Reto: Visualización

Rubén Bellido Cereijo

Índice de contenido

Pasos seguidos

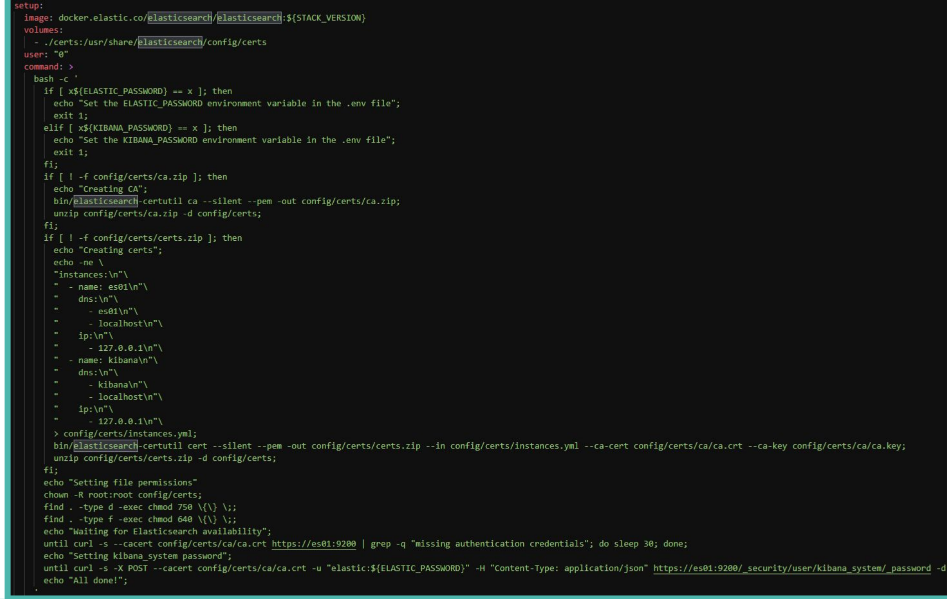
Planteamiento de la estructura del proyecto dentro de un entorno virtual:

```
.
├── certs
├── elasticsearch
│   └── data
├── kibana
│   └── data
├── generate_data.py
├── .env
└── docker-compose.yml
```

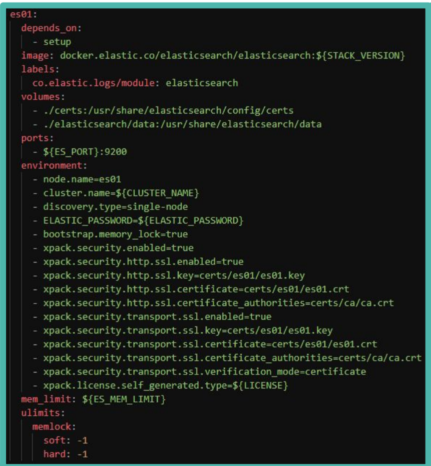
Pasos seguidos

Desarrollo del docker-compose encargado de crear los servicios:

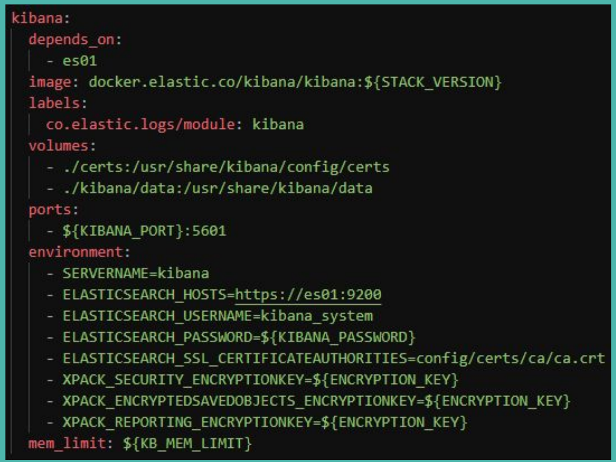
- ElasticSearch (8.12.1)
- Kibana (8.12.1)



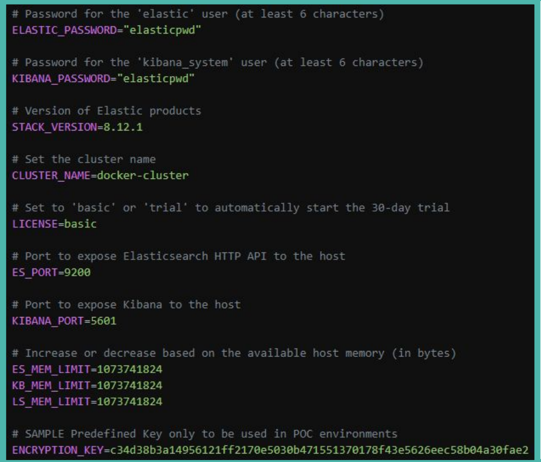
```yaml
setup:
  image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
  volumes:
    - ./certs:/usr/share/elasticsearch/config/certs
  user: "0"
  command: >
    bash -c '
      if [ x${ELASTIC_PASSWORD} == x ]; then
        echo "Set the ELASTIC_PASSWORD environment variable in the .env file";
        exit 1;
      elif [ x${KIBANA_PASSWORD} == x ]; then
        echo "Set the KIBANA_PASSWORD environment variable in the .env file";
        exit 1;
      fi;
      if [ ! -f config/certs/ca.zip ]; then
        echo "Creating CA";
        bin/elasticsearch-certutil ca --silent --pem -out config/certs/ca.zip;
        unzip config/certs/ca.zip -d config/certs;
      fi;
      if [ ! -f config/certs/certs.zip ]; then
        echo "Creating certs";
        echo -ne \
        "instances:\n"\
        "  - name: es01\n"\
        "    dns:\n"\
        "      - es01\n"\
        "      - localhost\n"\
        "    ip:\n"\
        "      - 127.0.0.1\n"\
        "  - name: kibana\n"\
        "    dns:\n"\
        "      - kibana\n"\
        "      - localhost\n"\
        "    ip:\n"\
        "      - 127.0.0.1\n"\
        > config/certs/instances.yml;
        bin/elasticsearch-certutil cert --silent --pem -out config/certs/certs.zip --in config/certs/instances.yml --ca-cert config/certs/ca/ca.crt --ca-key config/certs/ca/ca.key;
        unzip config/certs/certs.zip -d config/certs;
      fi;
      echo "Setting file permissions"
      chown -R root:root config/certs;
      find . -type d -exec chmod 750 \{\} \;;
      find . -type f -exec chmod 640 \{\} \;;
      echo "Waiting for Elasticsearch availability";
      until curl -s --cacert config/certs/ca/ca.crt https://es01:9200 | grep -q "missing authentication credentials"; do sleep 30; done;
      echo "Setting kibana_system password";
      until curl -s -X POST --cacert config/certs/ca/ca.crt -u "elastic:${ELASTIC_PASSWORD}" -H "Content-Type: application/json" https://es01:9200/_security/user/kibana_system/_password -d
      echo "All done!";
    '
```



```yaml
es01:
  depends_on:
    - setup
  image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
  labels:
    co.elastic.logs/module: elasticsearch
  volumes:
    - ./certs:/usr/share/elasticsearch/config/certs
    - ./elasticsearch/data:/usr/share/elasticsearch/data
  ports:
    - ${ES_PORT}:9200
  environment:
    - node.name=es01
    - cluster.name=${CLUSTER_NAME}
    - discovery.type=single-node
    - ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
    - bootstrap.memory_lock=true
    - xpack.security.enabled=true
    - xpack.security.http.ssl.enabled=true
    - xpack.security.http.ssl.key=certs/es01/es01.key
    - xpack.security.http.ssl.certificate=certs/es01/es01.crt
    - xpack.security.http.ssl.certificate_authorities=certs/ca/ca.crt
    - xpack.security.transport.ssl.enabled=true
    - xpack.security.transport.ssl.key=certs/es01/es01.key
    - xpack.security.transport.ssl.certificate=certs/es01/es01.crt
    - xpack.security.transport.ssl.certificate_authorities=certs/ca/ca.crt
    - xpack.security.transport.ssl.verification_mode=certificate
    - xpack.license.self_generated.type=${LICENSE}
  mem_limit: ${ES_MEM_LIMIT}
  ulimits:
    memlock:
      soft: -1
      hard: -1
```



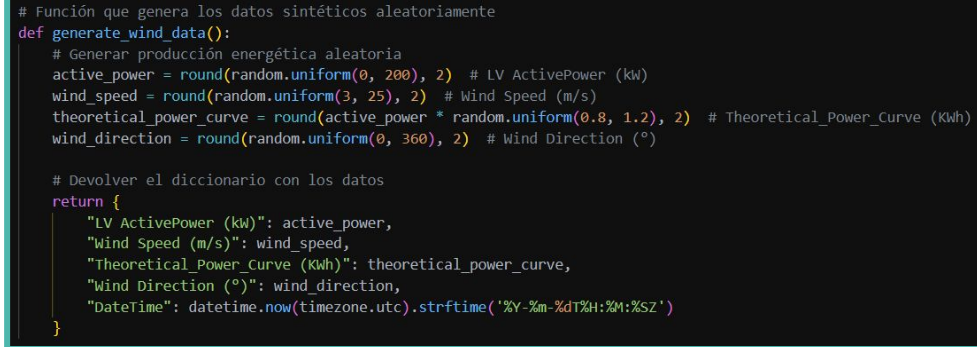
```yaml
kibana:
  depends_on:
    - es01
  image: docker.elastic.co/kibana/kibana:${STACK_VERSION}
  labels:
    co.elastic.logs/module: kibana
  volumes:
    - ./certs:/usr/share/kibana/config/certs
    - ./kibana/data:/usr/share/kibana/data
  ports:
    - ${KIBANA_PORT}:5601
  environment:
    - SERVERNAME=kibana
    - ELASTICSEARCH_HOSTS=https://es01:9200
    - ELASTICSEARCH_USERNAME=kibana_system
    - ELASTICSEARCH_PASSWORD=${KIBANA_PASSWORD}
    - ELASTICSEARCH_SSL_CERTIFICATEAUTHORITIES=config/certs/ca/ca.crt
    - XPACK_SECURITY_ENCRYPTIONKEY=${ENCRYPTION_KEY}
    - XPACK_ENCRYPTEDSAVEDOBJECTS_ENCRYPTIONKEY=${ENCRYPTION_KEY}
    - XPACK_REPORTING_ENCRYPTIONKEY=${ENCRYPTION_KEY}
  mem_limit: ${KB_MEM_LIMIT}
```



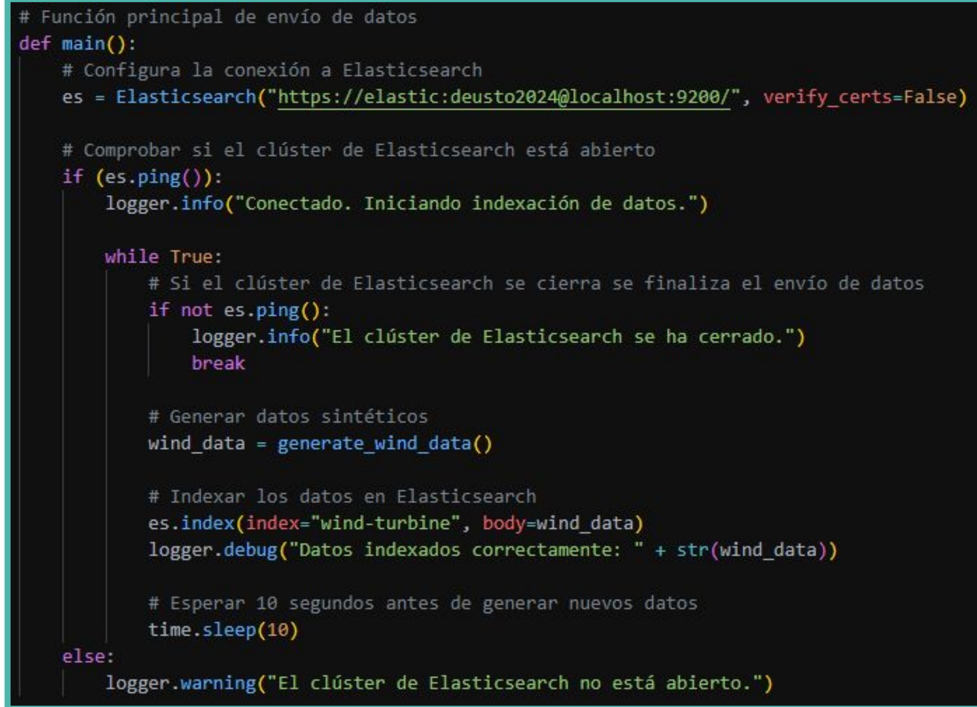
```
# Password for the 'elastic' user (at least 6 characters)
ELASTIC_PASSWORD="elasticpwd"

# Password for the 'kibana_system' user (at least 6 characters)
KIBANA_PASSWORD="elasticpwd"

# Version of Elastic products
STACK_VERSION=8.12.1

# Set the cluster name
CLUSTER_NAME=docker-cluster

# Set to 'basic' or 'trial' to automatically start the 30-day trial
LICENSE=basic

# Port to expose Elasticsearch HTTP API to the host
ES_PORT=9200

# Port to expose Kibana to the host
KIBANA_PORT=5601

# Increase or decrease based on the available host memory (in bytes)
ES_MEM_LIMIT=1073741824
KB_MEM_LIMIT=1073741824
LS_MEM_LIMIT=1073741824

# SAMPLE Predefined Key only to be used in POC environments
ENCRYPTION_KEY=c34d38b3a14956121ff2170e5030b471551370178f43e5626eec58b04a30fae2
```

Pasos seguidos

Elección de datos y desarrollo del script de indexación de datos.



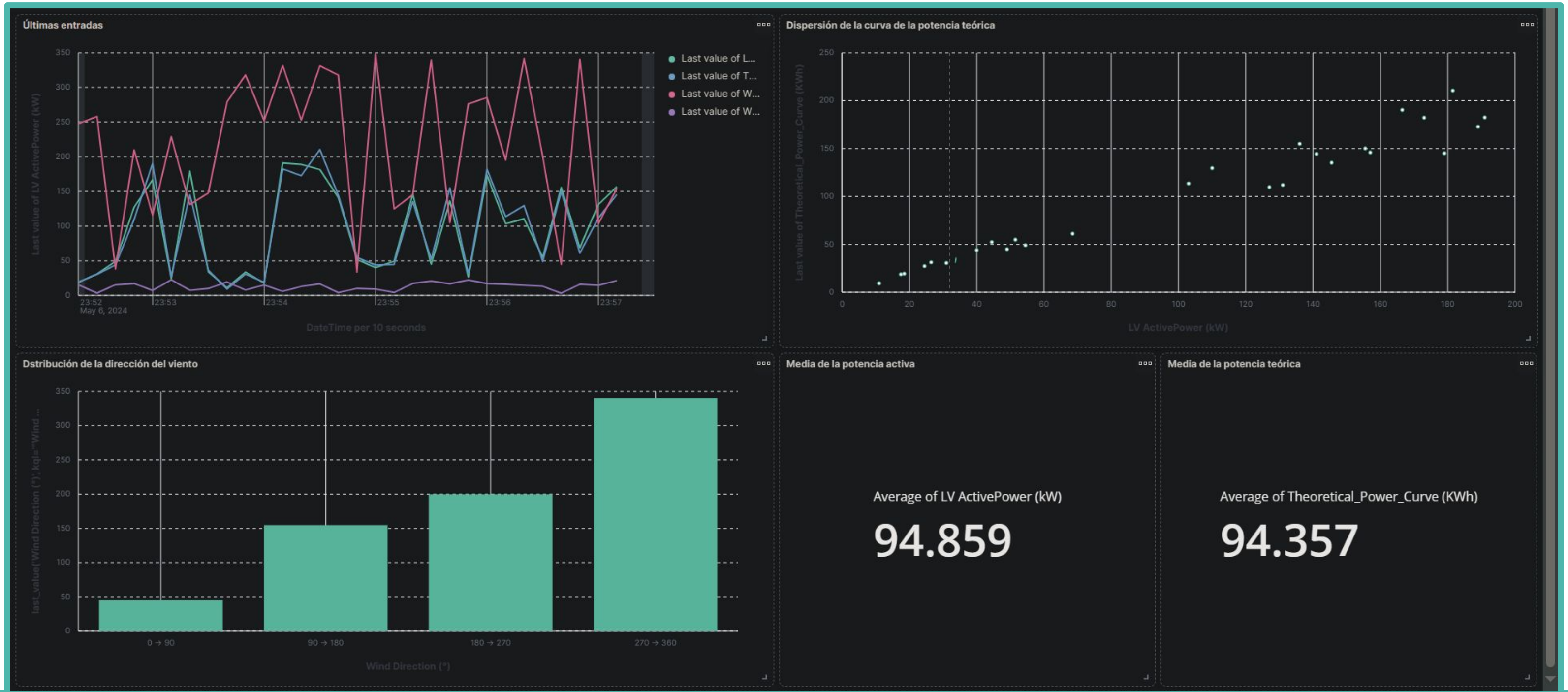
```python
# Función que genera los datos sintéticos aleatoriamente
def generate_wind_data():
    # Generar producción energética aleatoria
    active_power = round(random.uniform(0, 200), 2)  # LV ActivePower (kW)
    wind_speed = round(random.uniform(3, 25), 2)  # Wind Speed (m/s)
    theoretical_power_curve = round(active_power * random.uniform(0.8, 1.2), 2)  # Theoretical_Power_Curve (KWh)
    wind_direction = round(random.uniform(0, 360), 2)  # Wind Direction (°)

    # Devolver el diccionario con los datos
    return {
        "LV ActivePower (kW)": active_power,
        "Wind Speed (m/s)": wind_speed,
        "Theoretical_Power_Curve (KWh)": theoretical_power_curve,
        "Wind Direction (°)": wind_direction,
        "DateTime": datetime.now(timezone.utc).strftime('%Y-%m-%dT%H:%M:%SZ')
    }
```



```python
# Función principal de envío de datos
def main():
    # Configura la conexión a Elasticsearch
    es = Elasticsearch("https://elastic:deusto2024@localhost:9200/", verify_certs=False)

    # Comprobar si el clúster de Elasticsearch está abierto
    if (es.ping()):
        logger.info("Conectado. Iniciando indexación de datos.")

        while True:
            # Si el clúster de Elasticsearch se cierra se finaliza el envío de datos
            if not es.ping():
                logger.info("El clúster de Elasticsearch se ha cerrado.")
                break

            # Generar datos sintéticos
            wind_data = generate_wind_data()

            # Indexar los datos en Elasticsearch
            es.index(index="wind-turbine", body=wind_data)
            logger.debug("Datos indexados correctamente: " + str(wind_data))

            # Esperar 10 segundos antes de generar nuevos datos
            time.sleep(10)
    else:
        logger.warning("El clúster de Elasticsearch no está abierto.")
```

Pasos seguidos

Investigación de uso de Kibana y creación del dashboard con gráficos.



Últimas entradas
Last value of L...
Last value of T...
Last value of W...
Last value of W...
Last value of LV ActivePower (kW)
DateTime per 10 seconds
23:52 May 6, 2024
23:53
23:54
23:55
23:56
23:57
Dispersión de la curva de la potencia teórica
Last value of Theoretical_Power_Curve (KWh)
LV ActivePower (kW)
Dstribución de la dirección del viento
last_value(Wind Direction (°), kql='"Wind ...
0 → 90
90 → 180
180 → 270
270 → 360
Wind Direction (°)
Media de la potencia activa
Media de la potencia teórica

Average of LV ActivePower (kW)

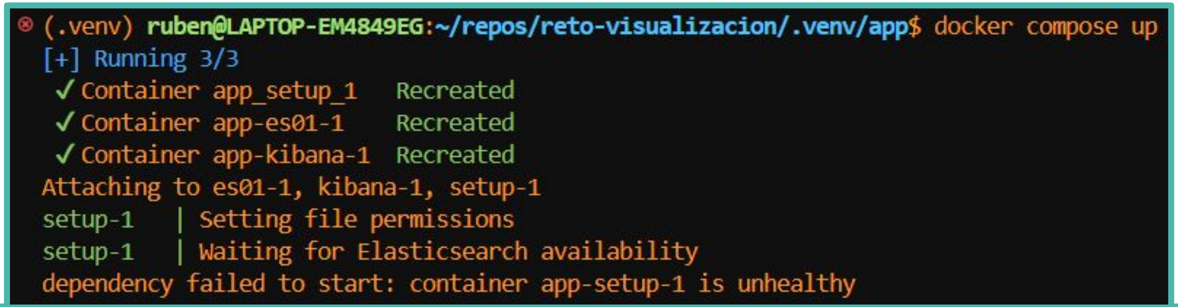
94.859

Average of Theoretical_Power_Curve (KWh)

94.357

Vías de mejora, problemas, retos

- Kibana no vinculandose a ElasticSearch
  - Al deshabilitar la seguridad: xpack.security.enabled=false
  - Los ElasticSearch hosts debían de referenciarse mediante HTTP
- Establecer la seguridad/certificados
  - Los servicios no pasaban el healthcheck
  - Ni ElasticSearch ni Kibana se levantaban correctamente
- Enviar datos de manera segura
  - El programa no encuentra/accede a los certificados



```
⊗ (.venv) ruben@LAPTOP-EM4849EG:~/repos/reto-visualizacion/.venv/app$ docker compose up
 [+] Running 3/3
  ✓Container app_setup_1   Recreated
  ✓Container app-es01-1    Recreated
  ✓Container app-kibana-1  Recreated
 Attaching to es01-1, kibana-1, setup-1
 setup-1    | Setting file permissions
 setup-1    | Waiting for Elasticsearch availability
 dependency failed to start: container app-setup-1 is unhealthy
```

Alternativas posibles

- Servicios de visualización
  - Grafana
  - Prometheus
  - …

DEMO

Bibliografía

https://realpython.com/python-virtual-environments-a-primer/

https://www.elastic.co/es/blog/getting-started-with-the-elastic-stack-and-docker-compose

https://www.elastic.co/guide/en/kibana/current/docker.html#_remove_docker_containers

https://www.elastic.co/es/blog/getting-started-with-the-elastic-stack-and-docker-compose