

Examen Kubernetes

¡Perfecto! Vamos a resolver la Opción 1 paso a paso aplicando las configuraciones exactas que has estado practicando.

1. Preparar el entorno y la imagen Docker

Primero, necesitamos crear el archivo HTML y el **Dockerfile** para empaquetar tu servidor web en Alpine.

1. **Crea el archivo **index.html**** con tu nombre. Puedes usar la estructura básica que utilizaste en el Taller 8 :
2. HTML

```
<html>
<body>
<h1>Antonio Muñoz Amores</h1>
</body>
</html>
```

- 3.
4. **Crea el archivo **Dockerfile**** en el mismo directorio. Usaremos la instalación de Apache sobre Alpine que hiciste en tus prácticas :
5. Dockerfile

```
FROM alpine:latest
RUN apk update && apk add apache2 nano
RUN mkdir -p /var/www/localhost/htdocs
COPY index.html /var/www/localhost/htdocs/index.html
EXPOSE 80
CMD ["httpd", "-D", "FOREGROUND"]
```

- 6.
7. **Construye la imagen** ejecutando el siguiente comando en tu terminal:
8. PowerShell

```
docker build -t apache-examen .
```

- 9.

2. Despliegue en Kubernetes con 2 réplicas

Ahora vamos a crear los manifiestos YAML para decirle a Kubernetes que despliegue esta imagen por duplicado y la esponja.

1. **Crea el fichero de despliegue (**apache-deploy.yml**)** adaptando la estructura de tu Taller 4 para que lance 2 réplicas :
2. YAML

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: apache-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: apache
  template:
    metadata:
      labels:
        app: apache
    spec:
      containers:
        - name: apache
          image: apache-examen
          ports:
            - containerPort: 80
```

3.

4. **Lanza el despliegue:**

+1

5. PowerShell

```
kubectl apply -f apache-deploy.yml
```

6.

7. **Crea el fichero del servicio ([apache-servicio.yml](#))** para poder acceder desde fuera, usando un NodePort :

+1

8. YAML

```
apiVersion: v1
kind: Service
metadata:
  name: apache-service
spec:
  selector:
    app: apache
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  type: NodePort
```

9.

10. **Lanza el servicio:**

11. PowerShell

```
kubectl apply -f apache-servicio.yml
```

12.

3. Comprobación (Alta Disponibilidad y Balanceo)

Para demostrar en el examen que todo funciona correctamente:

- **Balanceo de carga y réplicas:** Ejecuta `kubectl get pods`. Verás que hay dos pods distintos de Apache corriendo simultáneamente. Al acceder a la IP y puerto del servicio desde tu navegador, Kubernetes irá repartiendo las peticiones (balanceando la carga) entre esos dos contenedores de forma invisible.
- **Alta Disponibilidad:** Para comprobarlo, fuerza un fallo borrando uno de los pods manualmente con `kubectl delete pod <nombre-del-pod>`. Si vuelves a hacer un `kubectl get pods` rápidamente, verás que Kubernetes detecta la caída y levanta un pod nuevo automáticamente para mantener siempre las 2 réplicas que le pediste en el `apache-deploy.yml`.

¿Te gustaría que te redacte también los comandos exactos para borrar todo el entorno (limpieza) una vez termines de enseñarle el ejercicio al profesor?

COMPROBACION DE LOS LOGS, CON LOS FALLOS QUE ME HAN DADO, LO DEL FINAL ES LO QUE ME FUNCIONA

Ah, muy buen punto! Esa es la forma perfecta de demostrarle al profesor que el tráfico está llegando y que el balanceo de carga está funcionando de verdad.

Para ver las peticiones HTTP que llegan a tus contenedores de Apache, vamos a revisar los *logs* (registros) de Kubernetes. Sigue estos pasos:

Paso 1: Averiguar los nombres de tus pods Primero, necesitas saber cómo se llaman exactamente los dos pods que tienes corriendo. Escribe en tu terminal:

```
PowerShell
kubectl get pods
```

Copia el nombre de uno de ellos (será algo como `apache-deployment-xxxxx-yyyyy`).

Paso 2: Seguir los logs en tiempo real Para ver las peticiones en directo según van llegando, utiliza el comando `logs` con la bandera `-f` (que significa *follow* o seguir). Escribe esto sustituyendo el nombre por el tuyo:

```
PowerShell
kubectl logs -f <nombre-de-tu-pod>
```

Paso 3: Realizar la petición

- Deja esa terminal abierta.
- Ve a tu navegador y entra (o recarga pulsando **F5** un par de veces) en tu página:
`http://localhost:30080`
- Vuelve a la terminal. Deberías ver aparecer una línea nueva indicando la petición, algo parecido a esto: `GET / HTTP/1.1" 200`

Truco de pro para el examen: Si quieres dejar al profesor sin palabras con el balanceo de carga, abre **dos** ventanas de PowerShell. En una ejecuta el `kubectl logs -f` del primer pod, y en la otra el `kubectl logs -f` del segundo pod. Al pulsar **F5** rápidamente en el navegador varias veces, veréis cómo las peticiones (los `GET`) van saltando de una terminal a otra, demostrando que Kubernetes está repartiendo el trabajo entre los dos contenedores. Para salir de la vista de logs, solo tienes que pulsar **Ctrl + C** en la terminal.

¿Pasamos al enunciado del siguiente ejercicio o quieres los comandos para limpiar este despliegue antes de continuar?

¡No te preocupes! Ese mensaje no es un error, es solo una advertencia súper típica de Apache.

De hecho, si recuerdas tus apuntes del Taller 5, te salió exactamente ese mismo aviso al instalar Apache en el contenedor de Alpine: `AH00558: httpd: Could not reliably determine the server's fully qualified domain name....` Te indicaba que configurararas la directiva 'ServerName' para que no saliera el mensaje.

+1

El servidor está funcionando perfectamente. Ese mensaje solo te avisa de que no le has configurado un nombre de dominio (como `www.mipagina.com`), pero para este examen en local no hace ninguna falta.

¿Cómo ver las peticiones entonces?

Opción 1: Recarga el navegador Deja esa terminal tal cual está (con el comando en ejecución). Vete a tu navegador y pulsa **F5** varias veces en `http://localhost:30080`. Vuelve a la terminal y mira si han aparecido líneas nuevas debajo de esa advertencia indicando la conexión.

Opción 2: Leer el archivo de log interno (Recomendado para Alpine) En Alpine Linux, Apache suele estar configurado por defecto para guardar los registros de acceso en un archivo interno en lugar de lanzarlos directamente a la consola (`stdout`). Si en el paso anterior no viste aparecer nada nuevo al pulsar **F5**, haz lo siguiente:

1. Ve a tu terminal y pulsa **Ctrl + C** para cancelar el comando actual.
2. Ejecuta este comando para meterte en el archivo de *logs* de acceso directamente dentro del pod:
3. PowerShell

```
kubectl exec -it apache-deployment-55c9f69db-gwc9g -- tail -f /var/log/apache2/access.log
```

- 4.
5. Vuelve al navegador, recarga la página varias veces y mira la terminal. ¡Ahí sí que te van a salir los `GET / HTTP/1.1` cada vez que entres!

Pruébalo y me dices. ¿Te han salido los registros o pasamos directamente al siguiente ejercicio del examen?

¡Claro! Tiene todo el sentido del mundo que te pase esto, ¡y de hecho es la mejor prueba de que tu entorno está funcionando bien!

Acuérdate de que tienes el **balanceo de carga activado con 2 réplicas**. Es muy probable que, al recargar la página, Kubernetes esté mandando la petición al *otro* pod para no saturar al que estás vigilando.

Vamos a cazar esas peticiones para que se lo puedas enseñar al profesor:

Paso 1: Fuerzale el tráfico

1. Deja la terminal abierta con el comando ejecutándose (`... tail -f /var/log/apache2/access.log`).
2. Vete al navegador a tu página (`http://localhost:30080`) y **pulsa F5 (o el botón de recargar) de forma muy rápida y seguida unas 10 o 15 veces**.
3. Como el tráfico se reparte a la fuerza, algunas de esas peticiones caerán seguro en el pod que estás mirando y te empezarán a salir las líneas de registro en la terminal.

Paso 2: Vigila a los dos a la vez (El truco definitivo para el examen) Si quieres que el balanceo se vea perfecto:

1. Abre **dos** ventanas de tu terminal PowerShell.
2. Haz un `kubectl get pods` para ver el nombre exacto de tus 2 pods.
3. En la primera ventana ejecuta el comando para el pod 1: `kubectl exec -it <nombre-del-pod-1> -- tail -f /var/log/apache2/access.log`
4. En la segunda ventana ejecuta el comando para el pod 2: `kubectl exec -it <nombre-del-pod-2> -- tail -f /var/log/apache2/access.log`
5. Vete al navegador y dale a recargar varias veces. Verás cómo los registros van apareciendo saltando de una ventana a la otra. ¡Con eso le demuestras el balanceo de carga al profesor de forma impecable!

Prueba a darle F5 como un loco un par de veces y dime si ya te aparecen los logs. ¡Lo tienes casi listo!