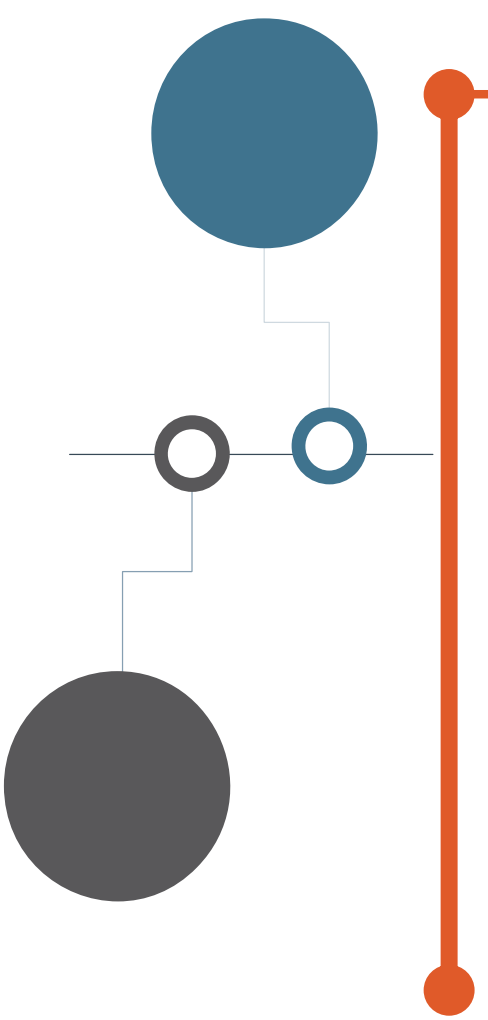
Servidores Web de Altas Prestaciones

Práctica 1: Servidores web y almacenamiento

ICAR
INGENIERÍA DE COMPUTADORES, AUTOMÁTICA Y ROBÓTICA

UNIVERSIDAD DE GRANADA

Índice

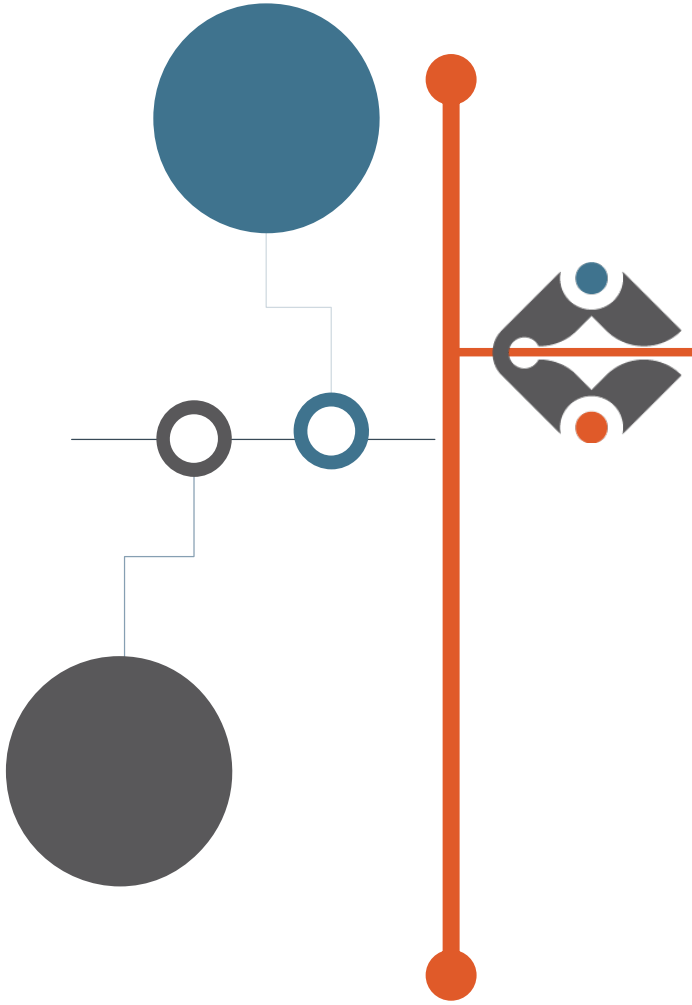


Objetivos

1. Aprender y aplicar conceptos fundamentales de Docker y contenedores.
2. Configurar servicios web utilizando Apache y PHP dentro de contenedores Docker.
3. Experimentar con la creación y gestión de redes Docker y asignar contenedores a estas redes.
4. Practicar la creación y uso de Dockerfile y docker-compose.yml para la configuración y despliegue de servicios.

La práctica se realizará de manera individual. Tiene un peso del **15%** del total de prácticas.

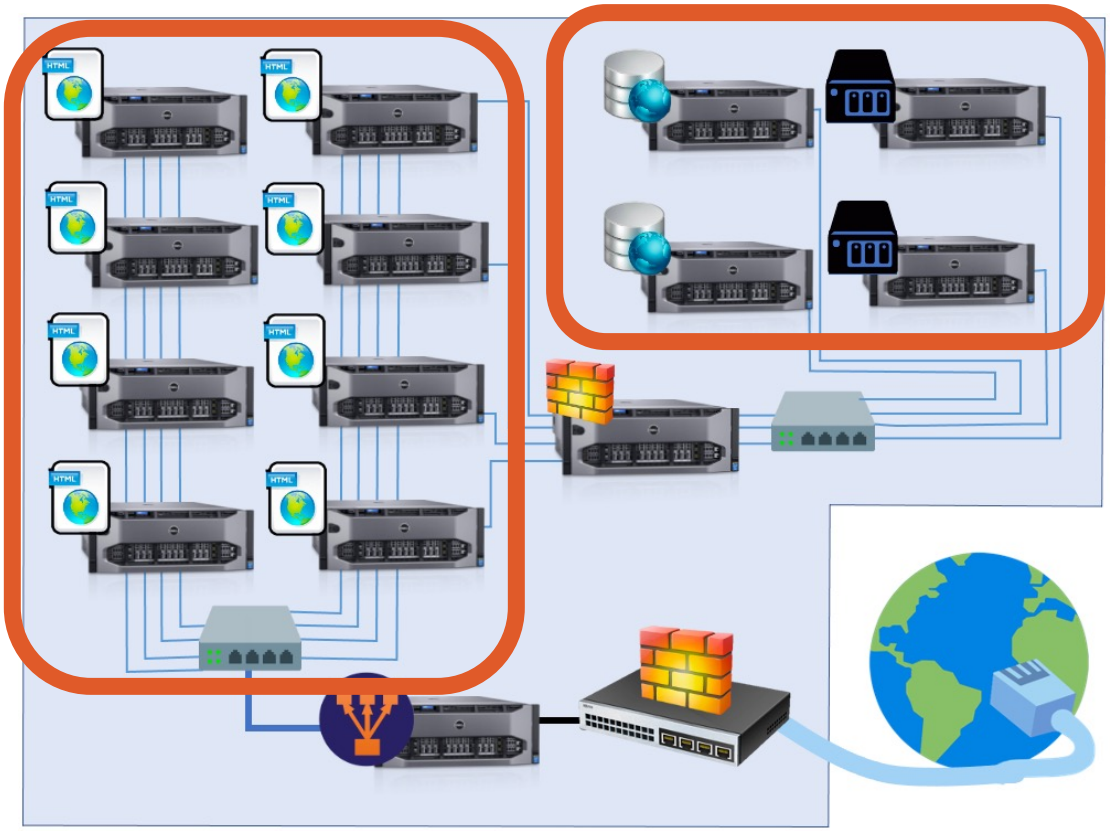
Índice



Práctica 1. Servidores web y almacenamiento

- Creación imágenes de servidor web (DockerFile)
- Creación de varias réplicas de servidores web
- Volúmenes de datos
- Gestión de redes
- Conectividad entre máquinas

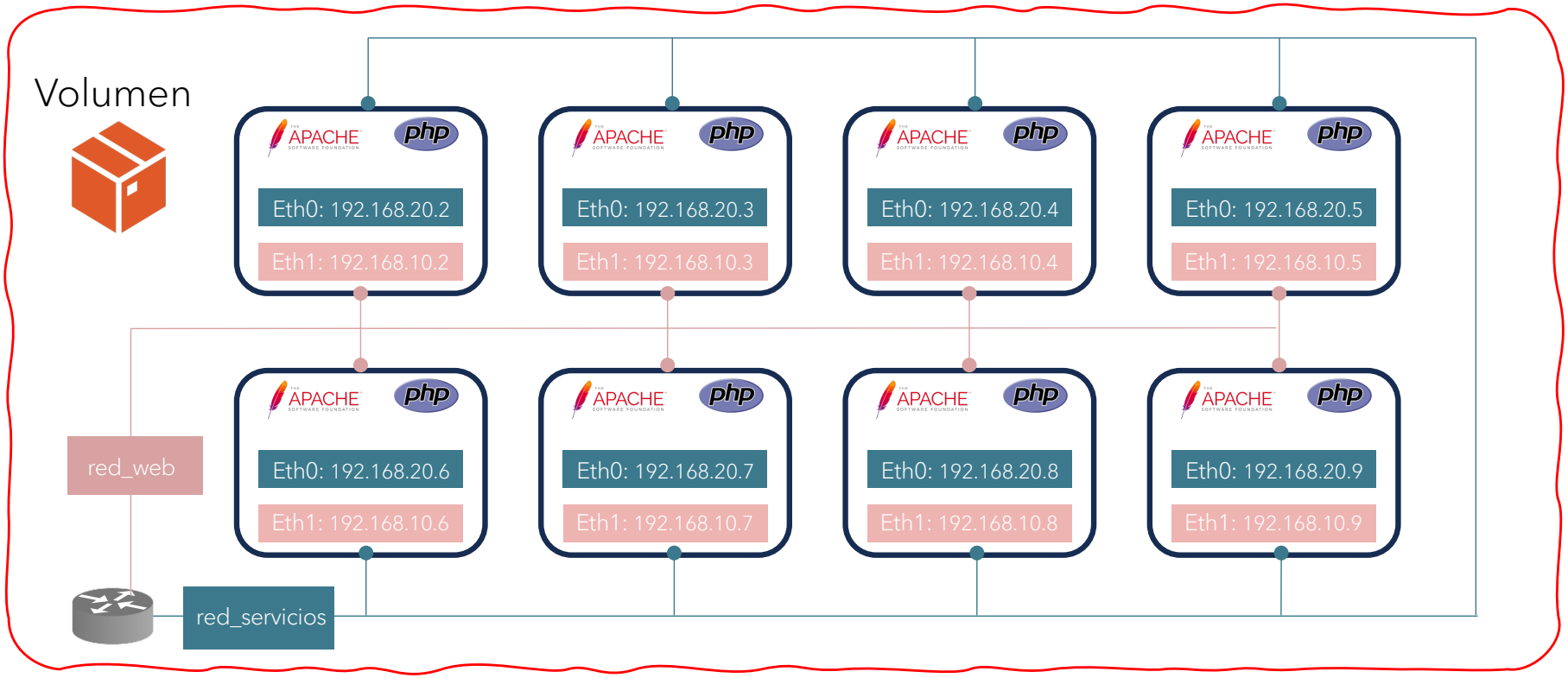
2 sesiones



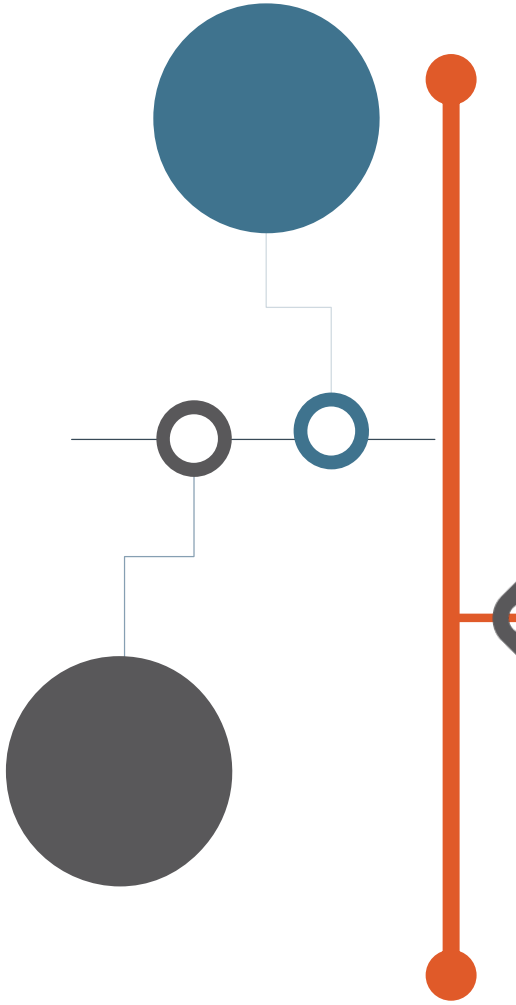
Requisitos Previos:

- Instalación de Docker y Docker Compose en el sistema.
- Conocimientos básicos de Docker, Apache, PHP, Dockerfile y Docker Compose.

Esquema general de la práctica



Índice



Desarrollo

Se pretende crear 8 réplicas de contenedores Docker que ejecuten Apache y PHP, conectados a dos redes específicas: **red_web** y **red_servicios**. Cada contenedor servirá una página web simple desde un directorio montado.

Parte 1: Creación del directorio y archivo index.php

- Crear un directorio en tu máquina local llamado **web_usuarioUGR**.
- Dentro de este directorio, crear un archivo **index.php** que muestre "SWAP - nombre del usuario" y la dirección IP del servidor Apache".

Parte 2: Creación del Dockerfile

- Crear un archivo **Dockerfile** en la raíz del proyecto llamado DockerfileApache_usuarioUGR.
- Usar una imagen base de Linux, instalar Apache, PHP y herramientas de red para comprobar conectividad entre máquinas.

Se pretende crear 8 réplicas de contenedores Docker que ejecuten Apache y PHP, conectados a dos redes específicas: **red_web** y **red_servicios**. Cada contenedor servirá una página web simple desde un directorio montado.

Parte 3: Configuración con Docker Compose

1. Crear un archivo **docker-compose.yml** con las siguientes características y siguiendo el esquema general de la práctica:
   - Crear una imagen llamada **`usuarioUGR-apache-image:p1`** a partir del Dockerfile DockerfileApache_usuarioUGR.
   - Crear 8 contenedores llamados webX, donde X es un número de 1 a 8 con volúmenes donde se monte el directorio "web_usuarioUGR" en el directorio raíz de Apache en el contenedor.
   - Añadir las dos redes al contenedor, una red llamada red_web con dirección 192.168.10.0/24 y otra red llamada red_servicios con dirección 192.168.20.0/24.

Se pretende crear 8 réplicas de contenedores Docker que ejecuten Apache y PHP, conectados a dos redes específicas: **red_web** y **red_servicios**. Cada contenedor servirá una página web simple desde un directorio montado.

Parte 4: Verificación y Pruebas

1. Verificar la Configuración de los Contenedores:
   - Lanzar los contenedores con **docker-compose up**
   - Utilizar **docker ps** para verificar que todos los contenedores estén en ejecución.
   - Comprobar que cada contenedor tiene una IP asignada en las redes **red_web** y **red_servicios**.
   - Comprobar conectividad entre los distintos contenedores.

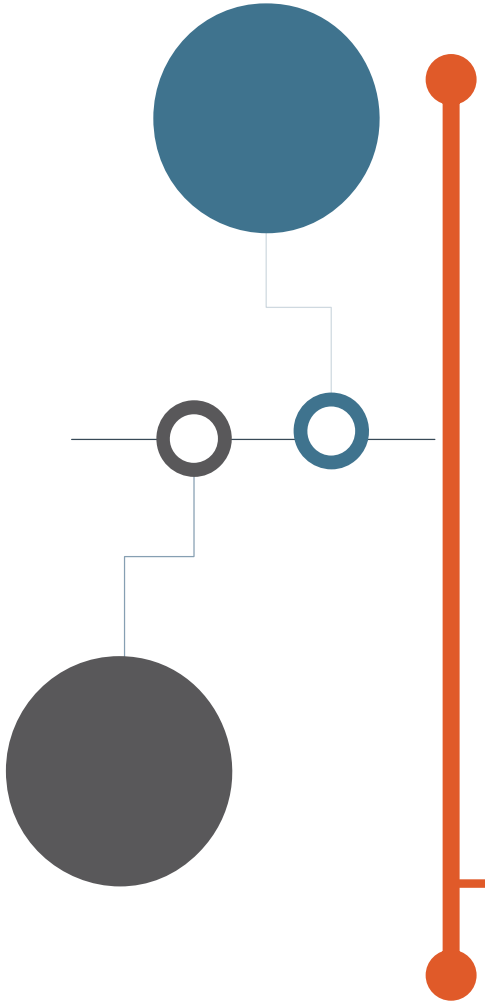
```
[sotillo@MBPM1deSoTiLLo P2 % docker compose exec web1 /bin/bash
[root@9d635d4ec410:/# ping 192.168.10.3
PING 192.168.10.3 (192.168.10.3) 56(84) bytes of data.
64 bytes from 192.168.10.3: icmp_seq=1 ttl=64 time=0.209 ms
64 bytes from 192.168.10.3: icmp_seq=2 ttl=64 time=0.159 ms
64 bytes from 192.168.10.3: icmp_seq=3 ttl=64 time=0.381 ms
^C
--- 192.168.10.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2082ms
rtt min/avg/max/mdev = 0.159/0.249/0.381/0.095 ms
root@9d635d4ec410:/#
```

Se pretende crear 8 réplicas de contenedores Docker que ejecuten Apache y PHP, conectados a dos redes específicas: **red_web** y **red_servicios**. Cada contenedor servirá una página web simple desde un directorio montado.

Parte 4: Verificación y Pruebas

2. Probar la Página Web:
    - Acceder a la página web de cada contenedor usando su dirección IP y verificar que muestra la información correcta.

Índice



Normas de entrega y evaluación

Para superar la práctica se deben realizar las siguientes tareas básicas:

B1. Configuración del Entorno:

- Instalar Docker y Docker Compose si aún no están instalados.
- Crear el directorio **web_usuarioUGR** y el archivo **index.php**.

B2. Creación del Dockerfile:

- Escribir un Dockerfile **DockerfileApache_usuarioUGR** que use una imagen base de Linux, e instale Apache, PHP y herramientas de red.

B3. Uso de Docker Compose:

- Escribir un archivo **docker-compose.yml** que defina la construcción de la imagen **usuarioUGR-apache-image:p1** y la creación de los 8 contenedores **webX**.

B4. Despliegue y verificación de Contenedores:

- Ejecutar **docker-compose up** para iniciar los contenedores.
- Usar **docker ps** para asegurarse de que todos los contenedores están en ejecución.
- Verificar que cada contenedor tiene una dirección IP asignada en las redes **red_web** y **red_servicios**.

B5. Pruebas Básicas:

- Acceder a la página web de cada contenedor usando su dirección IP y verificar que muestra la información correcta.

Normas de entrega y evaluación

Se proponen, opcionalmente, las siguientes tareas avanzadas:

A1. Personalización del Dockerfile:

- Modificar el Dockerfile para incluir configuraciones personalizadas de Apache o PHP.

A2. Creación de contendores con otros servidores web

- Crear contenedores con otros servidores web (nginx, lighttpd, etc.)

A3. Gestión Avanzada de Redes:

- Configurar reglas específicas de enrutamiento o restricciones de acceso entre las dos redes red_web y red_servicios.

A4. Automatización con Scripts:

- Crear scripts para tareas de mantenimiento automatizado, como limpieza de logs, monitoreo de la salud del contenedor, o actualizaciones automáticas de paquetes.
- Escribir scripts para automatizar la creación de contenedores o la configuración de la red.

A5. Monitoreo y Logging:

- Configurar herramientas de monitoreo y logging para rastrear el rendimiento y los eventos de los contenedores.
- Utilizar herramientas como **htop**, **netstat**, o **apache2ctl** dentro de los contenedores para monitorear y diagnosticar el estado del servidor.

Normas de entrega y evaluación

Se desarrollará un documento siguiendo el guion de la práctica y **detallando** e indicando, en su caso, los **aspectos básicos y avanzados realizados**, comandos de terminal ejecutados, resultados de ejecución, etc.

- Por ejemplo, si se ha realizado la tarea básica de configuración del entorno, el documento .pdf con la memoria de prácticas debe aparecer una sección titulada: *Tareas Básicas - B1. Configuración del Entorno* donde aparezcan detalladas las configuraciones y explicaciones sobre ellas. De igual forma, si por ejemplo, se han realizado tareas avanzadas sobre automatizaciones con Scripts, debe aparecer *Tareas Avanzadas - A4. Automatización con Scripts,* detalles de las configuraciones, explicaciones sobre ellas.

Se recomienda utilizar herramientas de control de Tiempo (por ejemplo, clockify) para contabilizar el tiempo de dedicado a la realización de la práctica.

Se deja a **libre elección** la **estructura y formato** del documento el cual reflejará el correcto desarrollo de la práctica a modo de diario/tutorial siguiendo los puntos descritos anteriormente. Asimismo, se recomienda incluir capturas de pantalla que reflejen el correcto desarrollo de los distintos apartados de la práctica. La **primera página** del documento debe incluir, al menos, **nombre**, **apellidos** y **tiempo dedicado a la práctica** medido con herramientas de control de tiempo.

Normas de entrega y evaluación

Para la entrega se habilitará una tarea en PRADO cuya entrega debe seguir OBLIGATORIAMENTE el formato especificado.

1. Un archivo **.pdf** con el documento desarrollado siguiendo el formato ApellidosNombreP1.pdf
2. Un archivo **.zip** con los distintos archivos de configuraciones, carpetas, etc. necesarios para la ejecución de la práctica siguiendo el formato ApellidosNombreP1.zip

Uso de Inteligencia Artificial Generativa

Para cada práctica es OBLIGATORIO usar herramientas de IA generativa (ChatGPT, Copilot u otras) e incluir enlace al chat/prompt utilizado. También se debe analizar y justificar el resultado que proporciona la herramienta con el resultado final que opta el estudiante para la práctica.

Es OBLIGATORIO incluir en el guion una sección titulada: "**Análisis propuesta IA**" donde se incluya enlace al chat/prompt con las consulta/as realizada/as, resultado que proporciona la IA y un párrafo con un análisis crítico y detallado del resultado proporcionado.

Normas de entrega y evaluación

La práctica se realizará de manera individual. Tiene un peso del **15%** del total de prácticas.

La práctica se evaluará mediante el uso de rúbrica específica (accesible por el estudiante en la tarea de entrega) y una defensa final de prácticas.

Cuestiones sobre la calificación obtenida en cada práctica se realizarán **UNICAMENTE** en la sesión dedicada a recuperación/defensa al final de curso.

La detección de prácticas copiadas implicará el suspenso inmediato de todos los implicados en la copia (tanto del autor del original como de quien las copió). OBLIGATORIO ACEPTAR LICENCIA EULA DE TURNITIN en la entrega. Si la memoria supera un 40% de copia Turnitin implicará el suspenso automáticamente.

Servidores Web de Altas Prestaciones

Práctica 1: Servidores web y almacenamiento

ICAR
INGENIERÍA DE COMPUTADORES, AUTOMÁTICA Y ROBÓTICA

UNIVERSIDAD DE GRANADA