

Docker Compose - Orquestación de Servicios IoT

Arquitectura de Contenedores para Plataforma IoT

Descripción General

Este archivo docker-compose.yml define un ecosistema completo de servicios para una plataforma IoT, integrando gestión de contenedores, proxy reverso, tunneling seguro, bases de datos, brokers de mensajería, y herramientas de visualización.

👉 Servicios de Infraestructura

Portainer - Gestión de Contenedores

yaml

portainer:

 image: portainer/portainer-ce:latest

 ports: ["9000:9000"]

 volumes:

 - /var/run/docker.sock:/var/run/docker.sock

 - ./portainer/data:/data

Propósito: Interfaz web para administración visual de contenedores Docker

Socket de Docker: Acceso completo a la API de Docker

Persistencia: Configuración almacenada en volumen local

Acceso: Puerto 9000 para interfaz administrativa

Nginx Proxy Manager - Reverse Proxy

yaml

nginx-proxy-manager:

 image: jc21/nginx-proxy-manager:latest

 ports: ["80:80", "443:443", "81:81"]

 environment:

 DB_SQLITE_FILE: "/data/database.sqlite"

 DISABLE_IPV6: "false"

Puertos:

80 (HTTP)

443 (HTTPS)

81 (Panel administrativo)

Base de datos: SQLite integrada para configuración

IPv6: Soporte habilitado para modern stack de red

Cloudflared - Túnel Seguro

yaml

cloudflared:

image: cloudflare/cloudflared:latest

command: tunnel --no-autoupdate run --token [TOKEN_EMBEBIDO]

Túnel embebido: Token directamente en comando (menos seguro)

Sin auto-update: Control manual de actualizaciones

Alternativa comentada: Configuración via archivo YAML

Servicios de Datos

MariaDB - Base de Datos Relacional

yaml

mariadb:

image: mariadb:latest

environment:

 MYSQL_ROOT_PASSWORD: \${MYSQL_ROOT_PASSWORD}

 MYSQL_DATABASE: \${MYSQL_DATABASE}

 MYSQL_USER: \${MYSQL_USER}

 MYSQL_PASSWORD: \${MYSQL_PASSWORD}

volumes:

 - ./services/silo/mariadb/data:/var/lib/mysql

Variables de entorno: Credenciales desde archivo .env

Persistencia: Datos de MySQL en volumen local

Usuario dedicado: Separación de privilegios

InfluxDB - Base de Datos de Series Temporales

yaml

influxdb:

image: influxdb:1.8

environment:

 - INFLUXDB_DB=\${INFLUXDB_DB}

 - INFLUXDB_ADMIN_USER=\${INFLUXDB_ADMIN_USER}

 - INFLUXDB_ADMIN_PASSWORD=\${INFLUXDB_ADMIN_PASSWORD}

volumes:

- ./services/silo/influxdb/data:/var/lib/influxdb

Versión específica: 1.8 para compatibilidad con Telegraf

Métricas IoT: Optimizado para datos de sensores

Persistencia: Datos de series temporales en volumen

 Servicios de Mensajería IoT

Mosquitto MQTT - Broker de Mensajes

yaml

mosquitto:

image: eclipse-mosquitto:2

ports:

- "1883:1883" # MQTT
- "9001:9001" # WebSocket

volumes:

- ./services/silo/mosquitto/config:/mosquitto/config
- ./services/silo/mosquitto/data:/mosquitto/data
- ./services/silo/mosquitto/log:/mosquitto/log

Protocolo MQTT: 1883 para clientes IoT tradicionales

WebSocket: 9001 para clientes web y aplicaciones modernas

Configuración persistente: Archivos de config, datos y logs

Telegraf - Agente de Métricas

yaml

telegraf:

image: telegraf:latest

volumes:

- ./services/silo/telegraf/config/telegraf.conf:/etc/telegraf/telegraf.conf:ro

depends_on:

- influxdb
- mosquitto

Configuración montada: Archivo de configuración externo

Dependencias: Requiere InfluxDB y Mosquitto funcionando

Solo lectura: Configuración inmodificable desde contenedor

Servicios de Visualización y Gestión

Node-RED - Flujos de Trabajo

yaml

```
nodered:  
  image: nodered/node-red:latest  
  ports: ["1880:1880"]  
  volumes:  
    - ./services/silo/nodered/data:/data  
  environment:  
    - NODE_RED_ENABLE_PROJECTS=true  
depends_on:  
  - mosquitto  
  - influxdb
```

Editor de flujos: Interfaz visual para orquestación IoT

Proyectos habilitados: Soporte para gestión de proyectos

Integraciones: Conectado a MQTT y base de datos

Grafana - Dashboards

yaml

```
grafana:  
  image: grafana/grafana:latest  
  volumes:  
    - ./services/silo/grafana/data:/var/lib/grafana  
  environment:  
    - GF_SECURITY_ADMIN_USER=${GRAFANA_ADMIN_USER}  
    - GF_SECURITY_ADMIN_PASSWORD=${GRAFANA_ADMIN_PASSWORD}  
    - GF_INSTALL_PLUGINS=grafana-clock-panel,grafana-simple-json-datasource  
depends_on:  
  - influxdb
```

Credenciales: Desde variables de entorno

Plugins preinstalados: Paneles adicionales para visualización

Persistencia: Configuración de dashboards en volumen

Adminer - Gestor de Bases de Datos

yaml

```
adminer:  
  image: adminer  
  depends_on:  
    - mariadb  
    - influxdb
```

Interfaz web: Administración visual de bases de datos

Multi-BD: Soporte para MariaDB e InfluxDB

Sin persistencia: Configuración temporal

Configuración de Red

Red Personalizada

yaml

```
networks:  
  silo-network:  
    driver: bridge
```

Red aislada: Comunicación segura entre contenedores

Driver bridge: Red privada dentro del host Docker

DNS automático: Resolución de nombres por nombre de servicio

Dependencias y Orden de Inicio

Estrategia de Dependencias

Infraestructura primero: Portainer y Proxy Manager

Bases de datos: MariaDB e InfluxDB antes de aplicaciones

Servicios dependientes: Telegraf, Node-RED, Grafana con depends_on

Políticas de Reinicio

yaml

restart: unless-stopped

Resiliencia: Reinicio automático ante fallos

Control manual: No reinicia si se detiene explícitamente

Producción: Comportamiento robusto para servicios críticos

Estrategia de Persistencia

Volúmenes Locales

text

```
./
└── portainer/data
└── nginx-proxy-manager/
    ├── data
    └── letsencrypt
└── services/silo/
    ├── mariadb/data
    ├── influxdb/data
    ├── mosquitto/{config,data,log}
    ├── nodered/data
    └── grafana/data
```

Datos Persistidos

Configuraciones: Nginx, Mosquitto, Telegraf

Bases de datos: MariaDB e InfluxDB

Aplicaciones: Node-RED flows, Grafana dashboards

Consideraciones de Seguridad

Exposición de Puertos

Mínimo necesario: Solo servicios que requieren acceso externo

Proxy centralizado: Nginx maneja terminación SSL

Red interna: Comunicación entre servicios en red aislada

Variables de Entorno

Credenciales: Externalizadas en archivo .env

Sensibles: Contraseñas de bases de datos y administración

Portainer socket: Acceso completo a Docker (restringir en producción)

Esta configuración docker-compose proporciona una plataforma IoT completa y production-ready, con todos los componentes necesarios para ingesta, procesamiento, almacenamiento y visualización de datos.