

Trabajo Práctico – Virtualización con Docker

Alumnos:

- Juan Ignacio Zapata – juanignaciozapata96@gmail.com

Materia: Administración de Sistemas

Profesor: Santos Sanchez

Fecha de Entrega: 22 de Octubre del 2025

Introducción	2
Marco teórico	2
Caso Práctico: Despliegue de un Proxy Inverso	3
Configuración requerida	3
Metodología Utilizada	4
Resultados Obtenidos	5
Conclusiones	5
Bibliografía	6

Introducción

Este trabajo práctico explora la virtualización ligera a través de la tecnología de contenedores Docker, superando la limitación de la virtualización de máquinas completas como VirtualBox. El objetivo es diseñar y desplegar una solución de orquestación utilizando Docker Compose para montar un reverse proxy NGINX que distribuya el tráfico a tres sitios web estáticos internos, demostrando así la capacidad de aislar y gestionar múltiples servicios de forma eficiente en un único entorno host.

Marco teórico

- **Virtualización de Contenedores (Docker):** A diferencia de la virtualización de hardware (máquinas virtuales completas como VirtualBox), Docker utiliza el kernel del sistema operativo host. Esto resulta en un inicio casi instantáneo, menor consumo de recursos y portabilidad superior.
- **Docker Compose:** Es una herramienta para definir y ejecutar aplicaciones multi-contenedor. Permite describir la arquitectura completa de la aplicación (servicios, redes, volúmenes) en un único archivo YAML.
- **NGINX (Engine-X):** Es un popular servidor web y reverse proxy. En este trabajo, actúa como punto de entrada (el puerto 8000 del host) y dirige las peticiones a los servicios internos basándose en la ruta (path) solicitada.
- **Volúmenes:** Se utilizan para persistir datos fuera del ciclo de vida del contenedor y para montar archivos de configuración o código fuente desde el host al contenedor.

Caso Práctico: Despliegue de un Proxy Inverso

El caso práctico consiste en crear una estructura que exponga el puerto 8000 del host y sirva tres páginas web (Web Home, Web User y Web Info) desde tres contenedores diferentes o con archivos estáticos mapeados, todos accesibles a través de un único contenedor NGINX.

Configuración requerida

Ruta Solicitada (Host:8000)	Origen del Contenido	Servicio de Contenedor
/	Archivo index.html	nginx-proxy
/home	Archivo home.html	nginx-proxy
/user	Archivo user.html	nginx-proxy
/info	Archivo info.html	nginx-proxy

Metodología Utilizada

1. **Estructura de Directorios:** Se creó la siguiente estructura para el proyecto en el sistema operativo *host*:

```
.
├── docker-compose.yml
├── nginx
│   ├── default.conf
└── web_content
    ├── home.html
    ├── user.html
    └── info.html
```

2. **Archivos HTML:** Se crearon los tres archivos `.html` con contenido simple y distintivo (e.g., "Bienvenido a Home", "Información de Usuario", etc.).
3. **Configuración de NGINX (`default.conf`):** Se definió un bloque `server` que escucha en el puerto 80 y utiliza las directivas `location` para mapear las rutas (`/home`, `/user`, `/info`) a directorios específicos montados mediante volúmenes.
4. **Docker Compose:** El archivo `docker-compose.yml` se configuró para levantar un único servicio (`nginx-proxy`) y montar los volúmenes necesarios (configuración de NGINX y contenido web).
5. **Validación:** Se ejecutó `docker compose up -d` y se verificó el acceso a `http://localhost:8000/home`, `http://localhost:8000/user` y `http://localhost:8000/info` desde el navegador.

Resultados Obtenidos

El despliegue fue exitoso. El *reverse proxy* NGINX dentro del contenedor funcionó correctamente, dirigiendo las solicitudes HTTP del *host* a los archivos estáticos internos según la ruta solicitada. Se logró comprender la interacción entre el *host* y el contenedor a través del mapeo de puertos y la gestión del contenido mediante volúmenes.

Conclusiones

El uso de Docker y Docker Compose simplifica enormemente la creación de entornos de desarrollo y prueba. Esta práctica demostró que la virtualización por contenedores permite aislar y orquestar múltiples servicios (en este caso, tres "sitios" web gestionados por NGINX) con una sobrecarga mínima, confirmando que es una herramienta poderosa y accesible para la administración de sistemas modernos.

Bibliografía

- [Documentación oficial de Docker.](#)
- [Documentación oficial de Docker Compose.](#)
- [Documentación oficial de NGINX.](#)