LAB 10

Estación Metereológica

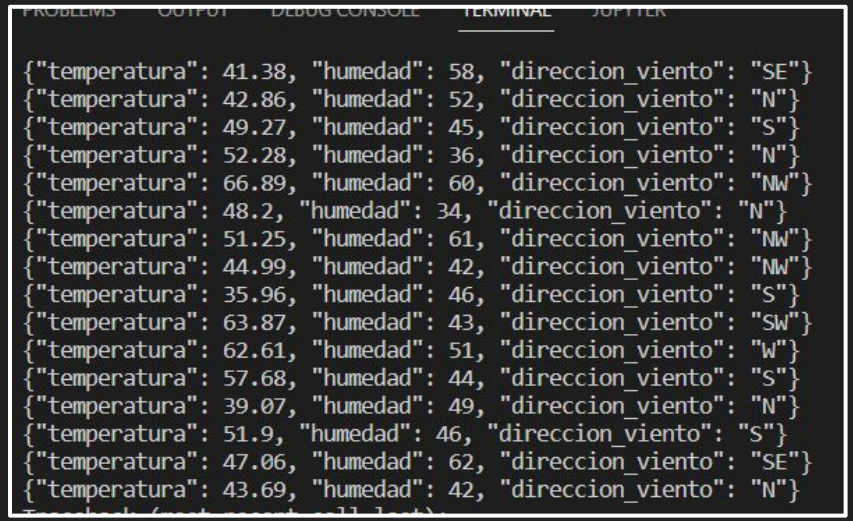
https://github.com/Olivverde/Lab-10-Redes.git

Oliver de León
Andres Quinto

Simulación de Sensor

```
s/sensors.py
{"temperatura": 47.83, "humedad": 20, "direccion_viento": "W"}
PS C:\Users\olive\Desktop\UVG\Lab-10-redes> & C:/Users/olive/App
s/sensors.py
{"temperatura": 72.06, "humedad": 46, "direccion_viento": "E"}
PS C:\Users\olive\Desktop\UVG\Lab-10-redes> & C:/Users/olive/App
s/sensors.py
s/sensors.py
{"temperatura": 57.18, "humedad": 51, "direccion_viento": "SE"}
PS C:\Users\olive\Desktop\UVG\Lab-10-redes> & C:/Users/olive/App
s/sensors.py
{"temperatura": 60.32, "humedad": 54, "direccion_viento": "NW"}
PS C:\Users\olive\Desktop\UVG\Lab-10-redes>
```

Fases tempranas de la simulación para corroboración de funcionamiento



Simulación de sensores cada 3 segs lista para implementación de 15 segs

➢ *Responda: ¿A qué capa pertenece JSON/SOAP según el Modelo OSI y porque?*

La condensación de datos JSON/SOAP pertenecen a la capa 6 del modelo OSI, la capa de "**Presentación**".

JSON/SOAP codifica diferentes tipos de datos y los representa en un **formato de codificación/serialización canonizado**. Al analizar la capa de presentación, encontraremos que se encarga de la "**presentación de la información**", por lo que JSON/SOAP fácilmente forma parte de esta.

➢ *Responda: ¿Qué beneficios tiene utilizar un formato como JSON/SOAP?*

Los beneficios que JSON/SOAP aportan pueden resumirse a continuación:

- Comprende un formato **compacto y eficiente**
- Es **fácil de leer**
- Es **ampliamente utilizado** en la comunidad de desarrollo de software
- Es **descriptivo** (presenta facilidad a la hora de distinguir tipos de datos e interpretar los mismos)
- Es un formato **flexible**

Envío de Datos al Edge Server

```
PS C:\Users\olive\Desktop\UVG\Lab-10-redes> & C:/Users/
s/IoT.py
<class 'str'>
<class 'kafka.producer.future.FutureRecordMetadata'>
<class 'str'>
<class 'kafka.producer.future.FutureRecordMetadata'>
Traceback (most recent call last):
```

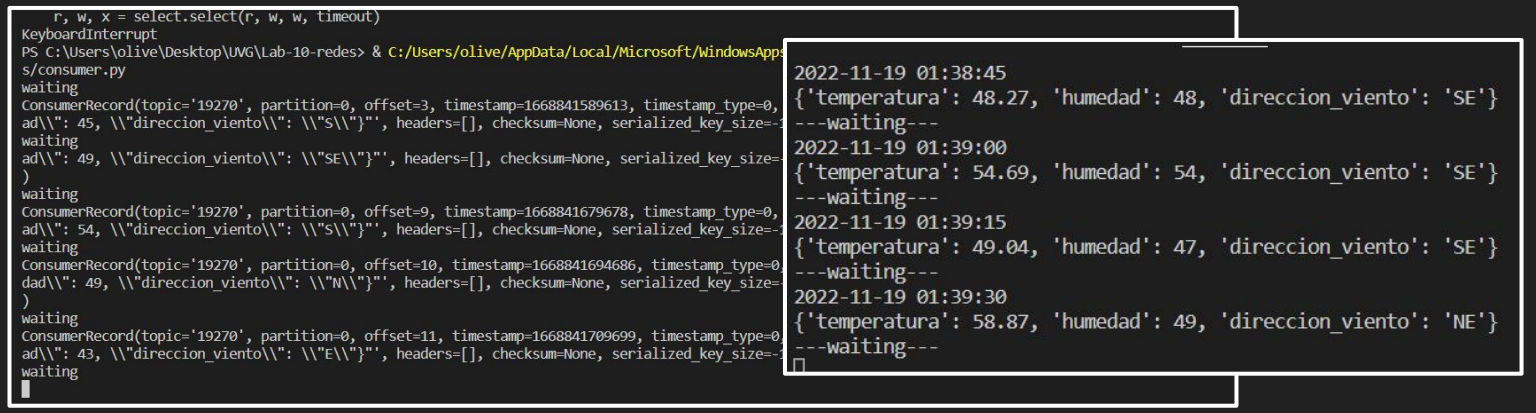
```python
def IoT_producer(self,mu,sigma,time):
    topic = '19270'
    producer = KafkaProducer(
        bootstrap_servers='lab10.alumchat.fun:9092',
        value_serializer=lambda v: json.dumps(v).encode('utf-8')
        )
    on = True
    while on:
        data = S(mu, sigma)
        mssg = data.IoT_params
        certif = producer.send(topic, mssg)
        print(type(certif))
        t.sleep(time)
```

Se realiza la conexión con el server y se procede a enviar mensajes en el formato JSON. Según la documentación de kafka, al imprimirse el la función producer.send, el retorno deberá de ser <FutureRecordMetadata> si la emisión fue exitosa

Consumir Datos

```
C:\Users\olive\Desktop\UVG\Lab-10-redes>python producer.py
```

Ejecución del archivo producer, donde se emiten la información de los sensores



Ejecución del archivo consumer, que actualmente escucha al emisor. (Captura de transmisión de 75 segundos)

- *Responda: ¿Qué ventajas y desventajas considera que tiene este acercamiento basado en Pub/Sub de Kafka?*
- *Responda: ¿Para qué aplicaciones tiene sentido usar Kafka? ¿Para cuáles no?*

+ Pub/Sub Kafka es más flexible y escalable por la metodología de suscripción. Es asincrónico, lo que se traduce en menos problemas al intercambiar datos.
- Solo podemos realizar solicitudes una por una, complicando los test. Muchos mensajes puede congestionar el broker

Se puede utilizar para determinar el tráfico de una red o web, además de procesar datos en tiempo real. No se debería usar en lugar de una base de datos, cuando solo se van a manejar muy poquitos datos, o cuando se necesita una simple cola de tareas, existen mejores herramientas para esto.

Transferencia de datos sin codificación

```
<BrokerConnection node_id=bootstrap-0 host=lab10.alumchat.fun:9092 <connected> [IPv4
ing connection.

Sending log: {'temperatura': 53.81, 'humedad': 56, 'direccion_viento': 'W'}

---Delay---

Sending log: {'temperatura': 55.99, 'humedad': 63, 'direccion_viento': 'N'}

---Delay---

Sending log: {'temperatura': 47.65, 'humedad': 37, 'direccion_viento': 'W'}

---Delay---

Sending log: {'temperatura': 57.25, 'humedad': 58, 'direccion_viento': 'E'}

---Delay---
```

```
mplete.

<BrokerConnection node_id=bootstrap-0 hos
ing connection.

--New message--

LOG: {'t': 53, 'h': 56, 'w': 'W'}

--New message--

LOG: {'t': 55, 'h': 63, 'w': 'N'}

--New message--

LOG: {'t': 47, 'h': 37, 'w': 'W'}

--New message--

LOG: {'t': 57, 'h': 58, 'w': 'E'}
```

Producer (izq) y Consumer (der)

Transferencia de datos con la codificación

```
<BrokerConnection node_id=bootstrap-0 host=lab10.alumchat.fun:9092 <connected> [IPv4 ('147.182.206.35', 9092)]>: Clos
ing connection.

Sending log: {'temperatura': 36.78, 'humedad': 46, 'direccion_viento': 'S'}

Sending encoded: b'$.h'

---Delay---

Sending log: {'temperatura': 50.6, 'humedad': 50, 'direccion_viento': 'E'}

Sending encoded: b'22j'

---Delay---

Sending log: {'temperatura': 49.15, 'humedad': 58, 'direccion_viento': 'SW'}

Sending encoded: b'1:g'

---Delay---

Sending log: {'temperatura': 50.38, 'humedad': 51, 'direccion_viento': 'NW'}

Sending encoded: b'23e'

---Delay---

Sending log: {'temperatura': 63.3, 'humedad': 61, 'direccion_viento': 'NW'}

Sending encoded: b'?=e'

---Delay---

Sending log: {'temperatura': 45.68, 'humedad': 52, 'direccion_viento': 'N'}

Sending encoded: b'-4d'

---Delay---
```

```
<BrokerConnection node_id=bootstrap-0 host=lab10.alumchat.fun:9092 <connected> [IPv4 ('147.182.206.35', 9092)]>: Clos
ing connection.

--New message--

Received message encoded: b'$.h'

LOG: {'t': 36, 'h': 46, 'w': 'S'}

--New message--

Received message encoded: b'22j'

LOG: {'t': 50, 'h': 50, 'w': 'E'}

--New message--

Received message encoded: b'1:g'

LOG: {'t': 49, 'h': 58, 'w': 'SW'}

--New message--

Received message encoded: b'23e'

LOG: {'t': 50, 'h': 51, 'w': 'NW'}

--New message--

Received message encoded: b'?=e'

LOG: {'t': 63, 'h': 61, 'w': 'NW'}

--New message--

Received message encoded: b'-4d'

LOG: {'t': 45, 'h': 52, 'w': 'N'}
```

Producer (izq) y Consumer (der)

- *Responda: ¿Qué complejidades introduce el tener un payload restringido (pequeño)?*
- *Responda: ¿Cómo podemos hacer que el valor de temperatura quepa en 14 bits?*

- La información que se envía debe ser reducida, añadiendo complejidad a la transmisión de la información.
- Multiplicamos por 100 y enviamos cómo binario.

- *Responda: ¿Qué sucedería si ahora la humedad también es tipo float con un decimal? ¿Qué decisiones tendríamos que tomar en ese caso?*
- *Responda: ¿Qué parámetros o herramientas de Kafka podrían ayudarnos si las restricciones fueran aún más fuertes?*

- En ese caso ocuparía más para nuestro payload, así que debemos aproximarnos.
- gzip, es un módulo integrado en kafka, nos permite comprimir y descomprimir, permitiéndonos enviar mas informacion ocupando menos.

Contenido graficado