



Implantación de un CMS Liferay Portal CE en alta disponibilidad

Jorge Russell Domínguez

Grado de Ingeniería Informática en Tecnologías de la Información
Área de Administración de redes y sistemas operativos

Profesor Consultor/a:

David Bañeras Besora

Nombre Profesor/a responsable de la asignatura:

Mario Prieto Vega

Fecha Entrega: 14 de enero de 2024



Esta obra está sujeta a una licencia de Reconocimiento-NoComercial [3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc/3.0/es/)

FICHA DEL TRABAJO FINAL

Título del trabajo:	<i>Descripción del trabajo</i>
Nombre del autor:	<i>Jorge Russell Domínguez</i>
Nombre del consultor/a:	<i>Fernando Pérez López</i>
Nombre del PRA:	<i>Mario Prieto Vega</i>
Fecha de entrega (mm/aaaa):	01/2024
Titulación:	<i>Grado de Ingeniería Informática en Tecnologías de la Información</i>
Área del Trabajo Final:	<i>Administración de redes y sistemas operativos</i>
Idioma del trabajo:	<i>Castellano</i>
Palabras clave	<i>CMS, web, liferay</i>
Resumen del Trabajo (máximo 250 palabras): <i>Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.</i>	
<p>La finalidad de este TFG es la de llegar a implantar un CMS Liferay Portal CE en alta disponibilidad con todas sus funcionalidades de una instalación base para supuestos usuarios y que, además, dé escalabilidad y rendimiento y que disponga de un almacenamiento de datos que disponga de información coherente y confiable. También, se implantará un sistema de monitorización para el mismo que nos permita detectar la disponibilidad de los servicios dentro de la misma y su rendimiento para poder gestionar de forma eficiente los recursos de esta infraestructura.</p> <p>El contexto surge de la necesidad de la propia empresa en la que trabajo en la cual, como otras empresas, requieren las posibilidades que les ofrecen los CMS para permitir a los usuarios de esta, crear, editar y publicar contenido a través de una web de forma eficiente. Los CMS permiten a sus usuarios gestionar archivos digitales (imágenes, videos y documentos), personalizar dicho contenido, control de versiones, integración con sistemas externos (como un CRM, bases de datos o ERPs) y permiten la búsqueda de dicha documentación para facilitar a los usuarios encontrar documentación específica dentro de la web y la empresa está interesada en disponer de estas funcionalidades. No obstante, en este TFG, abordaremos la manera de hacerlo con software open-source.</p> <p>La metodología utilizada será la de Investigación-Acción, ya que se trata de, mediante la investigación previa para resolver una situación específica, aplicar dichos hallazgos de forma práctica.</p> <p>El resultado es la puesta en marcha de dicha plataforma concluyendo de forma exitosa.</p>	

Abstract (in English, 250 words or less):

The purpose of this TFG is to implement a CMS Liferay Portal CE in high availability with all its functionalities in a base installation for supposed users and that, in addition, provides scalability and performance and that has a data storage that has consistent and reliable information. In addition, a monitoring system will be implemented for the same that allows us to detect the availability of the services within it and its performance to efficiently manage the resources of this infrastructure.

The context arises from the need of the company in which I work in which, like other companies, require the possibilities offered by CMS to allow users to create, edit and publish content through a website efficiently. CMSs allow their users to manage digital files (images, videos and documents), customize such content, version control, integration with external systems (such as CRM, databases or ERPs) and allow the search of such documentation to make it easier for users to find specific documentation within the website and the company is interested in having these functionalities. However, in this TFG, we will address how to do it with open-source software.

The methodology used will be Action-Research, as it is about, through previous research to solve a specific situation, applying those findings in a practical way.

The result is the successful implementation of this platform.

Índice

1. Introducción.....	3
1.1 Contexto y justificación del Trabajo.....	3
1.2 Objetivos del Trabajo.....	3
1.3 Enfoque y método seguido.....	9
1.4 Planificación del Trabajo.....	9
1.5 Breve sumario de productos obtenidos.....	11
1.6 Breve descripción de los otros capítulos de la memoria.....	12
1.6.1 Análisis y estudio tecnológico.....	12
1.6.2 Preparación del entorno.....	12
1.6.3 Desplegar una base de datos gestionada.....	13
1.6.4 Despliegue del portal digital Liferay funcional.....	13
1.6.5 Alta Disponibilidad, Escalabilidad y Rendimiento mediante una arquitectura en clúster.....	13
1.6.6 Seguridad.....	14
1.6.7 Segmentación de la red.....	14
1.6.8 Gestión de Backups (copias de respaldo).....	14
1.6.9 Monitorización del servicio.....	15
1.6.10 Documentación de explotación de servicio.....	15
1.6.11 Verificación de cumplimiento de objetivos y requisitos.....	15
2. Análisis, desarrollo y despliegue de un CMS en Alta Disponibilidad.....	16
2.1 Análisis y estudio tecnológico:.....	16
2.1.1 Comparativa de tecnologías de posible aplicación.....	16
2.1.2 Requisitos finales de software y hardware.....	17
2.2 Preparación del entorno.....	20
2.2.1 Preparación de sistema de virtualización y máquina virtual plantilla.....	20
2.2.2 Despliegue de la máquina virtual y configuración del Apache Webserver.....	20
2.2.3 Despliegue de la máquina virtual y configuración del HAProxy.....	21
2.2.4 Despliegue de la máquina virtual para el MySQL y configuración del NFS.....	23
2.3 Despliegue de una base de datos gestionada MySQL.....	23
2.3.1 Despliegue y configuración del servidor MySQL.....	23
2.3.2 Configuración del cliente de MySQL.....	24
2.4. Despliegue del portal digital Liferay funcional.....	24
2.4.1 Instalación de las Java Runtime JDK.....	25
2.4.2 Instalación del paquete Liferay Tomcat.....	25
2.4.3 Configuración del Data Library de Liferay en el NFS.....	27
2.4.4 Instalación y configuración del servidor de búsqueda Elasticsearch.....	28
2.4.5 Configuración redirección de tráfico del servidor web Apache al Liferay.....	29
2.4.6 Comprobación del correcto funcionamiento del Portal Liferay.....	30
2.5 Alta Disponibilidad, Escalabilidad y Rendimiento mediante una arquitectura en clúster.....	33
2.5.1 Configuración de una nueva máquina web y de aplicativos en clúster.....	33

2.5.2 Configuración de balanceo de carga y persistencia de sesión en los servidores web Apache y Tomcat que ejecutan Liferay.....	34
2.5.3 Configuración en clúster de los servidores de búsqueda Elasticsearch	35
2.5.4 Configuración en clúster del Portal Liferay.....	38
2.5.5 Pruebas de Alta Disponibilidad del clúster del Portal Liferay	38
2.6 Seguridad	39
2.6.1 Configuración del servicio de Firewall (Cortafuegos).....	39
2.6.2 Configuración del servicio de HAProxy para gestionar tráfico web cifrado	39
2.7 Segmentación de la red.....	41
2.7.1 Asignación de una nueva interfaz de red a la máquina HAProxy ...	41
2.7.2 Cambio de IP frontal del servicio de HAProxy	42
2.8 Gestión de Backups (copias de respaldo).....	42
2.8.1 Copias de respaldo con la herramienta AutoMySQL Backup	43
2.8.2 Copias de respaldo de ficheros con la herramienta tar	43
2.9 Monitorización del servicio	44
2.9.1 Configuración de agentes de Prometheus de extracción de métricas.	45
2.9.2 Configuración del servidor de métricas Prometheus.....	46
2.9.2 Configuración del servidor de análisis de métricas Grafana	47
2.10 Documentación de explotación de servicio	50
2.11 Verificación de cumplimiento de objetivos y requisitos.....	53
2.11.1 Verificación de correcto funcionamiento de la plataforma.....	54
2.11.2 Verificación de cumplimiento de objetivos y requisitos.	54
3. Conclusiones.....	54
3.1 Lecciones aprendidas.....	54
3.2 Objetivos	54
3.2 Planificación, metodología y cambios realizados	55
3.3 Líneas de trabajo futuro y dificultades	55
4. Glosario.....	56
5. Bibliografía	57
6. Anexos	61
Anexo 1. Instalación de máquinas virtuales con VirtualBox	61
Anexo 2. Clonación de máquinas virtuales.....	63
Anexo 3. Asignación de IPs estáticas de máquinas virtuales.....	65
Anexo 4. Configuración del servicio de Firewall (cortafuegos).....	67
Anexo 5. Instalación y configuración de un servidor web Apache.....	68
Anexo 6. Instalación y configuración de HAProxy	70
Anexo 7. Configuración de sistema de eventos mediante Syslog	73
Anexo 8. Configuración de un sistema NFS (sistema de ficheros en red)	75
Anexo 9. Instalación y configuración de una base de datos gestionada MySQL	79
Anexo 10. Instalación y configuración de un cliente de bases de datos MySQL	82
Anexo 11. Instalación y configuración de Java Runtime JDK en Linux	83
Anexo 12. Instalación y configuración de Liferay	85
Anexo 13. Configuración del Data Library de Liferay	90
Anexo 14. Instalación y configuración de Elasticsearch en Liferay	94

Anexo 15. Configuración de redirecciones de servidores web Apache a Liferay	99
Anexo 16. Entradas de DNS locales	102
Anexo 17. Comprobación de logs de accesos web en Syslog y Apaches ..	103
Anexo 18: Creación de sitios web en Liferay.....	104
Anexo 19. Gestión de contenido dentro de Liferay.....	110
Anexo 20. Configuración de balanceo de carga y persistencia de sesión entre servidores web Apache y servidores Tomcat.....	111
Anexo 21. Configuración en clúster de servidores de búsqueda Elasticsearch	113
Anexo 22. Configuración en clúster del Portal Liferay	116
Anexo 23. Pruebas de Alta Disponibilidad.....	117
Anexo 24. Creación de un certificado SSL autofirmado	120
Anexo 25. Configuración de HAProxy con HTTPS.....	122
Anexo 26. Configuración de Liferay con HTTPS	123
Anexo 27. Configuración de Red NAT en VirtualBox	124
Anexo 28. Configuración de backups con la herramienta AutoMySQL Backup	125
Anexo 29. Configuración de Agente de Prometheus Node Exporter.....	128
Anexo 30. Configuración de Agente de Prometheus HAProxy Exporter	130
Anexo 31. Configuración de Agente de Prometheus Apache Exporter	132
Anexo 32. Configuración de Agente de Prometheus JMX Exporter	135
Anexo 33. Configuración de Agente de Prometheus MySQL Exporter	137
Anexo 34. Configuración de Servidor Prometheus.....	140
Anexo 35. Configuración de Servidor Grafana	142

Lista de figuras

Figura 1. Cronograma Pte1	10
Figura 2. Cronograma Pte.2	11
Figura 3. 2022 State of Open Source Report	18
Figura 4. Página web principal de servidor web Apache	21
Figura 5. Página web principal de Apache vista a través de HAProxy	22
Figura 6. Servicio MySQL ejecutándose	23
Figura 7. Cliente MySQL conectándose al servidor MySQL	24
Figura 8. Test de detección de Java por el sistema	25
Figura 9. Página de Wizard de configuración de Liferay	26
Figura 10. Página web principal de Liferay funcionando en local	27
Figura 11. Contenido de Liferay generado en NFS	28
Figura 12. Elasticsearch en Liferay	29
Figura 13. Web del Portal de Liferay	30
Figura 14. Sitio web de prueba creado en Liferay	31
Figura 15. Otro sitio web de prueba creado en Liferay	32
Figura 16. Contenido subido a Liferay	33
Figura 17. Comprobación de persistencia de sesión	35
Figura 18. Nodos de Elasticsearch detectados en Liferay	37
Figura 19. Comprobación de persistencia de sesión en nodo secundario	38
Figura 20. Certificado SSL de la web de Liferay	41
Figura 21. Backups de base de datos MySQL	43
Figura 22. Métricas de agentes de Prometheus	46
Figura 23. Web de monitorización de Prometheus	47
Figura 24. Panel de control de Node Exporter con gráficas de rendimiento de la máquina	48
Figura 25. Panel de control de Apache Exporter con gráficas de rendimiento del servidor web	48
Figura 26. Panel de control de JMX Exporter con gráficas de rendimiento de la máquina virtual Java	49
Figura 27. Panel de control de MySQL Exporter con gráficas de rendimiento de la base de datos	50
Figura 28. Diagrama de red de la infraestructura	53
Figura 29. Instalación de VirtualBox	61
Figura 30. Web de descarga de máquinas virtuales CentOS	62
Figura 31. Configuración de red en VirtualBox	62
Figura 32. Máquina virtual creada en VirtualBox	63
Figura 33. Clonación de máquinas virtuales en VirtualBox	64
Figura 34. Configuración de red de CentOS	65
Figura 35. Configuración de IP de CentOS	66
Figura 36. Servicio web Apache ejecutándose	68
Figura 37. Página web de prueba del servidor Apache	69
Figura 38. Servicio HAProxy levantado y ejecutándose	72
Figura 39. Servicio de Syslog corriendo y ejecutándose	73
Figura 40. Servicio NFS corriendo y ejecutándose	75
Figura 41. Servicio MySQL corriendo y ejecutándose	79
Figura 42. Versión de Java instalada	84
Figura 43. Log de Liferay indicando su correcta inicialización	86

Figura 44. Configuración inicial de Liferay _____	86
Figura 45. Configuración de base de datos de Liferay _____	87
Figura 46. Log de creación de tablas en base de datos de Liferay _____	88
Figura 47. Web inicial de Liferay _____	88
Figura 48. Web inicial de Liferay tras login de usuario _____	89
Figura 49. Panel de control de Liferay _____	91
Figura 50. Configuración de almacenamiento de archivos de Liferay _____	92
Figura 51. Parametrización almacenamiento archivos de Liferay _____	92
Figura 52. Exportación datos Liferay _____	93
Figura 53. Web de estado de Elasticsearch _____	96
Figura 54. Configuración servicio de búsquedas en Liferay _____	97
Figura 55. Elasticsearch en Liferay _____	98
Figura 56. Reindexación de contenido en Liferay con Elasticsearch _____	98
Figura 57. Ajustes de sitio de Liferay _____	101
Figura 58. Servidores virtuales en Liferay _____	101
Figura 59. Fichero etc/hosts de nuestro host anfitrión _____	102
Figura 60. Logs HAPorxy _____	103
Figura 61. Logs servidor web Apache _____	103
Figura 62. Panel de control de Liferay, creación de sitios web _____	104
Figura 63. Creación de sitios web en Liferay _____	104
Figura 64. Plantillas web de sitios en Liferay _____	105
Figura 65. Nuevo sitio web creado en Liferay _____	105
Figura 66. Sitio web de pruebas creado en Liferay _____	105
Figura 67. Sitio web creado en Liferay _____	106
Figura 68. Configuración de sitios web en Liferay _____	106
Figura 69. URL Amigable de sitio web en Liferay _____	107
Figura 70. Sitio web de prueba final en Liferay _____	108
Figura 71. Sitio web de prueba secundario generado en Liferay _____	109
Figura 72. Gestión de contenido dentro de Liferay _____	110
Figura 73. Contenido publicado dentro de Liferay _____	110
Figura 74. Comprobación de la persistencia de sesión en el navegador _____	117
Figura 75. Comprobación de la persistencia de sesión en el navegador tras cambiar de servidor _____	118
Figura 76. Red NAT VirtualBox _____	124
Figura 77. Nueva IP anfitrión VirtualBox _____	124

1. Introducción

1.1 Contexto y justificación del Trabajo

Actualmente en la empresa en la que estoy trabajando surge la necesidad de que, como otras empresas, requieren las posibilidades que les ofrecen los CMS¹ (del inglés Content Management System, que significa Sistema de Gestión de Contenido) para permitir a los usuarios de esta, crear, editar y publicar contenido a través de una web de forma eficiente, además de poder desarrollar sitios web específicos dentro de dicha infraestructura.

Es un tema relevante ya que los CMS¹ permiten a sus usuarios gestionar archivos digitales (imágenes, videos y documentos), personalizar dicho contenido, control de versiones, integración con sistemas externos (como un CRM, bases de datos o ERPs) permiten la búsqueda de dicha documentación para facilitar a los usuarios encontrar documentación específica dentro de la web y la empresa está interesada en disponer de estas funcionalidades para mejorar la línea de negocio entre otras.

Estos CMS² facilitan la creación de sitios web, documentación, blogs, wikis y demás tipos de contenido útiles tanto para la gestión de documentación interna de la propia empresa como para posibles clientes. Además, estos gestores permiten gestionar archivos digitales (imágenes, videos y documentos), personalizar dicho contenido, control de versiones, integración con sistemas externos³ (como un CRM, bases de datos o ERPs) y permite la búsqueda de dicha documentación para facilitar a los usuarios encontrar documentación específica dentro de la web.

La empresa pretende resolver la falta de un repositorio donde se pueda gestionar ese contenido de forma centralizada de todos los archivos digitales para su línea de negocio y que se pueda disponer además de las capacidades antes mencionadas. Sobre todo, le interesa la de poder crear sitios web dentro de la misma, además de tener capacidad de gestión de contenido.

La aportación que se realiza es la instalación de un CMS con esas capacidades como es Liferay Portal CE que permite, también, crear sitios web y, además, añadir capacidad de alta disponibilidad.

1.2 Objetivos del Trabajo

El objetivo final del trabajo es llegar a implantar el entorno del portal digital, el cual actuará como un CMS y gestor de sitios web con todas sus

funcionalidades de una instalación base para supuestos usuarios y que, además, tenga capacidad de alta disponibilidad que dé escalabilidad y rendimiento, con almacenamiento de datos (el NFS) compartido entre las máquinas del portal que tenga información coherente y confiable. Todo ello de forma completamente transparente para el usuario, es decir, si parte del entorno se cae, como un servidor web, la parte del aplicativo y/o de búsqueda, que el resto de la infraestructura pueda redirigir las llamadas a sus instancias en clúster y que el usuario no se vea afectado y pueda seguir trabajando y que, en caso de necesidad por aumento de demanda, se pueda redirigir la carga entre nodos repartiendo la carga. Además, implantaremos también un sistema de monitorización de la plataforma.

Los objetivos primarios serán los siguientes:

- a) Implantar y habilitar un portal digital Liferay Portal Community Edition (CE): poner en marcha una instancia en clúster del portal digital Liferay Portal CE que permita creación, gestión de contenido y sitios web gracias a su capacidad como CMS y de creación de sitios web mediante plantillas.
- b) Implementar un entorno de Alta Disponibilidad (HA): garantizar que dicha instancia esté disponible para los posibles usuarios, minimizando tiempos de indisponibilidad mediante el uso de un balanceador de carga, como es el HA Proxy, y la configuración en clúster de las máquinas virtuales que contendrán las instancias web Apache, de los servidores del aplicativo Tomcat con Liferay y las instancias de búsqueda de Elasticsearch.
- c) Conseguir mediante esta instalación escalabilidad y rendimiento: Se asegura, mediante este tipo de instalación en clúster de máquinas virtuales, que el sistema pueda manejar de forma eficaz más carga de trabajo y que sea escalable a futuro, es decir, se podrían instalar más máquinas paralelas con la misma configuración de tal manera que podrían agregarse al clúster fácilmente para aumentar el rendimiento de todo el sistema.
- d) Establecer un almacenamiento compartido y confiable: este sistema se puede conseguir usando un NFS (Network File System) entre las máquinas en clúster que ejecutan la parte web y del aplicativo antes mencionadas, de tal manera que se garantice la coherencia y la replicación de los datos que se vuelquen en él
- e) Implementación de una base de datos gestionada: Mediante la configuración de un gestor de bases de datos en una de las máquinas virtuales para garantizar la persistencia y el acceso a datos para el portal digital Liferay Portal CE.

Los objetivos parciales o secundarios serán los siguientes:

- a) Seguridad: Configurar políticas de seguridad en un Firewall (cortafuegos) para proteger la infraestructura y sus datos de posibles amenazas y ataques desde el exterior e instalar un certificado en la parte web para que el tráfico HTTP esté cifrado.
- b) Segmentación de la red: Dicha segmentación agregará una capa extra de seguridad, de aislamiento de los componentes y para una gestión eficiente de los recursos.
- c) Gestión de backups (copias de respaldo) de datos: Establecer algún tipo de procedimiento para realizar copias de seguridad de los datos para garantizar la integridad de los mismos y la continuidad del servicio en caso de cualquier fallo relacionado, de tal manera que se puedan recuperar los mismos en tiempo y forma con el mínimo impacto sobre el servicio.
- d) Monitorización del servicio: implementar alguna herramienta que nos permita monitorizar la infraestructura que nos permita detectar la disponibilidad de los servicios dentro de la misma y su rendimiento para poder gestionar de forma eficiente los recursos de esta infraestructura.
- e) Documentación: Adjuntar en este trabajo la documentación suficiente y necesaria que describa la infraestructura con detalle, junto con un esquema de la red, y que proporcione una capacitación al usuario para asegurarse de que se pueda administrar y mantener toda la plataforma de forma eficiente y efectiva.
- f) Cumplimiento de objetivos y requisitos: Asegurar que la infraestructura diseñada cumpla que los objetivos y requisitos establecidos al inicio del trabajo, así como, la satisfacción y eficiencia de la misma.

Desglose de las tareas asociadas:

- Tarea 1: Análisis y estudio tecnológico:

Descripción de la tarea:

-Comparativa y análisis de las posibilidades tecnológicas del software open-source existente basado en los requerimientos de la empresa, de la matriz de compatibilidad de Liferay y resto de software para poder desplegar la infraestructura. Dicho análisis también comprenderá el estudio de los requerimientos hardware (CPU, Memoria, espacio en disco, etc) de todo el software implicado para poder ser ejecutado sin problemas.

Objetivo de la tarea:

-Tomar la decisión del software y versiones exactas del software que vamos a utilizar para poder desplegar la infraestructura completa necesaria para que el portal digital Liferay pueda funcionar en base a las

especificaciones del trabajo, así como tener los requerimientos del hardware necesario para desplegar la infraestructura.

- Tarea 2: Preparación del entorno:

Descripción de la tarea:

- Configurar y desplegar las máquinas virtuales necesarias para los servidores web y de aplicación de Liferay.
- Configurar un balanceador de carga HAProxy en una de ellas para poder distribuir el tráfico entre las distintas máquinas virtuales que alojarán la parte web y aplicativo.
- Configurar el almacenamiento compartido mediante un NFS para los datos necesarios de Liferay de tal manera que se garantice la alta disponibilidad si uno de los nodos se cae.

Objetivo de la tarea:

- Tener la infraestructura web, balanceador, de aplicativo y de almacenamiento necesaria para poder desplegar el Liferay.

- Tarea 3: Desplegar una base de datos gestionada:

Descripción de la tarea:

- Configurar una base de datos en una máquina virtual separada.

Objetivo de la tarea:

- Tener una base de datos necesaria para que el Liferay pueda almacenar sus datos.

- Tarea 4: Desplegar un portal digital Liferay funcional:

Descripción de la tarea:

- Instalar y configurar los servidores web en las máquinas virtuales que redirigirán el tráfico a los servidores del aplicativo.
- Instalar y configurar los servidores de aplicación junto con el Liferay que trabajarán en modo de alta disponibilidad.
- Instalar y configurar los servidores de búsqueda necesarios para Liferay que también trabajarán en modo de alta disponibilidad.
- Configurar el Liferay contra la base de datos gestionada para desplegar el contenido necesario por el aplicativo.
- Hacer pruebas básicas de funcionamiento del portal Liferay generando y/o configurando contenido dentro del mismo.

Objetivo de la tarea:

-Disponer de un portal de Liferay desplegado y asegurar su correcto funcionamiento.

- Tarea 5: Alta Disponibilidad, Escalabilidad y Rendimiento mediante una arquitectura en clúster

Descripción de la tarea:

-Establecer un clúster entre las máquinas virtuales del aplicativo mediante la configuración de los servidores web, los servidores del aplicativo y los servidores de búsqueda, de tal manera que se reparta la carga de trabajo a la vez que se da una alta disponibilidad al sistema en caso de fallos de alguno de los nodos.

-Realizar pruebas para verificar el correcto funcionamiento del clúster, desactivando algunos nodos y verificando que el sistema sigue funcionando.

Objetivo de la tarea:

-Dotar a la plataforma de alta disponibilidad, de escalabilidad y de un rendimiento adecuado.

- Tarea 6: Seguridad

Descripción de la tarea:

-Configurar políticas de seguridad de firewall para proteger la infraestructura y la plataforma de posibles amenazas exteriores.

-Instalar un certificado en la parte de los servidores web para que el tráfico HTTP esté cifrado, es decir, que sea HTTPS.

-Modificar el HAProxy para aceptar también tráfico cifrado HTTPS.

Objetivo de la tarea:

-Garantizar la confidencialidad e integridad de los datos y la estabilidad y disponibilidad del sistema y del servicio mediante las políticas de seguridad de firewall y el uso de tráfico cifrado para acceder a los servidores web.

- Tarea 7: Segmentación de la red

Descripción de la tarea:

-Establecer una segmentación de la red, definiendo una DMZ y una red de aplicativo/backend, mediante la configuración de interfaces de red extra en las máquinas virtuales en otras subredes y configurando rutas entre las mismas.

-Configurar los servicios web, del aplicativo y del servicio de búsqueda en sus correspondientes subredes.

Objetivo de la tarea:

-Aumentar el aislamiento de servicios de tal manera que se añada una capa extra de seguridad en caso de ataque desde el exterior y agregar una gestión eficiente de los recursos.

- Tarea 8: Gestión de Backups (copias de respaldo)

Descripción de la tarea:

-Establecer un sistema de gestión de copias de seguridad de los datos

Objetivo de la tarea:

-Garantizar la integridad de los datos y la continuidad del servicio en caso de cualquier fallo relacionado, de tal manera que se puedan recuperar los mismos en tiempo y forma con el mínimo impacto sobre dicho servicio.

- Tarea 9: Monitorización del servicio

Descripción de la tarea:

-Instalar y configurar en el host (u opcionalmente en una de las máquinas virtuales) un sistema de monitorización de los servicios desplegados.

Objetivo de la tarea:

-Asegurar que el sistema es estable, fiable y que proporciona los servicios para los que ha sido diseñado.

- Tarea 10: Documentación

Descripción de la tarea:

- Crear una documentación que describa la infraestructura con detalle y que proporcione una capacitación al usuario para el uso de la misma tal como el plan de contingencia en caso de fallos del sistema, así como un diagrama de red de la infraestructura.

Objetivo de la tarea:

- Proporcionar una capacitación al usuario para asegurarse de que se pueda administrar y mantener toda la plataforma de forma eficiente y efectiva.

- Tarea 11: Cumplimiento de objetivos y requisitos

Descripción de la tarea:

- Verificar todo el funcionamiento del sistema mediante pruebas simulando posibles fallos y situaciones de tal manera que el sistema siga funcionando adecuadamente.

Objetivo de la tarea:

-Asegurar que la infraestructura diseñada cumpla los objetivos y requisitos establecidos al inicio del trabajo, así como, la satisfacción y eficiencia de esta.

1.3 Enfoque y método seguido

El enfoque y método utilizada será la de Investigación-Acción, ya que se trata de, mediante la investigación previa para resolver una situación específica, aplicar dichos hallazgos de forma práctica. Por tanto, la estrategia a seguir consistirá en investigar y evaluar las posibilidades de software open-source existentes para conseguir desplegar una infraestructura completa con un CMS con capacidad de generar sitios web.

Se considera que esta es la estrategia apropiada para conseguir los objetivos ya que otras opciones existentes son de pago o en cloud y requieren de pagos de uso por dichas infraestructuras.

1.4 Planificación del Trabajo

El plan de trabajo con la programación semanal de las tareas se ha realizado en <https://www.onlinegantt.com/>

Se adjuntan dicha programación en imagen en este documento y en un archivo PDF aparte.

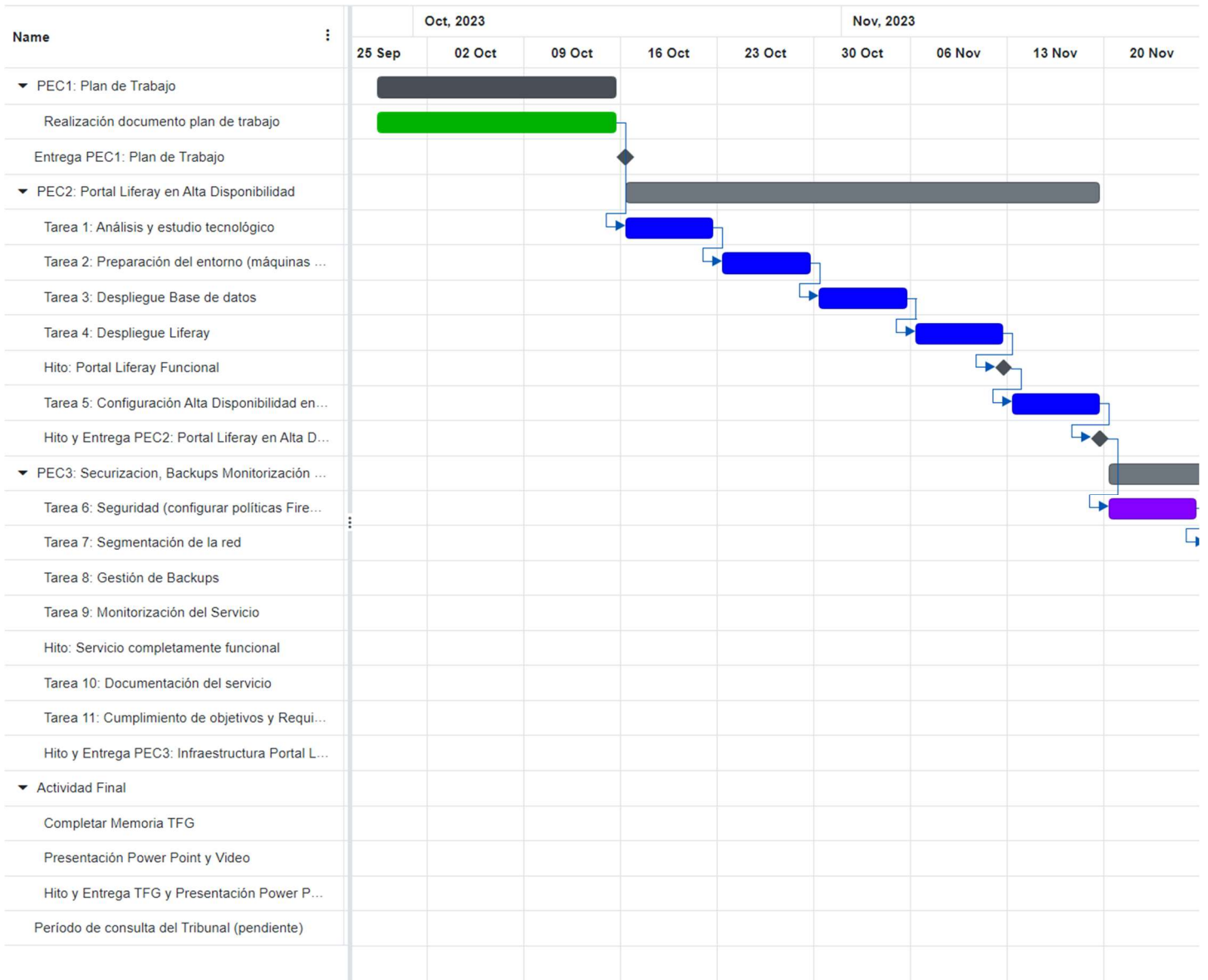


Figura 1. Cronograma Pte1

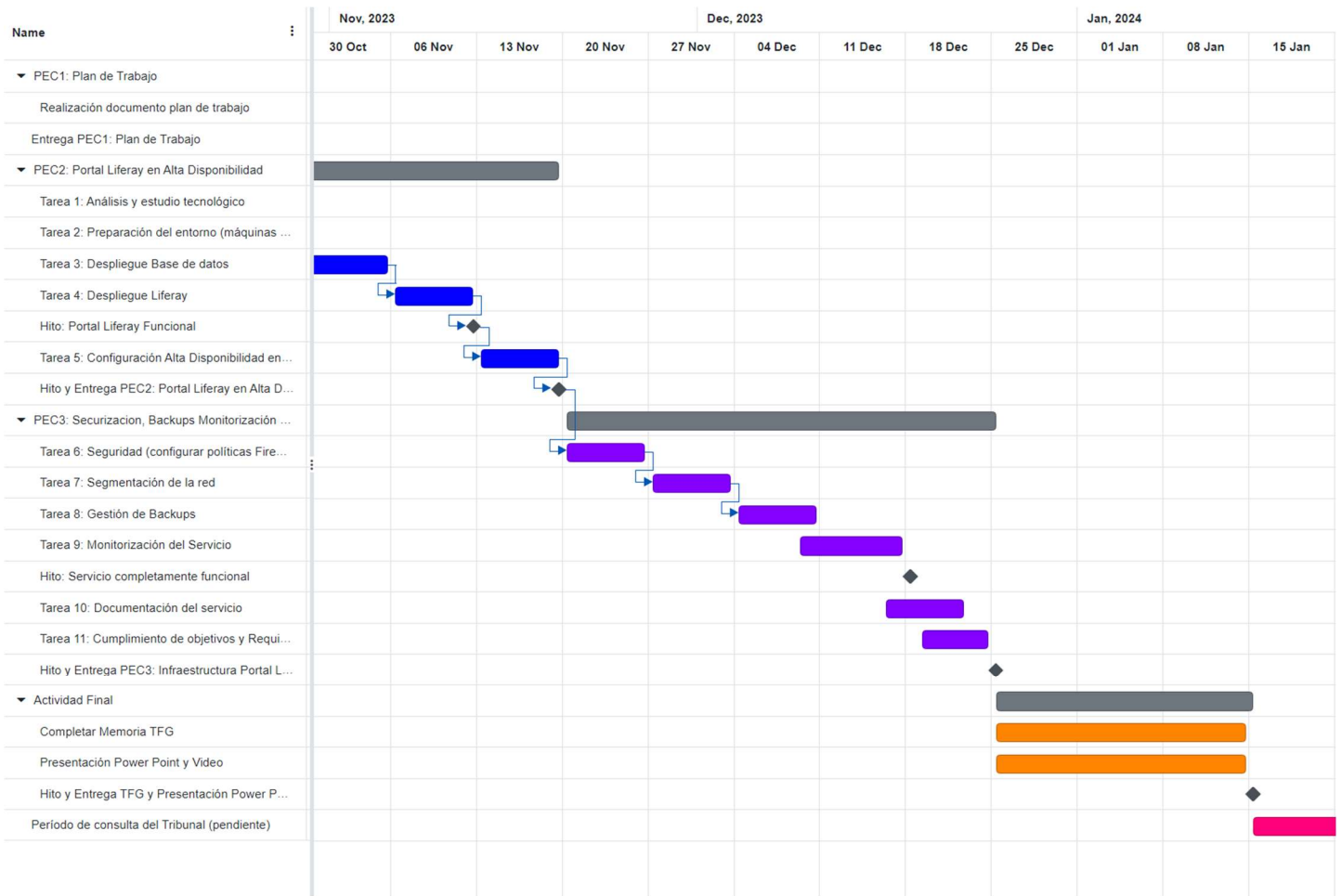


Figura 2. Cronograma Pte.2

1.5 Breve resumen de productos obtenidos

El producto obtenido es un portal digital con capacidad CMS basado en Liferay Portal CE en una infraestructura de instancia doble, ejecutándose en dos servidores virtuales distintos, tanto su parte aplicativa, como la parte web y de servicio de búsqueda, de tal manera, que se lo dota de una alta disponibilidad. Además, dispone de una seguridad perimetral proporcionada por un cortafuegos y un sistema de balanceo de carga que reparte esta entre las instancias web y asegura una conectividad mediante protocolo seguro. Por otro lado, está dotado de una base de datos e información coherente y confiable mediante un sistema NFS de compartición de ficheros en la red del portal, toda ella protegida con unas medidas de seguridad, integridad de datos y monitorización que garanticen la estabilidad del servicio.

La alta disponibilidad da una continuidad operacional ante posibles caídas de alguno de los sistemas, ya sean controladas o por eventos externos, y da una capacidad de rendimiento extra por el balanceo de carga que se va a producir entre la doble instancia.

Este portal dota a la empresa de la capacidad de que sus trabajadores puedan crear de sitios web mediante plantillas, documentación, blogs, wikis y demás tipos de contenido útiles tanto para la gestión de documentación interna de la propia empresa como para posibles clientes.

El servicio permite, no sólo gestionar archivos digitales (imágenes, videos y documentos), la personalización de este contenido, control de versiones, integración con sistemas externos (como CRMs, bases de datos o ERPs), la búsqueda de dicha documentación para facilitar a los trabajadores y usuarios autorizados encontrar documentación específica dentro de la web, sino que, además, incluye la capacidad de generar sitios webs internos mediante plantillas o desarrollos de los propios trabajadores de la empresa.

Por otro lado, es un repositorio donde se puede gestionar el contenido de forma centralizada de todos los archivos digitales para la línea de negocio de la empresa.

1.6 Breve descripción de los otros capítulos de la memoria

1.6.1 Análisis y estudio tecnológico

-Se procederá al análisis de las posibilidades tecnológicas del software open-source existente basado en los requerimientos y matriz de compatibilidad de Liferay y resto de software para poder desplegar la infraestructura. Dicho análisis también comprenderá el estudio de los requerimientos hardware (CPU, Memoria, espacio en disco, etc.) de todo el software implicado para poder ser ejecutado sin problemas.

-El objetivo es tomar la decisión del software y versiones exactas del software que vamos a utilizar para poder desplegar la infraestructura completa necesaria para que el portal digital Liferay pueda funcionar en base a las especificaciones del trabajo, así como tener los requerimientos del hardware necesario para desplegar la infraestructura.

1.6.2 Preparación del entorno

Esta preparación consistirá en:

-Desplegar y configurar las máquinas virtuales necesarias con VirtualBox para los servidores web y de aplicación de Liferay, para ello se instalará VirtualBox y se creará una máquina virtual que se configurará y actualizará. Esta máquina servirá de plantilla para distribuir el software necesario en cada una de ellas.

-Instalar y hacer una configuración base de un servidor web Apache en las máquinas destinadas al este menester y al servidor del aplicativo.

-Instalar y configurar un balanceador de carga HA Proxy en una de las máquinas virtuales para poder distribuir el tráfico entre el resto de las máquinas virtuales que alojarán la parte web y aplicativo.

-Desplegar una máquina en la que alojará una base de datos gestionada y configurar el almacenamiento compartido mediante un NFS para los datos necesarios de Liferay de tal manera que se garantice la alta disponibilidad si uno de los nodos se cae.

-El objetivo será tener la infraestructura básica de máquinas que tendrán los servicios web, balanceador, de aplicativo y de almacenamiento necesario para poder desplegar el Liferay.

1.6.3 Desplegar una base de datos gestionada

-Desplegaremos y configuraremos una base de datos en una máquina virtual separada.

-El objetivo es tener una base de datos necesaria para que el Liferay pueda almacenar sus datos.

1.6.4 Despliegue del portal digital Liferay funcional

-Instalaremos y configuraremos los servidores web en las máquinas virtuales que redirigirán el tráfico a los servidores del aplicativo.

-Instalaremos y configuraremos los servidores de aplicación junto con el Liferay que trabajarán en modo de alta disponibilidad.

-Instalaremos y configuraremos los servidores de búsqueda necesarios para Liferay que también trabajarán en modo de alta disponibilidad.

-Configuraremos el Liferay contra la base de datos gestionada para desplegar el contenido necesario por el aplicativo.

-Como paso final haremos pruebas básicas de funcionamiento del portal Liferay generando y/o configurando contenido web dentro del mismo.

-El objetivo final será disponer de un portal de Liferay desplegado y asegurar su correcto funcionamiento con una infraestructura dotada de balanceador, servidores web, de aplicativo, su base de datos y su sistema de ficheros en red completamente funcionales.

1.6.5 Alta Disponibilidad, Escalabilidad y Rendimiento mediante una arquitectura en clúster

-Estableceremos un clúster entre las máquinas virtuales del aplicativo mediante la configuración de los servidores web, los servidores del aplicativo y los servidores de búsqueda, de tal manera que se reparta la

carga de trabajo a la vez que se da una alta disponibilidad al sistema en caso de fallos de alguno de los nodos

-Realizaremos pruebas para verificar el correcto funcionamiento del clúster, desactivando algunos nodos y verificando que el sistema sigue funcionando.

-El objetivo de la tarea es dotar a la plataforma de alta disponibilidad, de escalabilidad y de un rendimiento adecuado.

1.6.6 Seguridad

-Configuraremos políticas de seguridad de firewall para proteger la infraestructura y la plataforma de posibles amenazas exteriores.

-Instalaremos un certificado en la parte de los servidores web o el HAProxy para que el tráfico HTTP esté cifrado, es decir, que sea HTTPS.

-Modificaremos el comportamiento de HA Proxy para aceptar también tráfico cifrado HTTPS.

-El objetivo de la tarea será garantizar la confidencialidad e integridad de los datos y la estabilidad y disponibilidad del sistema y del servicio mediante las políticas de seguridad de firewall y el uso de tráfico cifrado para acceder a los servidores web.

1.6.7 Segmentación de la red

-Estableceremos una segmentación de la red, definiendo una DMZ y una red de aplicativo/backend, mediante la configuración de interfaces de red extra en las máquinas virtuales en otras subredes y configurando rutas entre las mismas.

-Configuraremos los servicios web, de los aplicativos implicados en sus correspondientes subredes.

-El objetivo de la tarea es aumentar el aislamiento de servicios de tal manera que se añada una capa extra de seguridad en caso de ataque desde el exterior y agregar una gestión eficiente de los recursos.

1.6.8 Gestión de Backups (copias de respaldo)

-Estableceremos un sistema de gestión de copias de seguridad de los datos, tanto de la base de datos como del sistema de ficheros de red.

-El objetivo de la tarea será garantizar la integridad de los datos y la continuidad del servicio en caso de cualquier fallo relacionado, de tal manera que se puedan recuperar los mismos en tiempo y forma con el mínimo impacto sobre dicho servicio.

1.6.9 Monitorización del servicio

-Instalaremos y configuraremos en el host Windows (u opcionalmente en otra de las máquinas virtuales si tenemos suficientes recursos) un sistema de monitorización de los servicios desplegados.

-El objetivo de la tarea es asegurar que el sistema es estable, fiable y que proporciona los servicios para los que ha sido diseñado.

1.6.10 Documentación de explotación de servicio

-Crearemos una documentación que describa la infraestructura con detalle y que proporcione una capacitación al usuario para el uso de la misma tal como el plan de contingencia en caso de fallos del sistema, así como un diagrama de red de la infraestructura.

-El objetivo es proporcionar una capacitación al usuario para asegurarse de que se pueda administrar y mantener toda la plataforma de forma eficiente y efectiva.

1.6.11 Verificación de cumplimiento de objetivos y requisitos

-Verificaremos todo el funcionamiento del sistema mediante pruebas simulando posibles fallos y situaciones de tal manera que el sistema siga funcionando adecuadamente.

-El objetivo de la tarea será asegurar que la infraestructura diseñada cumpla los objetivos y requisitos establecidos al inicio del trabajo, así como, la satisfacción y eficiencia de esta.

2. Análisis, desarrollo y despliegue de un CMS en Alta Disponibilidad

2.1 Análisis y estudio tecnológico:

En este capítulo se procede al análisis de las posibilidades tecnológicas del software open-source existente basado en los requerimientos de una empresa para implantar un portal digital CMS con capacidad de creación de sitios web, en alta disponibilidad y monitorizado. Este análisis también comprenderá el estudio de los requerimientos hardware necesario para que el software implicado pueda ser ejecutado sin problemas.

Tendremos que tomar la decisión del software y sus versiones exactas que vamos a utilizar para poder desplegar la infraestructura completa necesaria para que el portal digital pueda funcionar en base a las especificaciones que un entorno corporativo, así como tener los requerimientos del hardware necesario para desplegar la infraestructura.

Partiremos de la premisa de que el objetivo es intentar implantar la infraestructura completa para el portal digital mediante software open-source, por tanto, limitaremos el estudio a este ámbito.

2.1.1 Comparativa de tecnologías de posible aplicación

Para poder implantar un portal digital de estas características mediante software open-source estaremos limitados a sólo unas pocas opciones. Además, deberemos tener en cuenta el software final instalado que tendrá que ser un portal digital con capacidad CMS y que se pueda instalar en servidores de aplicaciones Java, sobre Linux, para poder tener la capacidad de Alta Disponibilidad, escalabilidad y rendimiento mediante balanceo de carga.

Actualmente hay sólo cinco²⁹ productos que den capacidad de CMS y open-source: Alfresco, Liferay, LogicalDOC, OpenCMS y Magnolia.

No obstante, hay que tener en cuenta, que muchas empresas, como en nuestro caso, están evolucionando hacia plataformas de experiencia digital (Digital Experience Platforms o DXPs) las cuales integran funcionalidades que suelen estar más asociadas a los portales y que permiten, además de gestionar contenido, crear sitios web. Este tipo de productos buscan atender, también, otras necesidades de empresas que están en su proceso de transformación digital y cuyo objetivo es ofrecer mejores experiencias al cliente como determinados sitios web accesibles a los clientes.

En nuestro caso al producto que tiene esas capacidades, es Liferay Portal CE, ya que, por ejemplo, Alfresco, LogicalDOC, OpenCMS y Magnolia, sí que son CMS y permiten la gestión documental pero no tienen otras

capacidades que Liferay sí tiene, como la de creación de sitios web mediante plantillas y diseños internos.

Liferay Portal CE es un contenedor de portlets³⁰ y que, por defecto, viene con unos portlets que permiten gestionar contenidos. La parte de CMS tiene soporte para determinadas herramientas como flujos de trabajo, definición de estructuras de documentos, plantillas, definición de fechas de publicación y varias más que son interesantes para una empresa como en nuestro caso.

Todo ello hace que Liferay Portal CE tenga más ventajas que los otros CMS existentes y sea la opción preferida para hacer esta implantación.

2.1.2 Requisitos finales de software y hardware

Teniendo en cuenta que usaremos todo el software siendo este open-source y atendiendo a la matriz de compatibilidad de Liferay Portal CE 7.4⁴ (última versión) tendremos en cuenta dicho software y versionado. Además, hay que tener en cuenta que el paquete de Liferay Portal CE 7.4 se descarga en un paquete que ya incluye parte del software como la versión de Tomcat y de Elasticsearch a usar.

- Software:

-Plataforma de Virtualización: Oracle VM VirtualBox 7.0.12
(<https://www.virtualbox.org/wiki/Downloads>)

-Sistemas Operativos:

-Host anfitrión: Windows 10 EDU 64bits version 10.0.19045
-Máquinas virtuales: Linux CentOS 8.4
(<https://www.linuxvmimages.com/images/centos-8/#centos-842105>). Usaremos esta distribución porque es open-source, está soportada por Liferay, es la más similar a RedHat Enterprise Linux (que es con la que más experiencia dispongo) y porque entre ambas distribuciones están ampliamente extendidas según se puede ver en el gráfico adjunto⁵ del “*Ranking the Top Enterprise and Open Source Operating Systems of 2022*”.



Which of These Open Source Technologies Does Your Organization Use Today to Support Your Software Infrastructure?

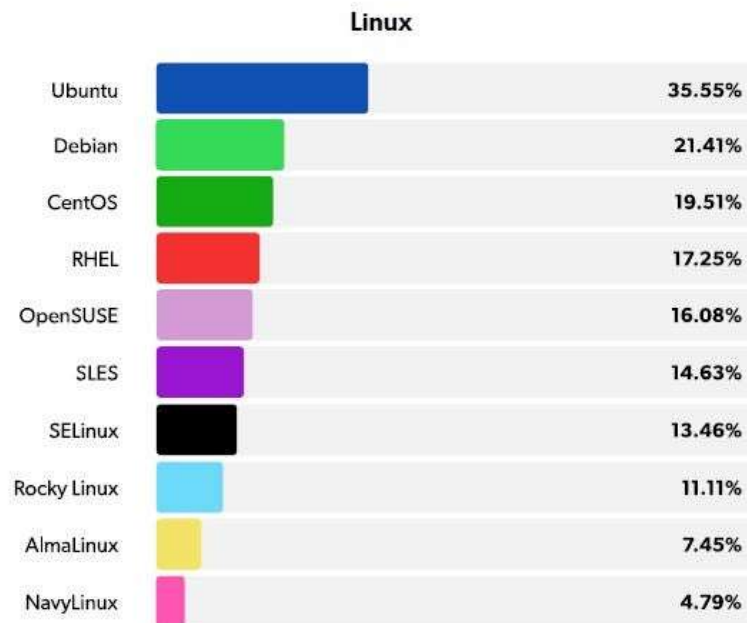


Figura 3. 2022 State of Open Source Report

-Servidores web: Apache Web Server 2.4
(<https://httpd.apache.org/>)

-Servidores de aplicativo: Apache Tomcat 9.0.80, ya que viene incluido en el paquete de Liferay CE que descargaremos.
(<https://tomcat.apache.org/>)

-Servidores de búsqueda: Elasticsearch 7.17, ya que viene incluido en el paquete de Liferay CE que descargaremos.
(<https://www.elastic.co/es/elasticsearch>)

-Portal Digital CMS: Liferay Portal Community Edition (CE) 7.4.3.99
(<https://www.liferay.com/es/downloads-community>)

-Gestor de Bases de Datos: MySQL Community Server 8.0.35, ya que es la versión gratuita y open-source.
(<https://dev.mysql.com/downloads/mysql/>)

-Balanceador: HAProxy 1.8
(<https://www.haproxy.org/>)

-Software Java: Oracle JDK 8u391
(<https://www.oracle.com/java/technologies/downloads/#java8>)

-Software monitorización: Grafana 10.2
(<https://grafana.com/grafana/download>)

-Software de extracción de métricas: Prometheus 2.48
(<https://prometheus.io/download/>)

A continuación, exponemos los requisitos de hardware necesarios basándonos en los requisitos recomendados de los productos open-source que vamos a instalar. También deberemos tener en cuenta que estaremos limitados por el host físico que vamos a utilizar que es un PC con una CPU AMD Ryzen 7 3700X con 8 cores (16 hilos lógicos), con una memoria de 32GB de RAM y un disco duro de 1 TB.

Los requerimientos⁶ de CentOS 8 nos dicen que como mínimo para la instalación se necesitan entre 768MB y 1.5Gb de memoria, aunque ajustaremos entre 4GB a 6GB para que el sistema pueda ejecutarse en las máquinas virtuales sin que el rendimiento se vea afectado y dependiendo de los procesos a ejecutar.

- Hardware

-Host anfitrión:

- CPU: 8 cores físicos (16 hilos lógicos)
- Memoria: 32 GB
- Disco: 220 a 500 GB aprox.

-Máquinas virtuales dependientes del host anfitrión:

-1 Máquina con el balanceador y firewall:

- CPU: 4 Cores lógicos
- Memoria: 4 GB
- Disco: 20 a 40 GB

-2 Máquinas con servicios web, aplicativo y servicio de búsqueda:

- CPU: 4 Cores lógicos
- Memoria: 6 GB
- Disco: 50 a 100 GB

-1 Máquina con gestor de bases de datos NFS:

- CPU: 4 Cores lógicos
- Memoria: 6 GB
- Disco: 100 a 200 GB

- Datos:

-Punto de montaje compartido NFS entre los 2 servidores del aplicativo de unos 100GB.

-Sistema de copia de seguridad del NFS, de la BBDD y del aplicativo.

- Usuarios:

- Se generarán usuarios locales en cada máquina para ejecutar cada servicio.
- Se generará un plan de contingencia en caso de fallos del sistema tales como caídas del servicio o recuperación de backups.

- Seguridad

- Habrá que generar un certificado para la parte web que puede ser uno gratuito o uno autofirmado.
- Un servicio de cortafuegos debe estar configurado al menos en la máquina que hará de balanceador.

2.2 Preparación del entorno

En este capítulo desplegaremos y configuraremos las máquinas virtuales necesarias, servicio de balanceo, sistema de compartición de archivos en red y servidores web. Todo ello junto con la posterior instalación de la base de datos gestionada nos dará la infraestructura básica de software para hacer funcionar el portal de Liferay.

2.2.1 Preparación de sistema de virtualización y máquina virtual plantilla

Para montar la infraestructura donde se instalará todo el software usaremos máquinas virtuales que se ejecutarán sobre VirtualBox en el host físico que será nuestra máquina con un sistema operativo Windows en ella (véase [Anexo 1](#)).

2.2.2 Despliegue de la máquina virtual y configuración del Apache Webserver

El servidor web es el encargado de recibir las peticiones HTTP para servir contenido estático, pero en este caso usaremos una de sus funcionalidades que es la de proxy inverso para redirigir dichas peticiones a los servidores de aplicaciones. Además, podrá repartir dichas peticiones entre todos los servidores de aplicaciones que configuremos.

El motivo de usar un servidor web por delante del servidor de aplicaciones es que agrega una capa extra de seguridad, gestionan mejor las peticiones web y liberan de carga al servidor de aplicaciones, que se puede encargar mejor de su tarea que es servir el contenido de la aplicación.

Con la plantilla de la máquina virtual, que creamos en el capítulo anterior, se procede a su clonación (véase [Anexo 2](#)) para crear la máquina virtual que alojará uno de los servidores web Apache (y que también alojará el Liferay junto con el Tomcat y el Elasticsearch). A esta máquina se la llamará “apacheliferay01” y haremos otra más adelante para tener la alta disponibilidad que se llame “apacheliferay02”.

Se tendrá también que fijar la IP de la máquina de forma estática (Véase [Anexo 3](#)), ya que la vamos a necesitar para más adelante para configurar

el HAProxy indicando las IPs de las máquinas a las que tiene que balancear. En nuestro caso, aplicamos la IP 192.168.1.20 para esta máquina y reservamos la 192.168.1.21 para la otra máquina con el Apache que configuraremos más tarde cuando activemos la alta disponibilidad. De esta manera ya tenemos las IPs para configurar más adelante en el HAProxy.

Desactivaremos también el servicio de cortafuegos (véase [Anexo 4](#)) para que no nos de problemas, no obstante, más adelante, en este TFG, se configurará la seguridad perimetral en la máquina de HAProxy.

Una vez instalado, configurado y levantado el servicio web Apache (véase [Anexo 5](#)) se podrá acceder mediante navegador web al mismo verificando su correcto funcionamiento.

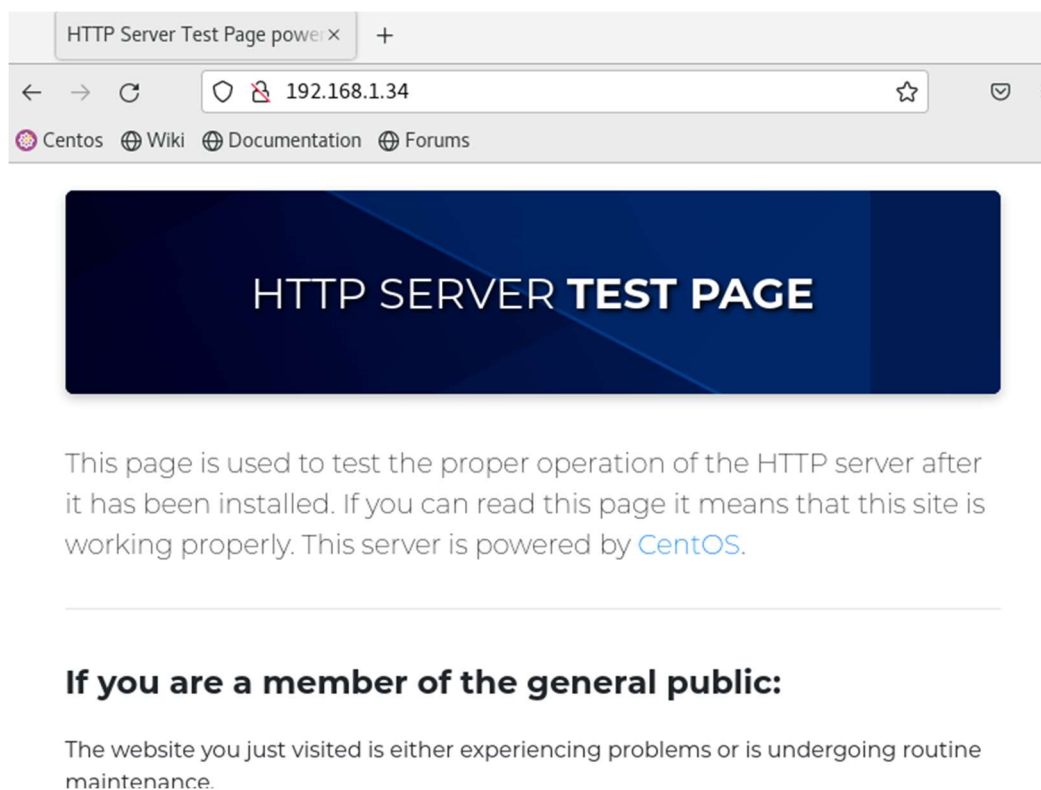


Figura 4. Página web principal de servidor web Apache

Volveremos más adelante a esta máquina a hacer más configuraciones sobre el servidor web Apache y a instalar el Tomcat junto con el Liferay y su servicio de búsqueda Elasticsearch.

2.2.3 Despliegue de la máquina virtual y configuración del HAProxy

El servicio de HAProxy nos proporcionará el servicio de balanceo y redirección de tráfico HTTP entre los servidores web Apache que estarán por debajo de ella además de darnos una capa extra de seguridad.

Procederemos a clonar la plantilla para crear la máquina virtual que alojará el HAProxy (véase [Anexo 2](#)) y le pondremos un nombre distinto para diferenciarla, en este caso “haproxy”

A continuación, se procede a instalar y configurar el paquete de haproxy¹¹ con el gestor de paquetes de “Dandified Yum”¹⁰ (véase [Anexo 6](#))

Desactivaremos también el servicio de Firewall (véase [Anexo 3](#)) para que no nos dé problemas de conectividad, aunque más adelante en este TFG lo reactivaremos y configuraremos en esta máquina la seguridad perimetral con este servicio.

Con el servicio de Apache webserver levantando en la máquina “apacheliferay01” y accediendo desde nuestro host a la IP de la máquina de “haproxy” con un navegador podemos comprobar como carga la página de test del Apache en la que se puede ver la IP de la máquina que tiene el servicio HAProxy.

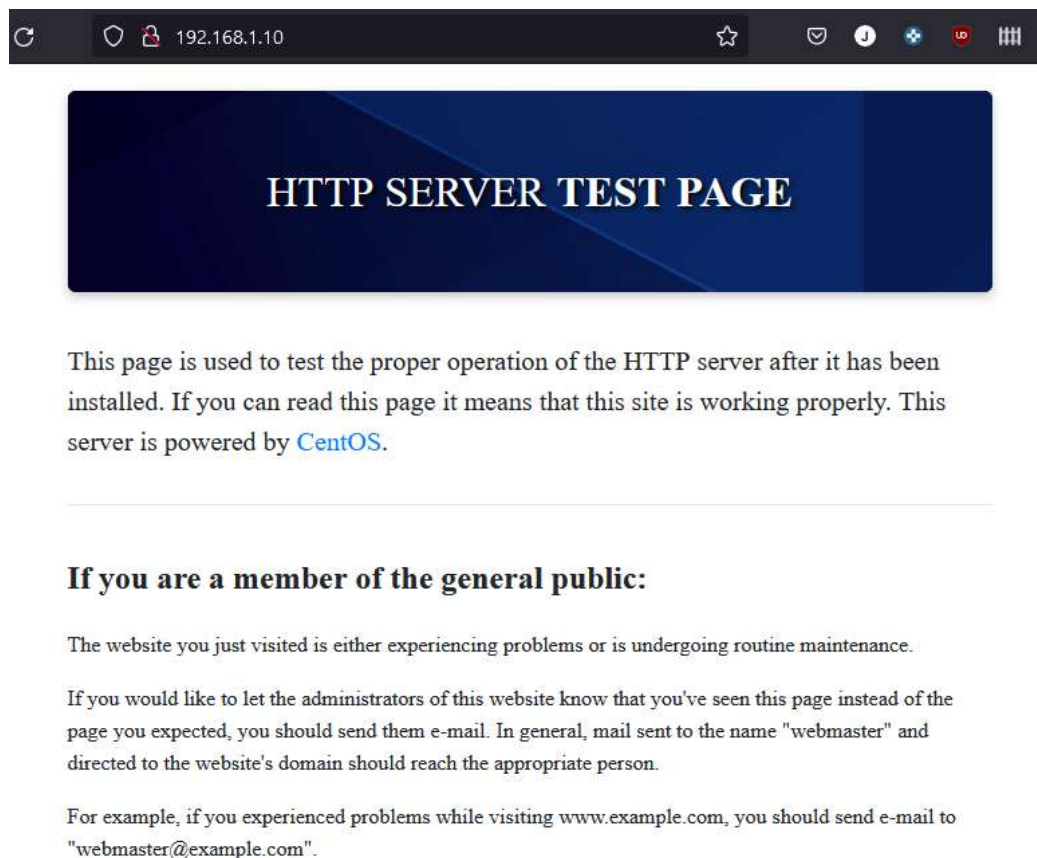


Figura 5. Página web principal de Apache vista a través de HAProxy

Esto nos indica que el balanceador HAProxy está redirigiendo el tráfico HTTP por el puerto 80 correctamente al Apache webserver como pretendíamos.

Es interesante que, además, configuremos un sistema de log de eventos mediante Syslog (véase [Anexo 7](#)) para que nos almacene las estadísticas de HAProxy. De esta manera podremos comprobar que los accesos que

se realizan y a que nodo de Apache Webserver se está llegando mediante dichos logs de eventos.

2.2.4 Despliegue de la máquina virtual para el MySQL y configuración del NFS

Esta máquina alojará la base de datos de Liferay y su sistema de ficheros accesible por red. De esta manera las máquinas donde están los liferays podrán acceder de forma remota a un sistema de ficheros compartido.

Procederemos a clonar la plantilla (véase [Anexo 2](#)) para crear la máquina virtual que alojará el MySQL y el servidor NFS y asignaremos un nombre a esta máquina que se llamará “mysql”.

En esta máquina fijaremos también su IP como estática 192.168.1.30 y el DNS de Google (8.8.8.8), como antes con las anteriores (véase [Anexo 3](#))

En el siguiente capítulo instalaremos la base de datos gestionada MySQL, pero antes, dejaremos configurado en este capítulo el NFS (véase [Anexo 8](#)) que usará el Liferay.

Para configurar el NFS (véase [Anexo 8](#)) tendremos que instalar la parte de servidor y la de cliente en cada máquina que lo vaya a usar, aquí haremos la configuración de la parte de servidor en la máquina “mysql” y la parte de cliente en la máquina “apacheliferay01”, pero también debe hacerse más adelante en la “apacheliferay02” cuando configuremos la alta disponibilidad y será análoga a esta.

2.3 Despliegue de una base de datos gestionada MySQL

2.3.1 Despliegue y configuración del servidor MySQL

Liferay necesita una base de datos gestionada, para ello se instalará (véase [Anexo 9](#)) en la máquina que hemos clonado antes y que hemos llamado “mysql”.

Tras la instalación y configuración de la misma se puede comprobar que está levantada y funcionando.

```
[centos@mysql ~]$ sudo systemctl status mysqld
● mysqld.service - MySQL 8.0 database server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor prese
   Active: active (running) since Sun 2023-11-12 20:38:33 CET; 8min ago
     Process: 1918 ExecStartPost=/usr/libexec/mysql-check-upgrade (code=exited, sta
     Process: 1055 ExecStartPre=/usr/libexec/mysql-prepare-db-dir mysqld.service (s
     Process: 986 ExecStartPre=/usr/libexec/mysql-check-socket (code=exited, statu
   Main PID: 1129 (mysqld)
   Status: "Server is operational"
     Tasks: 38 (limit: 36446)
    Memory: 459.8M
    CGroup: /system.slice/mysqld.service
            └─1129 /usr/libexec/mysqld --basedir=/usr

Nov 12 20:38:05 mysql systemd[1]: Starting MySQL 8.0 database server...
Nov 12 20:38:33 mysql systemd[1]: Started MySQL 8.0 database server.
```

Figura 6. Servicio MySQL ejecutándose

Como indica la documentación de Liferay¹⁵ tendremos que generar un esquema (que llamaremos “lportal”) y un usuario (que llamaremos “lportal_user”) para ese esquema con permisos totales para que Liferay pueda crear tablas, borrarlas y demás operativas necesarias para que pueda funcionar (véase [Anexo 9](#))

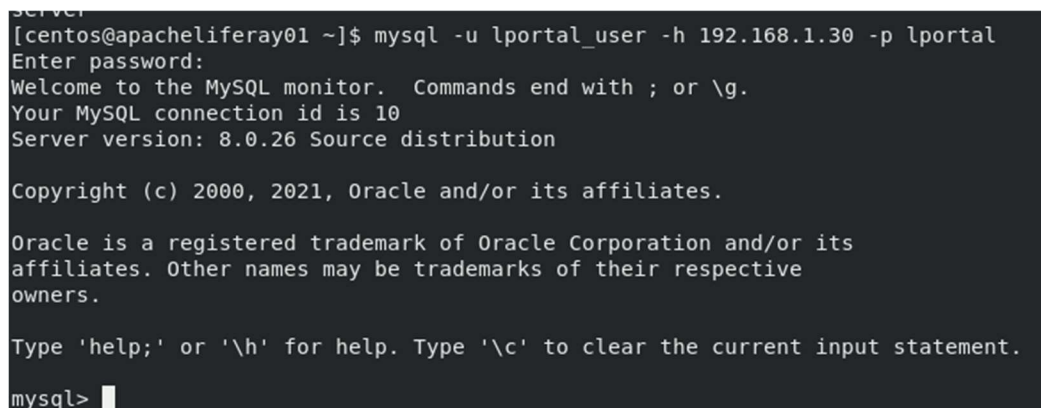
2.3.2 Configuración del cliente de MySQL

Se tiene que instalar el cliente de mysql (véase [Anexo 10](#)) en la máquina “apachelifeary01” con el objetivo de poder una prueba de conexión remota a la base de datos que hemos creado.

Una vez finalizada la instalación probamos la conexión con el usuario que hemos creado específicamente para Liferay:

```
mysql -u lportal_user -h 192.168.1.30 -p lportal
```

Y comprobamos que conecta adecuadamente



```
SERVER:
[centos@apacheliferay01 ~]$ mysql -u lportal_user -h 192.168.1.30 -p lportal
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.26 Source distribution

Copyright (c) 2000, 2021, 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>
```

Figura 7. Cliente MySQL conectándose al servidor MySQL

Con esto ya tendríamos la instalación mínima de base de datos para Liferay y además hemos asegurado que la conexión funciona desde el servidor del aplicativo.

2.4. Despliegue del portal digital Liferay funcional

Este despliegue nos proveerá del Portal Digital de Liferay de gestión y creación de contenido web, tales como sitios web.

Para hacer la instalación seguiremos la documentación oficial de Instalación de Liferay¹⁷, en concreto la de la instalación mediante un paquete conjunto de Liferay y Tomcat¹⁸. Para ello trabajaremos sobre la máquina “apacheliferay01” en base a esa documentación (aunque más adelante haremos lo mismo sobre la “apacheliferay02” que crearemos para cuando configuremos la alta disponibilidad).

2.4.1 Instalación de las Java Runtime JDK

Lo primero que nos piden como prerrequisito es la instalación de las Java Runtime JDK (véase [Anexo 11](#)), el problema es que las de Oracle ya son de pago a partir de la versión 8u202 para uso comercial¹⁹, por tanto, nos bajaremos esta versión, aunque podríamos utilizar la de IBM u otras que como indica la matriz de compatibilidad⁴ de Liferay, estén sean compatibles con las “Java Technical Compatibility Kit” de versiones 11 u 8.

Una vez instaladas y configuradas podemos comprobar que el sistema detecta el Java instalado.

```
liferay@apacheliferay01 ~]$ cd /opt/  
liferay@apacheliferay01 opt]$ java -version  
java version "1.8.0_202"  
Java(TM) SE Runtime Environment (build 1.8.0_202-b08)  
Java HotSpot(TM) 64-Bit Server VM (build 25.202-b08, mixed mode)  
liferay@apacheliferay01 opt]$
```

Figura 8. Test de detección de Java por el sistema

2.4.2 Instalación del paquete Liferay Tomcat

Aquí instalaremos y configuraremos el Portal Liferay (véase [Anexo 12](#)) junto al servidor de aplicaciones en un “bundle” (paquete) desde la web de Liferay en la máquina “apacheliferay01”.

Una vez instalado se podrá acceder localmente desde la máquina con un navegador usando la dirección:

<http://localhost:8080>

The image shows a web browser window with the address bar displaying 'localhost:8080/c/portal/setup_wizard'. The browser's address bar includes navigation icons (back, forward, refresh) and a star icon for bookmarks. Below the address bar, there are links for 'Centos', 'Wiki', 'Documentation', and 'Forums'. The Liferay logo is visible at the top left of the page content.

The main content area is titled 'Basic Configuration' and is divided into three sections: 'PORTAL', 'ADMINISTRATOR USER', and 'DATABASE'.

- PORTAL:**
 - Nombre del portal ***: Input field containing 'Liferay CE UOC'.
 - Idioma por defecto**: A dropdown menu showing 'español (España)' with a 'Cambiar' button next to it.
 - Zona horaria**: A dropdown menu showing '(UTC +01:00) hora estándar de Europa'.
- ADMINISTRATOR USER:**
 - Nombre ***: Input field containing 'Jorge'.
 - Apellido ***: Input field containing 'Russell'.
 - Correo electrónico ***: Input field containing 'jrussell@uoc.edu'.
- DATABASE:**
 - Default Database (Hypersonic)**: Text indicating the current database choice.
 - A note: 'This database is useful for development and demo'ing purposes, but it is not recommended for production use.' with a '(Change)' link below it.

At the bottom of the configuration area, there is a blue button labeled 'Finalizar Configuración'.

Figura 9. Página de Wizard de configuración de Liferay

En ella haremos las configuraciones básicas (véase [Anexo 12](#)), como conectar a la base de datos gestionada e introducir datos que se nos solicitan.

Una vez realizadas las tareas y configuraciones básicas, tendremos nuestro Liferay funcionando de forma local accesible mediante navegador como podemos ver en la figura a continuación.

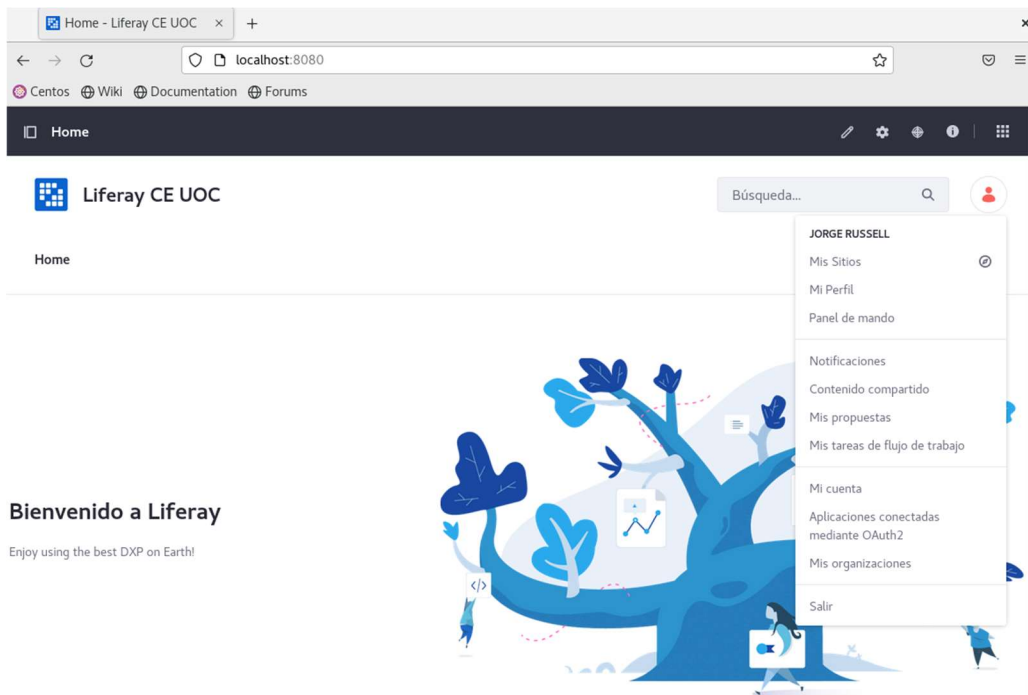


Figura 10. Página web principal de Liferay funcionando en local

2.4.3 Configuración del Data Library de Liferay en el NFS

Para poder disponer más adelante de la capacidad de Alta Disponibilidad tendremos que configurar el directorio donde guarda los datos el Liferay en un directorio accesible por red remotamente, como es un NFS, para así tener la información asegurada en otra máquina y accesible por todos los nodos de Liferay que se vayan a instalar. Este paso no sería necesario en una instalación con una instancia simple con un solo servidor, pero sí en nuestro caso.

Lo ideal sería que este directorio NFS estuviera montado en una cabina NAS²¹ (Network Attached Storage) que es un sistema de almacenamiento conectado en la red local que suele tener múltiples discos y un sistema interno de combinación de dichos discos para prevenir que se pierdan datos en caso de fallo de alguno de ellos. Estas cabinas NAS permiten el uso de NFS para tener almacenamiento en red accesible desde otros sistemas como el nuestro.

En nuestro caso simularemos que tenemos una NAS, pero en la máquina donde reside el MySQL, en la que hemos montado también un servidor NFS con un punto de montaje accesible remoto desde las máquinas donde estarán los servidores Liferay.

El Liferay guarda por defecto sus datos²² en su *home* en el subdirectorio `./data/document_library`

Según la documentación de Liferay²² podemos cambiar esta localización de ficheros de datos (véase [Anexo 13](#)).

Este punto de montaje cumple las necesidades para un entorno de alta disponibilidad, como el que montaremos más adelante, el cual estará accesible para ambos nodos, soportará peticiones concurrentes y bloqueo en exclusiva de ficheros, todas ellas características que aporta un sistema como es el NFS.

De hecho, hemos hecho una prueba de generación de contenido web y ya vemos como escribe ficheros en el directorio NFS que hemos configurado.

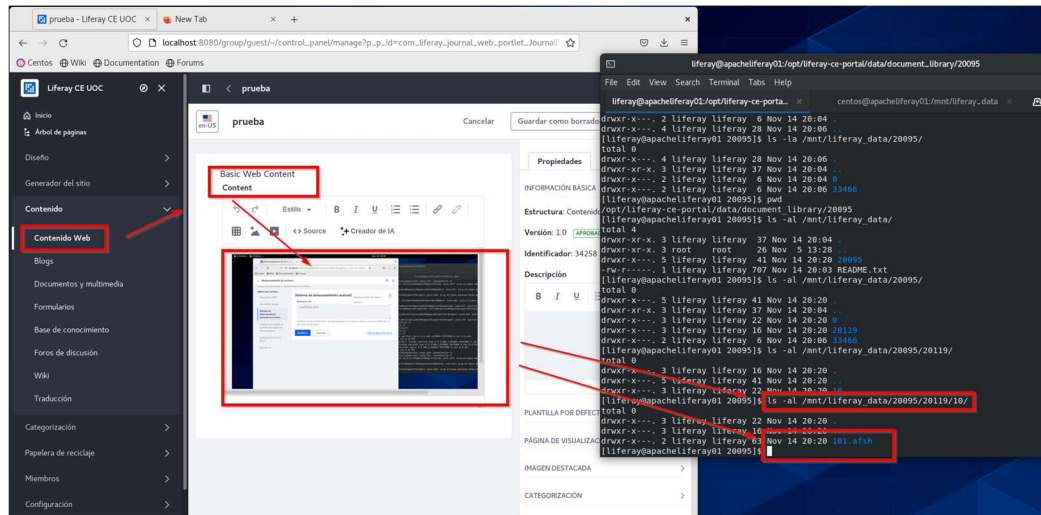


Figura 11. Contenido de Liferay generado en NFS

2.4.4 Instalación y configuración del servidor de búsqueda Elasticsearch

El Portal Liferay necesita un servicio de búsqueda que además indexe sus contenidos para que cuando se realice una búsqueda de los mismos estos se hagan de una forma rápida y eficiente. Para ello se configura un servicio que se llama Elasticsearch.

En este capítulo configuraremos ese servidor de búsqueda independiente Elasticsearch (véase [Anexo 14](#)) ya que el que lleva el paquete incluido no está recomendado para entornos productivos según la documentación de Liferay²³ a este efecto, además de darnos algunas de las instrucciones para hacer esta configuración²⁴.

Una vez instalado y configurado tendremos disponible dicho servicio desde el propio portal de Liferay que hemos instalado

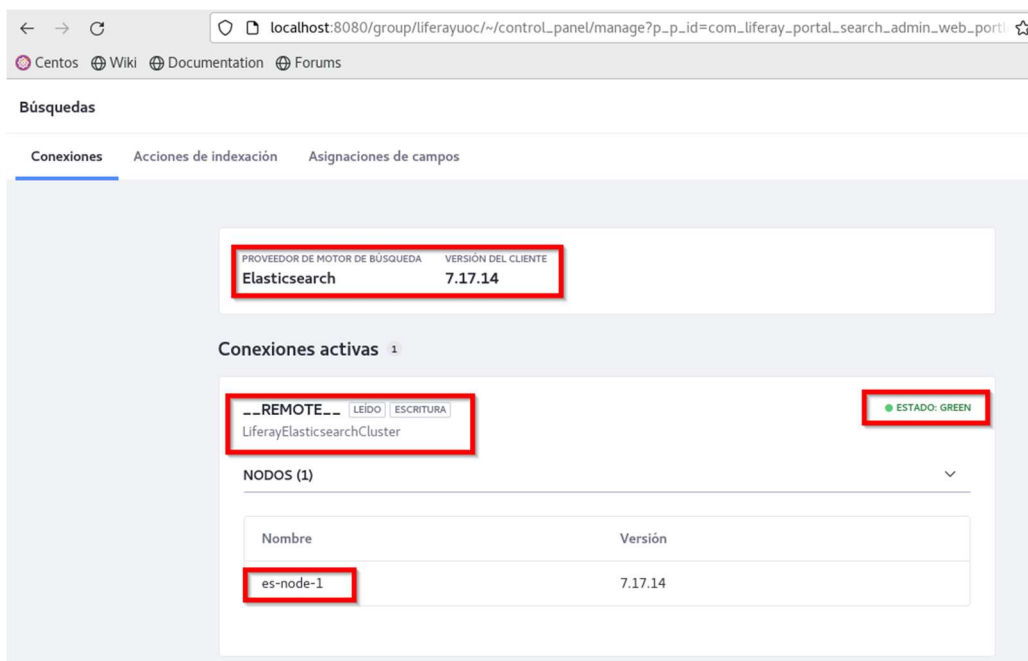


Figura 12. Elasticsearch en Liferay

2.4.5 Configuración redirección de tráfico del servidor web Apache al Liferay

Para poder acceder desde nuestro host hasta el Liferay a través de todas las capas, que incluyen, el HAProxy y el servidor web Apache tendremos que configurar dicho servidor web Apache (véase [Anexo 15](#)) para que pueda realizar la redirección de tráfico que envía el HAProxy hacia él.

Como hemos explicado antes hemos configurado servidores web Apache por varias razones, una de ellas es que gestionan más y mejor las llamadas HTTP que los servidores de aplicaciones y la otra, es por seguridad, al ser el servidor que recibe las peticiones aislamos al aplicativo sobre el Tomcat aislado de posibles ataques.

Para realizar esta tarea, no sólo tendremos que añadir configuración extra al servidor web Apache, sino que, además, tendremos que tener en cuenta que hay unos requerimientos de seguridad en Liferay y que las cabeceras HTTP lleguen correctamente al Liferay, lo cual implica que tanto el Apache como el HAProxy también tenga esta configuración (aunque esta configuración ya la hemos fijado antes en el caso del HAProxy como se ha podido ver en el capítulo [2.2.3](#))

Posteriormente y para poder acceder vía web al sitio web definido por Liferay desde nuestra máquina host Windows, hay que tener en cuenta que deberemos simular que existe un DNS para tal dominio, para ello habrá que editar nuestro fichero “etc/hosts” en nuestra máquina host Windows (véase [Anexo 16](#)) y de esta manera podremos usar las URL/Dominios que hemos configurado en Liferay para acceder a ellos. En este caso hemos definido estos tres ya que aparte del sitio web propio de Liferay, crearemos otros dos dominios aparte de pruebas:

<http://liferayuoc.org>

<http://liferaypruebasuoc.org>

<http://raylifed2cuoc.org>

2.4.6 Comprobación del correcto funcionamiento del Portal Liferay

En este capítulo podemos comprobar cómo se pueden crear correctamente sitios web usando plantillas de Liferay y gestionar contenido dentro del mismo.

Como se puede comprobar, al poner en el navegador de nuestro host Windows la URL de administración de Liferay lo que hará será ir al HAProxy que nos redirigirá al servidor web Apache y a su vez al Tomcat que tiene el Liferay realizando todo el circuito.

<http://liferayuoc.org>

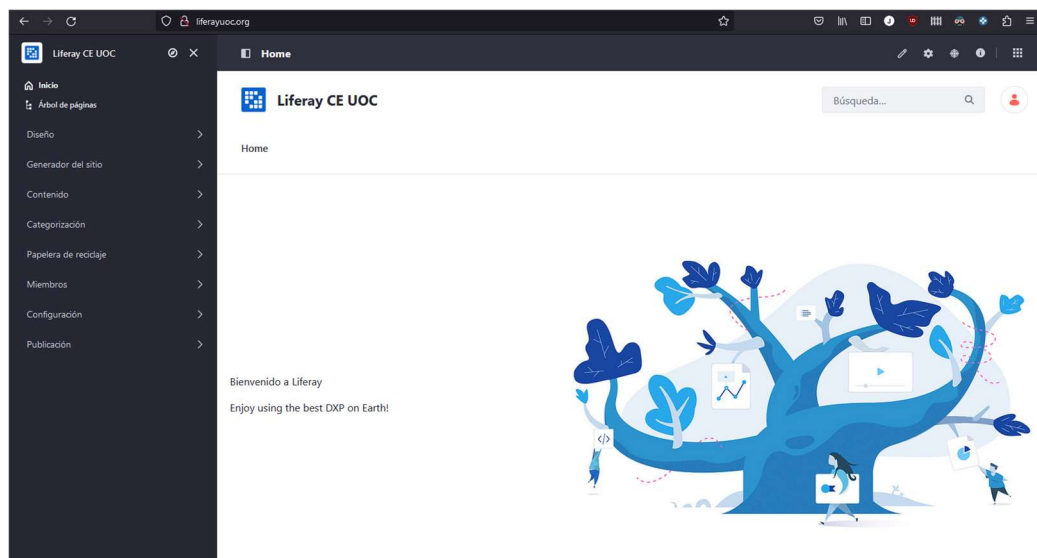


Figura 13. Web del Portal de Liferay

Además, y como hemos mencionado antes, crearemos dos sitios web con los dos dominios que hemos mencionado asociados a las mismas (véase [Anexo 18](#)).

Una vez creados estos dos sitios web podemos acceder a ellos con el navegador:

<http://liferaypruebasuoc.org/>

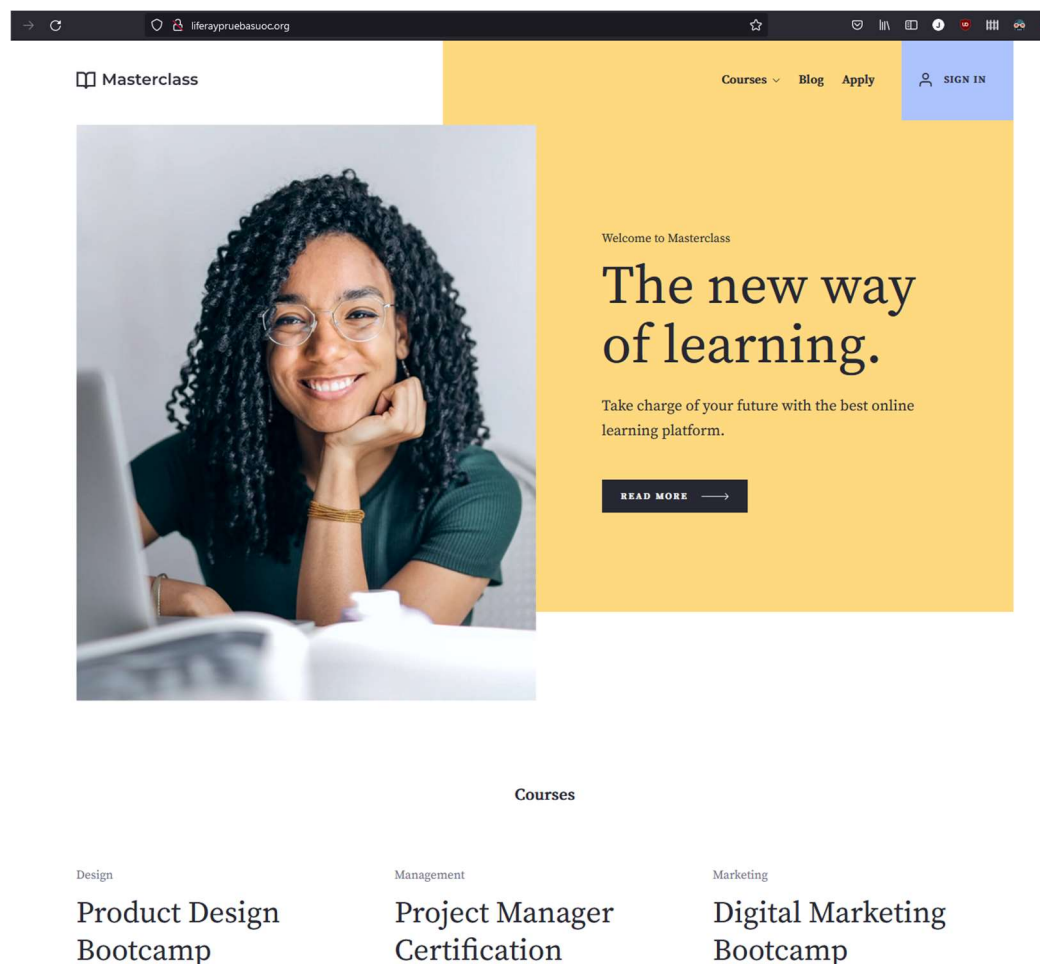


Figura 14. Sitio web de prueba creado en Liferay

<http://raylified2cuoc.org>

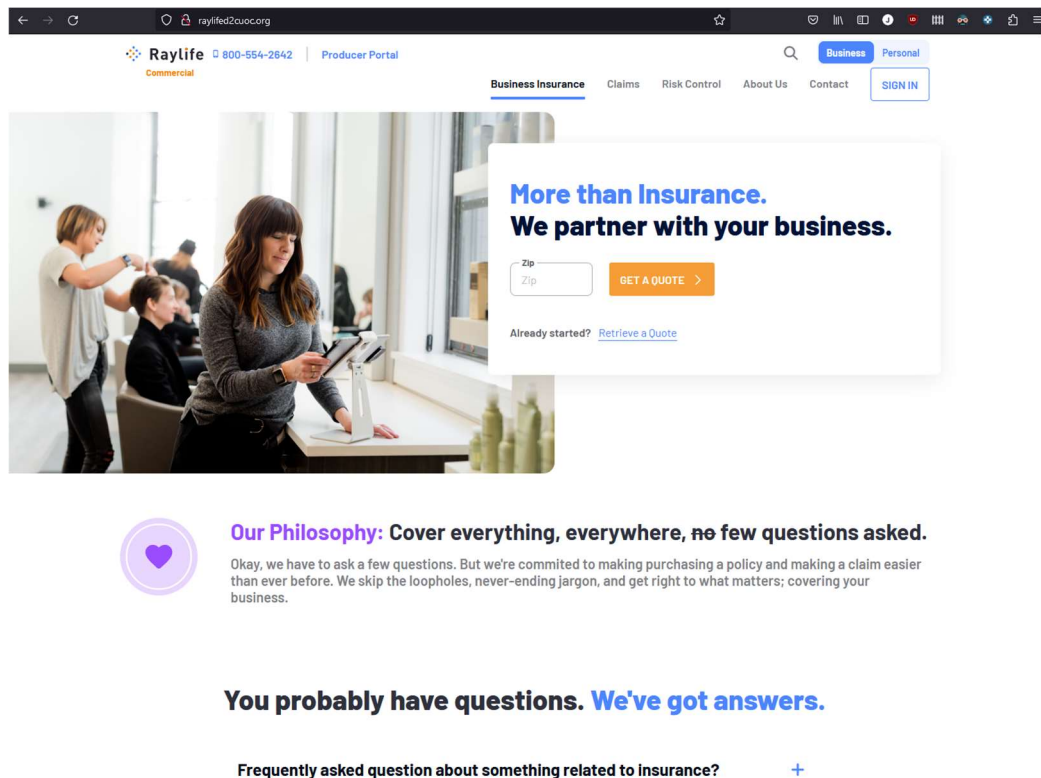


Figura 15. Otro sitio web de prueba creado en Liferay

Hay que tener en cuenta, que entrando por el dominio creado desaparece el menú de administración, lo cual es algo completamente normal, así se nos permite crear sitios para usuarios finales, mientras la administración queda aislada por otro dominio e incluso podría definirse para entrar desde una Intranet en vez desde Internet, con lo que tendríamos la administración aislada del exterior por seguridad.

Otra prueba que podemos hacer es la de añadir contenido, que era el otro objetivo principal final de la empresa (véase [Anexo 19](#)), desde la misma web como se puede ver más abajo:

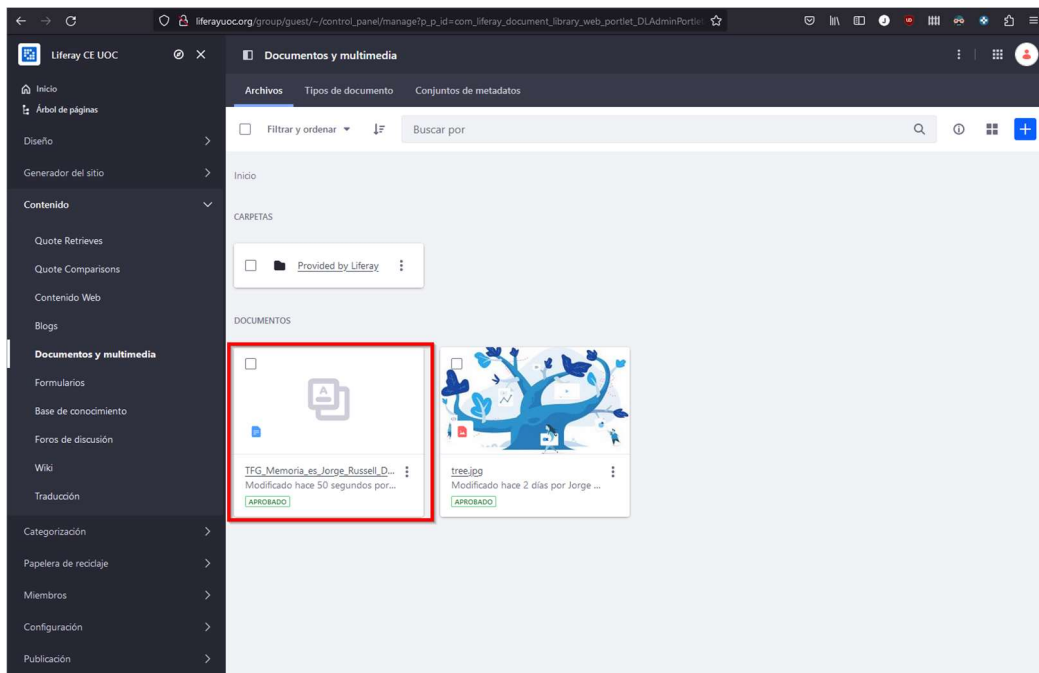


Figura 16. Contenido subido a Liferay

2.5 Alta Disponibilidad, Escalabilidad y Rendimiento mediante una arquitectura en clúster

En este capítulo montaremos la infraestructura necesaria para que exista la alta disponibilidad de la parte web y aplicativa de la plataforma. El software de balanceo HAProxy se encargará de repartir la carga entre los servidores web Apache y estos a su vez entre los servidores Tomcat que se encargan de ejecutar el Liferay en cada máquina. A su vez, también duplicaremos, o triplicaremos (véase [capítulo 2.5.3](#)), la capacidad del servidor de búsquedas, Elasticsearch, para dotarlo también de una alta disponibilidad.

Para lograr este objetivo habrá que reconfigurar las instancias simples de cada uno de los servicios y configurarlas para que sean dobles (o triples, en el caso del servidor de búsquedas) y que trabajen juntas y de forma coordinada. Para ello, cada uno de los servicios requiere configuraciones específicas para esta labor, ya sea para redirigir el tráfico, como el caso de los servidores web Apache, como para que las instancias se vean entre ellas y sincronicen datos, como los Liferay y los Elasticsearch.

A continuación, explicaremos como se debe configurar la infraestructura con el fin de conseguir este objetivo.

2.5.1 Configuración de una nueva máquina web y de aplicativos en clúster

Para conseguir un clúster debemos añadir una nueva máquina con las instancias web y aplicativas agregándola de tal manera que se forme ese clúster (grupo) con la otra máquina existente. Para ello, procederemos a clonar el servidor (véase [Anexo 2](#)) que contenía el servicio web Apache,

el Liferay y el Elasticsearch, “apacheliferay01” y asignaremos una nueva IP estática (véase [Anexo 3](#)) distinta en la misma subred y que antes ya habíamos reservado, la 192.168.1.21, de tal manera que no haya conflictos de red.

Levantaremos la máquina y procedemos a reconfigurar sus servicios como veremos en los siguientes capítulos.

2.5.2 Configuración de balanceo de carga y persistencia de sesión en los servidores web Apache y Tomcat que ejecutan Liferay

Para que haya una alta disponibilidad proporcionada por los servidores web Apache y que, a su vez, provean de la capacidad de balanceo de peticiones web (HTTP) a los servidores Tomcat que se ejecutan en las mismas máquinas, deberemos modificar la configuración de las instancias de los servidores web Apache (véase [Anexo 20](#)) de ambas máquinas (apacheliferay01 y apacheliferay02) para que puedan balancear la carga entre esos dos servidores del aplicativo Tomcat, que ejecutan sus respectivos portales Liferay. Además, tendremos que modificar ligeramente la configuración de los servidores Tomcat (véase [Anexo 20](#)) para mantener la persistencia de sesión³¹, de tal manera que, si un cliente está siendo servido por uno de ellos y hay un fallo en el servicio, el sistema de alta disponibilidad haga el cambio al otro Tomcat y se mantengan los datos de la sesión del usuario, así como su trabajo.

Gracias a esta configuración, y con el servicio de balanceo de HAProxy, por delante de ellos, se distribuirá la carga de las peticiones web que lleguen a esas instancias web que, a su vez, repartirán estas peticiones, se dotará de un sistema que detecta si se ha caído una instancia de Tomcat desde los Apaches y, de esta manera, los mismos, puedan dirigir las peticiones al nodo de Liferay que esté activo en ese momento. Todo ello sin que se pierda el trabajo que un usuario esté realizando en caso de fallo de cualquiera de los servidores (Apache o Tomcat), mientras haya alternativa ejecutándose.

Una vez arrancado el Tomcat se puede comprobar que la persistencia de sesión³¹ está funcionando, entrando en la web principal con un navegador, activando las herramientas de desarrollo (tecla F12). En las Cookies se puede observar el valor del Tomcat al que estamos conectados al ver el valor de JSESSIONID que es “jvm1”.

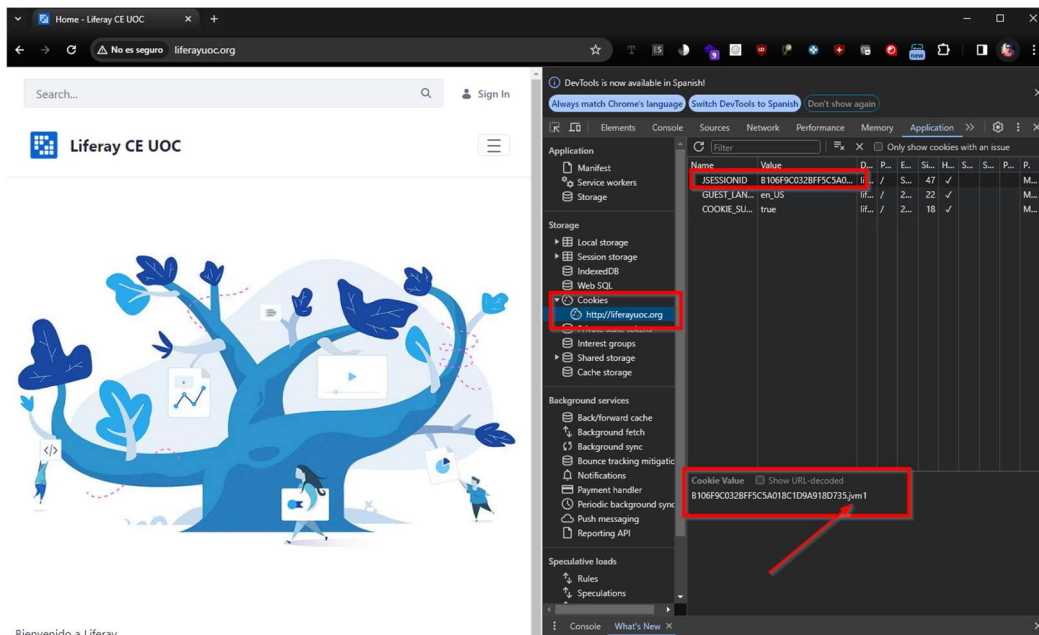


Figura 17. Comprobación de persistencia de sesión

Es decir, nos está sirviendo el contenido el Tomcat de la máquina “apacheliferay01”.

Si este Tomcat fallara por cualquier razón la Cookie con el JSESSIONID cambiaría a la otra máquina y mostraría ese número, pero acabado en “jmv2” y se mantendrían los datos de sesión de usuario.

2.5.3 Configuración en clúster de los servidores de búsqueda Elasticsearch

Para poder configurar en un clúster los servidores de búsqueda, para nuestro caso, y según la documentación de Elasticsearch³², deberemos tener en cuenta que Elasticsearch funciona de tal manera que uno de los nodos tiene que ser maestro y el resto funcionan como esclavos. El sistema de elección de un nodo maestro es equivalente al problema de los generales Bizantinos³³, es decir, se necesitan al menos tres nodos para poder elegir al nodo maestro y en nuestro caso tenemos dos nodos (uno en cada máquina “apacheliferay01” y “apacheliferay02”).

Elasticsearch permite funcionar con dos nodos, configurando uno de ellos como maestro y el otro como nodo de réplica de datos. Pero esta configuración tiene un problema, que sólo soportaría la caída del nodo de réplica de datos, si se cayese el nodo maestro, entonces el clúster de Elasticsearch dejaría de funcionar y afectaría al Liferay, que no podría tener todas las funcionalidades e incluso podría provocar que no funcionase en absoluto.

No obstante, esta problemática se puede resolver de dos maneras, instalando y configurando un tercer nodo y que los tres actúen con los tres roles idénticos, maestros, de datos y con capacidad de elegir otro maestro, y la otra forma, es que dos de los nodos sean maestros elegibles y de

réplica de datos, y el tercer nodo sea un maestro sin capacidad de réplica de datos, pero que actúe como nodo de desempate para elegir otro uno los maestros.

Elegiremos la segunda opción, usar el tercer nodo como nodo de desempate mientras los otros dos hacen de maestros elegibles y con capacidad de réplica de datos. La razón de elegir esta solución es que este nodo, al no replicar datos, usa menos recursos, por tanto, lo podemos instalar en la máquina en la que tenemos el MySQL sin que afecte demasiado a su rendimiento.

Para esto, lo que se hará es configurar Elasticsearch en clúster (véase [Anexo 21](#)) con un nodo maestro y de réplica de datos en “apachelifera01”, otro igual en “apachelifera02” y un tercero en la máquina “mysql”.

Una vez comprobado el correcto funcionamiento del clúster de Elasticsearch, se configurarán la instancia doble de Liferay (véase [Anexo 14](#)) en las máquinas “apachelifera01” y “apachelifera02” para que puedan conectarse a los tres nodos de Elasticsearch.

Cuando cualquiera de las máquinas de Liferay se levante se conectará a los nodos de Elasticsearch y trabajará con el nodo que sea elegido maestro. Esta configuración dota al Liferay de una Alta Disponibilidad del servicio de búsqueda de tal forma que mientras esté ejecutándose uno de los nodos elegibles como maestro, se podrá funcionar con normalidad.

Aquí podemos ver como Liferay detecta los 3 nodos del servicio de búsqueda:

The screenshot shows the Liferay administration interface for search configuration. At the top, there are tabs for 'Conexiones', 'Acciones de indexación', and 'Asignaciones de campos'. The 'Conexiones' tab is active. Below it, a summary box shows 'PROVEEDOR DE MOTOR DE BÚSQUEDA' as 'Elasticsearch' and 'VERSIÓN DEL CLIENTE' as '7.17.14'. Underneath, a section titled 'Conexiones activas' shows one active connection: 'LiferayElasticsearchCluster' with a status of 'ESTADO: GREEN'. This connection is labeled as 'REMOTE' and has 'LEÍDO' and 'ESCRITURA' permissions. A dropdown menu for 'NODOS (3)' is expanded, showing a table with the following data:

Nombre	Versión
es-node-1	7.17.14
es-node-2	7.17.14
es-node-3	7.17.14

Figura 18. Nodos de Elasticsearch detectados en Liferay

2.5.4 Configuración en clúster del Portal Liferay

La configuración en clúster con una instancia doble de Tomcat Liferay es relativamente, sencilla (véase [Anexo 22](#)). Se realiza una configuración mediante un enlace por UDP³⁴. Este enlace envía los datos entre los dos nodos para que se coordinen y sincronicen datos, como por ejemplo contenido web que se cree en uno de ellos y se refleje en el otro si otro usuario es redirigido a él.

Una vez configurados ambos servidores Tomcat-Liferay se pueden iniciar ambos servicios, y estos intercambiarán mensajes de coordinación de nodos en sus logs (véase [Anexo 22](#)), lo que nos indicará que está en correcto funcionamiento.

Una vez comprobado que ambos nodos de Tomcat están levantados si entramos con distintos navegadores, en algún momento se nos redirigirá a otro de los nodos como se puede ver en el valor de la Cookie de Liferay y su JSESSIONID que muestra que estamos en el nodo secundario al mostrar el valor “jvm2”

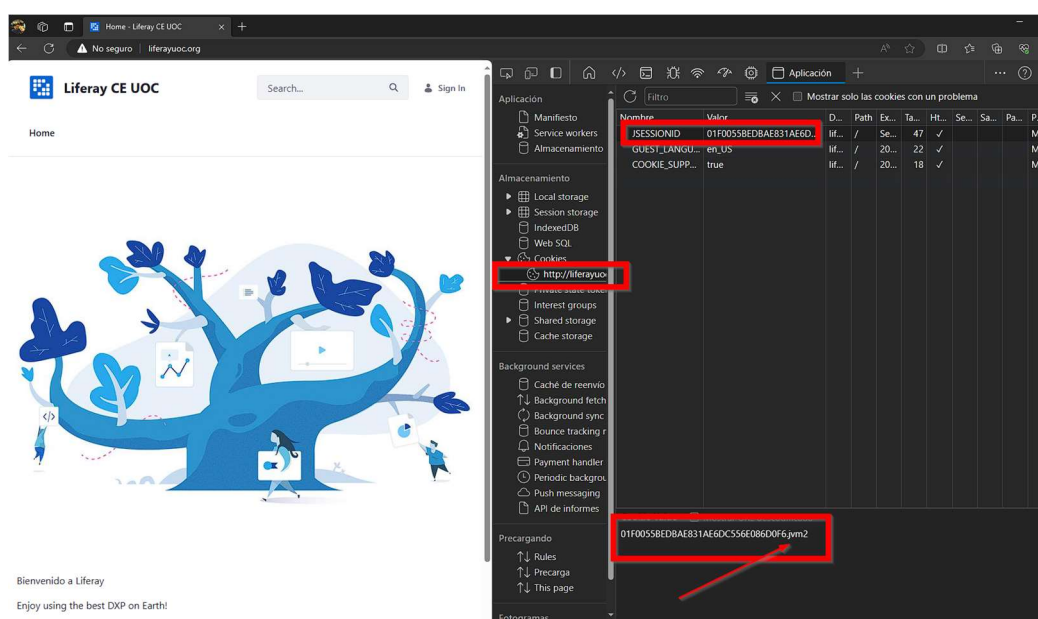


Figura 19. Comprobación de persistencia de sesión en nodo secundario

2.5.5 Pruebas de Alta Disponibilidad del clúster del Portal Liferay

Para comprobar el correcto funcionamiento basta con estar conectado por la web comprobar mediante la JSESSIONID al servidor que estamos conectados y posteriormente parar ese servidor, volver a recargar la web y verificar que cambia el valor, reflejando que el sistema realiza un balanceo del servicio.

Se puede hacer la misma prueba apagando uno de los servidores web, recargando la web y viendo que seguimos teniendo servicio y repitiendo lo mismo, pero al revés, levantamos el servidor web parado y apagamos el que estaba encendido. De esa forma se comprobará el balanceo del servicio.

Todo esto se verá reflejado en los logs (véase [Anexo 23](#)) de los sistemas mencionados en los que se podrá ver cómo, por ejemplo, en el log del HAProxy se ve como las peticiones van a un servicio web Apache u a otro en función de esa posible caída. Algo análogo pasaría en los logs de los servidores web Apache si se cayera uno de los Tomcat, y a su vez en los logs de Liferay también se vería la desconexión del otro nodo de Liferay.

Una vez realizadas estas pruebas queda garantizada la capacidad de Alta Disponibilidad del portal de Liferay.

2.6 Seguridad

En este capítulo trataremos de securizar el sistema de tal forma que se simule que está expuesto a Internet con las medidas de seguridad necesarias para proteger la infraestructura y la plataforma de posibles amenazas exteriores.

Para ello se configurarán, entre otras cosas, el servicio de Firewall de la máquina “haproxy”, ya que simula que es el balanceador y cortafuegos expuestos a Internet y además se configurará el servicio de HAProxy para aceptar comunicaciones HTTPS seguras y para ello se instalará un certificado mediante el cual se cifrarán dichas comunicaciones.

2.6.1 Configuración del servicio de Firewall (Cortafuegos)

Configuraremos políticas de seguridad de Firewall³⁶ (véase [Anexo 4](#)) que permitan sólo el tráfico HTTP y HTTPS en la máquina “haproxy”, ya que serán las que están involucradas en las comunicaciones desde un cliente web de un usuario exterior y el servicio web y Liferay que estarán corriendo dentro de la infraestructura y que gestiona dicha máquina.

Una vez establecidas estas reglas, el tráfico estará permitido a la par que el servicio de Firewall previene de cualquier otro intento de acceso a un puerto no autorizado y bloquear los mismos.

2.6.2 Configuración del servicio de HAProxy para gestionar tráfico web cifrado

En este capítulo vamos a configurar el servicio de HAProxy no sólo para que acepte tráfico HTTPS de tal manera que esté cifrado, sino que, además, será un punto de terminación SSL³⁷. Este tipo de terminaciones da una serie de ventajas, aparte de ser el punto único donde se encripta y desencripta el tráfico, lo cual nos da ventajas para gestionarlo, cuando operas con una granja de servidores a la que balancear el tráfico web,

como es nuestro caso, eliminamos el coste de rendimiento de encriptado y desencriptado en los servidores donde reside el servicio web, porque ya no requiere que se expongan estos servidores a Internet directamente, dejando que sea el HAProxy el que esté expuesto y porque ya no es necesario configurar cada servidor web (un VirtualHost en el puerto 443 con su certificado) para este tipo de comunicación.

El prerrequisito para gestionar el tráfico web cifrado es que debemos tener un certificado que se usa para cifrar los datos, por lo que tenemos dos opciones, o generamos uno nosotros autofirmado para tal efecto, o podríamos pedir uno a una autoridad certificadora (CA) de pago o de alguna que los provee de forma gratuita.

En nuestro caso generaremos un certificado autofirmado³⁸ (véase [Anexo 24](#)), por sencillez, y lo dejaremos instalado y configurado en la máquina de “haproxy” (véase [Anexo 25](#)).

El certificado con su parte pública y privada correspondiente lo unificamos en un fichero único (liferayuoc.org.pem) que dejaremos en “/etc/ssl/certs/” y que solo el usuario “root” pueda leerlo.

Posteriormente configuraremos el HAProxy (véase [Anexo 25](#)) para que, lea el certificado configurado para cifrar las peticiones web, que acepte también tráfico por HTTPS, aparte del HTTP, y para un extra de seguridad, obligaremos a redirigir cualquier petición por HTTP, a HTTPS. De esta manera se forzará que el tráfico esté cifrado siempre, lo cual nos da esa capa de seguridad que estábamos buscando.

Finalmente habrá que añadir la configuración a Liferay para especificar que el tráfico web va a ser a partir de ahora mediante HTTPS (véase [Anexo 26](#)).

Una vez reiniciado todos los servicios y si probamos a entrar en cualquiera de las webs que tenemos, incluso entrando por HTTP, se nos redirigirá a HTTPS (aunque nos dará una advertencia de seguridad al haber un certificado autofirmado, la cual tendremos que aceptar el riesgo y continuar) y se podrá acceder a la web.

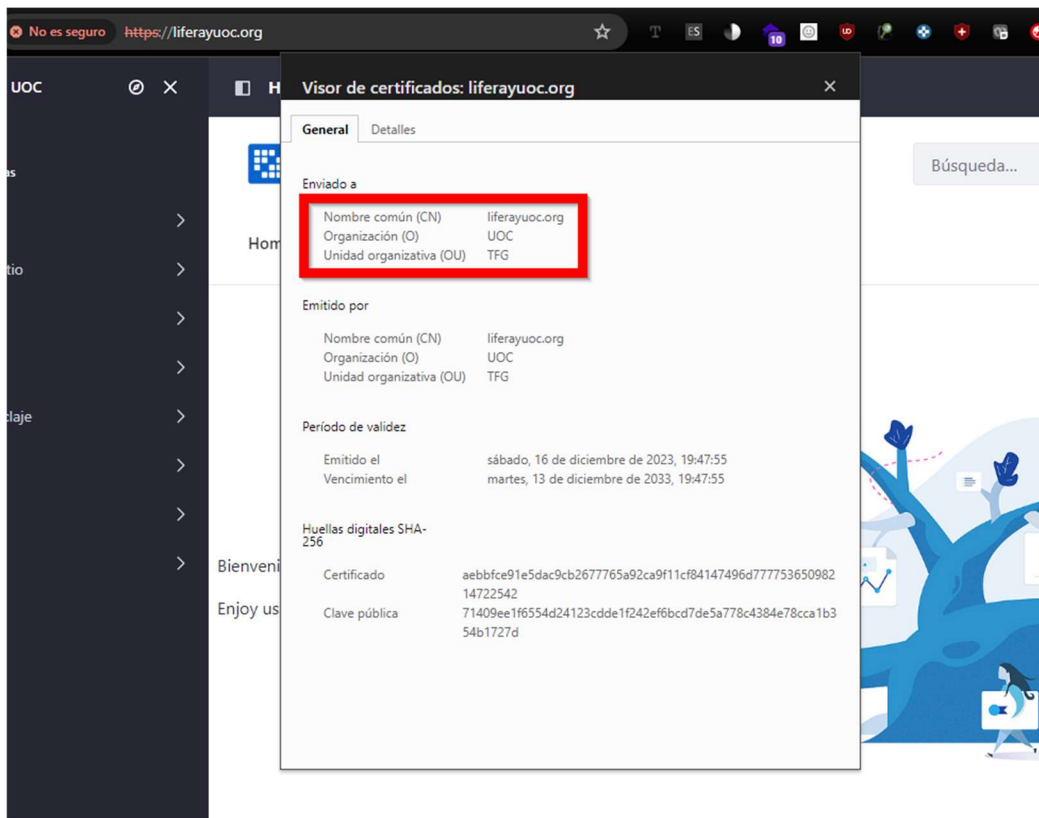


Figura 20. Certificado SSL de la web de Liferay

2.7 Segmentación de la red

En este capítulo estableceremos una segmentación de la red, definiendo una DMZ³⁹ (Demilitarized Zone) y una red de aplicativo/backend, mediante la configuración de interfaces de red extra en las máquinas virtuales. Para ello, configuraremos los servicios web, de los aplicativos implicados en sus correspondientes subredes.

De esta manera se aumentará el aislamiento de servicios que a su vez logran que se añada una capa extra de seguridad en caso de ataque desde el exterior y agregar una gestión eficiente de los recursos.

Lo ideal, en una red corporativa, sería que cada máquina virtual tuviera varios interfaces de red, cada uno en su segmento de red, de tal manera que habría un segmento para una DMZ³⁹ (Demilitarized Zone), que es una red perimetral que se implanta para proteger la LAN (red local), y luego segmentar internamente nuestra LAN con varias zonas en la que debe estar asignado cada servicio, como por ejemplo un segmento para servidores web, otro para servidores de aplicativo, otra para bases de datos y luego otras de administración, backup, etc.

2.7.1 Asignación de una nueva interfaz de red a la máquina HAProxy

En nuestro caso, por simplificación y ya que estamos limitados a que el servicio web esté ejecutándose en el mismo servidor que el aplicativo y

que necesitaremos poder monitorizar más adelante los servicios por las IPs actuales, vamos a definir un nuevo segmento de red 10.0.2.0/24 en VirtualBox (véase [Anexo 27](#)) que será nuestra DMZ por delante del servidor de HAProxy. A su vez, daremos al servidor de HAProxy una nueva IP 10.0.2.4/24 en una nueva interfaz de red que instalaremos para en ese nuevo segmento, posteriormente, tendremos que configurar una IP en nuestro host anfitrión dentro de esa red 10.0.2.1/24 y, por último, una ruta persistente en nuestro host físico para poder llegar a esa IP.

2.7.2 Cambio de IP frontal del servicio de HAProxy

Finalmente se reconfigurará el servicio de HAProxy para cambiar la IP que se usa para el balanceo para la nueva IP que se ha configurado.

También hay que tener en cuenta que el fichero “etc/hosts” de nuestro host anfitrión, debe reflejar el nuevo cambio de IP para las webs que hemos definido (véase [Anexo 16](#)) de tal manera que haya una entrada como esta:

```
10.0.2.4 liferayuoc.org liferaypruebasuoc.org raylifed2cuoc.org
```

Esta configuración lo que hace es que la IP frontal del balanceador esté en otra red que, en vez de esta, podría ser directamente una IP pública para dar acceso al servicio o una IP que se alcance mediante un NAT desde un Firewall, que es el que tiene la IP pública para exponer a Internet el servicio.

De esta manera tendremos el aislamiento deseado entre el servicio interno e Internet para añadir la capa de seguridad que estábamos buscando.

2.8 Gestión de Backups (copias de respaldo)

En este capítulo se establecerá un sistema de gestión de copias de seguridad de los datos, tanto de la base de datos como del sistema de ficheros de red con el fin de garantizar la integridad de los datos y la continuidad del servicio en caso de cualquier fallo relacionado, de tal manera, que se puedan recuperar los mismos en tiempo y forma con el mínimo impacto sobre dicho servicio.

Nos interesará principalmente hacer backups del esquema de Liferay de base de datos “lportal” en el MySQL y de los ficheros de datos de Liferay que se generan en el NFS que están en la máquina “mysql” en la ruta “/mnt/liferay_data”.

En un entorno corporativo lo ideal sería llevarnos estos backups a otro NFS externo o si no mediante una copia a otro servidor remoto usando herramientas del sistema operativo como “rsync”, “scp” o similares. No obstante, en nuestro caso, y al no disponer de más recursos, los haremos en la propia máquina sirviendo de ejemplo.

Para ello recurriremos a soluciones sencillas y open-source, como para el caso de la base de datos MySQL que usaremos la herramienta AutoMySQL Backup⁴⁰, y para el caso del NFS usaremos la herramienta de compresión de archivos, “tar”, que nos aporta el propio sistema operativo y que permite comprimir directorios recursivamente de manera rápida y eficiente a otro punto del sistema, preservando permisos y fechas de los mismos.

2.8.1 Copias de respaldo con la herramienta AutoMySQL Backup

Esta herramienta open-source se puede descargar y configurar (véase [Anexo 28](#)) para que haga un backup específico del esquema de Liferay “lportal” al directorio que le especifiquemos, en este caso “/var/backup/db/” y con los parámetros indicados de preservación de los mismos ya sean diarios, semanales, etc.

Posteriormente, se puede ejecutar con el usuario “root” directamente, lo cual nos generará un backup en la ruta configurada y, además, lo podemos programar mediante el crontab (el programador de tareas del sistema operativo). En nuestro caso lo dejaremos configurado para ejecutarse a diario a las 2 de la mañana.

```
[root@mysql db]# crontab -e
# Daily 2:00AM MySQL Backups
0 2 * * * /usr/local/bin/automysqlbackup
/etc/automysqlbackup/automysqlbackup.conf
```

Y como se puede ver tras ejecutarse manualmente se generan los backups del esquema de base de datos.

```
[root@mysql ~]# cd /var/backup/db/
[root@mysql db]# ls -la *
daily:
total 0
drwxr-xr-x. 3 root root 21 Dec 18 20:30 .
drwxr-xr-x. 9 root root 105 Dec 18 19:31 ..
drwxr-xr-x. 2 root root 112 Dec 18 19:31 lportal

fullschema:
total 120
drwxr-xr-x. 2 root root 118 Dec 18 19:31 .
drwxr-xr-x. 9 root root 105 Dec 18 19:31 ..
-rw-r--r--. 1 root root 57501 Dec 18 19:24 fullschema_daily_2023-12-18_19h24m_Monday.sql.gz
-rw-r--r--. 1 root root 57501 Dec 18 19:31 fullschema_daily_2023-12-18_19h31m_Monday.sql.gz
```

Figura 21. Backups de base de datos MySQL

2.8.2 Copias de respaldo de ficheros con la herramienta tar

Los backups de los ficheros de liferay se pueden hacer y programar de forma sencilla con la ejecución de un comando con la herramienta del sistema operativo “tar”.

Para ejecutar y guardar en la ruta “/var/backup/nfs/” un backup del NFS de los ficheros de Liferay en “/mnt/liferay_data/” con la fecha del día, basta con ejecutar:

```
tar -czvf /var/backup/nfs/backup_$(date +%Y%m%d').tar.gz
/mnt/liferay_data
```

Este comando copiará todo el directorio recursivamente preservando fechas, enlaces y permisos al destino y guardando el mismo en un fichero comprimido con la fecha del día en que se hizo.

Para programarlo, bastará introducirlo también en el crontab (el programador de tareas) de root, a la misma hora que los backups de base de datos, esto nos permitirá tener un backup diario a la misma hora del sistema de ficheros de Liferay.

```
[root@mysql db]# crontab -e
# Daily 2:00AM Liferay Data NFS Backups
0 2 * * * tar -czvf /var/backup/nfs/backup_$(date +%Y%m%d').tar.gz
/mnt/liferay_data
```

Además, para que no se nos llene el disco con backups es recomendable que se borren ficheros de más de 60 días, para ello lo dejamos también programado como tarea en el crontab.

```
# Delete Liferay Data backups older than 60 days
0 3 * * * find /var/backup/nfs/ -type f -mtime +60 -delete
```

2.9 Monitorización del servicio

En este capítulo instalaremos y configuraremos, tanto en el host anfitrión (Windows), como en las máquinas virtuales un sistema para extraer métricas de estas con las que dotar y alimentar un sistema de monitorización de las máquinas y servicios desplegados. De esta manera, nos aseguraremos de que el sistema es estable, fiable y que proporciona los servicios para los que ha sido diseñado.

Para llevar a cabo la tarea utilizaremos dos herramientas de monitorización, extracción de métricas y análisis de datos open-source, las cuales combinaremos para conseguir nuestro objetivo, Prometheus⁴⁰ y Grafana⁴¹.

Prometheus es una herramienta de monitorización y alertado que recoge métricas usando agentes desplegados en máquinas y servicios de los cuáles se pretenda extraer dicha información con su marca de tiempo para poder ser analizada posteriormente.

Grafana es una herramienta de análisis de datos derivados de una extracción de métricas, la cual da sentido a dichos datos. Por otro lado,

nos permite monitorizar aplicaciones y recursos hardware con una serie de paneles de control que se pueden personalizar al gusto.

Se procederá por tanto a instalar Prometheus y Grafana como servidores en nuestro host anfitrión los cuales extraerán los datos de agentes dedicados de Prometheus que desplegaremos para cada servicio que queremos monitorizar (HAProxy, Apache, JMX para Tomcat y MySQL) y para las máquinas. Estos agentes levantan un puerto al que se puede acceder por HTTP y proporcionan métricas que pueden ser consumidas luego por el servidor de Prometheus y luego proporcionarnos gráficas mediante el servidor Grafana, ambos instalados y configurados en nuestro host anfitrión.

2.9.1 Configuración de agentes de Prometheus de extracción de métricas.

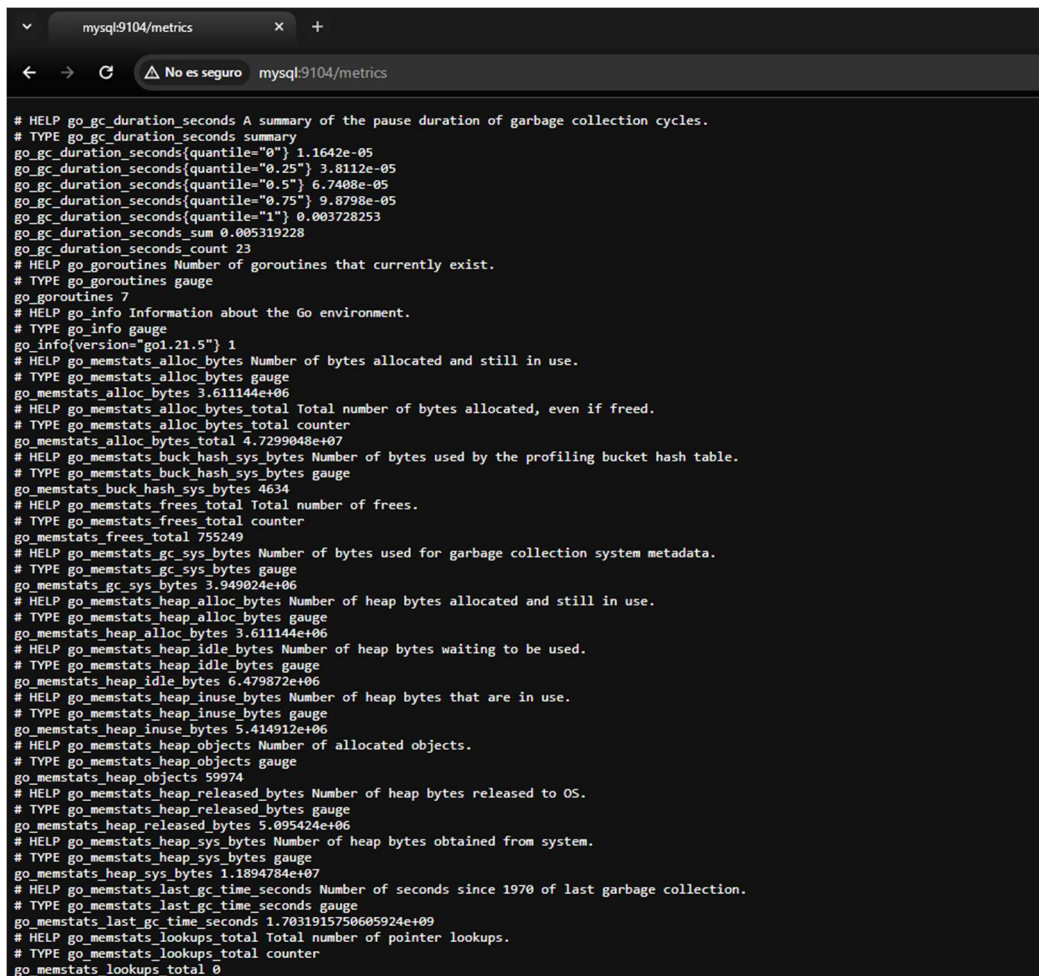
Prometheus dispone de distintos agentes en función de la necesidad de extracción de determinadas métricas, en nuestro caso instalaremos y configuraremos agentes, tanto para monitorizar los recursos del sistema operativo de las máquinas en cada máquina, como para el resto de los servicios, los cuales tienen distintos agentes en cada caso:

- Recursos de Sistema Operativo (CPU, Memoria, espacio en disco, etc): `node_exporter` (véase [Anexo 29](#))
- Recursos de balanceador HAProxy: `haproxy_exporter` (véase [Anexo 30](#))
- Recursos de servidor web Apache: `apache_exporter` (véase [Anexo 31](#))
- Recursos de Tomcat-Liferay: JMX Exporter (véase [Anexo 32](#))
- Recursos de MySQL: `mysql_exporter` (véase [Anexo 33](#))

El `node_exporter` se instalará en todas las máquinas para obtener las métricas de los recursos hardware de todas ellas y los otros se instalarán específicamente donde esté corriendo el servicio análogo, es decir, por ejemplo, el `apache_exporter` sólo se instalará en las máquinas que tienen los servidores web Apache y el JMX Exporter en las máquinas que tienen los servidores Tomcat.

Por otro lado, y como nota, decir que se podría configurar un agente para monitorizar Elasticsearch, pero no hay disponible binario de CentOS, aunque se podría compilar y la posterior instalación y configuración sería muy similar al del resto de agentes, pero como ya podemos ver el estado de Elasticsearch desde dentro de la consola de Liferay, no vemos necesaria esta instalación.

Una vez instalados estos agentes, se puede acceder a las métricas directamente vía web y se configuran estos puntos de acceso, más adelante en el servidor de Prometheus para poder hacer consultas y luego pasarlos a Grafana.



```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 1.1642e-05
go_gc_duration_seconds{quantile="0.25"} 3.8112e-05
go_gc_duration_seconds{quantile="0.5"} 6.7408e-05
go_gc_duration_seconds{quantile="0.75"} 9.8798e-05
go_gc_duration_seconds{quantile="1"} 0.003728253
go_gc_duration_seconds_sum 0.005319228
go_gc_duration_seconds_count 23
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 7
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.21.5"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 3.611144e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 4.7299048e+07
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 4634
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 755249
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 3.949024e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
go_memstats_heap_alloc_bytes 3.611144e+06
# HELP go_memstats_heap_idle_bytes Number of heap bytes waiting to be used.
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 6.479872e+06
# HELP go_memstats_heap_inuse_bytes Number of heap bytes that are in use.
# TYPE go_memstats_heap_inuse_bytes gauge
go_memstats_heap_inuse_bytes 5.414912e+06
# HELP go_memstats_heap_objects Number of allocated objects.
# TYPE go_memstats_heap_objects gauge
go_memstats_heap_objects 59974
# HELP go_memstats_heap_released_bytes Number of heap bytes released to OS.
# TYPE go_memstats_heap_released_bytes gauge
go_memstats_heap_released_bytes 5.095424e+06
# HELP go_memstats_heap_sys_bytes Number of heap bytes obtained from system.
# TYPE go_memstats_heap_sys_bytes gauge
go_memstats_heap_sys_bytes 1.1894784e+07
# HELP go_memstats_last_gc_time_seconds Number of seconds since 1970 of last garbage collection.
# TYPE go_memstats_last_gc_time_seconds gauge
go_memstats_last_gc_time_seconds 1.7031915750605924e+09
# HELP go_memstats_lookups_total Total number of pointer lookups.
# TYPE go_memstats_lookups_total counter
go_memstats_lookups_total 0
```

Figura 22. Métricas de agentes de Prometheus

2.9.2 Configuración del servidor de métricas Prometheus

El servidor Prometheus se encarga de recopilar las métricas que proveen los puntos de acceso de los agentes instalados en las máquinas virtuales. Se añaden dichos puntos al fichero de configuración con los nombres y etiquetas que veamos oportunos y esto nos permite luego usar filtros y órdenes de búsqueda para personalizar el tipo de métricas que se deseen ver, e incluso se podrían configurar alertados (vía web, mail, etc) con otros módulos.

La instalación de Prometheus (véase [Anexo 34](#)) se realiza sobre el host anfitrión Windows, se añaden a su fichero de configuración los puntos de acceso de métricas de los agentes instalados en las máquinas virtuales y al iniciar el servidor Prometheus, estos son visibles mediante la web que levanta en nuestro host. Desde ahí podemos consultar dichas métricas y aplicar los filtros y operaciones que veamos oportunas para consultar, por ejemplo, datos de consumo de CPU, memoria, etc.

Aquí podemos ver el servidor en marcha con los puntos de acceso a métricas de nuestros servidores y aplicativos e indicando su correcto funcionamiento.

The screenshot shows the Prometheus Targets page with a search bar and a table of targets. The table columns are Endpoint, State, Labels, Last Scrape, Scrape Duration, and Error. The targets are grouped by service name.

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
apache_server (2/2 up)					
http://apacheliferay02:9117/metrics	UP	instance="apacheliferay02:9117" job="apache_server"	5.910s ago	1.955ms	
http://apacheliferay01:9117/metrics	UP	instance="apacheliferay01:9117" job="apache_server"	14.208s ago	0.977ms	
haproxy_service (1/1 up)					
http://haproxy:9101/metrics	UP	instance="haproxy:9101" job="haproxy_service"	10.940s ago	0.978ms	
liferay_server (2/2 up)					
http://apacheliferay01:9010/metrics	UP	instance="apacheliferay01:9010" job="liferay_server"	849.000ms ago	19.543ms	
http://apacheliferay02:9010/metrics	UP	instance="apacheliferay02:9010" job="liferay_server"	5.696s ago	20.520ms	
mysql_server (1/1 up)					
http://mysql:9104/metrics	UP	instance="mysql:9104" job="mysql_server"	2.881s ago	122.143ms	
node_exporter_apacheliferay01 (1/1 up)					
http://apacheliferay01:9100/metrics	UP	instance="apacheliferay01:9100" job="node_exporter_apacheliferay01"	3.275s ago	8.794ms	
node_exporter_apacheliferay02 (1/1 up)					
http://apacheliferay02:9100/metrics	UP	instance="apacheliferay02:9100" job="node_exporter_apacheliferay02"	5.629s ago	9.772ms	
node_exporter_haproxy (1/1 up)					
http://haproxy:9100/metrics	UP	instance="haproxy:9100" job="node_exporter_haproxy"	11.498s ago	8.795ms	
node_exporter_mysql (1/1 up)					
http://mysql:9100/metrics	UP	instance="mysql:9100" job="node_exporter_mysql"	10.969s ago	8.795ms	

Figura 23. Web de monitorización de Prometheus

2.9.2 Configuración del servidor de análisis de métricas Grafana

El servidor Grafana se encarga de recopilar las métricas que obtiene previamente Prometheus para luego poder hacer un análisis de datos de dichas métricas mediante el uso de paneles de control personalizables. Por otro lado, nos permite monitorizar aplicaciones y recursos hardware mediante el uso de esos mismos paneles.

La instalación de Grafana (véase [Anexo 35](#)) se realiza sobre el host anfitrión Windows. Una vez instalado se accede a su interfaz vía web y se añaden desde él los orígenes de datos de Prometheus y los paneles de control (dashboards) que deseemos y, en este caso, agregaremos todos los que corresponden a los agentes instalados, es decir, uno para las máquinas (node exporter), para los apaches (apache exporter), para el HAProxy (haproxy exporter), para el MySQL (mysql exporter) y para la máquina virtual Java del Tomcat que ejecuta el Liferay (JMX Exporter).

Cuando los paneles personalizados están instalados, tenemos acceso al análisis de las métricas que se obtienen con Prometheus mediante gráficas que indican el estado de los aplicativos, servicios y hardware de las máquinas virtuales de una forma gráfica e intuitiva, como podemos ver más abajo para todos los paneles importados.

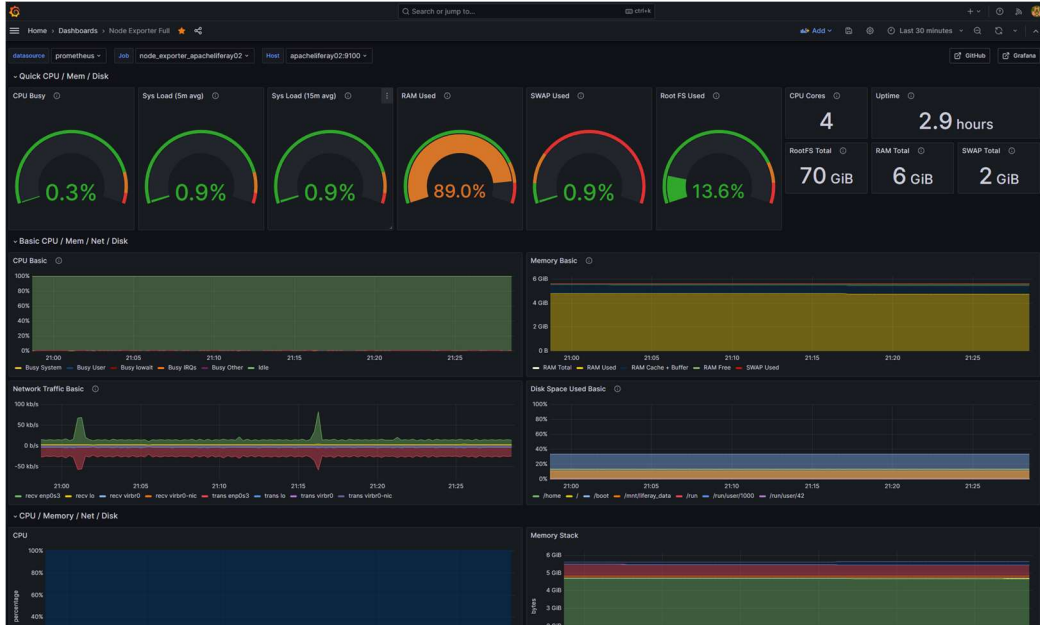


Figura 24. Panel de control de Node Exporter con gráficas de rendimiento de la máquina

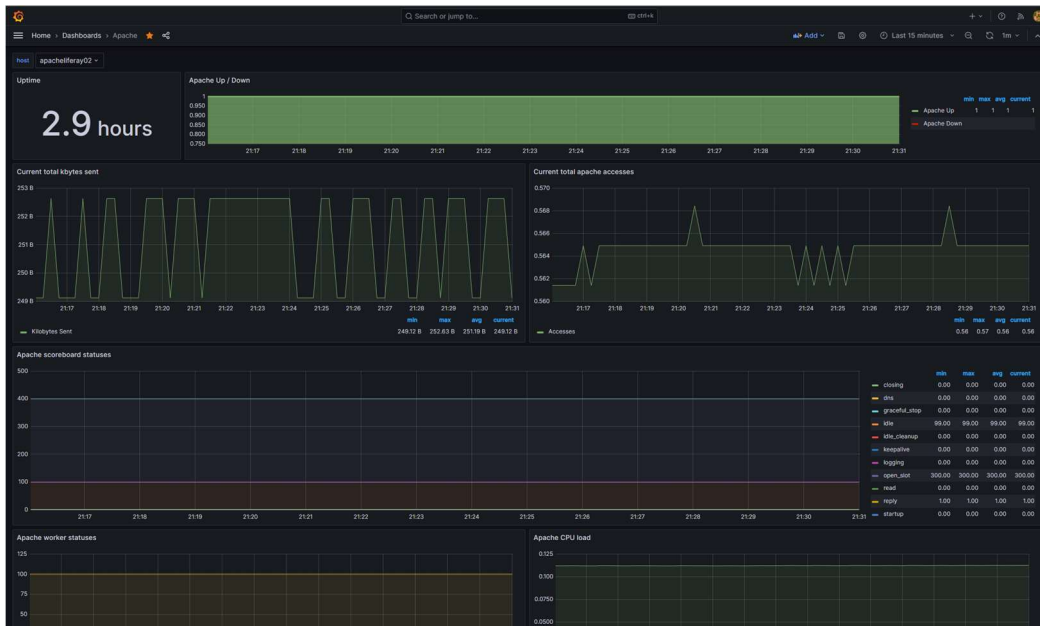


Figura 25. Panel de control de Apache Exporter con gráficas de rendimiento del servidor web

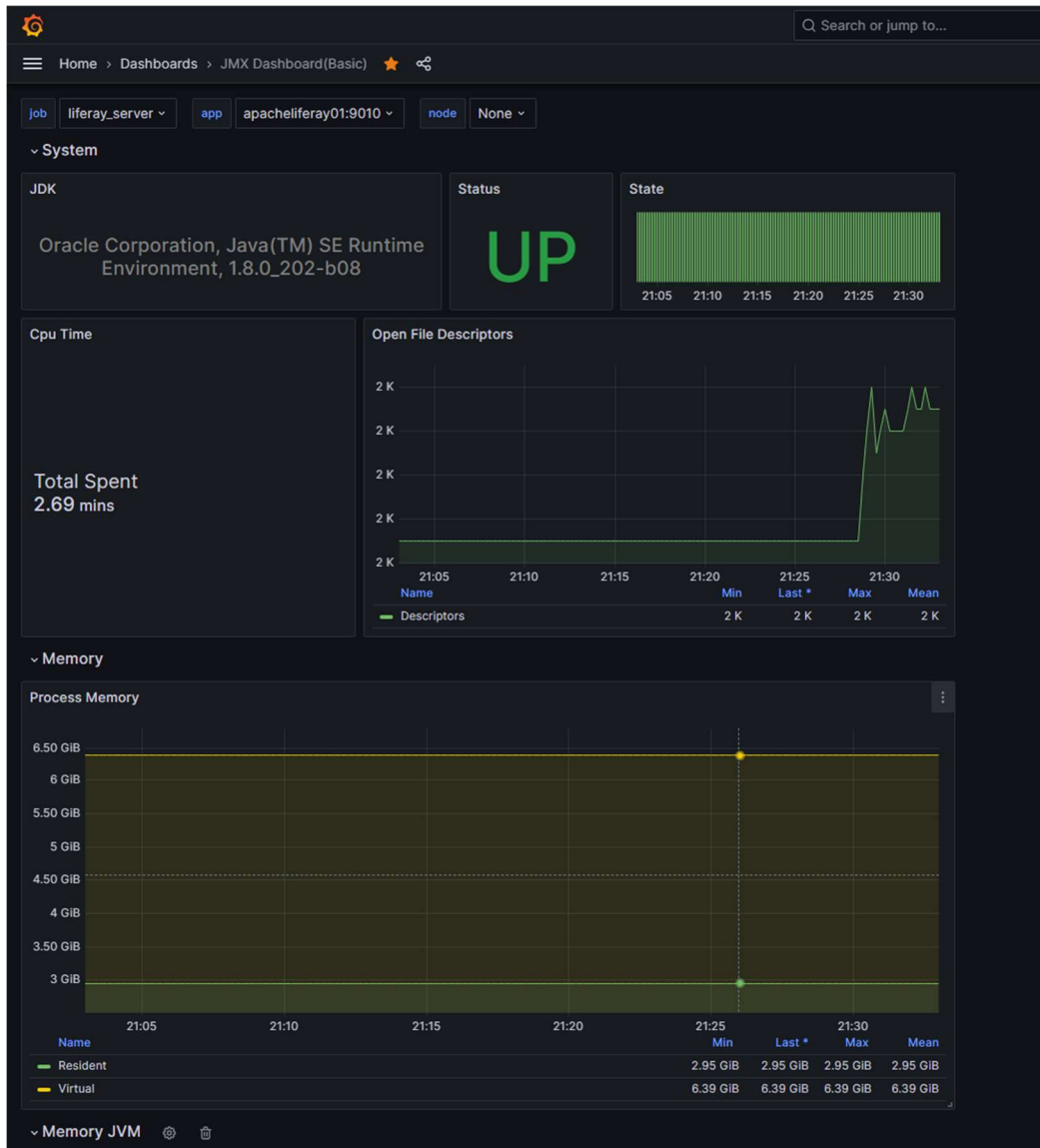


Figura 26. Panel de control de JMX Exporter con gráficas de rendimiento de la máquina virtual Java

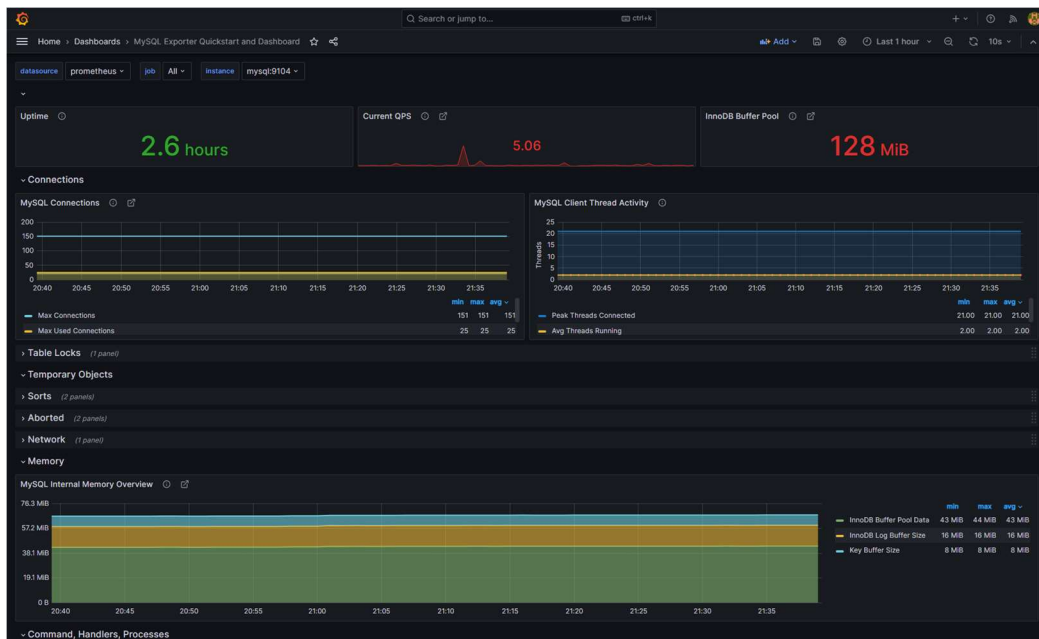


Figura 27. Panel de control de MySQL Exporter con gráficas de rendimiento de la base de datos

2.10 Documentación de explotación de servicio

En este capítulo daremos una documentación que describa la infraestructura con detalle y que proporcione una capacitación al usuario para el uso de esta, tal como el plan de contingencia en caso de fallos del sistema, así como un diagrama de red de la infraestructura.

De esta manera el usuario puede asegurarse de que se pueda administrar y mantener toda la plataforma de forma eficiente y efectiva.

La infraestructura está formada por:

- Balanceador de carga HAProxy:

Este balanceador está instalado en la máquina “haproxy”, recibe peticiones por una IP externa y se encarga de repartir las peticiones a las webs configuradas en los servidores web Apache que están por debajo suyo. Si uno de los servidores falla, el sistema lo detecta y enviará las mismas al otro sin pérdida de servicio ni de datos de sesión del usuario.

Tiene configurado un sistema de cortafuegos (firewall) para protegerlo de amenazas externas y tiene habilitado y forzado el tráfico cifrado mediante HTTPS.

Para verificar el correcto funcionamiento y la operación de servicio, véanse los anexos [4](#), [6](#), [7](#) y [17](#).

- Servidores web Apache:

Estos servidores web están instalados en las máquinas “apacheliferay01” y “apacheliferay02”, están encargados de recibir las peticiones web y servir las webs que están configuradas en los servidores de Liferay. También reparten carga y en caso de caída de uno de los servidores de Liferay pueden detectarlo y enviar las peticiones al otro sin pérdida de servicio ni de datos de sesión del usuario.

Para verificar el correcto funcionamiento y la operación de servicio de ambos, véanse los anexos [5](#) y [17](#).

- Servidores de aplicativo Tomcat-Liferay:

Estos servidores de aplicativo están instalados en las máquinas “apacheliferay01” y “apacheliferay02”, están encargados de servir el contenido dinámico del aplicativo del portal digital Liferay.

Están configurados para sincronizar contenido entre ambos y son capaces de asumir la caída de uno de sus nodos para seguir sirviendo dicho contenido sin pérdida de datos entre ellos y manteniendo los datos de sesión del usuario, si se diera el caso.

Para verificar el correcto funcionamiento de ambos servidores y su operación de servicio, véanse los anexos [12](#), [22](#) y [23](#).

- Servidores de búsqueda Elasticsearch:

Estos servidores de búsqueda están instalados en las máquinas “apacheliferay01”, “apacheliferay02” y “mysql”. Están encargados de proporcionar al Liferay del servicio de búsqueda e indexación de los contenidos que se generan en Liferay.

Están configurados para sincronizar contenido entre dos de ellos, siendo el tercero un nodo de desempate para elegir cual es el nodo maestro, y son capaces de asumir la caída de uno de sus nodos de datos y el de desempate, para seguir sirviendo los datos de indexación sin que haya una pérdida de servicio total.

Para verificar el correcto funcionamiento de estos servidores y su operación de servicio, véanse los anexos [14](#), [21](#) y [23](#).

- Servicio de ficheros compartidos en red NFS

Este servidor se encuentra en la máquina “mysql” y se encarga de proporcionar un sistema de ficheros compartidos en red (NFS) para que desde los servidores de Liferay, y mediante un cliente de NFS se pueda acceder y escribir contenido de ficheros que se generen por parte de estos servidores.

Para verificar el correcto funcionamiento y operación de este servicio, véase el [Anexo 8](#).

- Servicio de base de datos MySQL

Este servidor se encuentra en la máquina “mysql” y se encarga de proporcionar una base de datos gestionada y necesaria para el servidor de Liferay.

Para verificar el correcto funcionamiento y operación de este servicio, véanse los anexos [9](#) y [10](#).

A continuación, tenemos el diagrama de red de la infraestructura

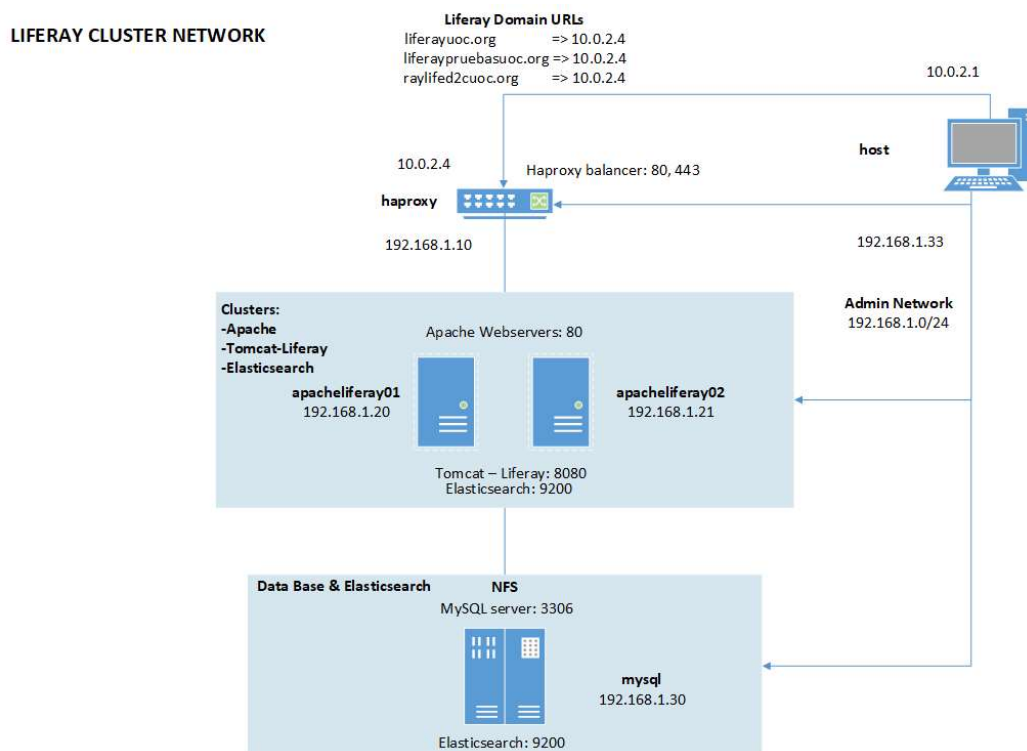


Figura 28. Diagrama de red de la infraestructura

La monitorización de servicios y máquinas y sus recursos se puede realizar a través de las consolas de Prometheus y Grafana (véanse anexos [34](#) y [35](#) y capítulo [2.9](#)).

En caso de contingencia con alguno de los servicios las máquinas pueden ser administradas vía SSH (utilizando el usuario y contraseña “centos:centos” y con algún cliente SSH como putty o MobaXterm) o directamente acceder a ellas vía consola de VirtualBox y verificar el funcionamiento de los servicios y operarlos como se indican en los anexos antes mencionados.

En el caso de que alguna de las máquinas estuviera caída, habría que comprobarlo y encenderla mediante la consola de VirtualBox.

Sólo se levantan manualmente los servicios de Elasticsearch y Liferay (el de Elasticsearch nos ha dado problemas configurarlo para que se arranque automáticamente y a causa de esto hemos dejado el de Liferay también de forma manual ya que depende de este) ya que, el resto lo hacen de forma automática al levantar la máquina.

2.11 Verificación de cumplimiento de objetivos y requisitos

En este capítulo se comprueba el funcionamiento del sistema mediante pruebas simulando posibles fallos y situaciones de tal manera que el

sistema siga funcionando adecuadamente para asegurar que la infraestructura diseñada cumpla los objetivos y requisitos establecidos al inicio del trabajo, así como, la satisfacción y eficiencia de esta.

2.11.1 Verificación de correcto funcionamiento de la plataforma

Estas pruebas se realizan exitosamente como se indica y se puede comprobar en el [Anexo 23](#) y con ellas se garantiza que el servicio funciona con la tolerancia a fallos esperada.

2.11.2 Verificación de cumplimiento de objetivos y requisitos.

Tras las anteriores pruebas y tras terminar todas las tareas queda verificado el cumplimiento de objetivos de todo el TFG, que consistía en conseguir poner en funcionamiento un portal digital con capacidades de CMS y de creación de sitios web, como es Liferay, y además que estuviera en Alta Disponibilidad y totalmente monitorizado.

3. Conclusiones

En este capítulo explicamos las conclusiones del TFG, describiendo las lecciones aprendidas, una reflexión crítica sobre el logro de los objetivos planteados, un análisis crítico sobre la planificación y metodología, evaluación de cambios realizados a lo largo del trabajo y las posibles líneas de trabajo futuro que no se han podido explorar y han quedado pendientes en el mismo.

3.1 Lecciones aprendidas

Se ha aprendido que es completamente posible implantar entornos de gestión de contenidos (CMS), en alta disponibilidad y con monitorización en entornos corporativos mediante software open-source, disminuyendo el coste al mínimo para el mantenimiento del mismo al ahórranos el coste derivado de posibles soluciones comerciales. Por otro lado, y obviamente, se ha aprendido a implantar y gestionar los sistemas relacionados con todo el entorno.

3.2 Objetivos

En cuanto a los objetivos, y como se ha comentado en anteriores apartados, queda verificado el cumplimiento de objetivos de todo el TFG, que consistía en conseguir poner en funcionamiento un portal digital con capacidades de CMS y de creación de sitios web, como es Liferay, y además que estuviera en Alta Disponibilidad y totalmente monitorizado.

Como nota hay que comentar que no se han conseguido, por falta de tiempo para la investigación, la manera de hacer que funcione un script

de arranque para Linux para Systemd, para que arranque el servicio de búsqueda en los servidores implicados (falla en el arranque de la máquina) y, por tanto, se ha dejado de forma manual, y como consecuencia, el de Liferay, ya que es dependiente del otro.

3.2 Planificación, metodología y cambios realizados

En cuanto a la planificación y metodología, se ha seguido la planificación expuesta tal y como quedó plasmada al principio del proyecto y la metodología de investigación-acción ha sido la idónea, ya que hemos investigado como realizar nuestras acciones para a continuación acometer las mismas y conseguir el éxito y objetivo de este TFG.

Ha habido cambios, como, por ejemplo, tener que introducir tres nodos, en vez de dos nodos, del servicio de búsqueda, para garantizar que ese servicio era el óptimo, pero no nos ha causado mayor problema más allá de la configuración extra de ese nodo.

3.3 Líneas de trabajo futuro y dificultades

Entre las líneas de trabajo futuro que no se han podido explorar, y han quedado pendientes, están la gestión de alertados para el servidor de Prometheus por falta de tiempo, un sistema en clúster para la base de datos de MySQL, también por falta de tiempo además de falta de recursos del host anfitrión (un clúster de MySQL necesita más máquinas y recursos hardware para funcionar de los que no disponíamos), una gestión más avanzada de los paneles de control de Grafana, también por falta de tiempo, y por último, poder finalizar unos scripts funcionales para arrancar los servicios de Elasticsearch y Liferay junto con las máquinas, que a pesar de haberlo intentado, nos ha faltado tiempo para implementarlos adecuadamente.

Por otro lado, se podría haber intentado instalar y configurar un sistema de detección de intrusos (IDS) como Snort⁴⁸ que se podría instalar en la máquina donde está el balanceador y cortafuegos, y la cual, nos daría información sobre intentos de ataques aparte de darnos la capacidad de prevenirlos mediante reglas en el propio software. No obstante, y como comentábamos antes, la falta de tiempo ha sido un factor común en todas ellas.

4. Glosario

Definición de los términos y acrónimos más relevantes utilizados dentro de la Memoria.

- CMS: Sistema de Gestión de Contenido
- ERP: Sistema de planificación de recursos empresariales
- Liferay Portal: solución software web para la gestión de contenidos y creación de sitios web
- NFS: Sistema compartido de ficheros en red
- Alta Disponibilidad: capacidad que tiene un sistema para ser accesible y confiable casi todo el tiempo
- HAProxy: Software de balanceo y repartición de carga
- Elasticsearch: Servicio de búsqueda e indexado
- Firewall: Sistema de cortafuegos y prevención de amenazas que bloquea puertos de escucha en los sistemas
- Apache webserver: Servidor web de contenido estático web que puede ser utilizado como proxy inverso para redirigir ese tráfico web.
- Tomcat: Servidor de servlets y aplicativos webs, tales como Liferay
- Máquina Virtual: host virtualizado por software que se ejecuta sobre un nodo físico
- Backup: Copia de respaldo
- Prometheus: Servicio de métricas y alertado
- Grafana: Servicio de análisis de métricas mediante paneles de control

5. Bibliografía

Lista numerada de las referencias bibliográficas utilizadas dentro de la memoria.

- 1- Coutinho, V.[Victor] (2020). Sistema de gestión de contenidos (CMS): ¿por qué implementarlo en tu empresa?, *Hackingarticles*. <https://www.hackingarticles.in/detect-nmap-scan-using-snort/>, (1/10/23)
- 2- Giménez, M.[Mónica] (2021). ¿Qué es Liferay y para qué sirve?, *Hiberus*. <https://www.hiberus.com/crecemos-contigo/que-es-liferay-y-para-que-sirve/> (1/10/23)
- 3- Mediavilla, J.[Jorge] (2020). Análisis de Liferay: un CMS, o mejor dicho DXP, que come en la mesa de los mejores, *CMS MAG*. <https://www.mejorcms.com/liferay/analisis-liferay/> (2/10/23)
- 4- Liferay DXP Quarterly Releases (7.4) Compatibility Matrix (2023), *Liferay*. <https://help.liferay.com/hc/en-us/articles/4411310034829-Liferay-DXP-Quarterly-Releases-7-4-Compatibility-Matrix>(2/10/23)
- 5- Ranking the Top Enterprise and Open Source Operating Systems of 2022, *Openlogic*. <https://www.openlogic.com/blog/top-open-source-operating-systems-2022>(3/10/23)
- 6- Centos System requirements reference (2022), *Centos* https://docs.centos.org/en-US/8-docs/standard-install/assembly_system-requirements-reference/#record-system-specifications_system-requirements-reference (5/10/23)
- 7- KeepCoding Team (2023), ¿Qué es y cómo usar el Comando Yum en Linux?, *keepcoding*. <https://keepcoding.io/blog/que-es-y-como-usar-el-comando-yum-en-linux/>(12/10/23)
- 8- How to Change or Set Hostname on CentOS 8 (2020), *PhoenixNAP*. <https://phoenixnap.com/kb/how-to-change-set-hostname-on-centos-8> (13/10/23)
- 9- How To Install the Apache Web Server on CentOS (2020),*DigitalOcean*. <https://www.digitalocean.com/community/tutorials/how-to-install-the-apache-web-server-on-centos-8> (14/10/23)
- 10-Dandified Yum (2015), Wikipedia. https://es.wikipedia.org/wiki/Dandified_Yum (17/10/23)
- 11-Pradeep, K.[Kumar], How to Install and Configure HAProxy on CentOS 8 / RHEL 8, *LinuxTechi*. <https://www.linuxtechi.com/install-configure-haproxy-centos-8-rhel-8/> (20/10/23)
- 12-Ramirez, N.[NickI] (2021), How to Enable Health Checks in HAProxy, *HAProxy*. <https://www.haproxy.com/blog/how-to-enable-health-checks-in-haproxy> (21/10/23)
- 13-Davidochobits (2020), Instalar y configurar un servidor NFS en Centos 8 y RHEL 8, *Ochobitshacenunbyte*. <https://www.ochobitshacenunbyte.com/2020/12/01/instalar-y-configurar-un-servidor-nfs-en-centos-8-y-rhel-8/> (23/10/23)

- 14-Drake, M.[Markl] (2020), How To Install MySQL on CentOS 8, DigitalOcean. <https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-centos-8> (1/11/23)
- 15-Configuring a Database (2023), Liferay. <https://learn.liferay.com/w/dxp/installation-and-upgrades/installing-liferay/configuring-a-database> (17/10/23)
- 16-Sverdlov, E.[Etel] (2012) How To Create a New User and Grant Permissions in MySQL, DigitalOcean. <https://www.digitalocean.com/community/tutorials/how-to-create-a-new-user-and-grant-permissions-in-mysql>(17/10/23)
- 17-Installing Liferay (2023), Liferay. <https://learn.liferay.com/w/dxp/installation-and-upgrades/installing-liferay>(18/10/23)
- 18-Installing a Liferay-Tomcat Bundle(2023), Liferay. <https://learn.liferay.com/w/dxp/installation-and-upgrades/installing-liferay/installing-a-liferay-tomcat-bundle> (19/10/23)
- 19-Java SE 8 Archive (2019), Oracle. <https://www.oracle.com/es/java/technologies/javase/javase8-archive-downloads.html> (20/10/23)
- 20-Running Liferay for the First Time (2023), Liferay. <https://learn.liferay.com/w/dxp/installation-and-upgrades/installing-liferay/running-liferay-for-the-first-time> (20/10/23)
- 21-Network Attached Storage (NAS): qué es y cómo funciona (2023), Ionos. <https://www.ionos.es/digitalguide/servidores/know-how/que-es-el-network-attached-storage-nas/> (21/10/23)
- 22-File Storage (2023), Liferay. <https://learn.liferay.com/w/dxp/system-administration/file-storage> (22/10/23)
- 23-Instalar un motor de búsqueda <https://learn.liferay.com/w/dxp/using-search/installing-and-upgrading-a-search-engine/installing-a-search-engine> (23/10/23)
- 24-Installing Elasticsearch (2023), Liferay. <https://learn.liferay.com/w/dxp/using-search/installing-and-upgrading-a-search-engine/elasticsearch/installing-elasticsearch> (24/10/23)
- 25-Matriz de soporte (2023), Elastic. https://www.elastic.co/es/support/matrix#matrix_jvm (25/10/23)
- 26-Connecting to Elasticsearch (2023), Liferay. <https://learn.liferay.com/w/dxp/using-search/installing-and-upgrading-a-search-engine/elasticsearch/connecting-to-elasticsearch> (28/10/23)
- 27-Fronting Liferay Tomcat with Apache HTTPd daemon Revisted (2022), Liferay. <https://liferay.dev/blogs/-/blogs/fronting-liferay-tomcat-with-apache-httpd-daemon-revisted> (2/11/23)
- 28-10 factores que debes considerar antes de elegir un CMS Open Source, Liferay. <https://www.liferay.com/es/blog/customer-experience/10-factores-que-debes-considerar-antes-de-elegir-un-cms-open-source> (19/10/2023)
- 29-CMS Liferay vs Alfresco (2008), Liferay. <https://liferay.dev/ask/questions/portal/cms-liferay-vs-alfresco-1> (19/10/2023)

- 30-Liferay Clustering conceptos Básicos, *Github*.
<https://github.com/dmcisneros/Apuntes/blob/master/clustering/cluster.MD> (27/10/2023)
- 31-Load Balancing with Apache 2 WebServer - Sticky Session (2015).
Motech Project.
https://docs.motechproject.org/en/latest/deployment/sticky_session_a_pache.html (29/10/2023)
- 32-Resilience in small clusters, *Liferay*.
<https://www.elastic.co/guide/en/elasticsearch/reference/current/high-availability-cluster-small-clusters.html#high-availability-cluster-design-two-nodes> (30/10/2023)
- 33-Problema de los generales bizantinos, *Wikipedia*.
https://es.wikipedia.org/wiki/Problema_de_los_generales_bizantinos (30/10/2023)
- 34-Configuring Cluster Link, *Liferay*.
<https://learn.liferay.com/w/dxp/installation-and-upgrades/setting-up-liferay/clustering-for-high-availability/configuring-cluster-link> (1/11/2023)
- 35-Configuring Unicast over TCP, *Liferay*.
<https://learn.liferay.com/w/dxp/installation-and-upgrades/setting-up-liferay/clustering-for-high-availability/configuring-unicast-over-tcp> (1/11/2023)
- 36- How To Set Up a Firewall Using firewalld on CentOS 8 (2020),
DigitalOcean <https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-using-firewalld-on-centos-8> (2/11/2023)
- 37- Ramirez, N.[Nick] (2019). HAProxy SSL Termination (Offloading) Everything You Need to Know, *HAProxy*.
<https://www.haproxy.com/blog/haproxy-ssl-termination> (5/11/2023)
- 38-Yeh, K.[Keith] (2017). How to create a self-signed SSL Certificate with SubjectAltName(SAN), *Github*.
<https://gist.github.com/KeithYeh/bb07cadd23645a6a62509b1ec8986bbc> (6/11/2023)
- 39- ¿Qué es una red DMZ?, *Fortinet*.
<https://www.fortinet.com/lat/resources/cyberglossary/what-is-dmz> (7/11/2023)
- 40-Install and Configure AutoMySQL Backup on Centos (2018),
InterServer. <https://www.interserver.net/tips/kb/configure-automysql-backup-centos/> (8/11/2023)
- 41-Prometheus Overview (2023), *Prometheus*.
<https://prometheus.io/docs/introduction/overview/> (9/11/2023)
- 42-Grafana OSS (2023), *Grafana*. <https://grafana.com/oss/grafana/> (9/11/2023)
- 43-Node-exporter setup with Systemd (2021), *Jaanhio*.
<https://jaanhio.me/blog/linux-node-exporter-setup/> (10/11/2023)
- 44-HAproxy-exporter-systemd (2018), *Github*.
<https://github.com/eosswedenorg/haproxy-exporter-systemd/blob/master/install.sh> (10/11/2023)
- 45-Mutai, J.[Josphat] (2022). Monitor Apache Web Server with Prometheus and Grafana in 5 minutes, *Computing for Geeks*.

- <https://computingforgeeks.com/monitor-apache-web-server-prometheus-grafana/> (10/11/2023)
- 46-Prince, M.[Meera] (2021). Liferay Portal Monitoring with Prometheus, *Liferay Savvy*. <http://www.liferay Savvy.com/2021/07/liferay-portal-monitoring-with.html> (10/11/2023)
- 47-Mutai, J.[Josphat] (2022). Configure Prometheus MySQL Exporter on Ubuntu 18.04 / CentOS 7, *Computing for Geeks*. <https://computingforgeeks.com/install-and-configure-prometheus-mysql-exporter-on-ubuntu-centos/>
- 48- Snort (2023), *Snort*. <https://www.snort.org/> (23/12/2023)

6. Anexos

Anexo 1. Instalación de máquinas virtuales con VirtualBox

-Instalaremos VirtualBox en el Host ejecutando el instalador previamente descargado de su web y dejamos las opciones por defecto y seleccionamos “Next”

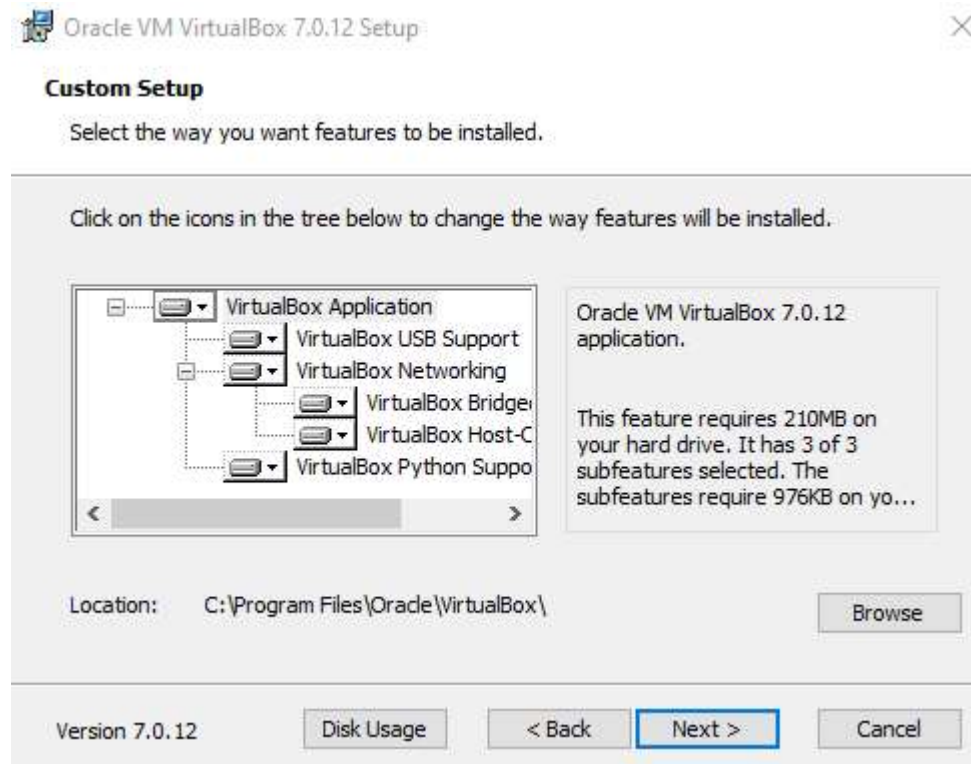


Figura 29. Instalación de VirtualBox

Finalizaremos la instalación pulsando “yes” y una vez instalado ejecutamos y tendríamos preparado el entorno listo para virtualizar máquinas que podremos ejecutar.

A continuación, descargaremos una máquina Virtual de CentOS 8 preparada para VirtualBox (ver enlace de descarga en el capítulo [2.1.2](#))

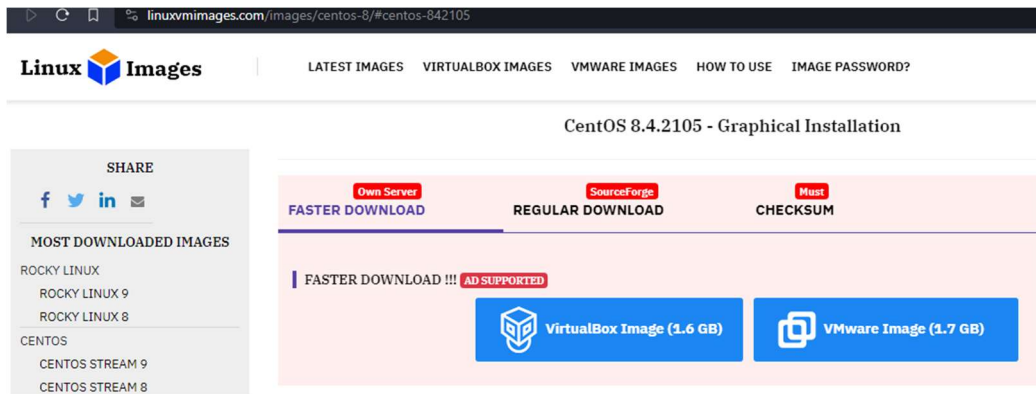


Figura 30. Web de descarga de máquinas virtuales CentOS

Una vez bajada la imagen la descomprimiremos en una carpeta del sistema que elijamos y, a continuación, la podemos “Añadir” a VirtualBox” seleccionando el archivo que hemos descargado.

Por el momento, dejamos configurado el “Adaptador puente” en “Red” para que use nuestra interfaz de red del host para conectarse a Internet directamente con una IP asignada por DHCP.

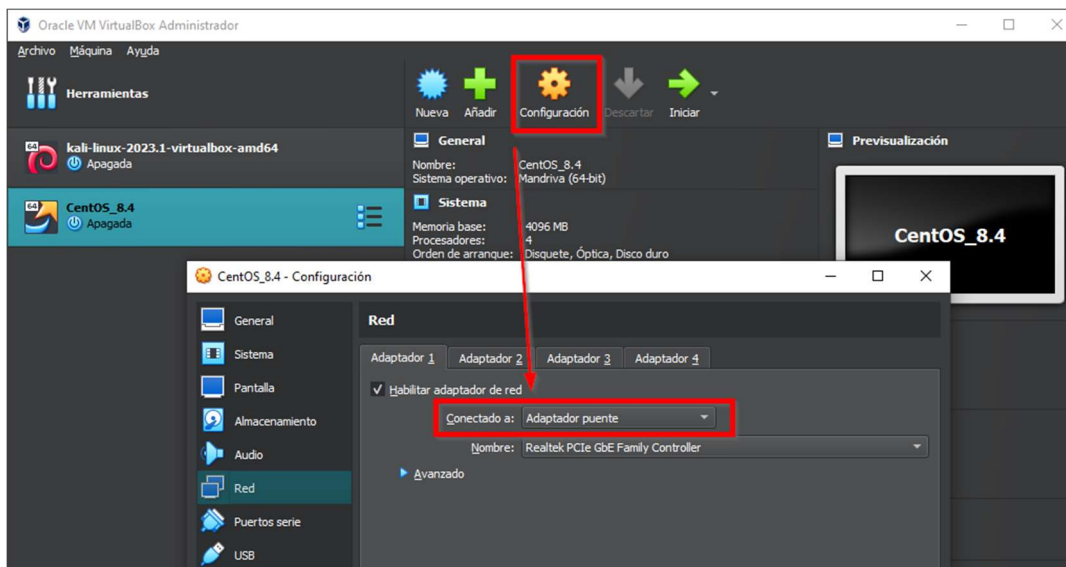


Figura 31. Configuración de red en VirtualBox

Aceptamos el cambio y arrancamos la máquina con el botón de “Iniciar” y, una vez arrancada, seleccionamos el usuario “centos” cuya contraseña es “centos” (como indica la propia web desde donde se descargan las imágenes) e introducimos usuario y contraseña para entrar al sistema por SSH o por consola gráfica de VirtualBox.

Para preparar la máquina simplemente la actualizamos desde el terminal con el comando yum⁷:

```
sudo yum update
```

Esto nos actualizará el sistema aplicando todas las correcciones y parches de seguridad para esta versión, e incluso sucederá que haya una actualización de la versión como podemos comprobar, al finalizar la misma, con el comando:

```
cat /etc/centos-release
```

Y veremos algo así:

```
CentOS Linux reléase 8.5.2111
```

Una vez terminada la actualización del sistema podemos apagar la máquina,

```
sudo init 0
```

Dejaremos la máquina guardada para usarla como plantilla y la renombraremos como tal incluyendo su nueva versión.

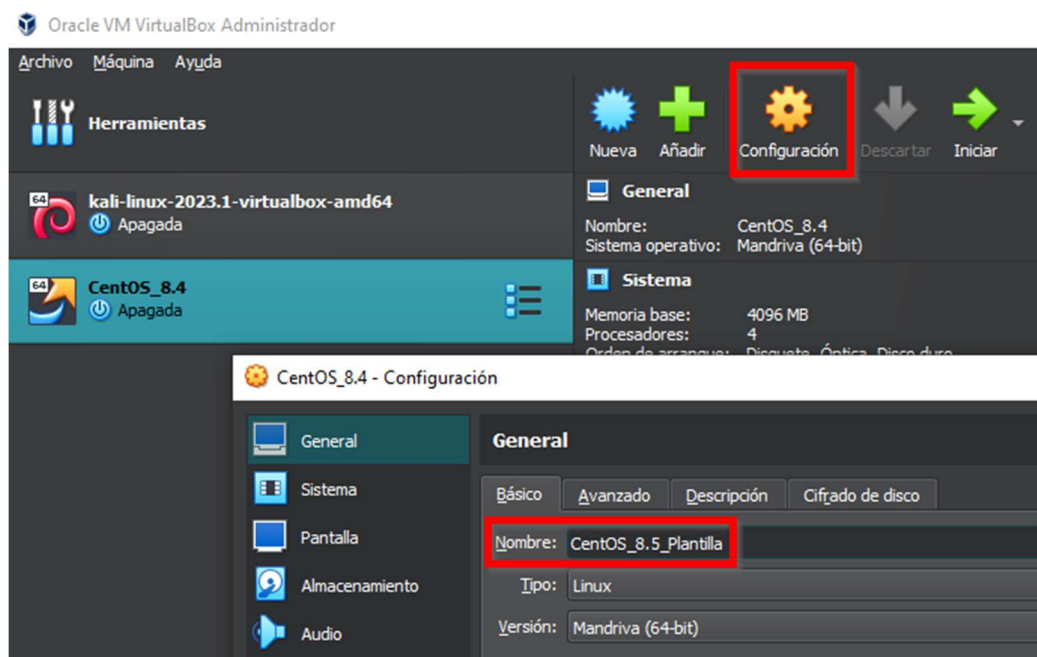


Figura 32. Máquina virtual creada en VirtualBox

Anexo 2. Clonación de máquinas virtuales

Para clonar una máquina en VirtualBox, damos a clic derecho encima de una de ellas y seleccionamos “clonar”.

Generaremos nuevas direcciones MAC para prevenir que haya problemas al asignar IPs en la red y el resto de opciones por defecto y le asignaremos un nombre. VirtualBox se encargará de generar una carpeta dentro de la unidad con el mismo nombre.

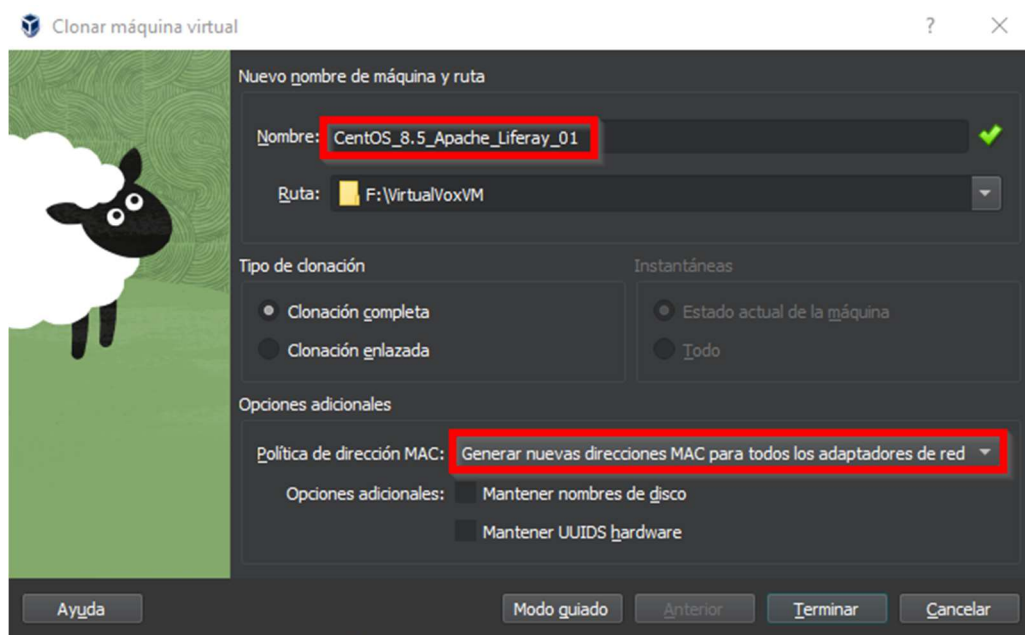


Figura 33. Clonación de máquinas virtuales en VirtualBox

En breve tendremos la máquina clonada y la arrancamos.

Lo primero que vamos a hacer es cambiar el nombre de la máquina para tenerla mejor identificada para cuando trabajemos con varias a la vez, para ello usaremos la terminal y el comando⁸ siguiente (el nombre de la máquina dependerá del que queramos darle):

```
sudo hostnamectl set-hostname apacheliferay01
```

Cerramos la terminal y al reabirla comprobamos que ya se ha cambiado el nombre.

Anexo 3. Asignación de IPs estáticas de máquinas virtuales

Para asignar alguna de las IPs habrá que hacerlo con una IP fuera del rango de DHCP de asignación dinámica de IPs de nuestro router, de tal manera que evitaremos posibles conflictos con otras IPs asignadas por DHCP dentro de la misma red.

En nuestro caso, el router dispone de un rango de IPs estáticas desde la 192.168.1.2/24 hasta la 192.168.1.32/24, ambas incluidas y siendo la 192.168.1.1 la del router, que, a su vez, será el *Default Gateway* (puerta de enlace) para las máquinas.

Para hacer esta operativa se puede hacer con la herramienta gráfica de configuración de CentOS en Settings => Network y seleccionamos el engranaje en la sección de "Wired".

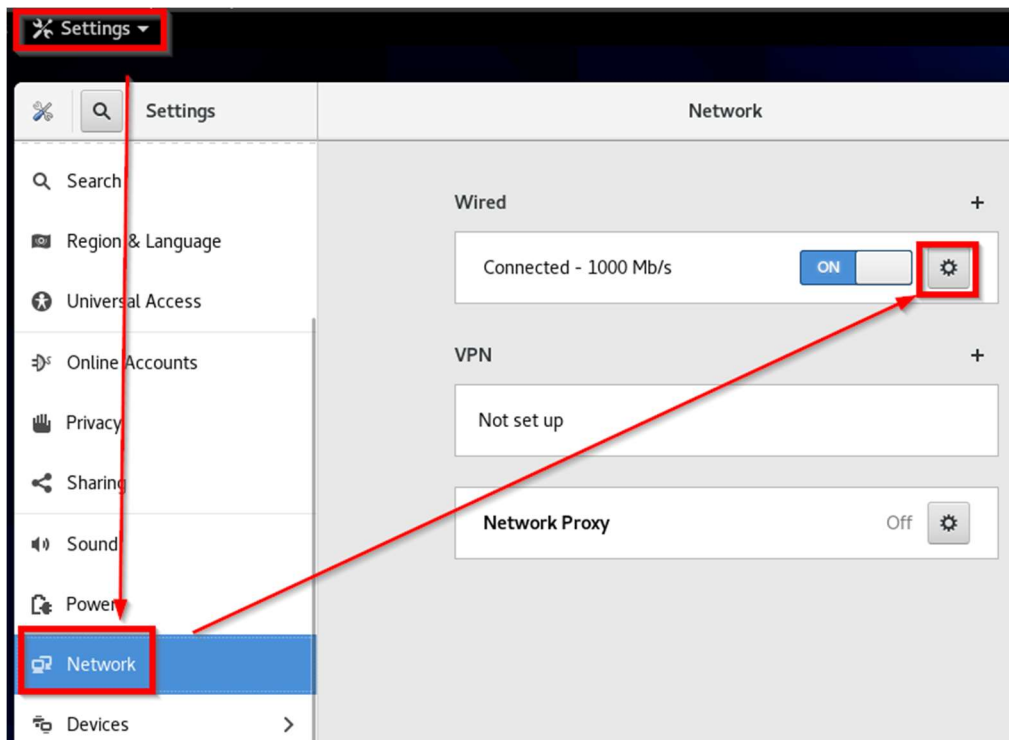


Figura 34. Configuración de red de CentOS

Luego se selecciona IPv4 y “Manual” y se fija una de esas IPs estáticas disponibles en nuestra red que esté libre que, en este caso de ejemplo, usaremos la 192.168.1.20 con su máscara de red 255.255.255.0 y el Gateway 192.168.1.1, que es nuestro Router. Además, configuraremos el DNS de Google (8.8.8.8) porque dependiendo del Router puede no asignarnos un DNS automáticamente.

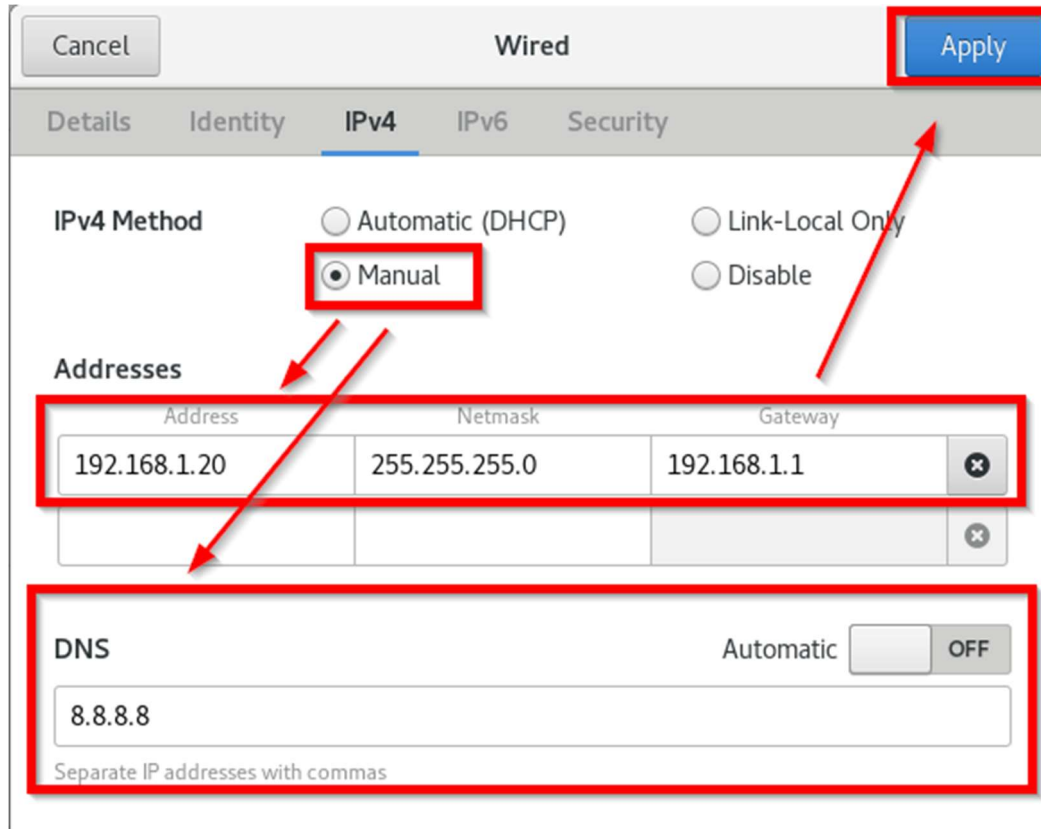


Figura 35. Configuración de IP de CentOS

Aplicamos y cerramos. La máquina, en el siguiente reinicio, cogerá los cambios y ya tendrá esa IP, pero también podemos apagar y encender la red ejecutando:

```
sudo nmcli networking off
sudo nmcli networking on
```

Anexo 4. Configuración del servicio de Firewall (cortafuegos)

El servicio de Firewall (cortafuegos) protege las máquinas de accesos por puertos no autorizados. Por defecto el servicio de Firewall de CentOS viene activado y restringe accesos a no ser que los permitamos.

Para configurar algún puerto de acceso como por ejemplo los de HTTP (puerto 80) y HTTPS (443) habría que ejecutar estos comandos que permiten dicho tráfico por la zona pública definida por defecto en el Firewall, hacen las mismas permanentes y luego se recarga la configuración de estas para que tengan efecto:

```
sudo firewall-cmd --zone=public --add-port=80/tcp --permanent
sudo firewall-cmd --zone=public --add-port=443/tcp --permanent
sudo firewall-cmd --reload
```

Estas dos reglas concretamente nos serán útiles para permitir el tráfico web a través de la máquina “haproxy” cuando configuremos su seguridad.

Para parar, arrancar y comprobar el servicio de Firewall se usan estos comandos:

```
sudo systemctl stop firewalld
sudo systemctl start firewalld
sudo systemctl status firewalld
```

En otras ocasiones nos interesará no tener el servicio de Firewall ya que estaremos en una red que se considerará segura.

Para desactivar el servicio de Firewall, tras pararlo, usaremos este comando

```
sudo systemctl disable firewalld
```

Anexo 5. Instalación y configuración de un servidor web Apache

Se deberá instalar el paquete del Apache⁹ Web server con el gestor de paquetes de “Dandified Yum”¹⁰ con el comando:

```
sudo dnf install httpd
```

Aceptamos y verificamos que queda instalado correctamente.

Ahora podemos levantar el proceso del Apache Webserver manualmente y luego verificar que se ha levantado con:

```
sudo systemctl start httpd  
ps -ef | grep apache
```

Como podremos ver, el proceso está arriba y corriendo con el usuario “apache”, que es el que se autoconfigura por defecto para correr el proceso, aunque, también podemos comprobar el estado del servicio con:

```
sudo systemctl status httpd
```

```
[centos@apacheliferay01 ~]$ sudo systemctl status httpd  
● httpd.service - The Apache HTTP Server  
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)  
  Active: active (running) since Sun 2023-10-29 22:03:02 CET; 39min ago  
    Docs: man:httpd.service(8)  
  Main PID: 4558 (httpd)  
  Status: "Total requests: 8; Idle/Busy workers 100/0;Requests/sec: 0.00341; Bytes served/sec: 514 B/sec"  
    Tasks: 213 (limit: 23544)  
  Memory: 44.0M  
  CGroup: /system.slice/httpd.service  
          └─4558 /usr/sbin/httpd -DFOREGROUND  
            └─4571 /usr/sbin/httpd -DFOREGROUND  
              └─4572 /usr/sbin/httpd -DFOREGROUND  
                └─4573 /usr/sbin/httpd -DFOREGROUND  
                  └─4574 /usr/sbin/httpd -DFOREGROUND
```

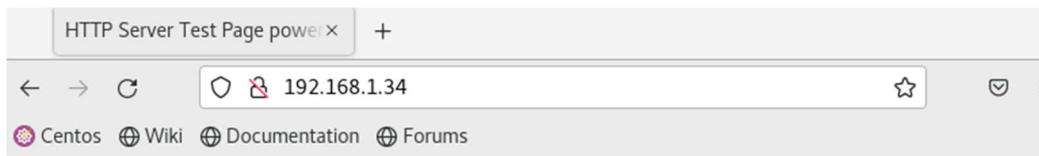
Figura 36. Servicio web Apache ejecutándose

Además, podemos hacer ejecutar un comando para comprobar que carga la página de prueba de Apache intentando acceder por http por la IP de la máquina (obtenida mediante comando) y el puerto 80 (http) con los comandos siguientes:

```
ip a  
curl http://192.168.1.34
```

Accediendo por http con el comando *curl*, se verifica que se ha cargado contenido web.

Por otro lado, podemos probar con el navegador Firefox local de la máquina a hacer la misma prueba. Si accedemos por IP, se puede ver que carga correctamente la página de test del servidor web Apache.



This page is used to test the proper operation of the HTTP server after it has been installed. If you can read this page it means that this site is working properly. This server is powered by [CentOS](#).

If you are a member of the general public:

The website you just visited is either experiencing problems or is undergoing routine maintenance.

Figura 37. Página web de prueba del servidor Apache

Ahora lo que haremos es dejar el servicio del Apache activado para cuando se arranque la máquina con:

```
sudo systemctl enable httpd
```

De esta manera cuando la máquina se inicie el Apache arrancará automáticamente.

Anexo 6. Instalación y configuración de HAProxy

Ahora procedemos a instalar el paquete de haproxy¹¹ con el gestor de paquetes de “Dandified Yum”¹⁰ con el comando:

```
sudo dnf install haproxy
```

Aceptamos pulsando “y” y verificamos que queda instalado correctamente.

Una vez hecho esto procedemos a hacer un backup del fichero de configuración de haproxy (manteniendo los permisos del fichero) antes de editarlo, el cual está en /etc/haproxy/:

```
sudo cp -p haproxy.cfg haproxy.cfg_back
```

Ahora podemos editar su configuración para que haga el balanceo entre los dos servidores web que tendremos en las otras máquinas que crearemos, pero por el momento solo configuraremos la IP en la que va a recibir las llamadas y la de una de las máquinas de Apache ya que no hemos instalado aún la otra máquina para la alta disponibilidad.

Para ello, previamente, fijaremos en esta máquina también su IP (véase [Anexo 3](#)) como estática 192.168.1.10 como antes con la de Apache y el DNS de Google (8.8.8.8).

Y ahora editamos el fichero del HAProxy para configurarlo con:

```
sudo vim /etc/haproxy/haproxy.cfg
```

En este fichero hacemos estos cambios:

- Fijamos la IP y el puerto 80 (HTTP) en la parte de “bind”.
- Añadimos las cabeceras HTTP que serán necesarias para el Liferay como veremos más tarde.
- Comentamos las reglas de “acl” para contenido estático ya que todo el contenido se redireccionará a los servidores web Apache.
- Definimos el tipo de balanceo que será “roundrobin” (una petición a cada servidor).
- Definimos la opción de comprobación de salud¹² contra los webserver con un “httpchk” (Check HTTP).
- Configuramos las IPs y puertos de las máquinas de los servidores web Apache que vamos a usar (para las pruebas podemos dejar uno de ellos comentado ya que al principio vamos a hacer pruebas y validaciones sólo con uno de ellos funcionando y este segundo solo se activará más adelante cuando configuremos la Alta Disponibilidad, no obstante, aquí dejamos ambos sin comentar como quedaría al final).

```

#-----
# main frontend which proxys to the backends
#-----
frontend liferay.local
    bind 192.168.1.10:80
    # Set headers
    http-request set-header X-Forwarded-Proto https if { ssl_fc }
    http-request add-header X-Real-IP %[src]
    option forwardfor

    #acl url_static      path_beg      -i /static /images /javascript
    /stylesheets
    #acl url_static      path_end      -i .jpg .gif .png .css .js

    #use_backend static      if url_static
    default_backend          apache_webservers

#-----
# static backend for serving up images, stylesheets and such
#-----
backend static
    balance      roundrobin
    server       static 127.0.0.1:4331 check

#-----
# round robin balancing between the various backends
#-----
backend apache_webservers
    balance      roundrobin
    option       httpchk

    server       apache-node01 192.168.1.20:80 check
    server       apache-node02 192.168.1.21:80 check

```

Pasamos a habilitar el HAProxy en la seguridad de SELinux (Security-Enhanced Linux) para que pueda conectarse a cualquier puerto del sistema de forma persistente con.

```
sudo setsebool -P haproxy_connect_any 1
```

Activamos el servicio haproxy para que pueda arrancar cuando arranque la máquina y lo arrancamos.

```
sudo systemctl enable haproxy
sudo systemctl start haproxy
```

Podemos ver si ha arrancado con este comando y comprobar que está arriba.

```
sudo systemctl status haproxy
```

```
[centos@haproxy ~]$ sudo systemctl status haproxy
● haproxy.service - HAProxy Load Balancer
   Loaded: loaded (/usr/lib/systemd/system/haproxy.service; enabled; vendor preset:
   Active: active (running) since Wed 2023-11-01 00:59:38 CET; 1s ago
     Process: 4573 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q $OPTIONS (code=exite
   Main PID: 4576 (haproxy)
      Tasks: 2 (limit: 23544)
     Memory: 4.1M
    CGroup: /system.slice/haproxy.service
            └─4576 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.
              └─4578 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.

Nov 01 00:59:38 haproxy systemd[1]: Starting HAProxy Load Balancer...
Nov 01 00:59:38 haproxy systemd[1]: Started HAProxy Load Balancer.
```

Figura 38. Servicio HAProxy levantado y ejecutándose

Anexo 7. Configuración de sistema de eventos mediante Syslog

Para configurarlo editamos el fichero de syslog:

```
sudo vim /etc/rsyslog.conf
```

Descomentamos estas líneas, guardamos y salimos

```
module(load="imtcp") # needs to be done just once
input(type="imtcp" port="514")
```

Creamos el fichero de haproxy.conf

```
sudo vim /etc/rsyslog.d/haproxy.conf
```

Agregamos estas dos líneas, guardamos y salimos

```
local2.=info      /var/log/haproxy-access.log
local2.notice     /var/log/haproxy-info.log
```

Reiniciamos el servicio de Syslog, lo habilitamos para que se inicie con el arranque de la máquina y comprobamos su estado con:

```
sudo systemctl restart rsyslog
sudo systemctl enable rsyslog
sudo systemctl status rsyslog
```

```
[centos@haproxy ~]$ sudo systemctl restart rsyslog
[centos@haproxy ~]$ sudo systemctl enable rsyslog
[centos@haproxy ~]$ sudo systemctl status rsyslog
● rsyslog.service - System Logging Service
   Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2023-11-05 15:38:21 CET; 45s ago
     Docs: man:rsyslogd(8)
           https://www.rsyslog.com/doc/
  Main PID: 4205 (rsyslogd)
    Tasks: 4 (limit: 11258)
   Memory: 1.6M
   CGroup: /system.slice/rsyslog.service
           └─4205 /usr/sbin/rsyslogd -n

Nov 05 15:38:21 haproxy systemd[1]: Starting System Logging Service...
Nov 05 15:38:21 haproxy rsyslogd[4205]: [origin software="rsyslogd" swVersion="8.2102.0-5.el8" x-pid="4205" x-info="https://www.rsyslog.com"] start
Nov 05 15:38:21 haproxy systemd[1]: Started System Logging Service.
Nov 05 15:38:21 haproxy rsyslogd[4205]: imjournal: journal files changed, reloading... [v8.2102.0-5.el8 try https://www.rsyslog.com/e/0 ]
[centos@haproxy ~]$
```

Figura 39. Servicio de Syslog corriendo y ejecutándose

Si entramos de nuevo desde el host con el navegador a la IP del balanceador que nos redirige al apache podemos ver en el fichero `/var/log/haproxy-access.log` los accesos que se realizan y a que nodo de Apache Webserver se está llegando, para ello ejecutamos:


```
sudo tail -200f /var/log/haproxy-access.log
```

Y vemos que efectivamente hay accesos y que se redirige correctamente las llamadas con llamadas GET HTTP 1.1 en dichos logs.

Anexo 8. Configuración de un sistema NFS (sistema de ficheros en red)

Para configurar el NFS tendremos que instalar la parte de servidor y la de cliente en cada máquina que lo vaya a usar, aquí haremos la configuración de la parte de servidor en la máquina “mysql” y la parte de cliente en la máquina “apacheliferay01” (también lo haremos más adelante en la “apacheliferay02” cuando configuremos la alta disponibilidad más adelante, pero será análogo a esta), para hacerlo usaremos este comando:

```
sudo dnf install nfs-utils
```

Que en este caso el CentOS 8 ya tiene el servicio de NFS instalado y por eso nos devuelve un “Nothing to do” tras el intento de instalación. No obstante, pasaremos a levantar el servicio, a habilitarlo con el arranque de la máquina y comprobar su status con estos 3 comandos:

```
sudo systemctl start nfs-server.service
sudo systemctl enable nfs-server.service
sudo systemctl status nfs-server.service
```

```
[centos@mysql ~]$ sudo systemctl start nfs-server.service
[centos@mysql ~]$ sudo systemctl enable nfs-server.service
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-server.service →
/usr/lib/systemd/system/nfs-server.service.
[centos@mysql ~]$ sudo systemctl status nfs-server.service
● nfs-server.service - NFS server and services
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; enabled; vendor
   Active: active (exited) since Sat 2023-11-04 19:52:59 CET; 13s ago
   Main PID: 6153 (code=exited, status=0/SUCCESS)
     Tasks: 0 (limit: 23544)
    Memory: 0B
   CGroup: /system.slice/nfs-server.service

Nov 04 19:52:59 mysql systemd[1]: Starting NFS server and services...
Nov 04 19:52:59 mysql systemd[1]: Started NFS server and services.
lines 1-10/10 (END)
```

Figura 40. Servicio NFS corriendo y ejecutándose

Podemos verificar el protocolo usado con el comando:

```
rpcinfo -p | grep nfs
```

Crearemos y exportaremos el recurso NFS que va a usar el Liferay. Para ello creamos el directorio que se va a exportar con:

```
sudo mkdir -p /mnt/liferay_data
```

También crearemos el usuario (y su grupo) que usará tanto el Liferay para ejecutarse como el que va a poder leer y escribir en ese recurso con los siguientes comandos, uno para crear el grupo “Liferay” y otro para el usuario “liferay” asignándolo a ese grupo:

```
sudo groupadd liferay
sudo useradd liferay -g liferay
```

Comprobamos que se ha creado y asignado al grupo correctamente con:

```
id liferay
```

Podemos, además, fijarle ya una contraseña (que será “Liferay_01”):

```
sudo passwd liferay
```

Haremos propietario a ese usuario del directorio que vamos a exportar y comprobamos con:

```
sudo chown -R liferay: /mnt/liferay_data/  
ls -al /mnt/ | grep liferay
```

Editamos ahora el fichero /etc/exports donde tenemos que declarar el recurso NFS que vamos a exportar.

```
sudo vim /etc/exports
```

Introducimos en ese fichero estos valores y guardamos:

```
/mnt/liferay_data 192.168.1.0/24(rw, sync, all_squash, anonuid=1001, anongid=1001)
```

Definiendo la red 192.168.1.0/24 se exportará el recurso a toda esa red. **rw**, exportará el directorio en modo lectura y escritura al host que haga de cliente.

sync, indica al servicio NFS que escriba operaciones cuando lo requiera.

all_squash, mapea todos los usuarios al usuario anónimo.

anonuid=1001, utilizará el usuario con ID 1001 (que es el de “liferay”)

anongid=1001, utilizará el grupo con ID 1001 (que es el de “liferay”)

Comprobamos que estamos exportando el recurso con:

```
sudo exportfs -arv
```

Con esto ya podríamos pasar a configurar la parte de NFS de cliente, pero antes, y como este servidor está en el backend, la seguridad perimetral se configurará más adelante en el servidor donde reside el HAproxy y que nos daría ahora problemas con el NFS si no configuramos unas políticas, vamos a parar y desactivar del arranque de la máquina el servicio de Firewall (véase [Anexo 4](#)).

Pasaremos entonces, a configurar el cliente NFS en la máquina “apacheliferay01” (y sería análogo en la “apacheliferay01”, para ello tendremos que crear también el usuario (y su grupo) que usará tanto el Liferay para ejecutarse como el que va a poder leer y escribir en ese recurso con los comandos que ejecutamos antes, uno para crear el grupo “liferay” y otro para el usuario “liferay” asignándolo a ese grupo:

```
sudo groupadd liferay
sudo useradd liferay -g liferay
```

Comprobamos que se ha creado y asignado al grupo correctamente con:

```
id liferay
```

Le daremos también la misma contraseña que antes (que será “Liferay_01”):

```
sudo passwd liferay
```

Instalamos la parte de utilidades y herramientas de NFS para poder acceder al directorio compartido del otro servidor con:

```
sudo dnf install nfs-utils nfs4-acl-tools
```

Pasamos a comprobar que desde la máquina que va a hacer de cliente se ve el directorio NFS exportado con:

```
showmount -e 192.168.1.30
```

Creamos el directorio local y asignamos como propietario al usuario “liferay” y comprobamos que está el directorio y permisos:

```
sudo mkdir -p /mnt/liferay_data/
sudo chown -R liferay: /mnt/liferay_data
ls -la /mnt/ | grep liferay
```

También tenemos que montar el recurso compartido, para eso editamos el fichero /etc/fstab/

```
sudo vim /etc/fstab
```

y añadimos esta línea:

```
192.168.1.30:/mnt/liferay_data /mnt/liferay_data nfs defaults 0 0
```

Guardamos y cerramos el fichero, esto hará que en el siguiente reinicio se monte automáticamente el recurso, no obstante, podemos montar ahora la unidad sin reiniciar y luego verificar que se ha montado con los comandos siguientes:

```
sudo mount -a
df -h /mnt/liferay_data/
```

También podemos ampliar la información sobre el recurso montado con:

```
sudo mount | grep -i nfs
```

Como prueba final intentaremos crear un fichero de prueba con el usuario "liferay" en el directorio montado desde la máquina cliente, para ello cambiamos al usuario, vamos al directorio y creamos un fichero vacío llamado "prueba.txt":

```
sudo su - liferay
cd /mnt/liferay_data/
touch prueba.txt
ls -al
```

Y veremos como se ha creado el fichero "prueba.txt"

Comprobamos que se ha creado también en la máquina que hace de servidor de NFS y se puede ver que se ha creado también ejecutando:

```
ls -al /mnt/liferay_data/
```

Y verificando que existe el archivo. Con esto aseguramos que nuestro servicio de NFS está funcionando correctamente entre ambas máquinas.

Anexo 9. Instalación y configuración de una base de datos gestionada MySQL

Para instalar la base de datos gestionada MySQL instalaremos el paquete `mysql`¹⁴ con el gestor de paquetes de “Dandified Yum”¹⁰ con el comando:

```
sudo dnf install mysql-server
```

Aceptamos con “y” y se instalará MySQL Server 8.0.26 con todas sus dependencias.

Ahora arrancamos el servicio y comprobamos que ha levantado con:

```
sudo systemctl start mysqld.service
sudo systemctl status mysqld
```

```
[centos@mysql ~]$ sudo systemctl start mysqld.service
[centos@mysql ~]$ sudo systemctl status mysqld
● mysqld.service - MySQL 8.0 database server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; disabled; vendor preset: disabl>
   Active: active (running) since Fri 2023-11-03 17:33:36 CET; 21s ago
     Process: 5410 ExecStartPost=/usr/libexec/mysql-check-upgrade (code=exited, status=0/SUCC>
     Process: 5279 ExecStartPre=/usr/libexec/mysql-prepare-db-dir mysqld.service (code=exited>
     Process: 5253 ExecStartPre=/usr/libexec/mysql-check-socket (code=exited, status=0/SUCCES>
   Main PID: 5363 (mysqld)
   Status: "Server is operational"
     Tasks: 38 (limit: 23544)
    Memory: 464.0M
    CGroup: /system.slice/mysqld.service
           └─5363 /usr/libexec/mysqld --basedir=/usr

Nov 03 17:33:31 mysql systemd[1]: Starting MySQL 8.0 database server...
Nov 03 17:33:31 mysql mysql-prepare-db-dir[5279]: Initializing MySQL database
Nov 03 17:33:36 mysql systemd[1]: Started MySQL 8.0 database server.
lines 1-16/16 (END)
```

Figura 41. Servicio MySQL corriendo y ejecutándose

Activaremos el servicio de MySQL para que cuando arranque la máquina levante el servicio de MySQL.

```
sudo systemctl enable mysqld
```

Procederemos a la mejorar la seguridad de la instalación de MySQL con el script que dispone para esta tarea.

```
sudo mysql_secure_installation
```

Se nos preguntará si queremos activar un plugin de verificación de robustez de la contraseña, si aceptamos se nos pedirá la medida, aceptaremos “Medium” que ya es suficientemente robusta y usaremos de contraseña “Liferay_01” (sin las comillas)

A continuación, nos dará la opción de borrar el usuario de test que puede hacer *login* sin contraseña, el cual borraremos también.

Dejaremos, no obstante, que se pueda hacer *login* remoto con la cuenta de root de MySQL porque es posible que lo necesitemos para la instalación de Liferay.

Eliminaremos la base de datos de test, que tampoco usaremos.

Finalmente recargamos permisos y quedará finalizada la securización de MySQL.

Pasaremos a hacer un test de conexión para ver si el servidor de MySQL funciona correctamente con este comando e introducimos la contraseña que creamos antes:

```
mysqladmin -u root -p version
```

Al entrar se puede ver que funciona correctamente y nos devuelve el número de versión.

A continuación, tendremos que crear un esquema de base de datos específico para Liferay. Para ello nos conectamos a MySQL con el siguiente comando:

```
mysql -u root -p
```

Y procedemos a crear la base de datos “lportal” para Liferay¹⁵ en el prompt de MySQL con el comando:

```
mysql> create database lportal character set utf8;
```

Como indica la documentación de Liferay¹⁵ tendremos que dar a este esquema permisos totales para que Liferay pueda crear tablas, borrarlas y demás operativas necesarias para que pueda funcionar, para ello crearemos un usuario¹⁶ y le daremos permisos totales con este comando que creará el usuario “lportal_user” con la contraseña “Liferay_01”:

```
mysql> CREATE USER 'lportal_user'@'%' IDENTIFIED BY 'Liferay_01';
```

Le damos todos los privilegios que son necesarios con:

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'lportal_user'@'%' WITH GRANT OPTION;
```

Y, por último, recargaremos permisos:

```
mysql> FLUSH PRIVILEGES;
```

Aprovechamos también para mostrar el puerto por defecto que está levantando para cuando nos tengamos que conectar a la base de datos con:

```
mysql> SHOW GLOBAL VARIABLES LIKE 'PORT';
```

Que nos devuelve el puerto de mysql, que es el 3306, como podemos ver. No obstante, y para asegurarnos de que levanta en todas las IPs editamos el fichero de configuración del servidor de mysql:

```
sudo vim /etc/my.cnf.d/mysql-server.cnf
```

Y agregamos esta línea que lo que hace es que levante el servidor MySQL para todas las IPs de la máquina:

```
bind-address=0.0.0.0
```

Finalmente reiniciamos el servicio de mysql y comprobamos que ha levantado con:

```
sudo systemctl stop mysqld.service  
sudo systemctl start mysqld.service  
sudo systemctl status mysqld.service
```


Anexo 10. Instalación y configuración de un cliente de bases de datos MySQL

Para instalar el cliente buscamos el paquete de cliente de “mysql” con la herramienta “dnf”:

```
sudo dnf search mysql
```

En la descripción nos indica que el cliente está en el paquete “mysql.x86_64”, por tanto, sabiendo que es el paquete del cliente, instalamos con:

```
sudo dnf install mysql.x86_64
```

Pulsamos “y” para terminar de instalar.

Anexo 11. Instalación y configuración de Java Runtime JDK en Linux

Descargaremos de aquí la versión que necesitamos:

<https://www.oracle.com/es/java/technologies/javase/javase8-archive-downloads.html>

En concreto la versión “jdk-8u202-linux-x64.tar.gz” (Nota: Es necesario crearse usuario de Oracle para la descarga, pero es gratuito hacérselo).

Y las descomprimos tras posicionarnos sobre el directorio /opt/ con:

```
sudo tar -xzvf /home/centos/Downloads/jdk-8u202-linux-x64.tar.gz
```

Cambiamos su propietario al usuario “liferay”

```
sudo chown -R liferay: jdk1.8.0_202
```

Y para simplificar y facilitar el acceso a este directorio crearemos un enlace simbólico al mismo:

```
sudo ln -s jdk1.8.0_202 java8
```

Podemos comprobar la versión dentro del directorio con:

```
./java -version
```

Tendremos que fijar la variable de entorno JAVA_HOME para el usuario que ejecuta el Liferay para que se use correctamente, para ello basta editar el archivo “.bashrc” del *home* del usuario y añadir estas tres líneas al final del fichero:

```
vim /home/liferay/.bashrc
```

```
export JAVA_HOME=/opt/java8
export PATH=$PATH:$JAVA_HOME/bin
export
CLASSPATH=.:$JAVA_HOME/jre/lib:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

Guardamos el fichero, hacemos *logout* del usuario, volvemos a entrar con ese usuario y comprobamos que detecta el Java en cualquier parte del sistema, por ejemplo, desde “/opt/” ejecutando:

```
java -version
```

```
[liferay@apacheliferay01 ~]$ cd /opt/  
[liferay@apacheliferay01 opt]$ java -version  
java version "1.8.0_202"  
Java(TM) SE Runtime Environment (build 1.8.0_202-b08)  
Java HotSpot(TM) 64-Bit Server VM (build 25.202-b08, mixed mode)  
[liferay@apacheliferay01 opt]$
```

Figura 42. Versión de Java instalada

Anexo 12. Instalación y configuración de Liferay

Para instalar Liferay descargamos el “bundle” (paquete) de Liferay Tomcat desde la web de Liferay:

<https://www.liferay.com/es/downloads-community>

Guardamos el fichero y una vez descargado lo descomprimiremos usando “tar” en el sistema sobre el directorio /opt/ y definiremos a este directorio como el “home” de Liferay. Además, asignaremos como propietario al usuario “liferay” de forma recursiva a todo el directorio. Para ello nos posicionamos en /opt/ y ejecutamos

```
sudo tar -xzvf /home/centos/Downloads/liferay-ce-portal-tomcat-7.4.3.102-ga102-20231109165213885.tar.gz
sudo chown -R liferay: liferay-ce-portal-7.4.3.102-ga102/
```

Por simplicidad y porque en caso de actualización o cambio de versión de Liferay a futuro, definimos un enlace simbólico para poder acceder al directorio con un nombre más corto.

```
sudo ln -s liferay-ce-portal-7.4.3.102-ga102 liferay-ce-portal
```

Cambiamos al usuario “liferay” y vamos al directorio de binarios del Tomcat dentro de Liferay

```
sudo su - liferay
```

Y editamos el fichero de “setenv.sh” del Tomcat para asegurarnos de que usa correctamente el Java que hemos definido, para ello editaremos:

```
vim /opt/liferay-ce-portal/tomcat-9.0.82/bin/setenv.sh
```

Y añadimos la ruta donde tenemos nuestro Java instalado a la variable JAVA_HOME:

```
JAVA_HOME=/opt/java8
```

A continuación, procedemos a ejecutar por primera vez²⁰ el Liferay CE con el comando siguiente dentro del “./tomcat-9.0.82/bin” del directorio de instalación de Liferay:

```
./startup.sh
```

Podemos ver como el Tomcat ha arrancado correctamente y verificarlo en los logs del Tomcat ejecutando un:

```
tail -200f /opt/liferay-ce-portal/tomcat-9.0.82/logs/catalina.out
```

```
2023-11-12 13:30:26.141 INFO [main][ModuleFrameworkImpl:273] Navigate to Control Panel > System > G
ogo Shell and enter "lb" to see all bundles

Starting Liferay Community Edition Portal 7.4.3.102 CE GA102 (November 10, 2023)

2023-11-12 13:30:32.675 INFO [main][StartupHelperUtil:96] There are no patches installed
2023-11-12 13:30:32.745 INFO [main][LoggingTimer:74] Starting com.liferay.portal.events.StartupHelp
erUtil#initResourceActions
2023-11-12 13:30:32.825 INFO [main][LoggingTimer:35] Completed com.liferay.portal.events.StartupHel
perUtil#initResourceActions in 80 ms
2023-11-12 13:30:34.086 INFO [main][AutoDeployDir:152] Auto deploy scanner started for /opt/liferay
-ce-portal-7.4.3.102-ga102/deploy
12-Nov-2023 13:30:44.239 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in [8
609] milliseconds
```

Figura 43. Log de Liferay indicando su correcta inicialización

Ya podremos entrar con un navegador local de la máquina “apacheliferay01” a su web que ejecutará su *Wizard* para configurar por primera vez el Liferay (si no hubiera entorno gráfico se podrían exportar las X11 del CentOS al Host con un servidor X instalado y ejecutar un Firefox).

<http://localhost:8080>

Configuramos el nombre del Portal, idioma, huso horario y administrador como vemos aquí

Figura 44. Configuración inicial de Liferay

Y también configuramos la base de datos con la IP de nuestro servidor “mysql”, el usuario que habíamos configurado “lportal_user” y su contraseña “Liferay_01”. Además, agregamos el “Sample Data” ya que nos va a agregar contenido de ejemplo lo cual nos vendrá bien para mostrar las capacidades de Liferay CE.

BASE DE DATOS

[« Usar la base de datos por defecto](#)

Tipo de la base de datos

MySQL

URL de JDBC *

jdbc:mysql://192.168.1.30/lportal?characterEncoding=UTF-8&dontTrackOpenResources=true&f

Nombre de clase del driver de JDBC *

com.mysql.cj.jdbc.Driver

Nombre de usuario

lportal_user

Contraseña

••••••••••

DATOS DE EJEMPLO

Añadir datos de ejemplo ?

Finalizar Configuración

Figura 45. Configuración de base de datos de Liferay

Pulsamos en “Finalizar Configuración”, se nos informará de que todos los datos se han guardado en el siguiente archivo y nos indicará que reiniciemos:

```
/opt/liferay-ce-portal-7.4.3.102-ga102/portal-setup-wizard.properties
```

Ahora paramos y arrancamos el Tomcat como se nos indica de nuevo desde el mismo directorio “./tomcat-9.0.82/bin” dentro del directorio de instalación de Liferay:

```
./shutdown.sh
```

Y arrancamos de nuevo

```
./startup.sh
```

En los logs en `/opt/liferay-ce-portal/tomcat-9.0.82/logs/catalina.out` podemos ver como se están creando las tablas en la base de datos ejecutando:

```
tail -200f /opt/liferay-ce-portal/tomcat-9.0.82/logs/catalina.out
```

```
Loading file:/opt/liferay-ce-portal-7.4.3.102-ga102/portal-setup-wizard.properties
2023-11-12 20:30:32.220 INFO [main][PortalContextLoaderListener:114] JVM arguments: -Djava.util.lo
gging.config.file=/opt/liferay-ce-portal/tomcat-9.0.82/conf/logging.properties -Djava.util.logging.
manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -Djava.protocol.han
dler.pkgs=org.apache.catalina.webresources -Dorg.apache.catalina.security.SecurityListener.UMASK=00
27 -Dfile.encoding=UTF-8 -Djava.locale.providers=JRE,COMPAT,CLDR -Djava.net.preferIPv4Stack=true -D
user.timezone=GMT -Xms2560m -Xmx2560m -XX:MaxNewSize=1536m -XX:MaxMetaspaceSize=768m -XX:MetaspaceS
ize=768m -XX:NewSize=1536m -XX:SurvivorRatio=7 -Dignore.endorsed.dirs= -Dcatalina.base=/opt/liferay
-ce-portal/tomcat-9.0.82 -Dcatalina.home=/opt/liferay-ce-portal/tomcat-9.0.82 -Djava.io.tmpdir=/opt
/liferay-ce-portal/tomcat-9.0.82/temp
2023-11-12 20:30:39.358 INFO [main][DialectDetector:144] Using dialect org.hibernate.dialect.MySQL
8Dialect for MySQL 8.0
2023-11-12 20:30:39.462 INFO [main][DBInitUtil:94] Create tables and populate with default data
```

Figura 46. Log de creación de tablas en base de datos de Liferay

Una vez veamos en los logs que el Liferay ha levantado, volvemos a entrar en la web (tarda unos 40 minutos en arrancar la primera vez tras ejecutar el *Wizard*) y vemos que ya tenemos la instalación base de Liferay CE funcionando:

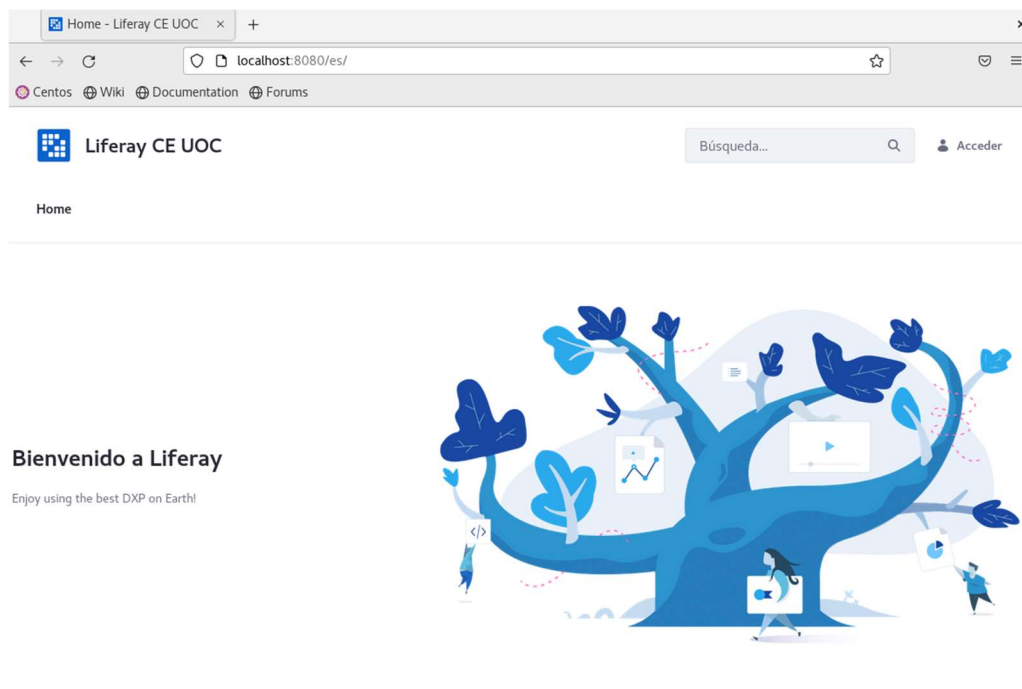


Figura 47. Web inicial de Liferay

Ahora podemos hacer login en el portal usando de usuario el mail que hemos creado, "jrussell@uoc.edu" y como contraseña "test" (que nos la crea así por defecto), aceptamos las condiciones de uso, respondemos una de las preguntas de seguridad de recuperación de contraseña y ya tendríamos todo listo con nuestro portal funcionando en local.

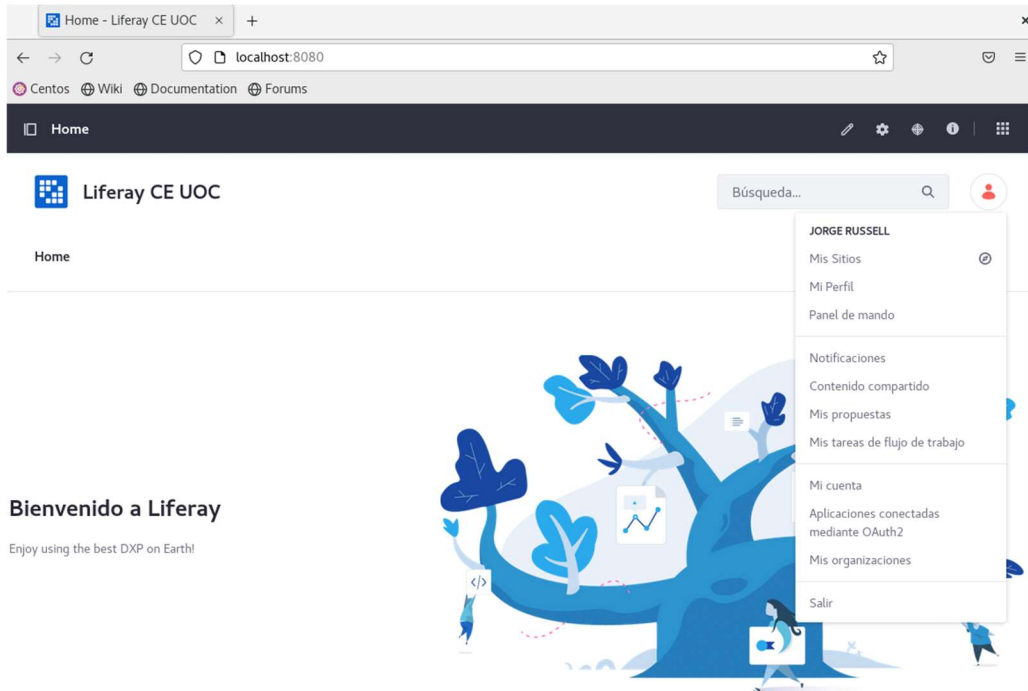


Figura 48. Web inicial de Liferay tras login de usuario

Anexo 13. Configuración del Data Library de Liferay

Para modificar la ruta del Data Library de Liferay, creamos el fichero `portal-ext.properties` en el directorio raíz, con:

```
vim portal-ext.properties
```

Y añadimos esta línea

```
dl.store.impl=com.liferay.portal.store.file.system.AdvancedFileSystems  
tore
```

Guardamos, cerramos y reiniciamos Liferay desde:

```
/opt/liferay-ce-portal/tomcat-9.0.82/bin/
```

Ejecutando:

```
./shutdown.sh
```

```
./startup.sh
```

Y entramos en la web y vamos al Panel de Control => Configuración del Sistema => Almacenamiento de archivos => Sistema de almacenamiento avanzado de archivos e introducimos nuestro punto de montaje NFS

<http://localhost:8080/>

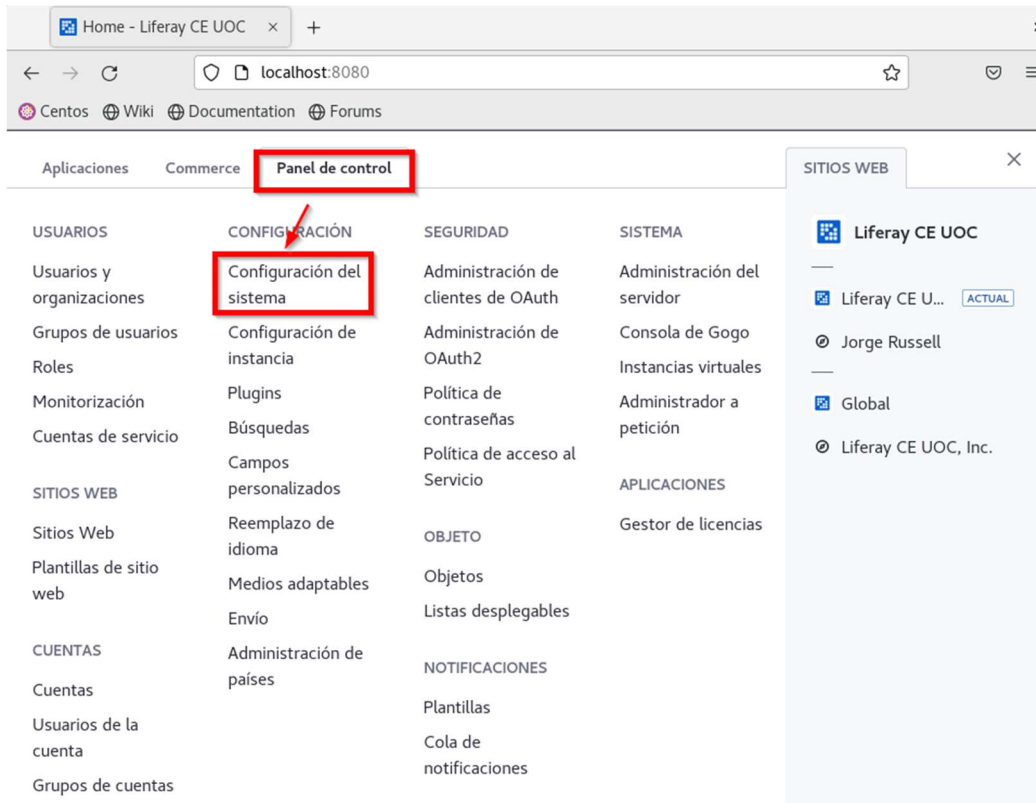


Figura 49. Panel de control de Liferay

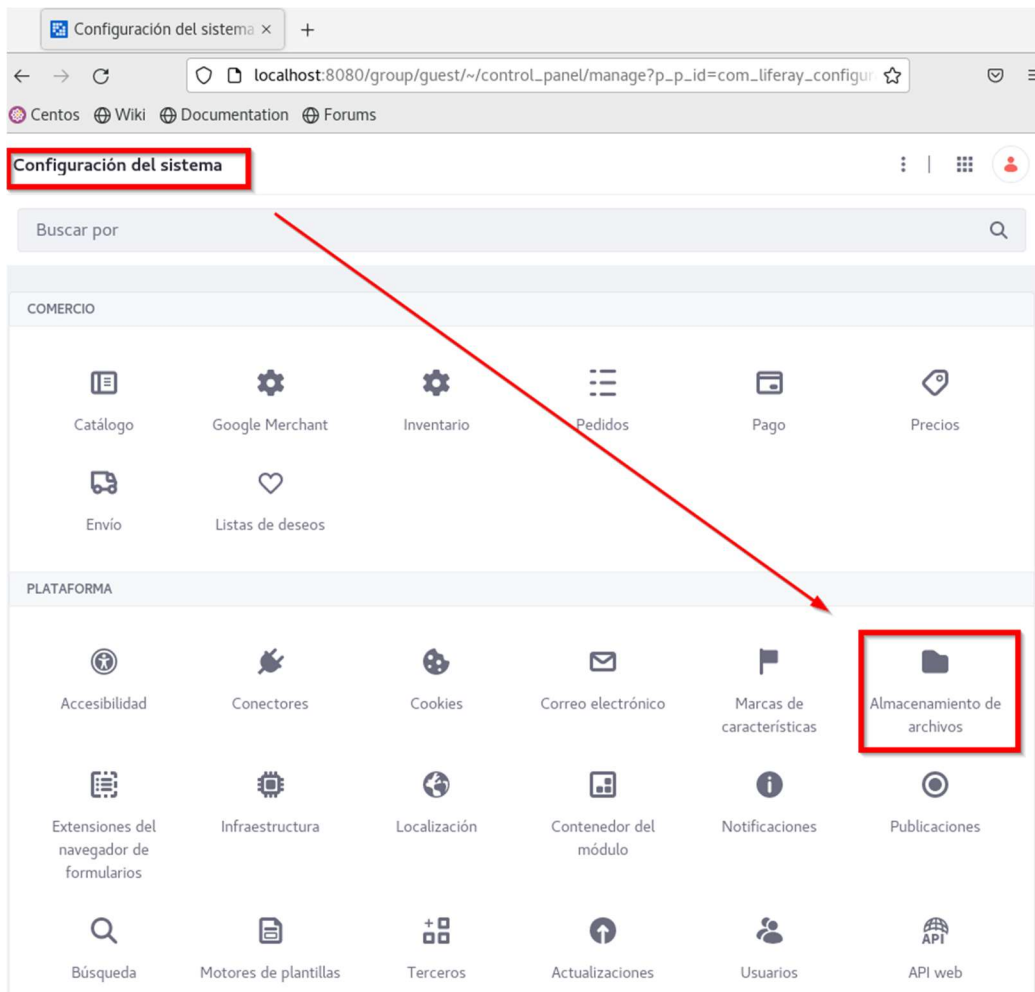


Figura 50. Configuración de almacenamiento de archivos de Liferay

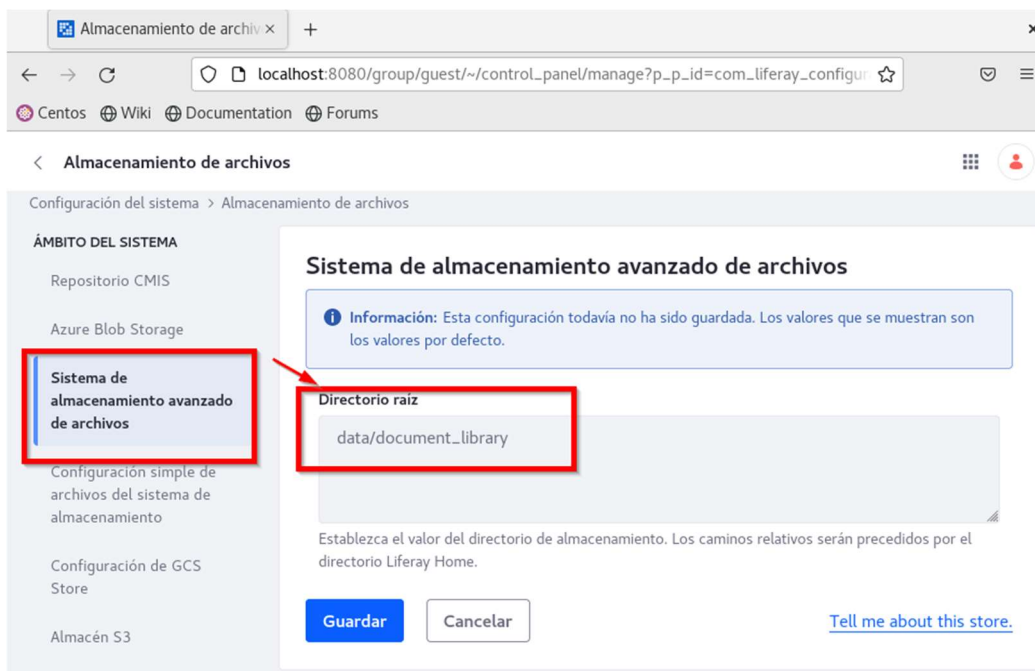


Figura 51. Parametrización almacenamiento archivos de Liferay

Cambiamos el valor por el absoluto con el punto de montaje del NFS que será /mnt/liferay/ y guardamos y exportamos el fichero.

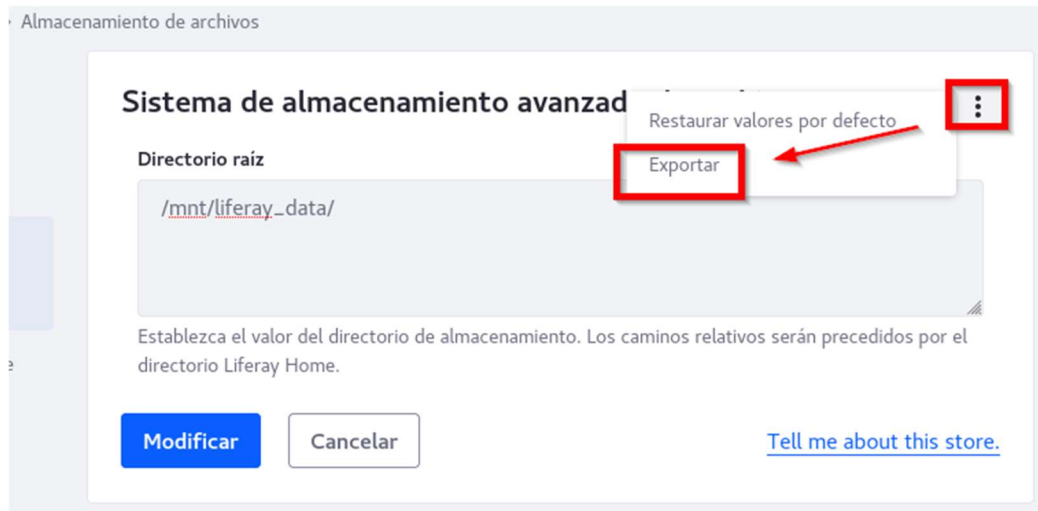


Figura 52. Exportación dantos Liferay

Lo cual nos exportará este fichero y hay que guardarlo ese fichero exportado en el subdirectorio de Liferay de “./osgi/configs”

```
com.liferay.portal.store.file.system.configuration.AdvancedFileSystemStoreConfiguration.config
```

Su contenido indica el nuevo directorio para que guarde los ficheros de su Document Library.

```
rootDir="/mnt/Liferay_data/"
```

De esta manera Liferay ya estaría usando ese directorio para escribir.

Anexo 14. Instalación y configuración de Elasticsearch en Liferay

Para evitar problemas con la versión exacta de Liferay, vamos a descargar la misma versión que viene incluida en el Paquete de Liferay Tomcat que podemos ver en esta ruta:

```
/opt/liferay-ce-portal/elasticsearch-sidecar
```

En dicho directorio se puede comprobar que la versión es la 7.17.14 y esta versión la tenemos aquí para descargar:

<https://www.elastic.co/es/downloads/past-releases/elasticsearch-7-17-14>

Seleccionamos la versión comprimida y la guardamos. Una vez guardada la descomprimimos en /opt/, hacemos propietario al usuario "liferay" y creamos un enlace simbólico para que sea más fácilmente accesible ejecutando estos comandos en /opt/

```
sudo tar -xzf /home/centos/Downloads/elasticsearch-7.17.14-linux-x86_64.tar.gz
sudo chown -R liferay: elasticsearch-7.17.14/
sudo ln -s elasticsearch-7.17.14 elasticsearch
```

Es recomendable fijar también la variable de entorno ES_JAVA_HOME, lo cual podemos hacer como hicimos antes editando el fichero ".bashrc" del *home* del usuario Liferay y añadiendo esta línea:

```
export ES_JAVA_HOME=/opt/java8
```

Para seguir con la instalación según la documentación²⁴ tendremos que ejecutar este comando, que permite que el sistema tenga procesos con un área de asignación de memoria mayor que las de por defecto y que es requerida por el proceso de Elasticsearch.

```
sudo sysctl -w vm.max_map_count=262144
```

Este cambio también debe hacerse de forma permanente en el sistema para que se apliquen cuando la máquina se reinicie también, para ello editamos el fichero "/etc/sysctl.conf" y añadimos la línea a continuación:

```
sudo vim /etc/sysctl.conf
```

```
vm.max_map_count=262144
```

En la documentación²⁴ se nos especifica que es necesario que instalemos una serie de *plugins* desde dentro del subdirectorio ".bin" del Elasticsearch, que son estos y ejecutando estos comandos dentro de "/opt/elasticsearch/bin" con el usuario liferay:

```
./elasticsearch-plugin install analysis-icu
```

```
./elasticsearch-plugin install analysis-kuromoji
./elasticsearch-plugin install analysis-smartcn
./elasticsearch-plugin install analysis-stempel
```

Nos da un aviso de que versiones futuras de Elasticsearch no soportarán Java 1.8 pero la actual si lo soporta según su matriz de compatibilidad²⁵. Además, se ve como instala esos *plugins* correctamente.

Pasamos ahora a realizar la configuración del fichero principal de Elasticsearch en “/opt/elasticsearch/config” con:

```
vim elasticsearch.yml
```

Y hacer una configuración como esta para el caso de un solo nodo (más adelante al configurar la alta disponibilidad cambiaremos esto). Aquí lo que tenemos es el nombre del clúster de Elasticsearch, indicamos que es un solo nodo, que usa los puertos 9200 y 9300, indicamos el nombre del host de la red, el nombre del nodo y desactivamos la seguridad de xpack, como indica la documentación.

```
cluster.name: LiferayElasticsearchCluster

discovery.type: single-node
http.port: 9200
network.host: es-node-1
node.name: es-node-1
transport.port: 9300

xpack.security.enabled: false
```

Por otro lado, como hemos indicado el nombre del host de la red, tenemos que editar el “/etc/hosts” de la máquina para que el sistema pueda asociar ese nombre a la IP de la máquina, para ello editamos y añadimos la línea a continuación a ese archivo:

```
sudo vim /etc/hosts
```

```
192.168.1.20    apacheliferay01    es-node-1
```

También nos recomienda meter este parámetro en el fichero “./config/jvm.options” para casos como ahora en el que solo hay un nodo de búsqueda:

```
-Des.enforce.bootstrap.checks=true
```

Arrancamos el Elasticsearch como demonio desde “/opt/elasticsearch/bin” ejecutando como usuario “liferay”:

```
./elasticsearch -d
```

Y podemos comprobar en los logs que arranca correctamente viendo los logs con:

```
tail -200f /opt/elasticsearch/logs/LiferayElasticsearchCluster.log
```

Veremos algo así en ellos:

```
[o.e.c.s.MasterService ] [es-node-1] elected-as-master
```

También lo podemos comprobar accediendo a la web que levanta desde nuestra máquina local (o por IP si accediéramos desde el host físico):

<http://es-node-1:9200/>

o

<http://192.168.1.20:9200/>

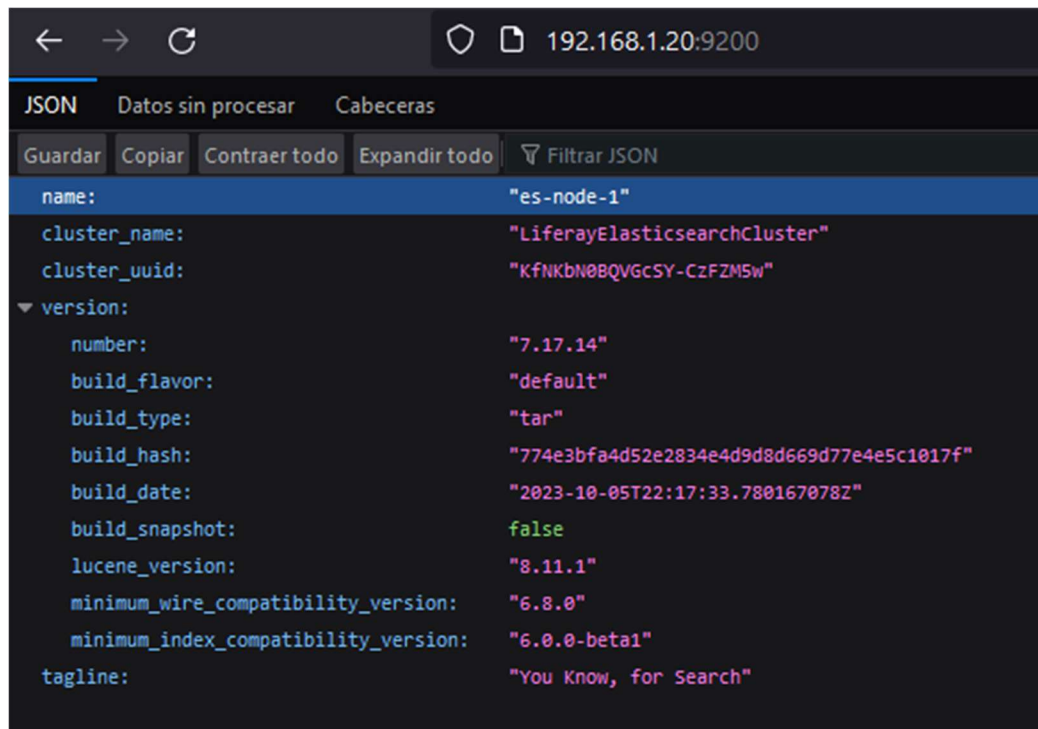


Figura 53. Web de estado de Elasticsearch

Pasaremos entonces a conectar el Liferay con el motor de búsqueda Elasticsearch, para ello podemos hacerlo de dos maneras²⁶. Una de ellas creando el fichero de configuración dentro del directorio de instalación del Liferay en “./osgi/configs/” y la otra es desde el interfaz web. Lo haremos editando el fichero por simplicidad:

```
vim
com.liferay.portal.search.elasticsearch7.configuration.ElasticsearchCo
nfiguration.config
```

Y añadimos estas dos líneas:

```
productionModeEnabled=B"true"  
networkHostAddresses=["http://es-node-1:9200"]
```

En el caso de que fuera un clúster, como se hará más adelante, habrá que añadir el resto de máquinas a esa lista, por ejemplo, así:

```
networkHostAddresses=["http://es-node-1:9200", "http://es-node-  
2:9200", "http://es-node-3:9200"]
```

Guardamos y Podemos arrancar el Liferay

```
./startup.sh
```

Y ahora si vamos a la web local de Liferay al Panel de control => Configuración => Búsquedas

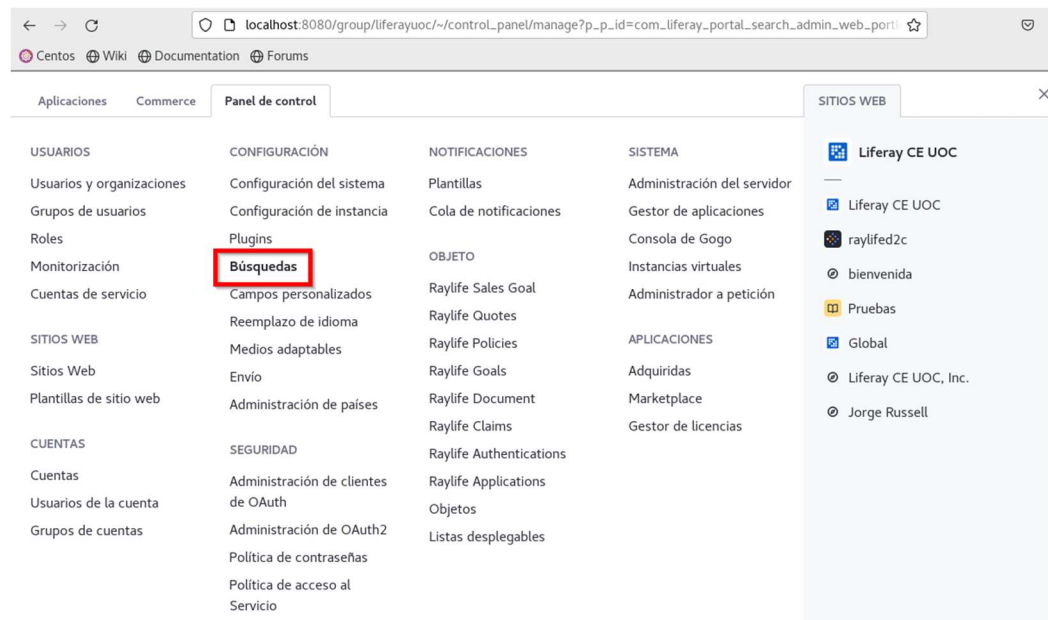


Figura 54. Configuración servicio de búsquedas en Liferay

Comprobamos que ha conectado correctamente al Elasticsearch

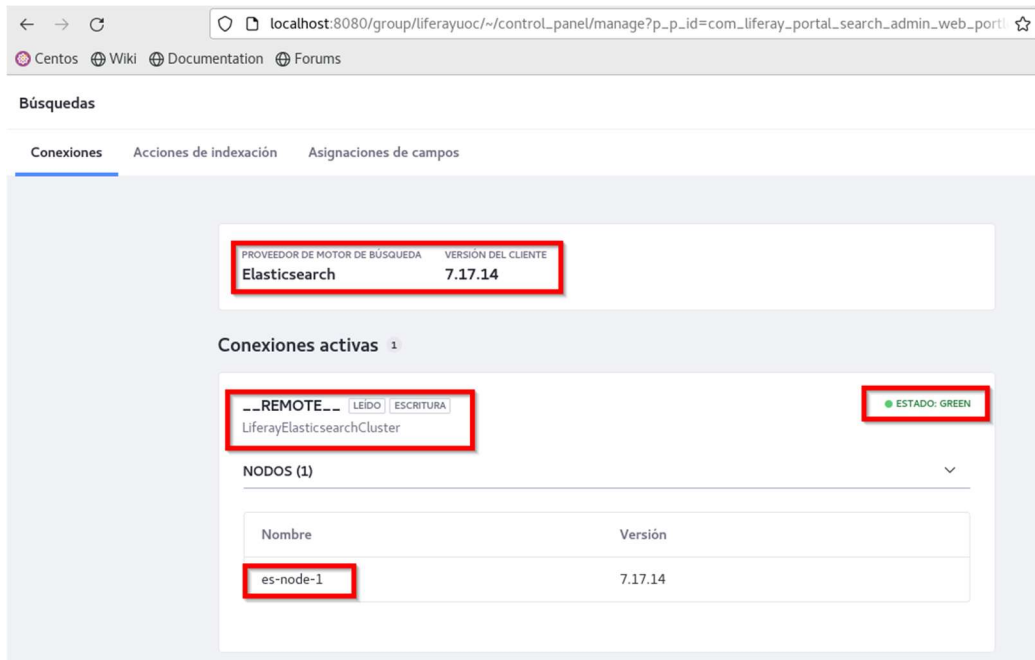


Figura 55. Elasticsearch en Liferay

Por otro lado, la documentación nos especifica que es necesario reindexar los índices de búsqueda y los de correcciones ortográficas, por tanto, pasamos a hacerlo.



Figura 56. Reindexación de contenido en Liferay con Elasticsearch

Confirmamos cada una de ellas en los logs de Liferay en "/opt/liferay-ce-portal/logs", donde se pueden ver estas reindexaciones, y con esto habríamos terminado de configurar el motor de búsqueda en Liferay

Anexo 15. Configuración de redirecciones de servidores web Apache a Liferay

Empezaremos añadiendo la configuración extra al servidor web Apache²⁷, para ello editaremos en la máquina “apacheliferay01” el fichero siguiente y añadimos la configuración a continuación:

```
sudo vim /etc/httpd/conf/httpd.conf
```

```
<VirtualHost *:80>
    # Set the header for the http protocol
    RequestHeader set X-Forwarded-Proto "http"

    # Serve /excluded from the local httpd data
    ProxyPass /excluded !

    # Preserve the host when invoking tomcat
    ProxyPreserveHost on

    # Pass all traffic to a localhost tomcat.
    ProxyPass / http://localhost:8080/
    ProxyPassReverse / http://localhost:8080/

    # This would be the configuration to invoke a tomcat on
    another server
    # ProxyPass / http://192.168.1.21:8080/
    # ProxyPassReverse / http://192.168.1.21:8080/
</VirtualHost>
```

Esta configuración añade un nuevo VirtualHost escuchando en el puerto 80 en cualquier IP, las cabeceras HTTP, hace una configuración de exclusión a /excluded en caso de que quisiéramos usar ese directorio para tener contenido estático (nos valdría para hacer pruebas aunque es opcional y se podría comentar), se mantiene el host al invocar al Tomcat ya que es necesario para Liferay, y se haría la redirección de todo el tráfico que entre al “/” (ProxyPass) del servidor web Apache al Tomcat para que le pase las peticiones y nos devuelva la web del Liferay.

Hemos dejado, además, la configuración extra para un nuevo servidor Liferay Tomcat con su IP para cuando tengamos que montar la alta disponibilidad, aunque la hemos dejado por el momento comentada. Esa configuración en el otro Apache tendría la IP 192.168.1.20 en vez de la 192.168.1.21.

Reiniciamos el servidor web Apache para que apliquen los cambios:

```
sudo systemctl restart httpd.service
```

Pasamos a aplicar unas propiedades^{27.28} al fichero “portal-ext.properties” de Liferay que permitirán las redirecciones, balanceos desde de los servidores web Apache (host, puerto, protocolo) , y hosts válidos o el Liferay no se nos abrirá vía web desde otro host remoto, para ello editamos el fichero en el *home* de Liferay y añadimos la siguiente configuración al final:

```
vim portal-ext.properties
```

```
# Redirect
redirect.url.security.mode=ip
redirect.url.ips.allowed=127.0.0.1,192.168.1.10,192.168.1.20,192
.168.1.21

# Balancing
web.server.forwarded.host.enabled=true
web.server.forwarded.port.enabled=true
web.server.forwarded.protocol.enabled=true

# Valid hosts
virtual.hosts.valid.hosts=localhost,127.0.0.1,[::1],[0:0:0:0:0:0
:0:1],192.168.1.10,192.168.1.20,192.168.1.21,liferayuoc.org,life
raypruebasuoc.org,raylifed2cuoc.org
```

Nótese que hemos añadido como hosts válidos tres dominios que usaremos luego para definir el acceso externo a la web por defecto de Liferay y otras dos de prueba que crearemos y configuraremos más adelante.

Guardamos, cerramos y reiniciamos el Liferay desde “/opt/liferay-ce-portal/ tomcat-9.0.82/bin/”:

```
./shutdown.sh
./startup.sh
```

Una vez hemos reiniciado tendremos que definir un VirtualHost en Liferay para su web principal, para ello vamos a los ajustes del sitio

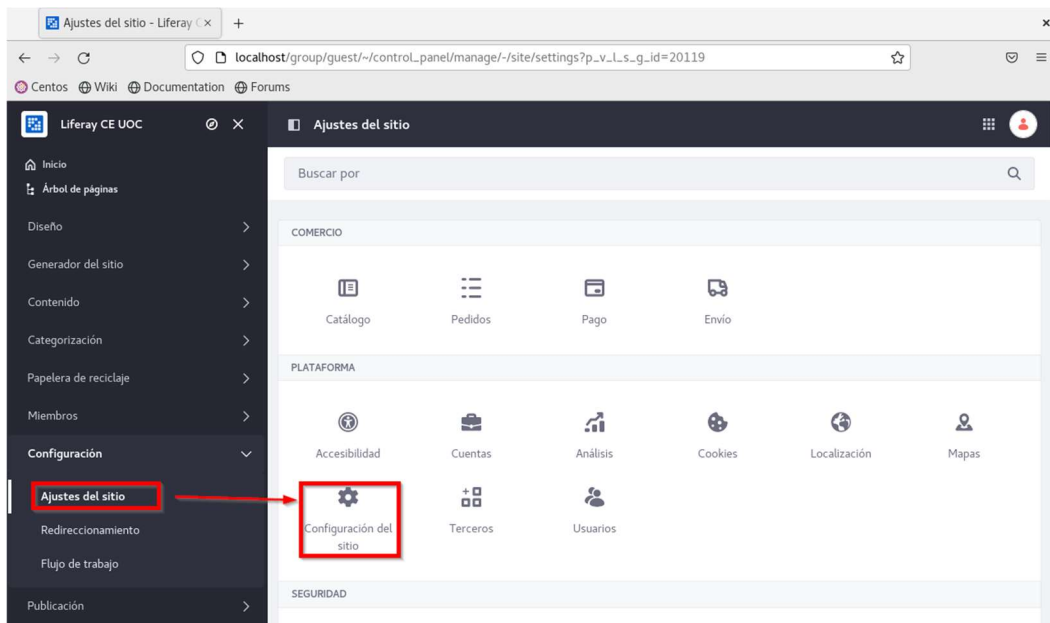


Figura 57. Ajustes de sitio de Liferay

Definimos sus servidores virtuales al menos como <http://liferayuoc.org> porque es la que usaremos luego para acceder externamente, la URL amigable la podemos dejar como está y guardaremos cambios.

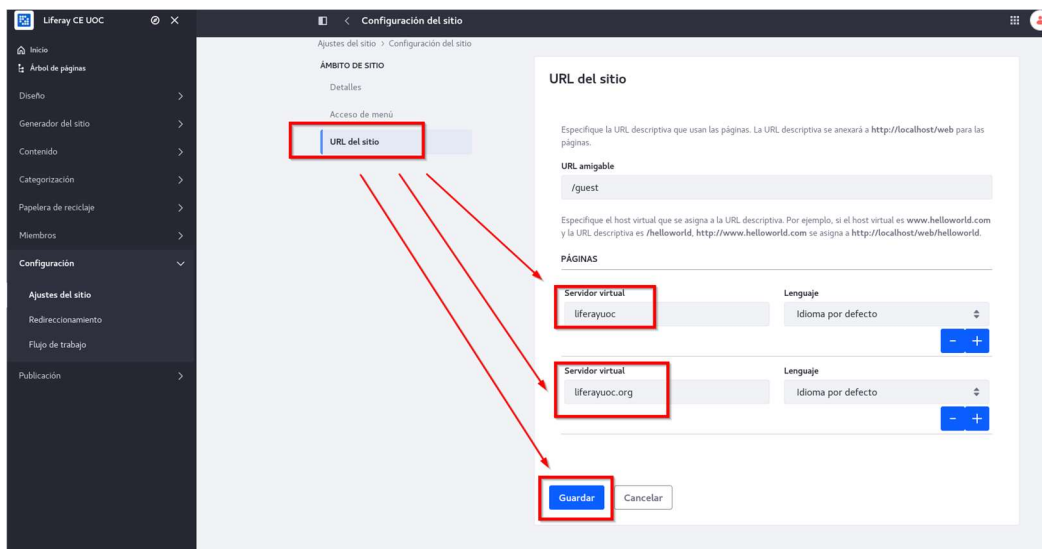


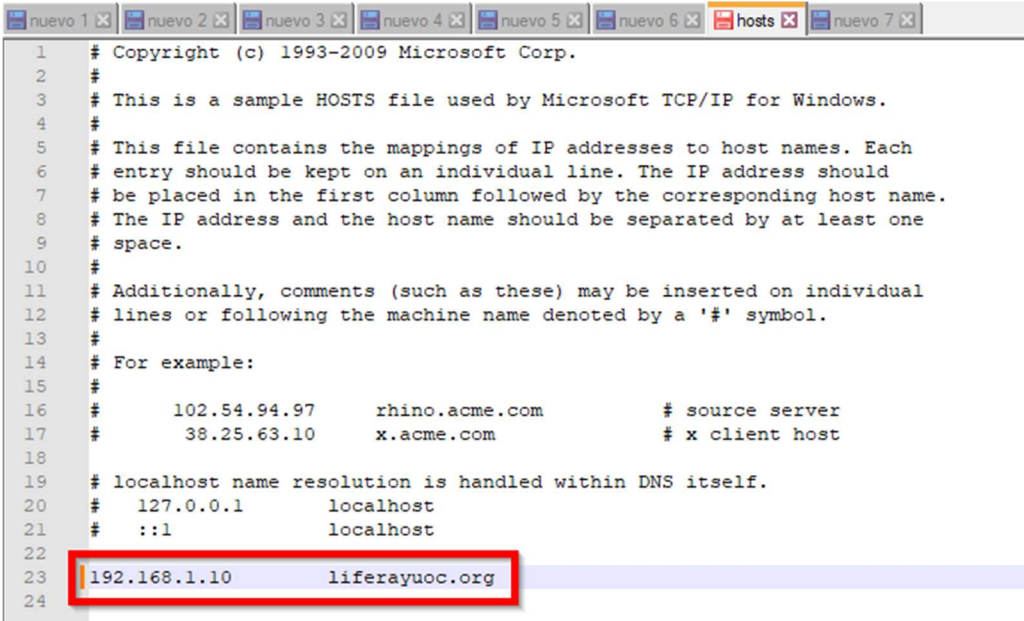
Figura 58. Servidores virtuales en Liferay

Anexo 16. Entradas de DNS locales

Para añadir dominios DNS de forma local editamos el fichero de nuestra máquina host Windows que normalmente estará en:

C:\Windows\System32\drivers\etc\hosts

Y añadiremos esta entrada que es la IP de nuestra máquina que tiene el HAProxy.



```
1 # Copyright (c) 1993-2009 Microsoft Corp.
2 #
3 # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
4 #
5 # This file contains the mappings of IP addresses to host names. Each
6 # entry should be kept on an individual line. The IP address should
7 # be placed in the first column followed by the corresponding host name.
8 # The IP address and the host name should be separated by at least one
9 # space.
10 #
11 # Additionally, comments (such as these) may be inserted on individual
12 # lines or following the machine name denoted by a '#' symbol.
13 #
14 # For example:
15 #
16 #       102.54.94.97       rhino.acme.com           # source server
17 #       38.25.63.10      x.acme.com             # x client host
18
19 # localhost name resolution is handled within DNS itself.
20 #   127.0.0.1       localhost
21 #       ::1         localhost
22
23 192.168.1.10       liferayuoc.org
24
```

Figura 59. Fichero etc/hosts de nuestro host anfitrión

De tal manera que ahora al poner en el navegador de nuestro host Windows la URL siguiente lo que hará será ir al HAProxy que nos redirigirá al servidor web Apache y a su vez al Tomcat que tiene el Liferay realizando todo el circuito.

<http://liferayuoc.org>

Anexo 17. Comprobación de logs de accesos web en Syslog y Apaches

Se pueden comprobar en los logs del HAProxy en la máquina “haproxy” como se ven las llamadas a los Apaches desde la IP del host Windows a las URLs del portal de Liferay que se han configurado como la de:

<http://liferayuoc.org>

```
sudo tail -200f /var/log/haproxy-access.log
```

```
liferay /exports/@clayui$slider.js HTTP/1.1"
Nov 18 15:57:54 localhost haproxy[4969]: 192.168.1.33 52627 [18/Nov/2023:15:57:54.542] liferay.local
apache_websevers/apache-node01 0/0/0/4/4 304 217 - - - - 5/5/0/0/0 0/0 "GET /combo/?browserId=firef
ox&minifierType=&languageId=es_ES&t=1699579706221&/o/frontend-js-ai-web/liferay/navigation.js&/o/fro
ntend-js-ai-web/liferay/store.js&/o/frontend-js-ai-web/liferay/layout.js HTTP/1.1"
Nov 18 15:58:53 localhost haproxy[4969]: 192.168.1.33 52634 [18/Nov/2023:15:58:53.947] liferay.local
apache_websevers/apache-node01 0/0/0/2/2 304 217 - - - - 1/1/0/0/0 0/0 "GET /combo/?browserId=firef
ox&minifierType=&languageId=es_ES&t=1699579706221&/o/frontend-js-ai-web/ai/ai-sortable-layout/asse
ts/skins/sam/ai-sortable-layout.css HTTP/1.1"
Nov 18 15:58:53 localhost haproxy[4969]: 192.168.1.33:52634 [18/Nov/2023:15:58:53.954] liferay.local
apache_websevers/apache-node01 0/0/0/16/16 304 217 - - - - 1/1/0/0/0 0/0 "GET /combo/?browserId=fir
efox&minifierType=&languageId=es_ES&t=1699579706221&/o/frontend-js-ai-web/ai/dd-ddm-base/dd-ddm-bas
e-min.js&/o/frontend-js-ai-web/ai/dd-drag/dd-drag-min.js&/o/frontend-js-ai-web/ai/event-resize/ev
ent-resize-min.js&/o/frontend-js-ai-web/ai/dd-ddm/dd-ddm-min.js&/o/frontend-js-ai-web/ai/dd-ddm-d
rop/dd-ddm-drop-min.js&/o/frontend-js-ai-web/ai/dd-drop/dd-drop-min.js&/o/frontend-js-ai-web/ai/d
d-drop-plugin/dd-drop-plugin-min.js&/o/frontend-js-ai-web/ai/dd-delegate/dd-delegate-min.js&/o/fron
tend-js-ai-web/ai/dd-proxy/dd-proxy-min.js&/o/frontend-js-ai-web/ai/ai-sortable-layout/ai-sorta
ble-layout-min.js&/o/frontend-js-ai-web/ai/dd-constrain/dd-constrain-min.js&/o/frontend-js-ai-web/
ai/dd-scroll/dd-scroll-min.js&/o/frontend-js-ai-web/liferay/layout_column.js HTTP/1.1"
```

Figura 60. Logs HAPorxy

Podemos hacer lo mismo con los logs del servidor web Apache en la máquina “apacheliferay01” y “apacheliferay01”. En ellos se pueden ver los GET que se realizan al Liferay desde la IP de la máquina de HAProxy con y se ven los mensajes de “HTTP/1.0 200” que indican una carga correcta de la web:

```
sudo tail -200f /var/log/httpd/access_log
```

```
192.168.1.10 - - [18/Nov/2023:16:03:16 +0100] "GET /combo/?browserId=firefox&minifierType=&languageId
&/o/frontend-js-ai-web/liferay/navigation.js&/o/frontend-js-ai-web/liferay/store.js&/o/frontend-js
t.js HTTP/1.1" 304 - "http://liferayuoc.org/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) G
/119.0"
192.168.1.10 - - [18/Nov/2023:16:03:16 +0100] "GET /combo/?browserId=firefox&minifierType=&languageId
&/o/frontend-js-ai-web/ai/ai-sortable-layout/assets/skins/sam/ai-sortable-layout.css HTTP/1.1" 3
bc.org/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
192.168.1.10 - - [18/Nov/2023:16:03:16 +0100] "GET /combo/?browserId=firefox&minifierType=&languageId
&/o/frontend-js-ai-web/ai/dd-ddm-base/dd-ddm-base-min.js&/o/frontend-js-ai-web/ai/dd-drag/dd-dra
js-ai-web/ai/event-resize/event-resize-min.js&/o/frontend-js-ai-web/ai/dd-ddm/dd-ddm-min.js&/o/fr
dd-ddm-drop/dd-ddm-drop-min.js&/o/frontend-js-ai-web/ai/dd-drop/dd-drop-min.js&/o/frontend-js-ai-
h/dd-drop-plugin-min.js&/o/frontend-js-ai-web/ai/dd-delegate/dd-delegate-min.js&/o/frontend-js-ai-
roxy-min.js&/o/frontend-js-ai-web/ai/ai-sortable-layout/ai-sortable-layout-min.js&/o/frontend-js-
ain/dd-constrain-min.js&/o/frontend-js-ai-web/ai/dd-scroll/dd-scroll-min.js&/o/frontend-js-ai-web/
js HTTP/1.1" 304 - "http://liferayuoc.org/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Ge
19.0"
192.168.1.10 - - [18/Nov/2023:16:03:17 +0100] "OPTIONS / HTTP/1.0" 200 - "-" "-"
```

Figura 61. Logs servidor web Apache

Anexo 18: Creación de sitios web en Liferay

Entrando en la web de <http://liferayuoc.org> y desde el Panel de Control => Sitios Web:

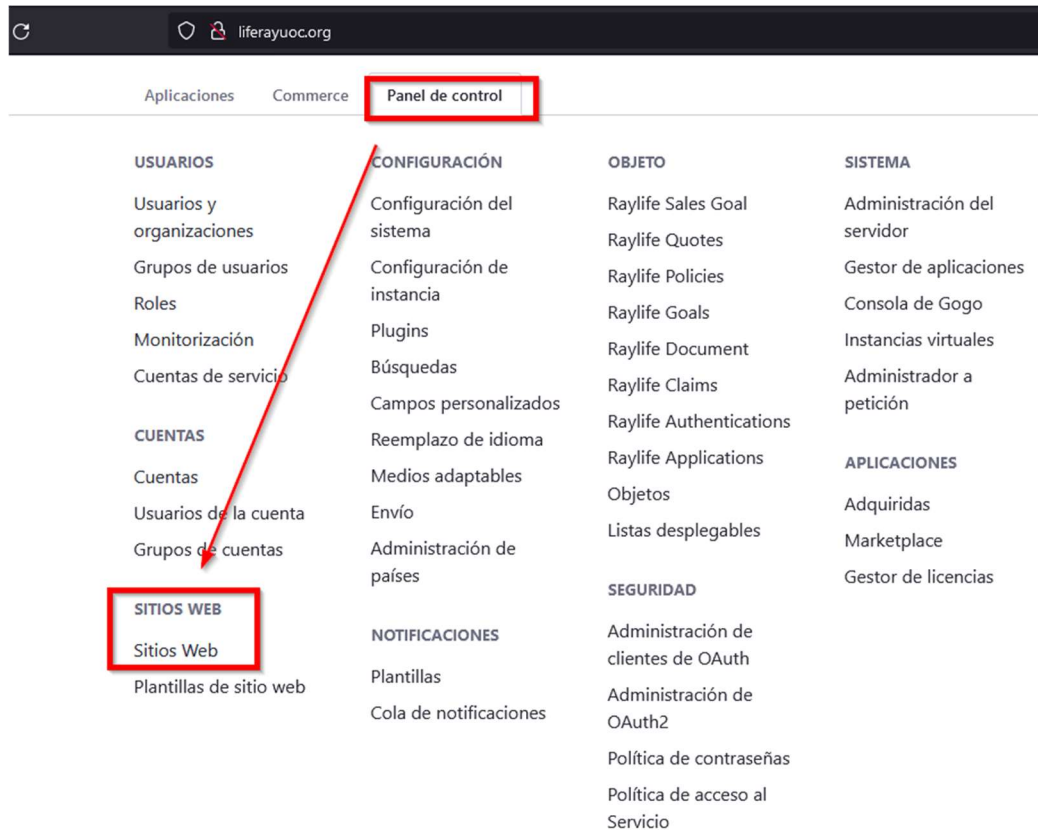


Figura 62. Panel de control de Liferay, creación de sitios web

Y pulsamos en el signo + para crear un nuevo sitio

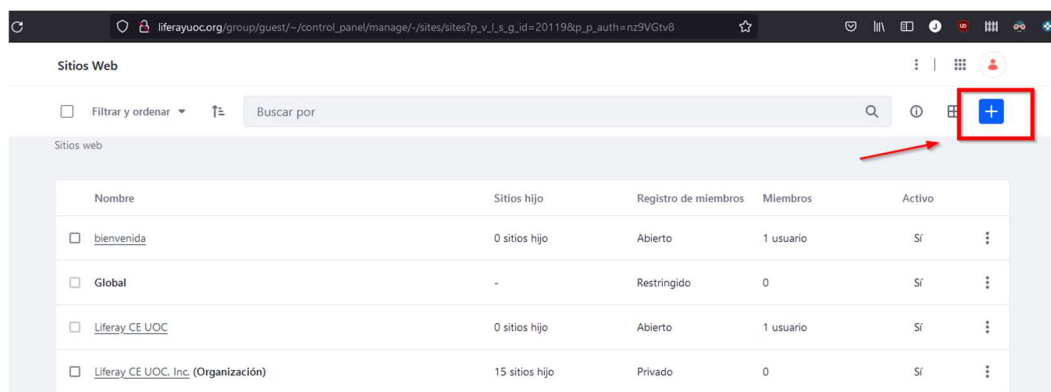


Figura 63. Creación de sitios web en Liferay

Seleccionamos una plantilla, como por ejemplo la de Masterclass

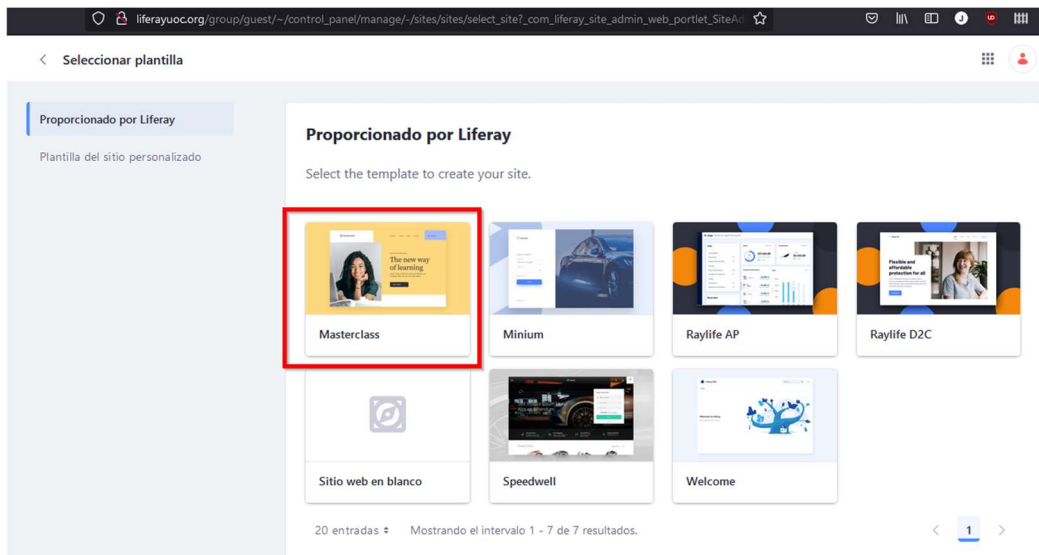


Figura 64. Plantillas web de sitios en Liferay

Usamos de nombre “Pruebas” y pulsamos “Añadir”

Y tendremos un nuevo sitio creado

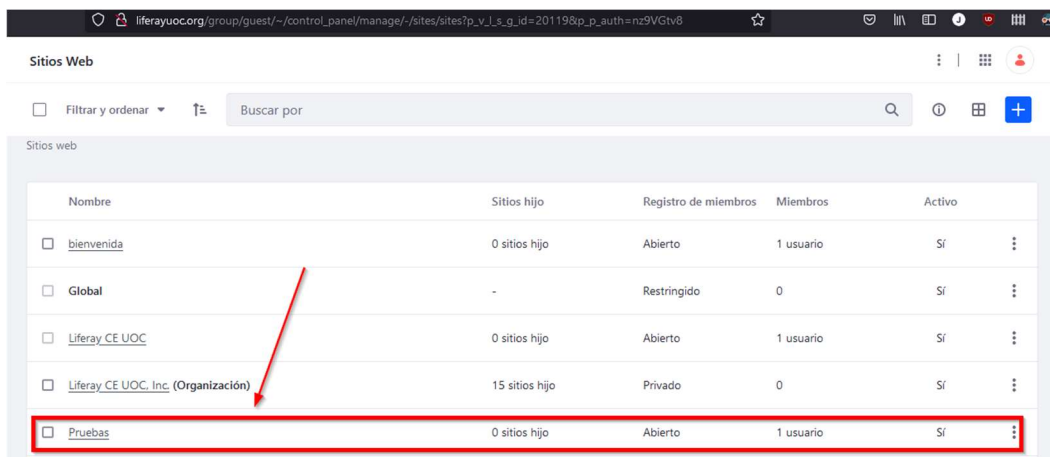


Figura 65. Nuevo sitio web creado en Liferay

Entramos en el nuevo sitio



Figura 66. Sitio web de pruebas creado en Liferay

Y veremos que tenemos un sitio con contenido básico que se puede editar, añadir más contenido, más páginas, etc.

Lo que sucede es que de momento se abre con la URL <http://liferayuoc.org/web/pruebas> pero podemos asignar uno de los dominios que definimos antes como hosts aceptados. Para Esto vamos a “Ajustes del Sitio” en el menú de la izquierda.

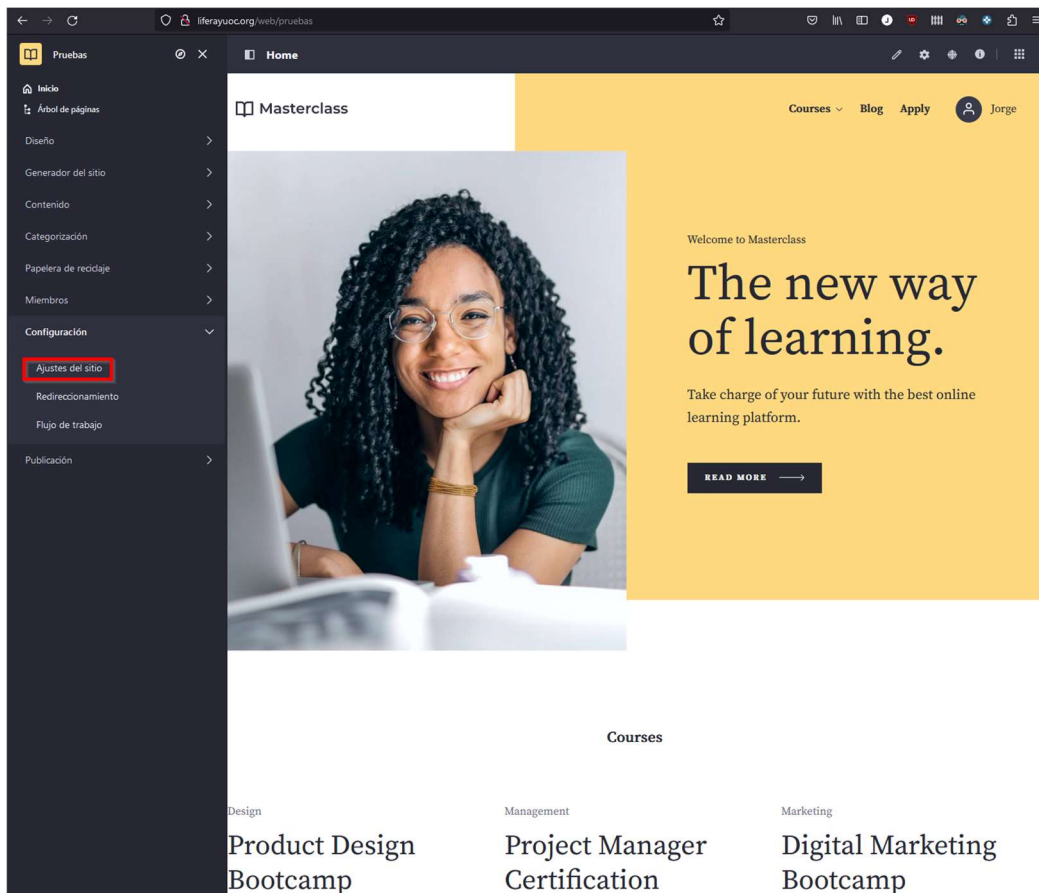


Figura 67. Sitio web creado en Liferay

Luego a “Configuración del sitio”

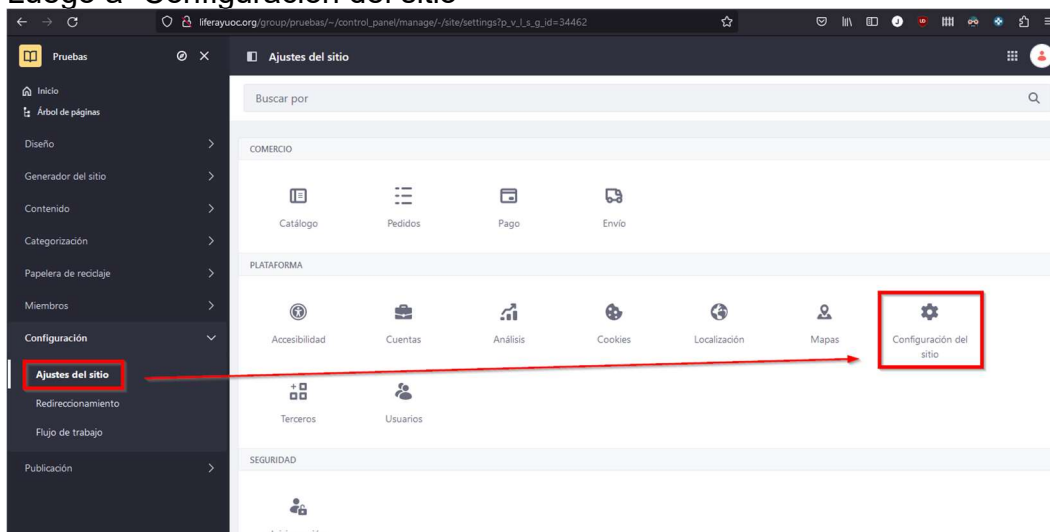


Figura 68. Configuración de sitios web en Liferay

Y luego a “URL del sitio”, cambiamos si hace falta la URL amigable (que por defecto deja el nombre que creamos para la web), el nuevo dominio y guardamos.

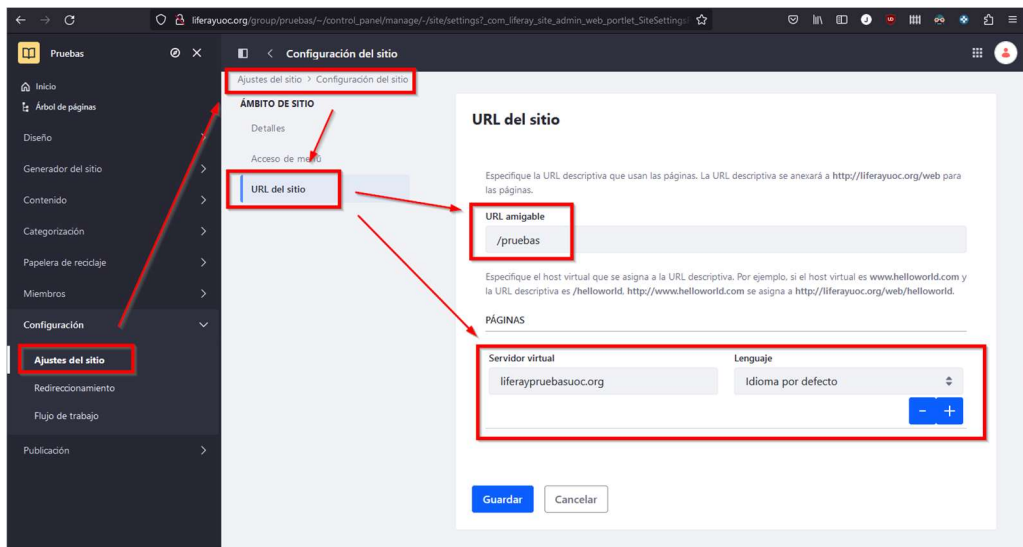
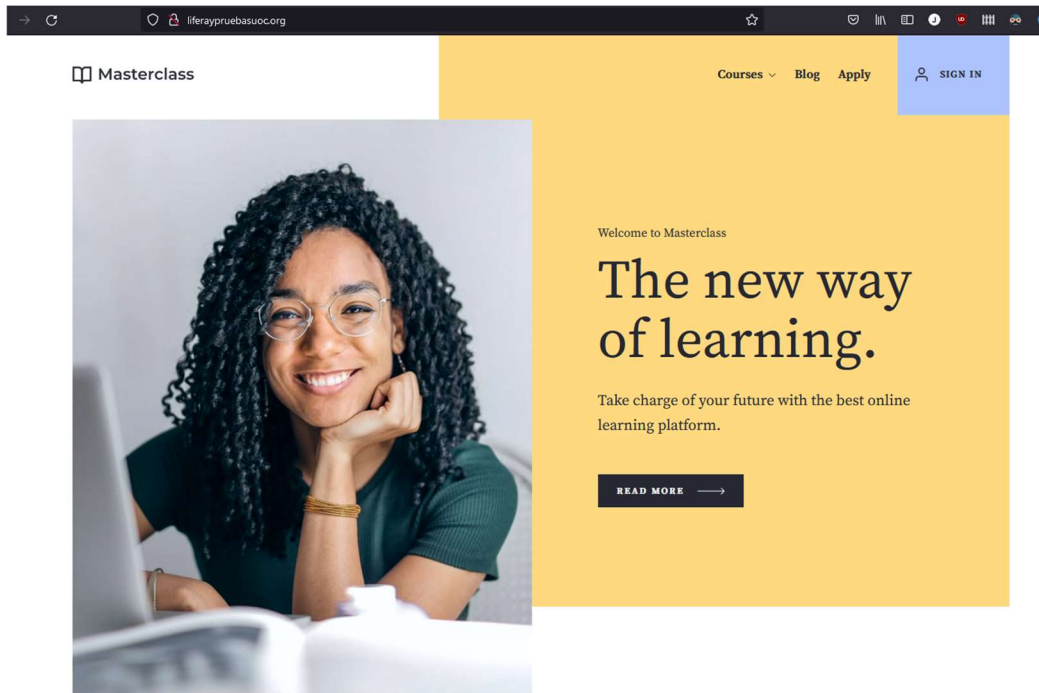


Figura 69. URL Amigable de sitio web en Liferay

Y ya podremos entrar usando esa URL tras editar el fichero de “etc/hosts” de nuestro host Windows (véase [Anexo 16](#)) y añadirla como alias, como hemos hecho antes con la principal, de tal manera que ya podemos acceder con la URL <http://liferaypruebasuoc.org/>



Courses

Design

Product Design
Bootcamp

Management

Project Manager
Certification

Marketing

Digital Marketing
Bootcamp

Figura 70. Sitio web de prueba final en Liferay

Repetimos pasos creando otra web con otra plantilla y con el otro dominio <http://raylified2cuoc.org>

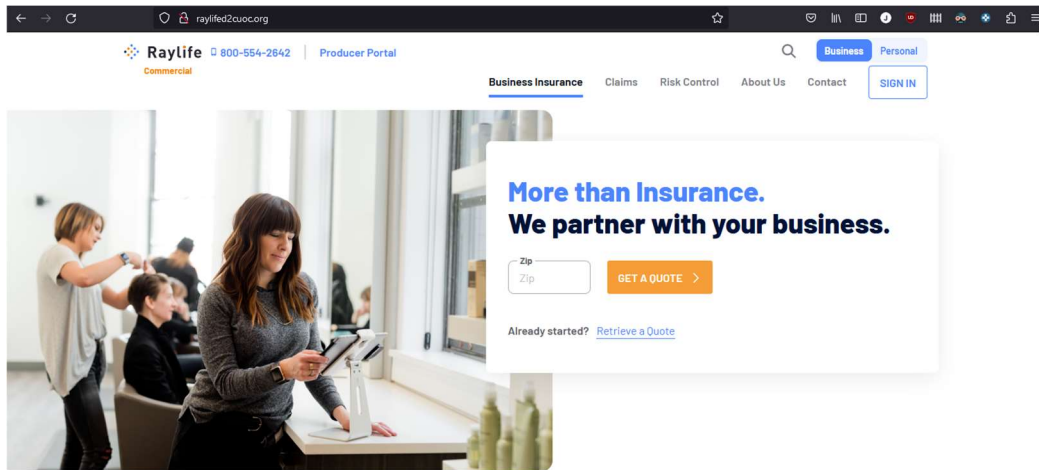


Figura 71. Sitio web de prueba secundario generado en Liferay

Hay que tener en cuenta que entrando por el dominio creado desaparece el menú de administración, lo cual es algo completamente normal, así se nos permite crear sitios para usuarios finales, mientras la administración queda aislada por otro dominio e incluso podría definirse para entrar desde una Intranet en vez desde Internet, con lo que tendríamos la administración aislada del exterior por seguridad.

Anexo 19. Gestión de contenido dentro de Liferay

Para gestionar contenido vamos a la web principal de administración de <http://liferayuoc.org> y agregamos a contenido de prueba como una carpeta o un fichero, por ejemplo, este mismo documento.

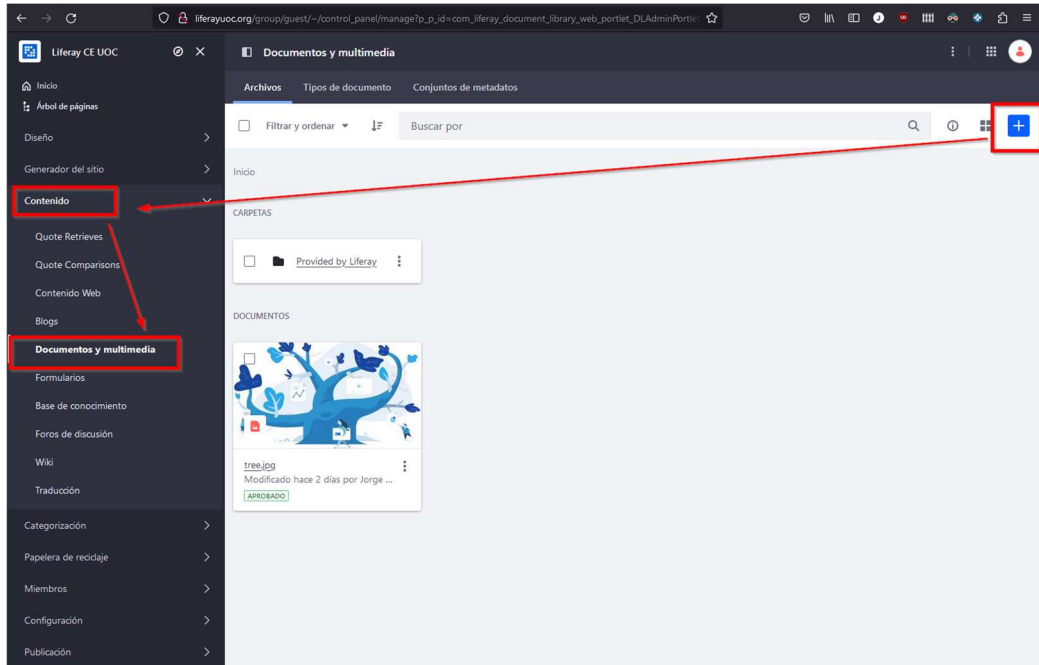


Figura 72. Gestión de contenido dentro de Liferay

Se pueden definir varios parámetros como una URL amigable, pero lo dejamos todo por defecto y pulsamos abajo en “Publicar”

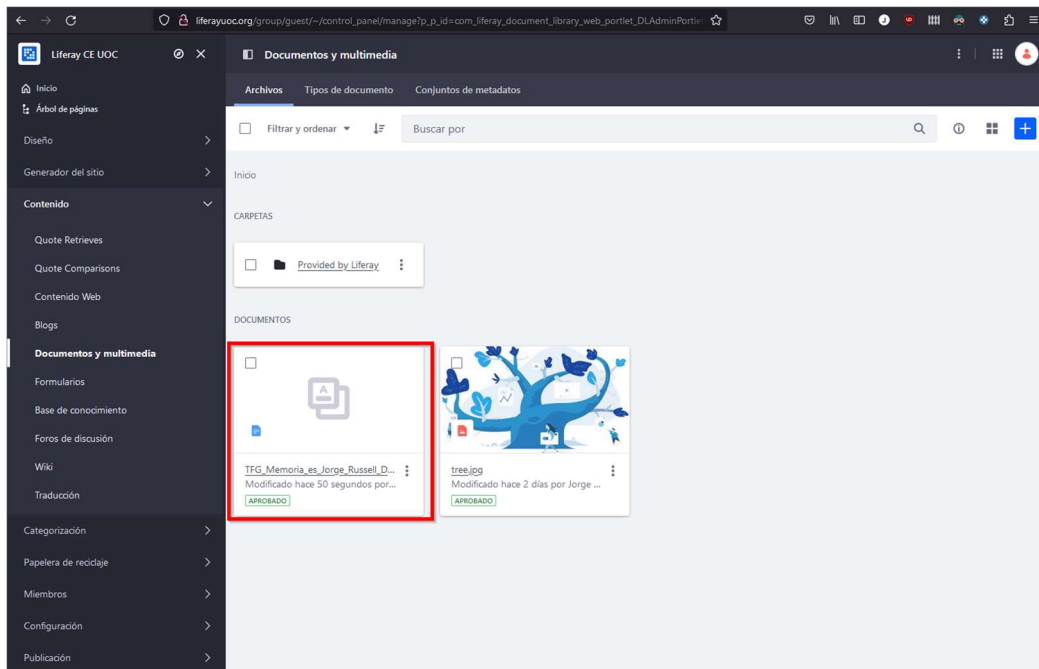


Figura 73. Contenido publicado dentro de Liferay

Anexo 20. Configuración de balanceo de carga y persistencia de sesión entre servidores web Apache y servidores Tomcat

Cambiaremos ligeramente la configuración extra el servidor web Apache. para ello editaremos en la máquina “apacheliferay01” y “apacheliferay02”, el fichero que modificamos anteriormente y cambiamos la configuración a continuación:

```
sudo vim /etc/httpd/conf/httpd.conf
```

```
<VirtualHost *:80>
  # Set the header for the http protocol
  RequestHeader set X-Forwarded-Proto "http"

  # Serve /excluded from the local httpd data
  ProxyPass /excluded !

  # Preserve the host when invoking tomcat
  ProxyPreserveHost on

  # Allows URL rewrite so they are more friendly
  RewriteEngine On

  # ProxyPass and ProxyPassReverse directives to redirect HTTP
  traffic to balanced members
  ProxyPass "/" "balancer://liferay/"
  ProxyPassReverse "/" "balancer://liferay/"

  # Balancer members with jvmRoute and sticky session to
  perssit sessions, connection timeout and lbmethod
  <Proxy "balancer://liferay">
    BalancerMember "http://192.168.1.20:8080" route=jvm1
    loadfactor=1 max=64 connectiontimeout=1200
    BalancerMember "http://192.168.1.21:8080" route=jvm2
    loadfactor=1 max=64 connectiontimeout=1200
    ProxySet stickysession=JSESSIONID
    ProxySet lbmethod=byrequests
  </Proxy>
</VirtualHost>
```

Esta configuración es análoga a la que teníamos antes (véase [Anexo 15](#)) manteniendo el VirtualHost escuchando en el puerto 80 en cualquier IP, las cabeceras HTTP, la configuración de exclusión a /excluded en caso de que quisiéramos usar ese directorio para tener contenido estático (nos valdría para hacer pruebas aunque es opcional y se podría comentar), se mantiene el host al invocar al Tomcat ya que es necesario para Liferay, y se haría la redirección de todo el tráfico que entre al “/” (ProxyPass) del

servidor web Apache al Tomcat mediante un clúster “Balancer” al que hemos llamado “liferay” para que le pase las peticiones, al clúster de máquinas dentro del apartado “Proxy” y nos devuelva la web del Liferay usando el grupo “Balancer” que será el grupo de servidores de aplicativo de Tomcat que ejecutan el Liferay.

Esta configuración mantiene la persistencia de sesión (para que el usuario no pierda el trabajo que está realizando en caso de fallo de uno de los servidores y se salte al otro) mediante la directiva “stickysession” con JSESSIONID combinada con el parámetro “route” y balancea la carga equitativamente entre los Tomcat a la par que provee de la alta disponibilidad que estamos buscando.

No obstante, hay que usar el parámetro “route” para cambiar también la configuración en los Tomcat para que usando otro parámetro “jvmRoute” sean idénticos. Para ello editamos el fichero de configuración en las máquinas “apacheliferay01” y “apacheliferay02, del Tomcat y agregamos ese parámetro en cada uno de ellos y que coincida con lo configurado en el de “route” de los Apache

```
vim /opt/liferay-ce-portal/tomcat-9.0.82/conf/server.xml
```

Y modificamos esta línea en cada máquina de tal manera que quede así

apacheliferay01:

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="jvm1">
```

apacheliferay02:

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="jvm2">
```

De esta manera se conseguirá la persistencia de sesión.

Aplicaremos los cambios al reiniciar ambos servidores web Apache en ambas máquinas con:

```
sudo systemctl restart httpd.service
```

Aplicaremos también los cambios al reiniciar los servidores Tomcat en ambas máquinas yendo al directorio:

```
/opt/liferay-ce-portal/tomcat-9.0.82/bin/
```

Y ejecutando la parada (si procede) y el arranque:

```
./shutdown.sh
```

```
./startup.sh
```

Anexo 21. Configuración en clúster de servidores de búsqueda Elasticsearch

Para poder configurar Elasticsearch para su funcionamiento en clúster, el prerequisite es tener instaladas y configuradas las Java Runtime JDK (véase [Anexo 11](#)) en las máquinas.

Posteriormente, se hará una instalación y configuración base de Elasticsearch (véase [Anexo 14](#)) en las máquinas que vayan a formar parte del clúster.

Por último, configuraremos los servicios de Elasticsearch de cada una de las máquinas para que funcionen en clúster, en modo, dos maestros que harán a su vez de réplica de datos (apacheliferay01 y apacheliferay02), y un tercer nodo que ejercerá de nodo de desempate (mysql) para elegir maestro en caso de que alguno de los nodos se caiga.

Para ello, editamos el fichero de configuración en el directorio “/opt/elasticsearch/config/” de las máquinas y lo dejamos de la siguiente manera en cada una de ellas:

```
vim elasticsearch.yml
```

Las máquinas que harán de maestros elegibles y réplicas de datos:

apacheliferay01

```
# Use a descriptive name for your cluster:
cluster.name: LiferayElasticsearchCluster
# Use a descriptive name for the node:
node.name: es-node-1
# address here to expose this node on the network:
network.host: 192.168.1.20
# By default Elasticsearch listens for HTTP traffic on the first
free port it
# finds starting at 9200. Set a specific HTTP port here:
http.port: 9200
# Transport.port
transport.port: 9300
# Pass an initial list of hosts to perform discovery when this
node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
discovery.seed_hosts: ["es-node-1", "es-node-2", "es-node-3"]
# Disable security
xpack.security.enabled: false
# Disable geoip Downloader
ingest.geoip.downloader.enabled: false
```


apacheliferay02:

```
# Use a descriptive name for your cluster:
cluster.name: LiferayElasticsearchCluster
# Use a descriptive name for the node:
node.name: es-node-2
# address here to expose this node on the network:
network.host: 192.168.1.21
# By default Elasticsearch listens for HTTP traffic on the first
free port it
# finds starting at 9200. Set a specific HTTP port here:
http.port: 9200
# Transport.port
transport.port: 9300
# Pass an initial list of hosts to perform discovery when this
node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
discovery.seed_hosts: ["es-node-1", "es-node-2", "es-node-3"]
# Disable security
xpack.security.enabled: false
# Disable geoip Downloader
ingest.geoip.downloader.enabled: false
```

Estas dos configuraciones indican al Elasticsearch el nombre del clúster, el nombre del nodo, su IP, los puertos http de consulta y transporte, la lista de nodos del clúster, la seguridad que la dejaremos deshabilitada y que no se descargue de Internet datos de geolocalización de IPs, ya que no son necesarias y consumen recursos en el arranque.

Por defecto, si no indicamos el rol de las máquinas, ambas tendrán rol de maestro y de réplica de datos.

Por otro lado, en la máquina que hará de nodo de desempate (nodo de voto sólo), le especificaremos con el parámetro “node.roles”, que es la máquina encargada del desempate mediante la especificación “master” y “voting_only”, que a su vez indican que es un maestro que no forma parte de los maestros elegibles pero que sí que puede votar por los maestros elegibles, que son las otras dos máquinas:

mysql:

```
# Use a descriptive name for your cluster:
cluster.name: LiferayElasticsearchCluster
# Use a descriptive name for the node:
node.name: es-node-3
# address here to expose this node on the network:
network.host: 192.168.1.30
# By default Elasticsearch listens for HTTP traffic on the first
free port it
# finds starting at 9200. Set a specific HTTP port here:
http.port: 9200
# Transport.port
transport.port: 9300
# Pass an initial list of hosts to perform discovery when this
node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
discovery.seed_hosts: ["es-node-1", "es-node-2", "es-node-3"]
# Voting-only master-eligible node
node.roles: [ master, voting_only ]
# Disable security
xpack.security.enabled: false
```

Arrancamos los Elasticsearch, como demonios, en las máquinas para aplicar los cambios desde “/opt/elasticsearch/bin” ejecutando como usuario “liferay”:

```
./elasticsearch -d
```

Anexo 22. Configuración en clúster del Portal Liferay

Para configurar Liferay en clúster lo haremos mediante una configuración de “Cluster Link” por UDP³⁴. Para ello sólo hay que añadir las siguientes propiedades en el directorio “/opt/liferay-ce-portal/”, editando con usuario “liferay” en ambas máquinas el fichero:

```
vim portal-ext.properties
```

Y añadiendo estas líneas:

```
#cluster
cluster.link.enabled=true
cluster.link.autodetect.address=mysql:3306
```

Esas propiedades activan el link para formar el clúster y se usa como dirección de vínculo común la base de datos (mysql y puerto 3306), pero valdría otro común al que lleguen ambas máquinas, como por ejemplo www.google.es:80, pero usamos el vínculo interno por si las máquinas no tuvieran acceso a Internet.

En el caso de que tuviéramos un clúster geográfico o máquinas separadas por segmentos de red distintos con un cortafuegos en medio, habría que configurar el enlace mediante *Unicast* por TCP³⁵, no obstante, no es el caso, pero lo dejamos indicado.

Aplicaremos los cambios al reiniciar los servidores Tomcat en ambas máquinas yendo al directorio:

```
/opt/liferay-ce-portal/tomcat-9.0.82/bin/
```

Y ejecutando la parada (si procede) y el arranque:

```
./shutdown.sh
```

```
./startup.sh
```

Podemos comprobar en los logs como muestra la coordinación de ambos nodos:

```
tail -200f /opt/liferay-ce-portal/logs/liferay.2023-12-16.log
```

Veremos un mensaje así en el que se ven ambos nodos:

```
[jgroups-11,liferay-channel-control,apacheliferay01-54863][JGroupsReceiver:89] Accepted view [apacheliferay01-54863|1] (2)
[apacheliferay01-54863, apacheliferay02-26726]
```

Anexo 23. Pruebas de Alta Disponibilidad

Para hacer esta prueba basta con conectarse a la web de Liferay de administración, verificar a que nodo web Apache estamos entrando con el log de la máquina de “haproxy”:

```
tail -200f /var/log/haproxy-access.log
```

Se podrá ver algo así:

```
localhost haproxy[1055]: 192.168.1.33:57500 [16/Dec/2023:17:27:21.685]  
liferay.local apache_webservers/apache-node02 0/0/0/17/17 304
```

Como se puede apreciar estamos conectados al nodo 02 de Apache que corresponde a la máquina “apacheliferay02”

A su vez mediante la Cookie de Liferay y su JSESSIONID nos dirá a qué Tomcat nos estamos conectando que es la “jvm1” que se corresponde con la máquina “apacheliferay02”

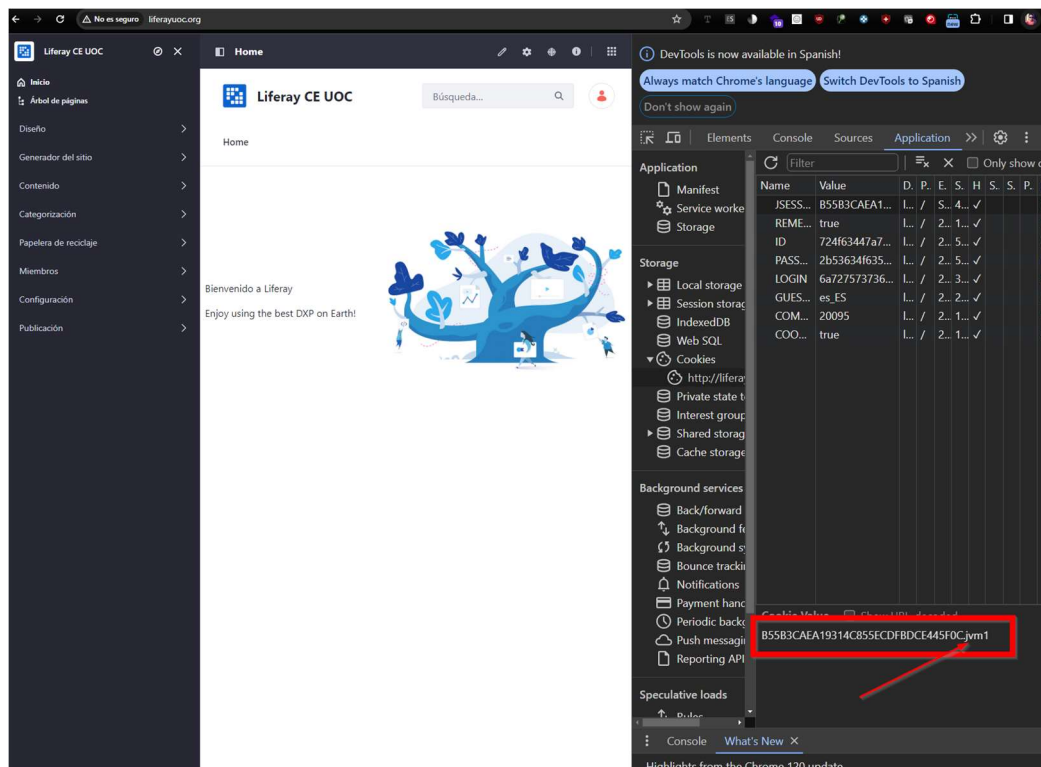


Figura 74. Comprobación de la persistencia de sesión en el navegador

Con esta información haremos dos cosas, paramos el servidor web de “apacheliferay02” y el Tomcat de la “apacheliferay01” y recargamos la web.

Comprobamos el log del HAProxy y podemos ver que ya se conecta al nodo 01 de Apache de la máquina “apacheliferay01”

```
haproxy[1055]: 192.168.1.33:57999 [16/Dec/2023:17:34:02.921]
liferay.local apache_webserver/apache-node01 0/0/1/16/18 200 45441
```

Y que el Tomcat que está sirviendo ahora es el que corresponde a la “jvm2” de la máquina “apacheliferay02” y que no se ha perdido la sesión de usuario, ya que el mismo sigue conectado al portal, es decir, se mantienen los datos de la sesión.

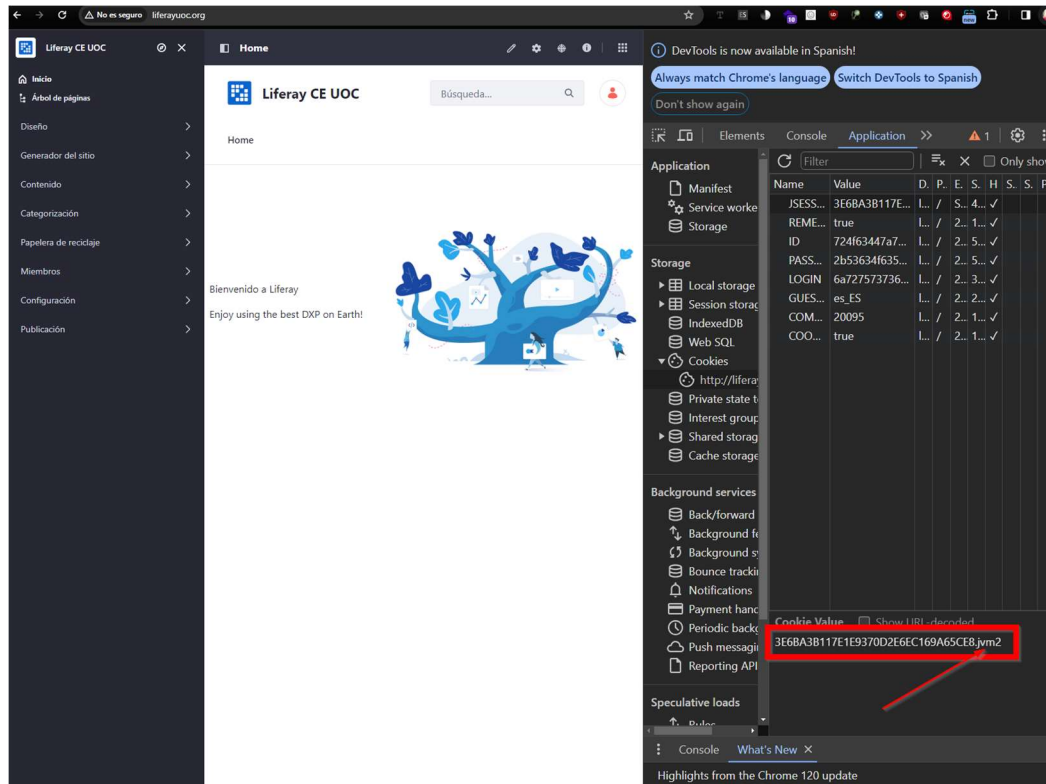


Figura 75. Comprobación de la persistencia de sesión en el navegador tras cambiar de servidor

Por otro lado, se puede comprobar el estado de los servidores de búsqueda y parando alguno viendo cómo se comporta el clúster:

```
tail -200f /opt/elasticsearch/logs/LiferayElasticsearchCluster.log
```

Veremos algo como esto si todo funciona bien:

```
[es-node-1] elected-as-master ([2] nodes joined){es-node-1}{2gf6g3LAQeGfdXkCf359WA}{waD3dfBaSSGc7oA8eFnzzg}{192.168.1.20}{192.168.1.20:9300}{cdfhilmrstw} elect leader, {es-node-3}{yyKfTyvOTpqpPYoNez4K7A}{NOqtnjHVTs6T143YaLVILw}{192.168.1.30}{192.168.1.30:9300}{mv} elect leader, _BECOME_MASTER_TASK_, _FINISH_ELECTION_], term: 30, version: 442, delta: master node changed {previous [], current [{es-node-1}{2gf6g3LAQeGfdXkCf359WA}{waD3dfBaSSGc7oA8eFnzzg}{192.168.1.20}{192.168.1.20:9300}{cdfhilmrstw}]}, added {es-node-3}{yyKfTyvOTpqpPYoNez4K7A}{NOqtnjHVTs6T143YaLVILw}{192.168.1.30}{192.168.1.30:9300}{mv}}
```

```
[o.e.c.r.a.AllocationService] [es-node-1] Cluster health status
changed from [RED] to [GREEN] (reason: [shards started [[.ds-ilm-
history-5-2023.11.16-000001][0], [.ds-.logs-deprecation.elasticsearch-
default-2023.11.16-000001][0]]]).
```

Este log que indica que el clúster está en “verde” y funcionando correctamente, aunque uno de los nodos maestros (el nodo secundario) esté caído y se ha elegido como nodo maestro al nodo principal, el “es-node-1” que está en la máquina “apacheliferay01”.

Anexo 24. Creación de un certificado SSL autofirmado

Crearemos la parte privada del certificado para la web de Liferay y su petición de firma:

```
openssl genrsa -des3 -out liferayuoc.org.key 2048
openssl req -new -key liferayuoc.org.key -out liferayuoc.org.csr
```

A continuación, se nos pedirá rellenar estos datos

```
Enter pass phrase for liferayuoc.org.key:
You are about to be asked to enter information that will be
incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or
a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:ES
State or Province Name (full name) []:Madrid
Locality Name (eg, city) [Default City]:Madrid
Organization Name (eg, company) [Default Company Ltd]:UOC
Organizational Unit Name (eg, section) []:TFG
Common Name (eg, your name or your server's hostname)
[]:liferayuoc.org
Email Address []:jrussell@uoc.edu
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Eliminamos la contraseña de la parte privada

```
cp liferayuoc.org.key liferayuoc.org.key.tmp
openssl rsa -in liferayuoc.org.key.tmp -out liferayuoc.org.key
```

Editamos el fichero externo con las propiedades que tendrá el certificado, incluyendo las SAN (dominios que hemos generado y además hemos añadido un *wildcard* para que valga cualquier web creada con nombre *uoc.org)

```
vim san.txt
```

```
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer:always
basicConstraints = CA:TRUE
keyUsage = digitalSignature, nonRepudiation,
keyEncipherment, dataEncipherment, keyAgreement, keyCertSign
```

```
subjectAltName      = DNS:liferayuoc.org,  
DNS:liferaypruebasuoc.org, DNS:raylifed2cuoc.org, DNS:*uoc.org  
issuerAltName       = issuer:copy
```

Firmamos el certificado final con una validez de 10 años

```
openssl x509 -req -in liferayuoc.org.csr -signkey liferayuoc.org.key -  
out liferayuoc.org.crt -days 3650 -sha256 -extfile san.txt
```

Una vez creados ambos archivos, que son la parte pública (liferayuoc.org.crt) y la parte privada (liferayuoc.org.key) los podemos combinar en un solo fichero (liferayuoc.org.pem) para manejarlo con facilidad.

Anexo 25. Configuración de HAProxy con HTTPS

Editaremos el fichero de configuración de HAProxy en la máquina “haproxy”

```
sudo vim /etc/haproxy/haproxy.cfg
```

y añadiremos estas líneas:

```
frontend liferay.local
    bind 192.168.1.10:80
    bind 192.168.1.10:443 ssl crt /etc/ssl/certs/liferayuoc.org.pem
    ssl-min-ver TLSv1.2
    http-request redirect scheme https unless { ssl_fc }
```

Se añadirá el bind al puerto 443 que es el HTTPS, se especificará la ruta del certificado con su parte pública y privada, se obligará a que como mínimo se acepte TLSv1.2 y finalmente se hará una redirección de HTTP a HTTPS si la llamada no se hace por HTTPS.

Guardamos, cerramos y reiniciamos el servicio de HAProxy

```
sudo systemctl restart haproxy
```

Anexo 26. Configuración de Liferay con HTTPS

Editaremos el fichero de propiedades en el directorio `"/opt/liferay-ce-portal/"` del Liferay de ambas máquinas

```
vim portal-ext.properties
```

Y agregamos estas líneas

```
web.server.protocol=https  
web.server.http.port=80  
web.server.https.port=443
```

Aplicaremos los cambios al reiniciar los servidores Tomcat en ambas máquinas yendo al directorio:

```
/opt/liferay-ce-portal/tomcat-9.0.82/bin/
```

Y ejecutando la parada (si procede) y el arranque:

```
./shutdown.sh
```

```
./startup.sh
```

Anexo 27. Configuración de Red NAT en VirtualBox

VirtualBox permite la configuración de Redes NAT de tal manera que los hosts virtualizados y nuestro host físico puedan ser configurados para acceder a ellas y haya comunicación entre dichos hosts implicados.

En nuestro caso configuraremos en “Herramientas => Administrador de Red” una nueva Red NAT 10.0.2.0/24.

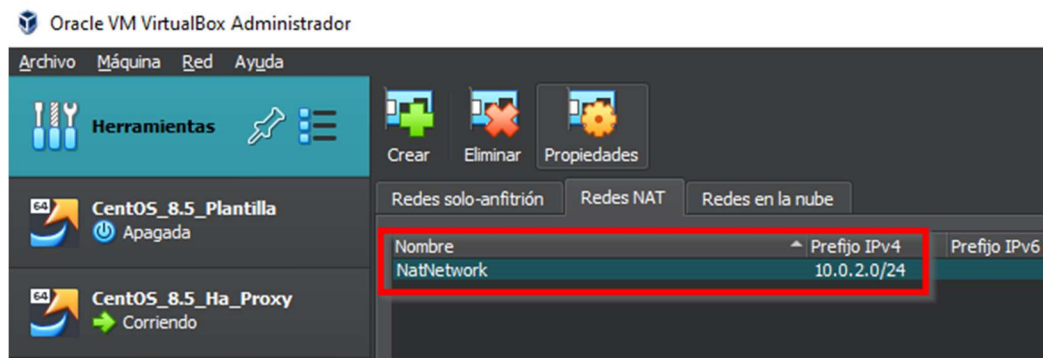


Figura 76. Red NAT VirtualBox

Añadiremos una IP 10.0.2.1/24 en esta misma red para el host anfitrión con VirtualBox

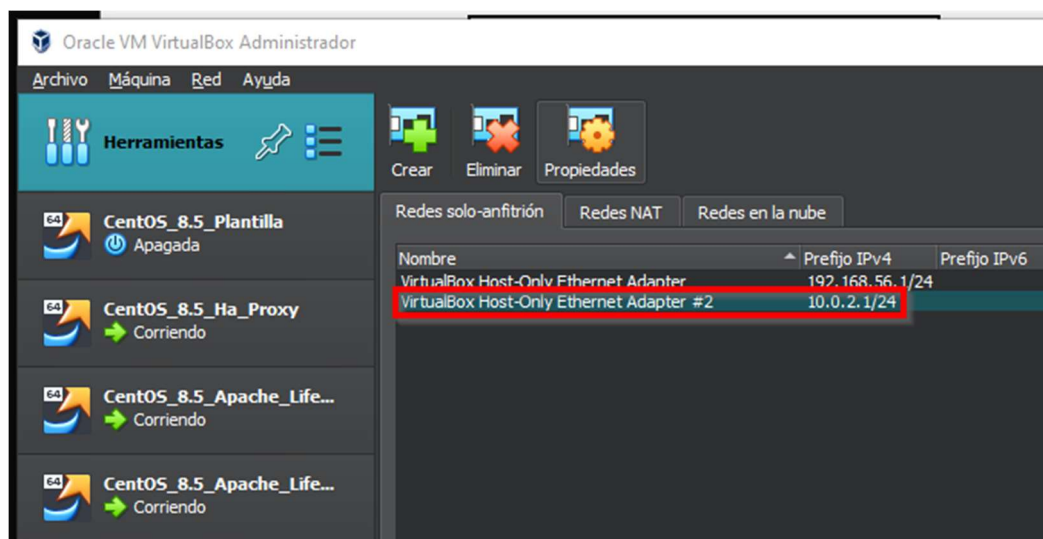


Figura 77. Nueva IP anfitrión VirtualBox

Configuraremos en el servidor “haproxy” una IP en esa red, 10.0.2.4/24 (véase [Anexo 3](#)).

Por último, añadiremos una ruta estática persistente en nuestro host físico para que se llegue a la IP que hemos configurado en “haproxy”

```
route -p add 10.0.2.0 MASK 255.255.255.0 10.0.2.4 METRIC 20
```

Anexo 28. Configuración de backups con la herramienta AutoMySQL Backup

Nos bajaremos el software de la web oficial del proyecto en <https://sourceforge.net/projects/automysqlbackup/> a un directorio que crearemos con root para el mismo.

```
sudo mkdir -p /opt/mysqlbackup
sudo tar zxvf automysqlbackup-v3.0_rc6.tar.gz -C /opt/mysqlbackup
```

Una vez creado podremos instalarlo ejecutando dentro

```
./install.sh
```

Dejamos las opciones por defecto y se nos creará un archivo de configuración que editaremos con los datos de nuestra base de datos, su usuario, el puerto, el directorio donde se guardarán los backups y los parámetros que queremos de persistencia de existencia de backups

```
vim /etc/automysqlbackup/automysqlbackup.conf
```

A continuación, indicamos los parámetros que hemos configurado:

```

# Basic Settings
# Username to access the MySQL server e.g. dbuser
CONFIG_mysql_dump_username='root'
# Password to access the MySQL server e.g. password
CONFIG_mysql_dump_password='Liferay_01'
# Host name (or IP address) of MySQL server e.g localhost
CONFIG_mysql_dump_host='localhost'
# Backup directory location e.g /backups
CONFIG_backup_dir='/var/backup/db'
# List of databases for Daily/Weekly Backup e.g. ( 'DB1' 'DB2'
'DB3' ... )
# set to (), i.e. empty, if you want to backup all databases
CONFIG_db_names=(lportal)
# Which day do you want monthly backups? (01 to 31)
# If the chosen day is greater than the last day of the month, it
will be done
# on the last day of the month.
# Set to 0 to disable monthly backups.
CONFIG_do_monthly="01"
# Which day do you want weekly backups? (1 to 7 where 1 is
Monday)
# Set to 0 to disable weekly backups.
CONFIG_do_weekly="5"
# Set rotation of daily backups. VALUE*24hours
# If you want to keep only today's backups, you could choose 1,
i.e. everything older than 24hours will be removed.
CONFIG_rotation_daily=6
# Set rotation for weekly backups. VALUE*24hours
CONFIG_rotation_weekly=35
# Set rotation for monthly backups. VALUE*24hours
CONFIG_rotation_monthly=150
# Server Connection Settings
# Set the port for the mysql connection
CONFIG_mysql_dump_port=3306
# Use ssl encryption with mysqldump?
CONFIG_mysql_dump_usessl='no'
# Choose Compression type. (gzip or bzip2)
CONFIG_mysql_dump_compression='gzip'

```

Guardamos y cerramos y si ejecutamos

```
automysqlbackup
```

Se nos generará un backup diario y un log indicando que se ha hecho el backup.

```
Backup Start Time Mon Dec 18 19:31:53 CET 2023
```

```
=====
```

```
Daily Backup ...
Daily Backup of Database ( lportal )
Rotating 6 day backups for lportal
-----
Backup End Time Mon Dec 18 19:31:56 CET 2023
=====
Total disk space used for backup storage...
Size - Location
3.1M /var/backup/db
```

Nota: puede que se generen logs de errores, pero es normal porque está la contraseña de la base de datos en el fichero en claro y es debido a esto. Para mantener la seguridad el fichero de configuración sólo debe tener permisos de lectura y escritura para el usuario "root"

Finalmente, se puede dejar programado con un crontab (para root) la ejecución diaria de esta herramienta, para ello ejecutamos

```
[root@mysql ~]# crontab -e
```

Dejando esta configuración en una línea, que indica que se ejecuta todos los días a las 2 de la mañana:

```
# Daily 2:00AM MySQL Backups
0 2 * * * /usr/local/bin/automysqlbackup
/etc/automysqlbackup/automysqlbackup.conf
```

Anexo 29. Configuración de Agente de Prometheus Node Exporter

Para configurar el agente de node_exporter⁴³ lo primero será bajar la última versión de la web que podemos ver en:

https://github.com/prometheus/node_exporter/releases/

La podemos descargar desde línea de comandos en todas las máquinas donde lo vayamos a instalar con:

```
wget
https://github.com/prometheus/node_exporter/releases/download/v1.7.0/n
ode_exporter-1.7.0.linux-amd64.tar.gz
```

Descomprimos luego el fichero bajado

```
tar -xzvf node_exporter-1.7.0.linux-amd64.tar.gz
```

Entramos en el directorio descomprimido y copiamos el ejecutable a los directorios del sistema

```
sudo cp node_exporter /usr/local/bin/
```

Añadimos el usuario sin capacidad de hacer login para que lo ejecute

```
sudo useradd -M -s /bin/false node_exporter
```

Y ahora configuramos el servicio para que se arranque con el sistema

```
sudo vim /etc/systemd/system/node_exporter.service
```

Y añadimos esto al fichero

```
[Unit]
Description=Node Exporter, Prometheus
Documentation=https://prometheus.io/docs/guides/node-exporter/
Wants=network-online.target
After=network-online.target

[Service]
User=node_exporter
Group=node_exporter
Type=simple
Restart=on-failure
ExecStart=/usr/local/bin/node_exporter

[Install]
WantedBy=multi-user.target
```

Hacemos una recarga de los demonios, activamos el servicio y lo iniciamos

```
sudo systemctl daemon-reload
sudo systemctl enable node_exporter.service
sudo systemctl start node_exporter
```

Una vez iniciado el servicio se podrá acceder vía web a las métricas con una URL como la siguiente

http://IP_MAQUINA:9100/metrics

Esa URL habrá que configurarla luego en el servidor Prometheus para acceder a las métricas

Anexo 30. Configuración de Agente de Prometheus HAProxy Exporter

Para configurar el agente de haproxy_exporter⁴⁴ lo primero será bajar la última versión de la web que podemos ver en:

https://github.com/prometheus/haproxy_exporter/releases

La podemos descargar desde línea de comandos en todas las máquinas donde lo vayamos a instalar con:

```
wget
https://github.com/prometheus/haproxy_exporter/releases/download/v0.15.0/haproxy_exporter-0.15.0.linux-amd64.tar.gz
```

Descomprimos luego el fichero bajado

```
tar -xzf haproxy_exporter-0.15.0.linux-amd64.tar.gz
```

Entramos en el directorio descomprimido y copiamos el ejecutable a los directorios del sistema

```
sudo cp haproxy_exporter /usr/local/bin/
```

Añadimos el usuario sin capacidad de hacer login para que lo ejecute

```
sudo useradd -M -s /bin/false haproxy_exporter
```

En este caso tendremos que hacer una pequeña configuración y verificación en el servicio de HAProxy

```
sudo vim /etc/haproxy/haproxy.cfg
```

Nos aseguramos de que está esta línea, la cual es la que vuelca al socket las estadísticas con permisos de lectura sobre el mismo y que para mandar órdenes sobre el servicio hacen falta permisos de administración.

```
stats socket /var/lib/haproxy/stats mode 660 level admin
```

Habrá que reiniciar el servicio de HAProxy si ha habido cambios (véase [Anexo 6](#))

Ahora configuramos el servicio del haproxy_exporter para que se arranque con el sistema:

```
sudo vim /etc/systemd/system/haproxy_exporter.service
```

Y añadimos esto al fichero especificando la ruta de donde se conseguirán las estadísticas de HAProxy y que hemos definido antes.

```
[Unit]
Description=HAProxy Exporter, Prometheus
Documentation=https://github.com/prometheus/haproxy_exporter/
Wants=network-online.target
After=network-online.target

[Service]
EnvironmentFile=/etc/prometheus/haproxy_exporter.conf
Type=simple
Restart=on-failure
User=haproxy_exporter
Group=haproxy_exporter
ExecStart=/usr/local/bin/haproxy_exporter --haproxy.scrape-uri=unix:/var/lib/haproxy/stats

[Install]
WantedBy=multi-user.target
```

Hacemos una recarga de los demonios, activamos el servicio y lo iniciamos

```
sudo systemctl daemon-reload
sudo systemctl enable haproxy_exporter.service
sudo systemctl start haproxy_exporter
```

Una vez iniciado el servicio se podrá acceder vía web a las métricas con una URL como la siguiente

http://IP_MAQUINA:9101/metrics

Esa URL habrá que configurarla luego en el servidor Prometheus para acceder a las métricas.

Anexo 31. Configuración de Agente de Prometheus Apache Exporter

Para configurar el agente de apache_exporter⁴⁵ lo primero será bajar la última versión de la web que podemos ver en:

https://api.github.com/repos/Lusitaniae/apache_exporter/releases

```
wget
https://github.com/Lusitaniae/apache_exporter/releases/download/v1.0.3
/apache_exporter-1.0.3.linux-amd64.tar.gz
```

Descomprimos luego el fichero bajado

```
tar -xzf apache_exporter-1.0.3.linux-amd64.tar.gz
```

Entramos en el directorio descomprimido y copiamos el ejecutable a los directorios del sistema

```
sudo cp apache_exporter /usr/local/bin/
```

Añadimos el usuario sin capacidad de hacer login para que lo ejecute

```
sudo useradd -M -s /bin/false apache_exporter
```

En este caso tendremos que hacer un cambio en la configuración y del servicio web de Apache para que exponga el “status” del servidor de forma local y sea excluida esa parte del balanceo que nos lleva a los servidores de Liferay. De esta manera el agente de Prometheus podrá consultar el estado del servidor web.

```
sudo vim /etc/httpd/conf/httpd.conf
```

La configuración quedaría ahora de esta manera:

```

<VirtualHost *:80>
  # Set the header for the http protocol
  RequestHeader set X-Forwarded-Proto "http"

  # Serve /server-status from the local httpd data
  ProxyPass /server-status !

  # Preserve the host when invoking tomcat
  ProxyPreserveHost on

  # Allows URL rewrite so they are more friendly
  RewriteEngine On

  # ProxyPass and ProxyPassReverse directives to redirect HTTP
  traffic to balanced members
  ProxyPass "/" "balancer://liferay/"
  ProxyPassReverse "/" "balancer://liferay/"

  # Balancer members with jvmRoute and sticky session to
  perssit sessions, connection timeout and lbmethod
  <Proxy "balancer://liferay">
    BalancerMember "http://192.168.1.20:8080" route=jvm1
    loadfactor=1 max=64 connectiontimeout=1200
    BalancerMember "http://192.168.1.21:8080" route=jvm2
    loadfactor=1 max=64 connectiontimeout=1200
    ProxySet stickysession=JSESSIONID
    ProxySet lbmethod=byrequests
  </Proxy>

  # Server-Status web
  <Location /server-status>
    SetHandler server-status
    Require ip 127.0.0.1
    Require ip ::1
  </Location>

</VirtualHost>

```

Habrá que reiniciar el servicio web Apache al haber habido cambios (véase [Anexo 5](#))

Ahora configuramos el servicio del apache_exporter para que se arranque con el sistema:

```
sudo vim /etc/systemd/system/apache_exporter.service
```

Y añadimos esto al fichero

```
[Unit]
Description=Apache exporter, Prometheus
Documentation=https://github.com/Lusitaniae/apache_exporter
Wants=network-online.target
After=network-online.target

[Service]
Type=simple
User=apache_exporter
Group=apache_exporter
ExecReload=/bin/kill -HUP $MAINPID
ExecStart=/usr/local/bin/apache_exporter

SyslogIdentifier=apache_exporter
Restart=always

[Install]
WantedBy=multi-user.target
```

Hacemos una recarga de los demonios, activamos el servicio y lo iniciamos

```
sudo systemctl daemon-reload
sudo systemctl enable apache_exporter.service
sudo systemctl start apache_exporter
```

Una vez iniciado el servicio se podrá acceder vía web a las métricas con una URL como la siguiente

http://IP_MAQUINA:9117/metrics

Esa URL habrá que configurarla luego en el servidor Prometheus para acceder a las métricas.

Anexo 32. Configuración de Agente de Prometheus JMX Exporter

Para configurar el agente de java JMX Exporter⁴⁶ lo primero será bajar la última versión de la web que podemos ver aquí además de darnos información para la instalación:

https://github.com/prometheus/jmx_exporter/

```
wget
https://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/0.19.0/jmx_prometheus_javaagent-0.19.0.jar
```

También hay que descargar o copiar el archivo de configuración específico para Tomcat el cual podemos ver directamente aquí

https://github.com/prometheus/jmx_exporter/blob/main/example_configs/tomcat.yml

Copiamos ambos ficheros en

```
/opt/jmx_exporter/
```

En este caso tendremos que hacer un cambio en la configuración del fichero de configuración de entorno de Tomcat

```
vim /opt/liferay-ce-portal/tomcat-9.0.82/bin/setenv.sh
```

Y añadimos esta línea

```
Export JAVA_OPTS="-
javaagent:/opt/jmx_exporter/jmx_prometheus_javaagent-
0.19.0.jar=9010:/opt/jmx_exporter/tomcat.yml $JAVA_OPTS"
```

Reiniciamos los Tomcat para que se aplique el cambio

Una vez iniciado el agente se podrá acceder vía web a las métricas con una URL como la siguiente

http://IP_MAQUINA:9010/metrics

Esa URL habrá que configurarla luego en el servidor Prometheus para acceder a las métricas.

Anexo 33. Configuración de Agente de Prometheus MySQL Exporter

Para configurar el agente de `mysql_exporter`⁴⁷ lo primero será bajar la última versión de la web que podemos ver aquí, además de darnos información para la instalación:

https://github.com/prometheus/mysqld_exporter

```
wget
https://github.com/prometheus/mysqld_exporter/releases/download/v0.15.1/mysqld_exporter-0.15.1.linux-amd64.tar.gz
```

Descomprimos luego el fichero bajado:

```
tar -xzf mysqld_exporter-0.15.1.linux-amd64.tar.gz
```

Entramos en el directorio descomprimido y copiamos el ejecutable a los directorios del sistema:

```
sudo cp mysql_exporter /usr/local/bin/
```

Creamos un usuario de bases de datos para que pueda leer del interior de la misma, para ello entramos como administradores al MySQL:

```
mysql -u root -p
```

Y ejecutamos esto:

```
CREATE USER 'mysqld_exporter'@'localhost' IDENTIFIED BY 'Liferay_01'
WITH MAX_USER_CONNECTIONS 2;
GRANT PROCESS, REPLICATION CLIENT, SELECT ON *.* TO
'mysqld_exporter'@'localhost';
FLUSH PRIVILEGES;
EXIT
```

Añadimos el usuario sin capacidad de hacer login para que ejecute el agente:

```
sudo useradd -M -s /bin/false mysql_exporter
```

Creamos un fichero de configuración que leerá el agente con los datos de la base de datos

```
sudo vim /etc/.mysqld_exporter.cnf
```

Con estos datos

```
[client]
user=mysqld_exporter
password=Liferay_01
```


Habr  que asegurar tambi n el fichero para que s lo lo pueda leer root y el usuario de mysql_exporter asign ndoles como propietarios y dando s lo los permisos de lectura y escritura para root y el de lectura para el mysql_exporter

```
sudo chown root:mysql_exporter /etc/.mysqld_exporter.cnf
sudo chmod 640 /etc/.mysqld_exporter.cnf
```

Ahora configuramos el servicio del `mysql_exporter` para que se arranque con el sistema:

```
sudo vim /etc/systemd/system/mysql_exporter.service
```

Y añadimos esto al fichero

```
[Unit]
Description=MySQL Exporter, Prometheus
After=network.target
User=mysql_exporter
Group=mysql_exporter

[Service]
Type=simple
Restart=always
ExecStart=/usr/local/bin/mysqld_exporter --config.my-
cnf="/etc/.mysqld_exporter.cnf" --collect.global_status --
collect.info_schema.innodb_metrics --
collect.auto_increment.columns --collect.info_schema.processlist
--collect.binlog_size --collect.info_schema.tablestats --
collect.global_variables --
collect.info_schema.query_response_time --
collect.info_schema.userstats --collect.info_schema.tables --
collect.perf_schema.tablelocks --collect.perf_schema.file_events
--collect.perf_schema.eventswaits --
collect.perf_schema.indexiowaits --
collect.perf_schema.tableiowaits --collect.slave_status --
web.listen-address=:9104

[Install]
WantedBy=multi-user.target
```

Hacemos una recarga de los demonios, activamos el servicio y lo iniciamos

```
sudo systemctl daemon-reload
sudo systemctl enable mysql_exporter.service
sudo systemctl start mysql_exporter
```

Una vez iniciado el servicio se podrá acceder vía web a las métricas con una URL como la siguiente

http://IP_MAQUINA:9104/metrics

Esa URL habrá que configurarla luego en el servidor Prometheus para acceder a las métricas.

Anexo 34. Configuración de Servidor Prometheus

La instalación del servidor de Prometheus sobre Windows se puede realizar bajando el fichero en su web:

<https://prometheus.io/download/>

En concreto usaremos la última versión estable

<https://github.com/prometheus/prometheus/releases/download/v2.48.1/prometheus-2.48.1.windows-amd64.zip>

La bajamos y la descomprimos en un directorio, luego configuramos los puntos de acceso de métricas dentro del fichero prometheus.yml en su directorio raíz dejando la configuración así para las métricas:

```
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any
  # timeseries scraped from this config.
  - job_name: "prometheus"
    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.
    static_configs:
      - targets: ["localhost:9090"]
  - job_name: 'haproxy_service'
    static_configs:
      - targets: ['haproxy:9101']
  - job_name: 'node_exporter_haproxy'
    static_configs:
      - targets: ['haproxy:9100']
  - job_name: 'node_exporter_apacheliferay01'
    static_configs:
      - targets: ['apacheliferay01:9100']
  - job_name: 'node_exporter_apacheliferay02'
    static_configs:
      - targets: ['apacheliferay02:9100']
  - job_name: 'node_exporter_mysql'
    static_configs:
      - targets: ['mysql:9100']
  - job_name: 'apache_server'
    static_configs:
      - targets: ['apacheliferay01:9117','apacheliferay02:9117']
  - job_name: 'liferay_server'
    static_configs:
      - targets: ['apacheliferay01:9010','apacheliferay02:9010']
  - job_name: 'mysql_server'
    static_configs:
      - targets: ['mysql:9104']
```

Una vez configurado el servidor procedemos a arrancarlo con una ventana de comandos de esta manera:

```
D:\prometheus-2.48.1>prometheus.exe --config.file=prometheus.yml
```

Y ya podremos entrar en el servidor desde nuestro anfitrión mediante un navegador con la URL:

<http://localhost:9090>

Anexo 35. Configuración de Servidor Grafana

La instalación del servidor de Grafana sobre Windows se puede realizar bajando el fichero en su web:

<https://grafana.com/grafana/download?platform=windows>

Usaremos la última versión estable open-source

<https://dl.grafana.com/oss/release/grafana-10.2.3.windows-amd64.zip>

La bajamos y la descomprimimos en un directorio del sistema y luego simplemente entrando en el directorio de binarios procedemos a arrancarlo con una ventana de comandos de esta manera:

```
D:\grafana-v10.2.3\bin>grafana-server.exe
```

Y ya podremos entrar en el servidor desde nuestro anfitrión mediante un navegador con la URL:

<http://localhost:3000>

Se nos pedirá usuario y contraseña que son “admin” y “admin”, respectivamente.

Desde dentro de la pantalla agregamos un nuevo “datasource” desde el menú de “connections”, buscamos “prometheus”, damos los datos del servidor (su URL local) y de esta manera Grafana ya podrá acceder a los datos que le sirve el propio Prometheus.

Para ver estos datos tendremos que importar nuevos “dashboards” para visualizar en gráficas esos datos importándolos mediante su ID o fichero json en nuestro servidor aquí:

<http://localhost:3000/dashboard/import>

Esos IDs, o ficheros json, se pueden conseguir desde la web de Grafana destinada a los mismos:

<https://grafana.com/grafana/dashboards/>