[root@pcl-tft ~]# ping www.xataka.com
PING d2t8dj4tr3q9od.cloudfront.net (108.157.96.112) 56(84) bytes of data.
64 bytes from server-108-157-96-112.mad56.r.cloudfront.net (108.157.96.112): icmp_seq=1 ttl=245 time=30.2 ms
64 bytes from server-108-157-96-112.mad56.r.cloudfront.net (108.157.96.112): icmp_seq=2 ttl=245 time=30.2 ms
64 bytes from server-108-157-96-112.mad56.r.cloudfront.net (108.157.96.112): icmp_seq=3 ttl=245 time=30.2 ms
64 bytes from server-108-157-96-112.mad56.r.cloudfront.net (108.157.96.112): icmp_seq=4 ttl=245 time=30.1 ms
^C
--- d2t8dj4tr3q9od.cloudfront.net ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 30.127/30.186/30.210/0.034 ms
```

Ilustración 74: Verificación de la conexión a internet con la *bridge*

En este momento, ya se encuentra disponible la nueva interfaz puente en el sistema anfitrión. Por lo cual, el siguiente paso es añadir la red *bridge* a la máquina virtual, del mismo modo que se ha hecho cuando se han añadido los recursos de almacenamiento, con la utilidad “*virt-manager*”. Se selecciona la máquina a la que se le va a añadir la red, “prueba.bridge” y desde el menú superior, en la pestaña “Editar”, se accede a “Detalles de la máquina virtual” y se abre una ventana emergente. En ella, se hace clic en “Agregar hardware” y se selecciona el tipo de red “*Bridge device*”, se añade el nombre de la interfaz puente “*bridge0*” y se selecciona el modelo de dispositivo “*virtio*” (Ver ilustración 75).

Ilustración 75: Agregando la *bridge* a la máquina virtual

El siguiente paso es añadir la dirección MAC de la máquina virtual seleccionada en la ilustración 75, al router donde se encuentra conectado el sistema anfitrión. La dirección IP asignada a la máquina es la **10.13.15.80** y el *hostname* de la máquina que se ha indicado es **llariadna**. Como se puede apreciar, la red de dicha dirección IP coincide con la red del sistema anfitrión, cuya dirección IP es 10.13.15.47.

Tras esto, se comprueba si se ha asignado automáticamente los ajustes de la dirección IP a la máquina virtual:

```
[prueba@llariadna ~]$ nmcli device show
GENERAL.DEVICE:                enp1s0
GENERAL.TYPE:                  ethernet
GENERAL.HWADDR:                52:54:00:32:69:B3
GENERAL.MTU:                   1500
GENERAL.STATE:                 100 (conectado)
GENERAL.CONNECTION:            Conexión cableada 1
GENERAL.CON-PATH:              /org/freedesktop/NetworkManager/ActiveConnection/37
WIRED-PROPERTIES.CARRIER:     activado
IP4.ADDRESS[1]:                10.13.15.80/24
IP4.GATEWAY:                   10.13.15.1
IP4.ROUTE[1]:                  dst = 0.0.0.0/0, nh = 10.13.15.1, mt = 100
IP4.ROUTE[2]:                  dst = 10.13.15.0/24, nh = 0.0.0.0, mt = 100
IP4.DNS[1]:                    193.145.138.100
IP4.DNS[2]:                    193.145.138.200
IP4.DOMAIN[1]:                 ciber.ulpgc.es
IP6.ADDRESS[1]:                fe80::af76:e39a:fa64:16a0/64
IP6.GATEWAY:                   --
IP6.ROUTE[1]:                  dst = fe80::/64, nh = ::, mt = 100
```

Ilustración 76: Verificación de la dirección IP en la máquina virtual

Tal y como se puede ver en la ilustración 76, se ha añadido la dirección IP 10.13.15.80 a la interfaz **enp1s0**, la puerta de enlace (10.13.15.1), los DNS (servidores de nombres de dominios de la ULPGC) y la dirección MAC, que si se aprecia en la ilustración 75 se puede ver que son las mismas.

Finalmente, para comprobar que el puente funciona como debería, se comprueba que desde el exterior, se pueda llegar a esta máquina y que la máquina, tiene conectividad al exterior por direcciones IP y por resolución de nombres:

```
[prueba@llariadna ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=29.8 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=116 time=29.9 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=116 time=30.0 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=116 time=30.0 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 29.839/29.913/29.977/0.057 ms
[prueba@llariadna ~]$ ping www.xataka.com
PING d2t8dj4tr3q9od.cloudfront.net (108.157.96.78) 56(84) bytes of data:
64 bytes from server-108-157-96-78.mad56.r.cloudfront.net (108.157.96.78): icmp_
seq=1 ttl=245 time=30.2 ms
64 bytes from server-108-157-96-78.mad56.r.cloudfront.net (108.157.96.78): icmp_
seq=2 ttl=245 time=30.3 ms
64 bytes from server-108-157-96-78.mad56.r.cloudfront.net (108.157.96.78): icmp_
seq=3 ttl=245 time=30.4 ms
^C
--- d2t8dj4tr3q9od.cloudfront.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 30.237/30.291/30.371/0.057 ms
```

Ilustración 77: Verificación de la conectividad de la máquina hacia el exterior

ANEXO VIII: Instalación y configuración de la pila LAMP

Para instalar la pila LAMP en los nodos que ofrece el servicio Apache (**Nodo1-servicio** y **Nodo2-servicio**) desde los repositorios propios de Fedora, se recomienda actualizar los paquetes disponibles con el siguiente comando:

```
[root@nodo1-tft ~]# dnf -y update
```

Como ya se tiene instalado los servicios de Apache y MariaDB, se instala el repositorio Remi para PHP en **ambos nodos** de la siguiente manera:

```
[root@nodo1-tft ~]# dnf -y install  
→ https://rpms.remirepo.net/fedora/remi-release-35.rpm
```

Una vez instalado el repositorio Remi, se activa dicho repositorio en los **dos nodos** para poder instalar después el paquete de PHP con la versión preferida. Para activar el repositorio se ejecuta el comando posterior:

```
[root@nodo1-tft ~]# dnf config-manager --set-enabled remi
```

Seguidamente, se actualiza la información del repositorio volviendo a ejecutar el primer comando y se instalan los paquetes PHP que se necesiten. En este caso se instalarán los siguientes servicios de PHP especificados con la versión 7.4 en los nodos “**Nodo1-servicio**” y “**Nodo2-servicio**”:

```
[root@nodo1-tft ~]# dnf -y install php74-php-fpm php74-php-cli  
→ php74-php-mysqlnd
```

Cuando haya finalizado la instalación de PHP, se inicia y activa el servicio **php74-php-fpm** y se recarga el servicio Apache (httpd):

```
[root@nodo1-tft ~]# systemctl enable --now php74-php-fpm  
[root@nodo1-tft ~]# systemctl reload httpd
```

Se debe tener en cuenta que el servicio HTTP debe estar agregado en el *firewall*. Como en pasos anteriores se ha instalado el servicio Apache y configurado el *firewall* añadiendo dicho servicio en los nodos “**Nodo1-servicio**” y “**Nodo2-servicio**”, solo se debe verificar que se tenga abierto.

Es conveniente inspeccionar la versión recién instalada de PHP para verificar que esté activa, esto se puede realizar ejecutando el nuevo comando disponible cuando se ha instalado el paquete de PHP 7.4:


```
[root@nodo1-tft ~]# php74 -v
PHP 7.4.30 (cli) (built: Jun 7 2022 08:38:19) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
```

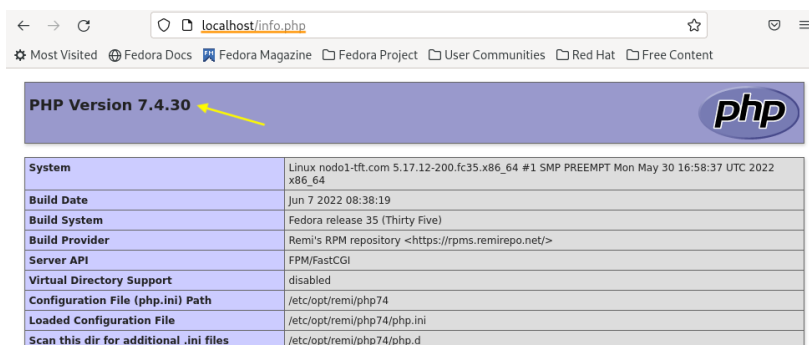
Ilustración 78: Versión de PHP instalado

A continuación, se prueba que funciona el entorno LAMP creando un script PHP en la ruta `/var/www/html/` para poder ser accedido a él desde el navegador de los nodos “**Nodo1-servicio**” y “**Nodo2-servicio**”. Al script se le ha denominado **info.php**, donde se agregará la línea que se aprecia en la ilustración 79:

```
[root@nodo1-tft ~]# cat /var/www/html/info.php
<?php phpinfo();
```

Ilustración 79: Información del fichero info.php

La creación del fichero se lleva a cabo en los dos nodos que ofrecen el servicio (**Nodo1-servicio** y **Nodo2-servicio**). Entonces, para comprobar el correcto funcionamiento del servidor LAMP, es decir, de los servicios de Apache, MariaDB y PHP y, al mismo tiempo, verificar la versión de PHP, se accede desde el navegador *Firefox* de los nodos a la siguiente ruta: **localhost/info.php** (Ver ilustración 80).



PHP Version 7.4.30	
System	Linux nodo1-tft.com 5.17.12-200.fc35.x86_64 #1 SMP PREEMPT Mon May 30 16:58:37 UTC 2022 x86_64
Build Date	Jun 7 2022 08:38:19
Build System	Fedora release 35 (Thirty Five)
Build Provider	Remi's RPM repository <https://rpms.remirepo.net/>
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/opt/remi/php74
Loaded Configuration File	/etc/opt/remi/php74/php.ini
Scan this dir for additional .ini files	/etc/opt/remi/php74/php.d

Ilustración 80: Comprobación del servidor LAMP

Para finalizar, se modifican cuatro parámetros del fichero de configuración de PHP **ini.php** en el directorio `/etc/opt/remi/php74/ini.php` de los nodos “**Nodo1-servicio**” y “**Nodo2-servicio**” para establecer la configuración exacta de la zona horaria y la configuración de la información en detalle de los posibles errores.

Por un lado, se busca la directiva `date.timezone`, se descomenta eliminando el carácter “;” y, en este caso, se le asigna el valor de la zona horaria de las Islas Canarias (Ver ilustración 81).

```
[Date]
; Defines the default timezone used by the date functions
; http://php.net/date.timezone
date.timezone = Atlantic/Canary
```

Ilustración 81: Primera parte de la modificación del fichero ini.php

Por otro lado, se buscan las tres directivas “*display_errors*”, “*display_startup_errors*” y “*error_reporting*” se les asigna el valor “*ON*” a las dos primeras y el valor “*E_ALL*” a la última (Ver ilustración 82).

```
display_errors = On
; Default Value: On
; Development Value: On
; Production Value: Off

display_startup_errors = On
; Default Value: Off
; Development Value: On
; Production Value: Off

error_reporting = E_ALL
; Default Value: E_ALL & ~E_NOTICE & ~E_STRICT & ~E_DEPRECATED
; Development Value: E_ALL
; Production Value: E_ALL & ~E_DEPRECATED & ~E_STRICT
```

Ilustración 82: Segunda parte de la modificación del fichero ini.php

Estas directivas se han activado para obtener información en detalle de los posibles errores que se produzcan para así poder ayudar a solventarlos rápidamente.

Se guardan los cambios y se recarga el servicio PHP para aplicarlos en **ambos nodos**:

```
[root@nodo1-tft ~]# systemctl reload php74-php-fpm
```

ANEXO IX: Instalación de *WordPress*

Para comenzar con el proceso de instalación de *WordPress*, se accede desde el anfitrión o desde cualquier nodo, a la dirección IP **192.168.122.246** o al nombre de dominio **balanceador-tft.com** y rápidamente, se redirecciona a la *url* “**192.168.122.246/wp-admin/setup-config.php**”, la página principal de configuración del *WordPress*, donde se detalla la información necesaria a disponer para continuar.



Ilustración 83: Página inicial de configuración de WordPress

Desde la página principal de configuración, se hace clic en el botón “¡Vamos a ello!” para comenzar con la instalación. Se redirecciona a otra página donde se muestra un formulario a rellenar con datos necesarios sobre la conexión de la base de datos. Estos datos son los que se han creado en el apartado “**Creando una base de datos y un usuario para WordPress**” de la sección “Instalación y configuración de *WordPress*”: el nombre de la base de datos (wordpress), el nombre de usuario asociado (wordpress), la contraseña (passwordpress), el servidor de la base de datos (192.168.122.246) y el prefijo de tabla, que se deja la que se establece por defecto (**Ver ilustración 84**).

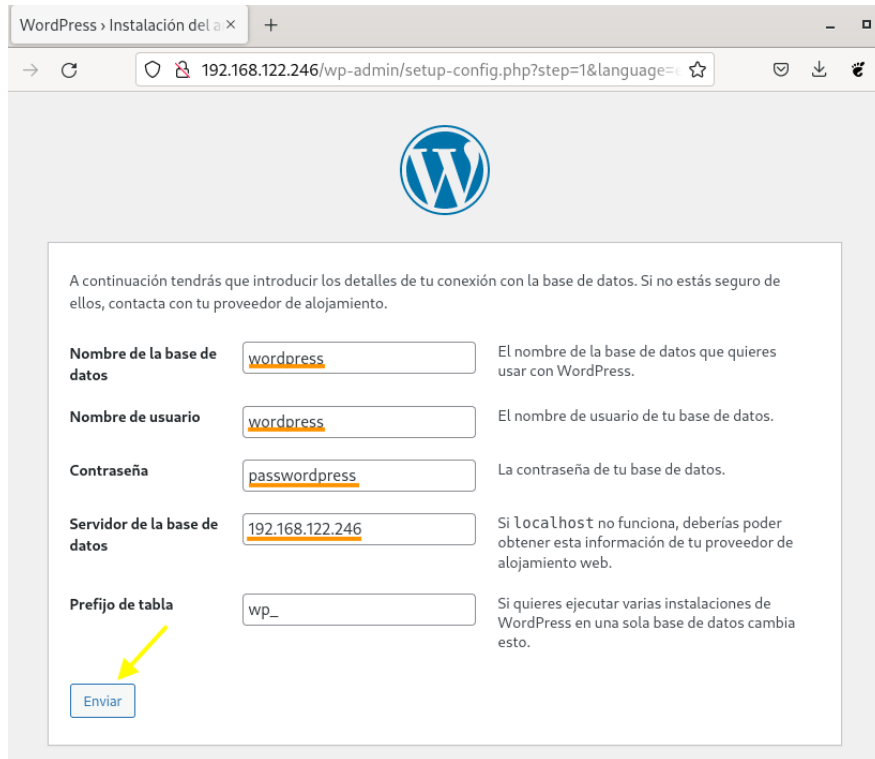


Ilustración 84: Paso 1 - Formulario sobre la conexión de la base de datos

Una vez rellenado el formulario, se hace clic en el botón “Enviar” y, en este punto, ya se ha terminado la primera parte de la configuración de instalación y se hace clic en el botón “Ejecutar la instalación” (**Ver ilustración 85**).



Ilustración 85: Paso 2 - Finalización de la primera parte de la instalación

La siguiente parte de la configuración de instalación es la incorporación de la información necesaria del sitio web a crear en *WordPress*. Los campos que se deben rellenar son: el título del sitio (Blog TFT), el nombre del usuario (admin), la contraseña y el correo electrónico. Además, si se requiere que el sitio web sea indexable por los robots de los motores de búsqueda, se marca la casilla del campo “Visibilidad en los motores de búsqueda” (en este caso no se

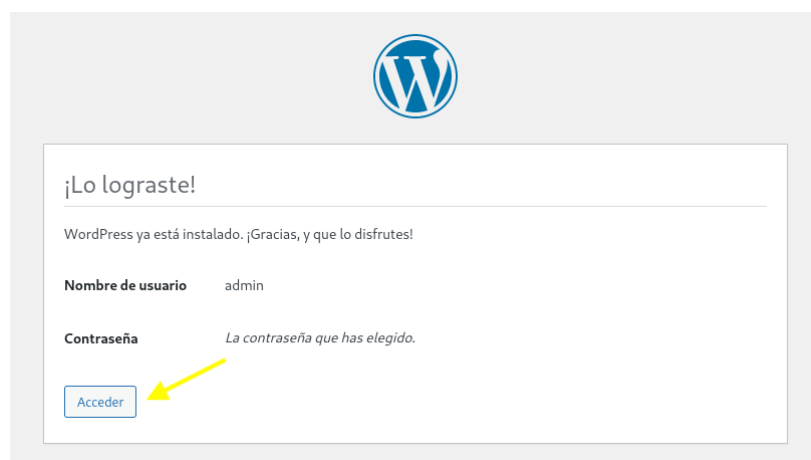
ha marcado). Finalmente, se hace clic en el botón “Instalar WordPress” para proceder con la instalación (**Ver ilustración 86**).



The screenshot shows the WordPress installation configuration page. At the top is the WordPress logo. Below it, a greeting "Hola" is followed by a welcome message: "¡Bienvenido al famoso proceso de instalación de WordPress en cinco minutos! Simplemente completa la información siguiente y estarás a punto de usar la más enriquecedora y potente plataforma de publicación personal del mundo." The section "Información necesaria" contains several fields: "Título del sitio" with the value "Blog TFT"; "Nombre de usuario" with the value "admin"; "Contraseña" with a masked password and a "Mostrar" button; and "Tu correo electrónico" with the value "ariadna.del103@alu.ulpc.es". There is also a checkbox for "Visibilidad en los motores de búsqueda" which is unchecked. At the bottom, a yellow arrow points to the "Instalar WordPress" button.

Ilustración 86: Paso 3 - Configuración de la información del sitio web

A tal efecto, *WordPress* ya se encuentra instalado en el “**Nodo2-servicio**”. Para probar que se puede acceder al panel de administración o al blog TFT (sitio web configurado que ofrecerá *WordPress*), se hace clic en el botón “Acceder” (**Ver ilustración 87**).



The screenshot shows the WordPress installation completion page. At the top is the WordPress logo. Below it, a message "¡Lo lograste!" is followed by "WordPress ya está instalado. ¡Gracias, y que lo disfrutes!". The "Nombre de usuario" is listed as "admin" and the "Contraseña" is listed as "La contraseña que has elegido.". At the bottom, a yellow arrow points to the "Acceder" button.

Ilustración 87: Paso 4 - Finalización de la instalación de *WordPress*

Se redirecciona a la página de inicio de sesión donde se debe ingresar el usuario recién creado y la contraseña para poder acceder al panel de administración de *WordPress* (Ver **ilustración 89**).



Ilustración 88: Paso 5 - Página de inicio de sesión de *WordPress*

Cuando se haya iniciado sesión, se iniciará la página oficial de administración de *WordPress* (Ver **ilustración 89**).

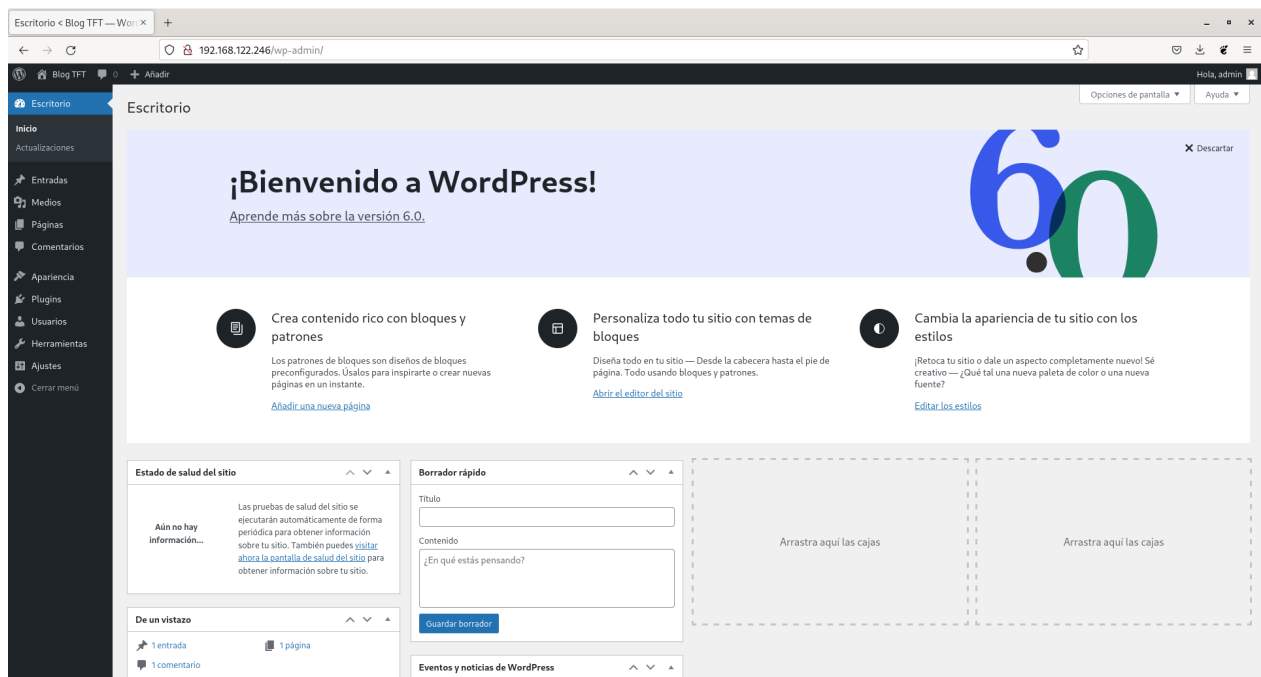


Ilustración 89: Página oficial de administración de *WordPress*