

Práctica Docker Compose

Documentar correctamente y de forma clara el proceso de creación de un sistema de monitorización con prometheus y grafana de peticiones a endpoints de un servidor nodejs.

El sistema deberá componerse de los siguientes servicios:

- (1,5 puntos) Aplicación

Este servicio se encargará de arrancar un contenedor (myapp_practica) en el que, partiendo de un Dockerfile, se pondrá en marcha un servidor express muy sencillo (el app.js y el package.json se adjuntan en la tarea). Las características del Dockerfile serán:

- Partirá de una imagen de node (versión alpine3.10)
- Establecerá un directorio de trabajo "myapp" donde residirá el código de la aplicación.
- Expondrá el puerto publicado por el servidor express.
- Ejecutará como comando la instrucción necesaria para arrancar el servidor express.

Por otra parte, este servicio asociado a la aplicación, se publicará en el puerto 83 y pertenece a una red común a todos los servicios denominada "network_practica"

- (1,5 puntos) Prometheus

Prometheus es una aplicación que nos permite recoger métricas de una aplicación en tiempo real. Como veréis en el ejemplo de app.js, se incluye una dependencia en el código (prom-client) que permite crear contadores de peticiones que podemos asociar fácilmente a nuestros endpoints de manera que podemos saber cuántas veces se ha llamado a una función de nuestra api.

En nuestro caso, el servicio de prometheus se encargará de arrancar en el puerto 9090 de nuestro host un contenedor (prometheus_practica) basado en la imagen prom/prometheus:v2.20.1. Para poder configurar correctamente este servicio, será necesario realizar además dos acciones:

- Copiar el fichero adjunto prometheus.yml al directorio /etc/prometheus del contenedor
- Ejecutar el comando --config.file=/etc/prometheus/prometheus.yml

Por supuesto, para que este servicio funcione correctamente, el servicio responsable de arrancar la aplicación debe ejecutarse antes y el servicio deberá pertenecer a la red común "network_practica".

- (2 puntos) Grafana

Este servicio será el encargado de graficar todas las métricas creadas por el servicio de Prometheus. Por tanto, siempre arrancará tras él. En nuestro caso, el servicio de grafana se encargará de arrancar en el puerto 3500 de nuestro host un contenedor (grafana_practica) basado en la imagen grafana/grafana:7.1.5 que, además, se caracterizará por:

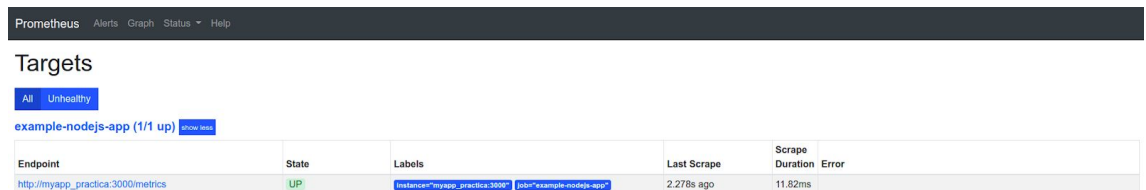
- Establecer las variables de entorno necesarias para:
 - Deshabilitar el login de acceso a Grafana
 - Permitir la autenticación anónima
 - Que el rol de autenticación anónima sea Admin

- Que instale el plugin grafana-clock-panel 1.0.1
- Pertenece a la red común “network_practica”
- Dispondrá de un volumen nombrado (myGrafanaVol) que permitirá almacenar los cambios en el servicio ya que se asociará con el directorio /var/lib/grafana

Además, para una correcta configuración de Grafana, será necesario realizar la copia del fichero adjunto datasources.yml al directorio del contenedor /etc/grafana/provisioning/datasources/.

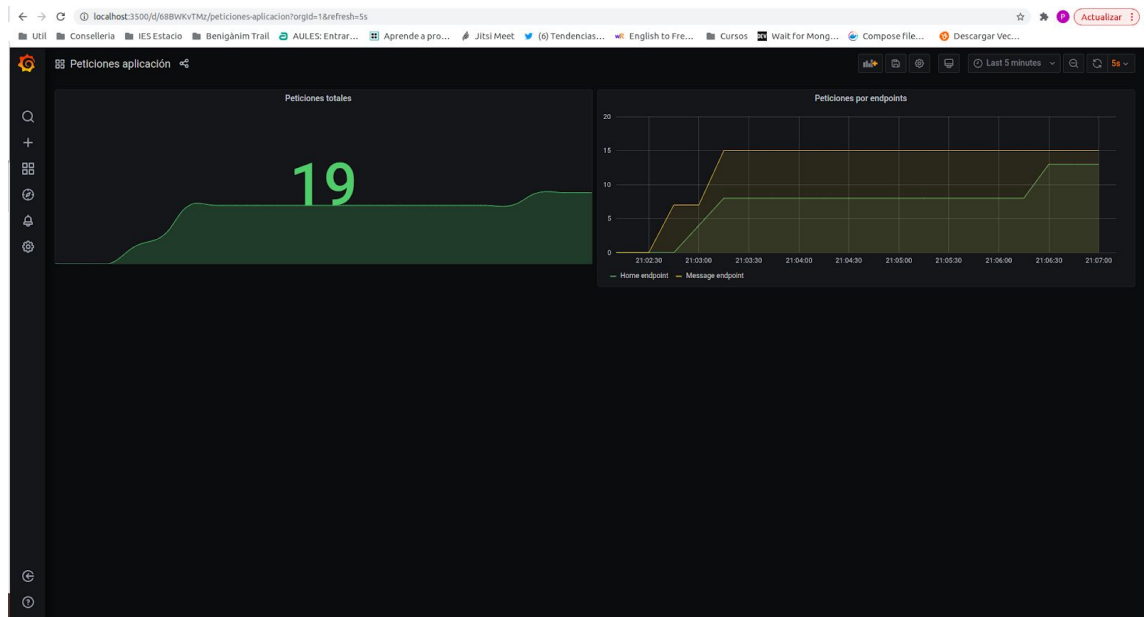
Verificar que, tras implementar todo lo necesario, lo siguiente:

- Se accede correctamente a la aplicación que se ejecuta en el contenedor a través del puerto 83
- Se accede correctamente a Prometheus en el puerto 9090 y que en el apartado Status -> Targets se muestra el acceso correcto a las métricas capturadas en la app.



Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://myapp_practica:3000/metrics	UP	instance="myapp_practica:3000" job="example-nodejs-app"	2.278s ago	11.82ms	

- (1 punto) Se accede correctamente a la aplicación de Grafana en el puerto 3500 y se puede crear un nuevo dashboard para poder incluir paneles en los que mostrar las métricas recogidas en la app. Averiguar cómo incluir un panel con las peticiones asociadas a cada endpoint y otro con un contador de peticiones totales a endpoints.



(2 punto) La entrega y documentación de la práctica se realizará utilizando un repositorio de github (Metrics_Prometheus_Grafana_Nodejs) en el que:

- Existirá una rama gh-pages para la documentación. Esta deberá incluir una introducción teórica sobre docker y docker-compose, explicar el propósito de la práctica así como cada una de las tecnologías implicadas (básicamente prometheus y grafana) y, como no, el proceso de desarrollo detallado.
- Existirá una rama master con todo el código necesario distribuido de la siguiente forma:
 - Una carpeta (src) con el código de la aplicación
 - Una carpeta grafana con el fichero de configuración de grafana datasources.yml
 - Una carpeta prometheus con el fichero de configuración de prometheus prometheus.yml
 - Fichero .dockerignore utilizado en el dockerfile
 - Fichero Dockerfile utilizado en el primer servicio
 - Fichero docker-compose.yml

(2 puntos). Intentar, analizando el código de app.js y los ficheros de configuración de prometheus y grafana, realizar los cambios necesarios para poder monitorizar las peticiones de endpoints de la aplicación de Yolanda.