MQTT KAFKA BRIDGE

ANDREAS MOSSLJUNG

CADEC 2020.01.23 & 2020.01.29 | CALLISTAENTERPRISE.SE

CALLISTA

AGENDA

- A real-world example of a Callista project at Volvo Cars
- What we will try to solve
- Introduction to MQTT and Kafka
- Building the bridge
- Demo

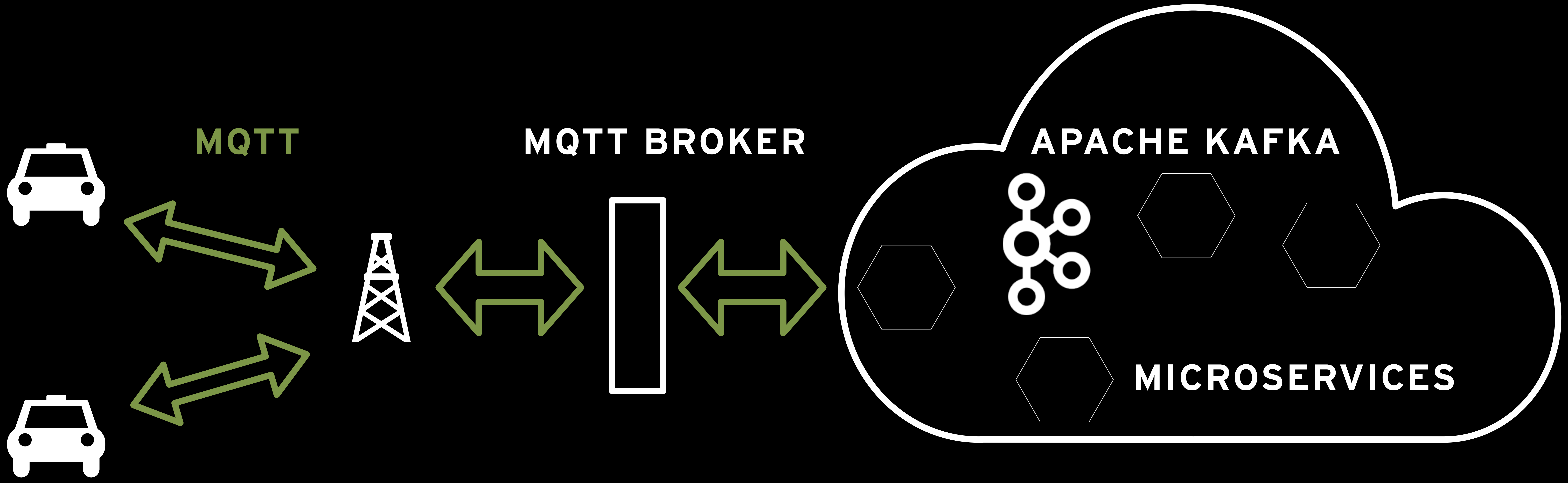
CALLISTA

CONNECTED CARS

SLIPPERY ROAD ALERT
Part of Connected Safety
THE CLOUD

CALLISTA

OVERVIEW



- Near realtime system
- All messages pass through cloud, never directly client to client via broker
- High message rate
- Need multiple clustered brokers

RAPID GROWTH EXPECTED

- > 2 000 messages per second today
- > 700 000 Volvos sold last year
- Larger part of these connected
- New services in vehicle require connection

CALLISTA

CLUSTERED MQTT BROKERS

- Clustered MQTT brokers exist
- Features of MQTT not in use by us needs state in the broker, makes existing solutions inefficient
- Cloud platforms like AWS IoT, Azure IoT Hub and Google Cloud IoT Core has some support
- Do not want our micro services to speak MQTT
- And we already have a clustered platform in Kafka and a scalable way of deploying micro services in Kubernetes 🤔

HIVEMQ

verneMQ

APACHE ACTIVEMQ™

IDEA FROM BOOTCAMP

- Bi annual activity

NIKLAS ANTONCIC

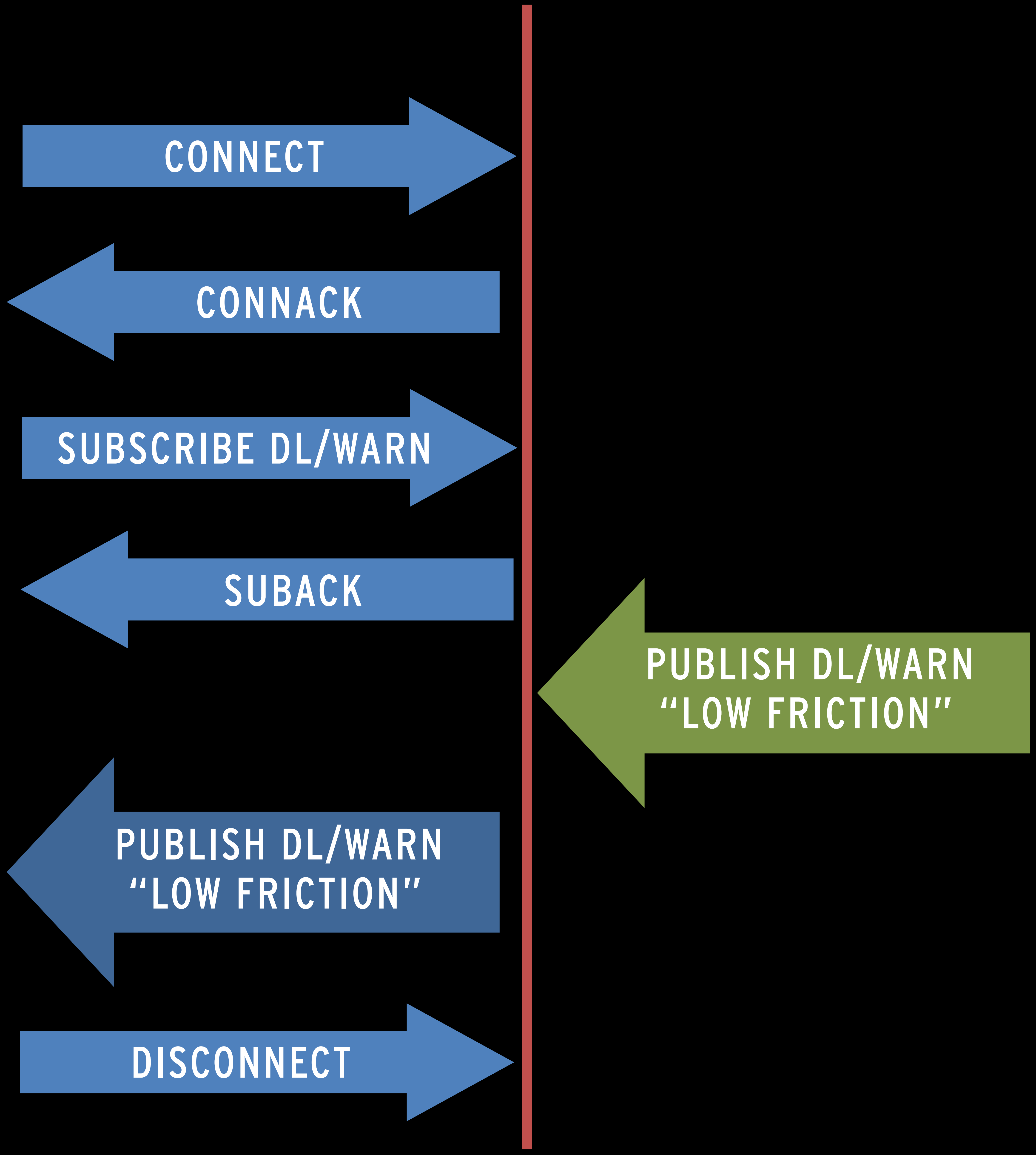
BJÖRN GYLLING

ANDREAS MOSSLJUNG

CALLISTA

WHAT IS MQTT?

- Publish - Subscribe
- Over TCP/IP (for example)
- Lightweight, suitable for IoT
- Quality of service:
  - At most once (0)
  - At least once (1)
  - Exactly once (2)

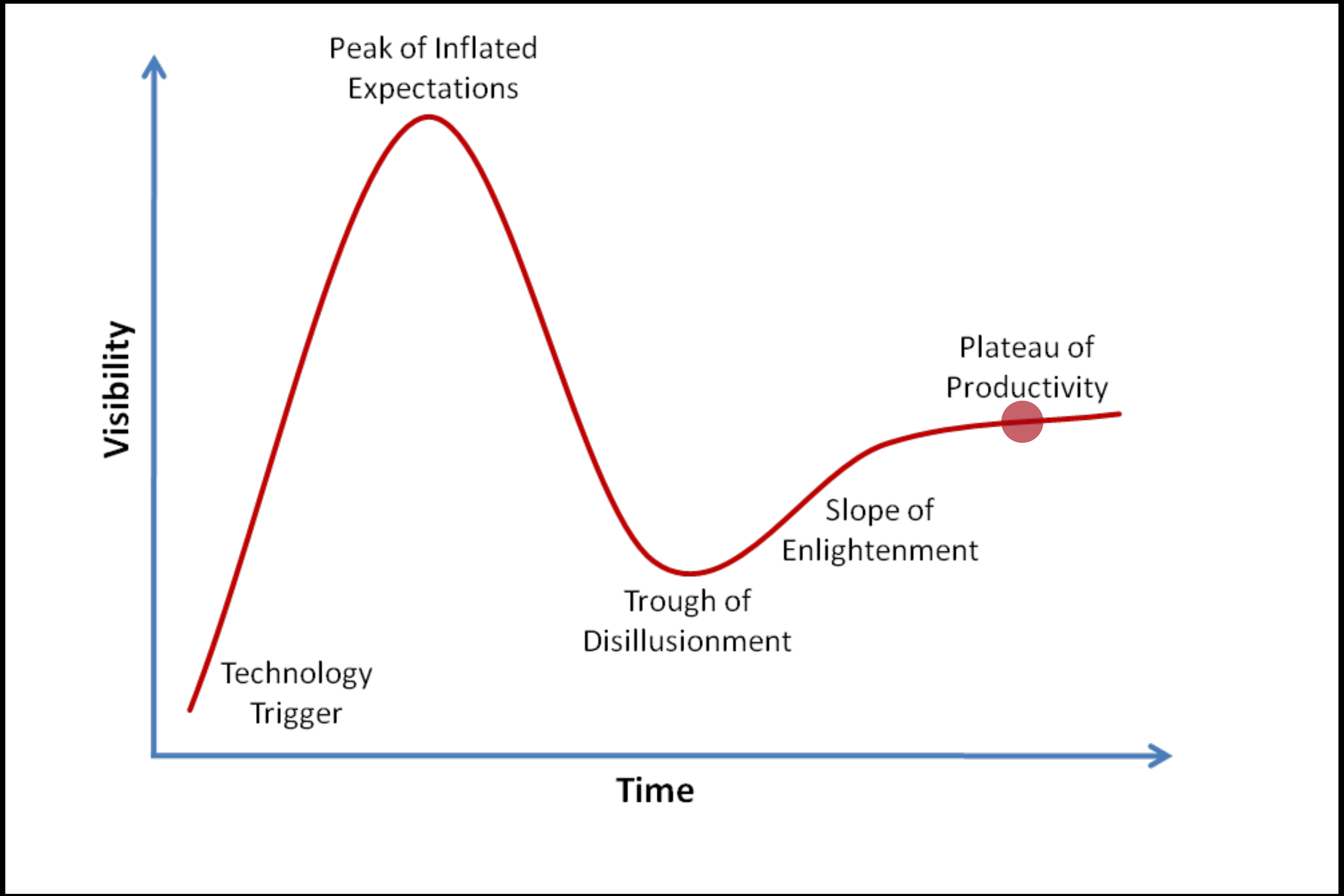

CONNECT
CONNACK
SUBSCRIBE DL/WARN
SUBACK
PUBLISH DL/WARN "LOW FRICTION"
PUBLISH DL/WARN "LOW FRICTION"
DISCONNECT

WHAT WE DON'T USE OF MQTT

- Version 3.1.1, not version 5
- Only QoS 0 (at most once)
  - Ack and resend implemented in application layer as needed
  - No message expiry in version 3.1.1
- No last will or retained messages, all messages are realtime data
- No communication directly between the vehicles via broker, always through backend

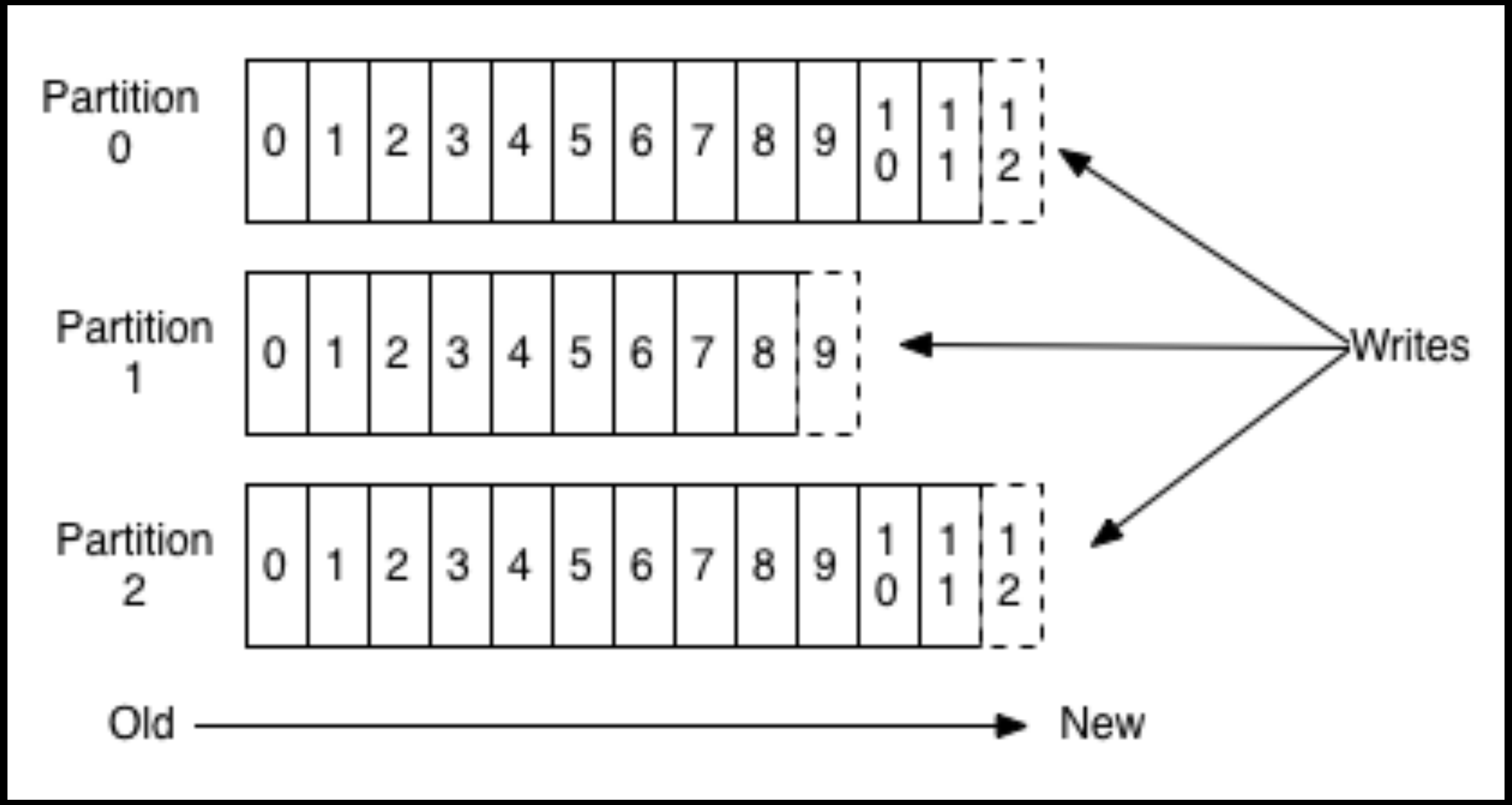
WHAT IS APACHE KAFKA?

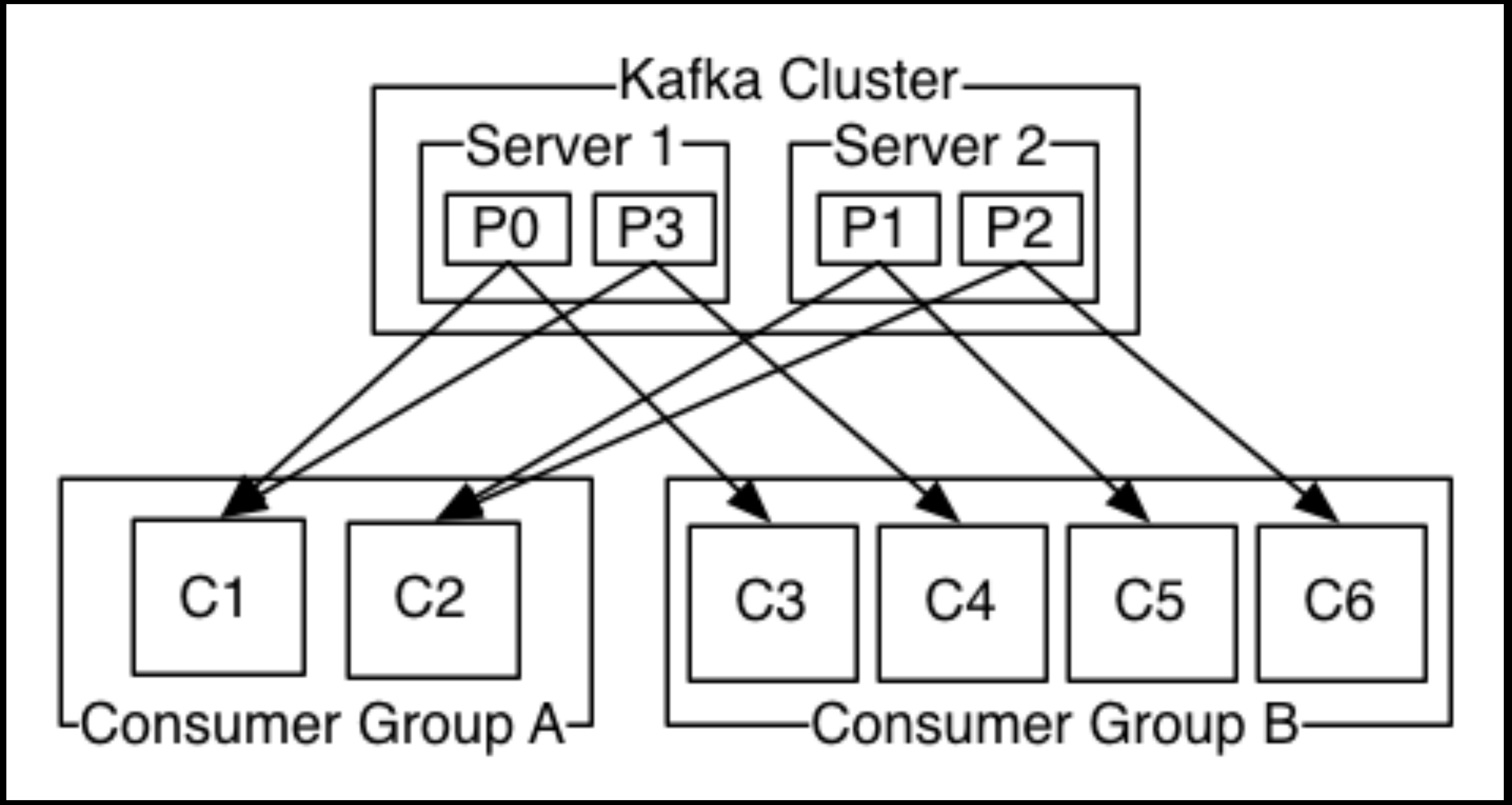
- A distributed streaming platform used for building real-time data pipelines and streaming apps.
- Open-source
- Horizontally scalable, fault-tolerant and fast.
- Familiar to Cadec regulars
- Far to the right on the Gartner hype curve by now



WHAT IS APACHE KAFKA?

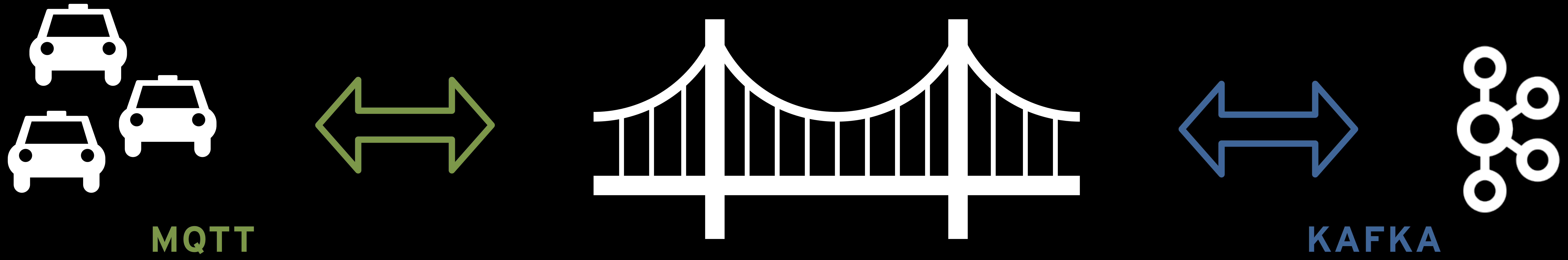
- Publish and subscribe to streams of records
- Also acts as a messaging system and a storgage system
- Streams of records are stored in categories called topics
- Topics are partitioned
- Consumer groups: Each record published to a topic is delivered to one consumer instance within each subscribing consumer group





CALLISTA

BUILDING A BRIDGE



- No MQTT broker
- Connect directly to a micro service running on Kubernetes
- Use Kafka
- Should scale up to the limits of the Kubernetes and Kafka clusters

MQTT TOPICS VS KAFKA TOPICS

MQTT

- UTF-8 characters
- Max 65535 bytes
- Payload can be any binary data
- Typically hierarchical, levels separated by /
- + and # used as wildcards when subscribing

KAFKA

- Alpha-numeric
- Max 255 characters
- Payload is a key-value pair
- Key and value can be any binary data

THE BRIDGE MUST MAP TOPIC NAMES

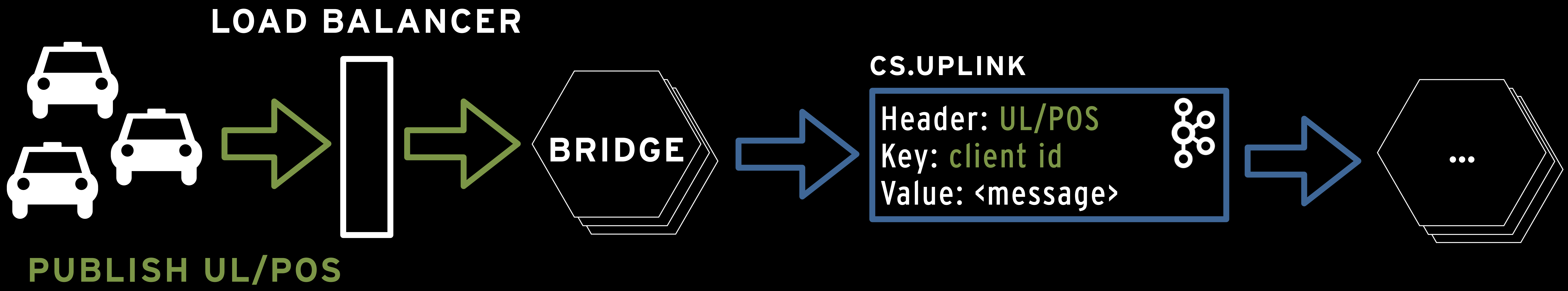
MAPPING OF TOPIC NAMES

- Let a list of regular expressions transform MQTT topic names into Kafka topic names
- This allows multiple MQTT topics to end up one the same Kafka topic
- Add the full MQTT topic as a Kafka header on the message, might contain needed information (example /temperature/roof)
- Use the MQTT client id as Kafka key. For us this is the identity of the car.
- Pass on the MQTT payload as the value of the Kafka message

IMPLEMENTING THE BRIDGE

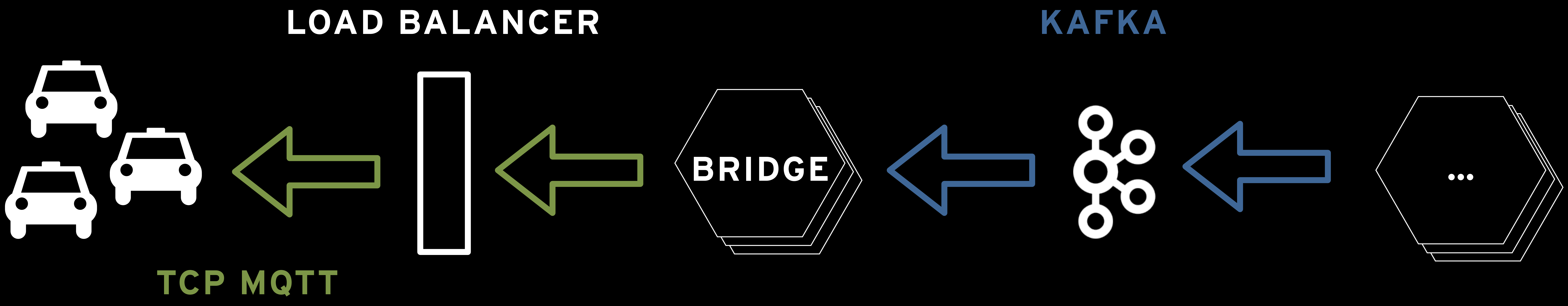
- Only parts of the specification implemented
- Usage of unimplemented features results in closing the TCP connection
- Use Netty to serialise and deserialise MQTT messages
- Non blocking, excellent performance
- About 400 lines of Java code to implement the MQTT protocol

UPLINK



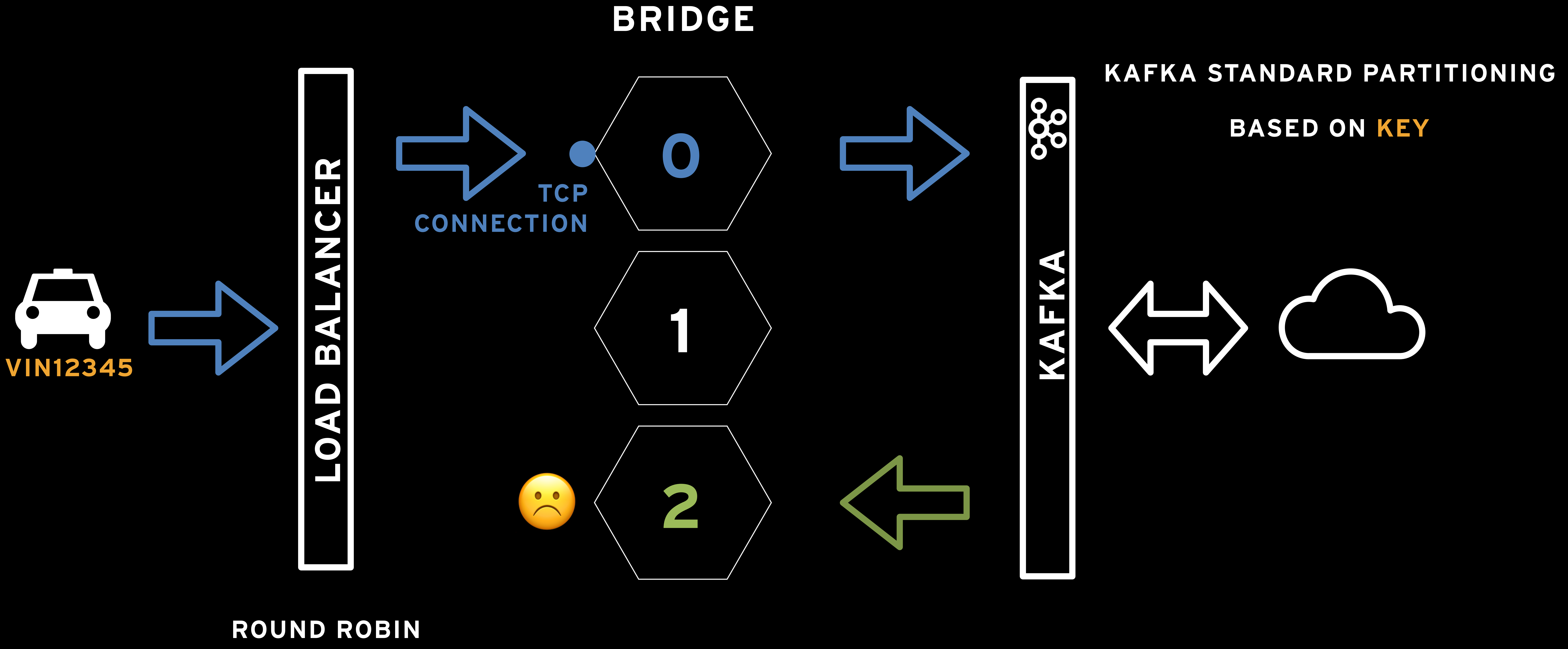
- Client connects to any bridge instance and maintains TCP connection
- Bridge writes to mapped Kafka topic, use client id as key (and partition by it)
- Consumed by the instance of micro service that the broker assigned

DOWNLINK



- Microservices need to publish messages to specific clients
- We never broadcast messages. Publish to Kafka, use client id as key just like uplink
- But the client is connected to the instance of Bridge that was assigned by the load balancer, probably not the same as Kafka assigned the partition

STANDARD PARTITIONING



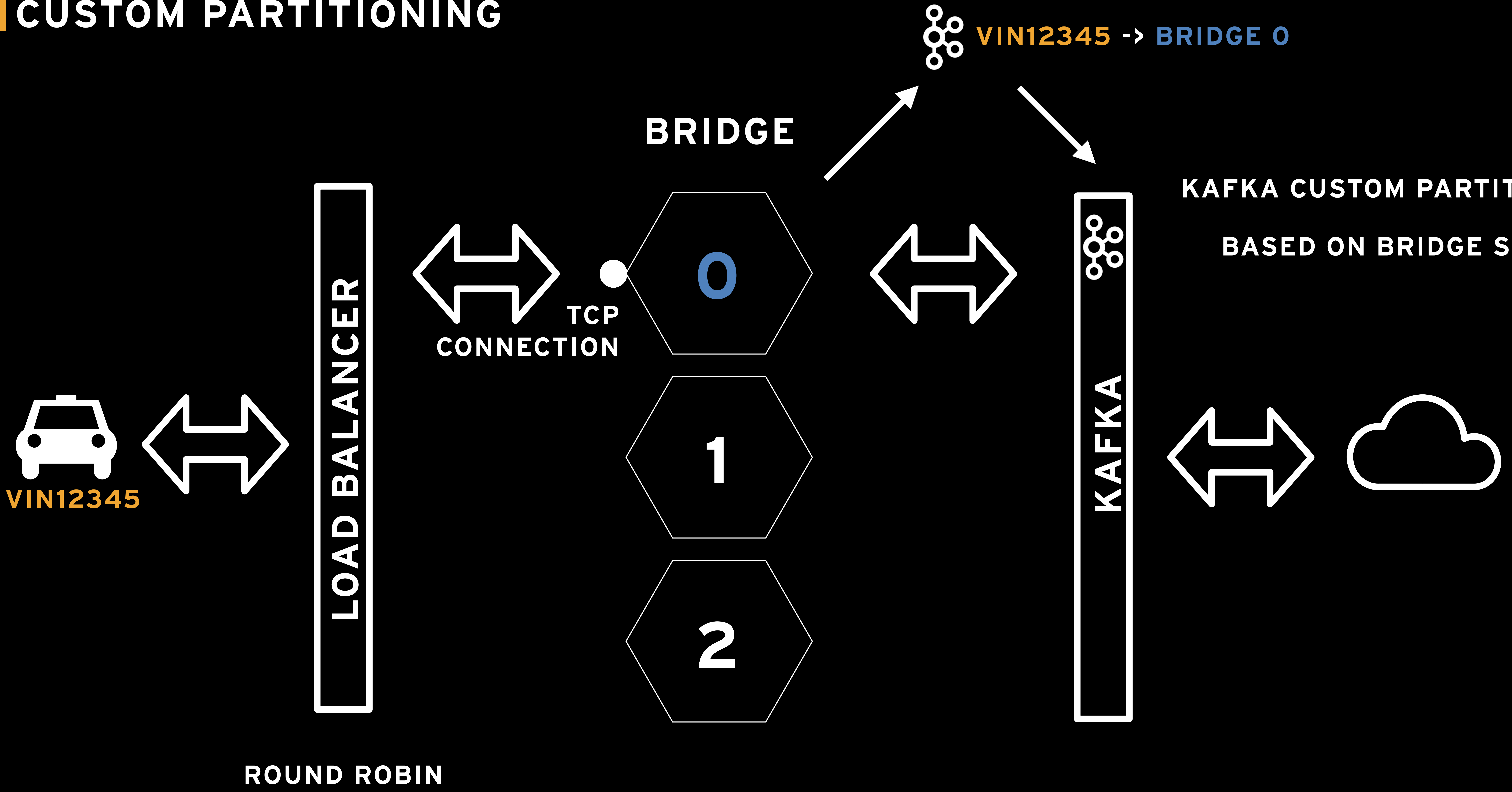
CALLISTA

A MOMENT OF REFLECTION

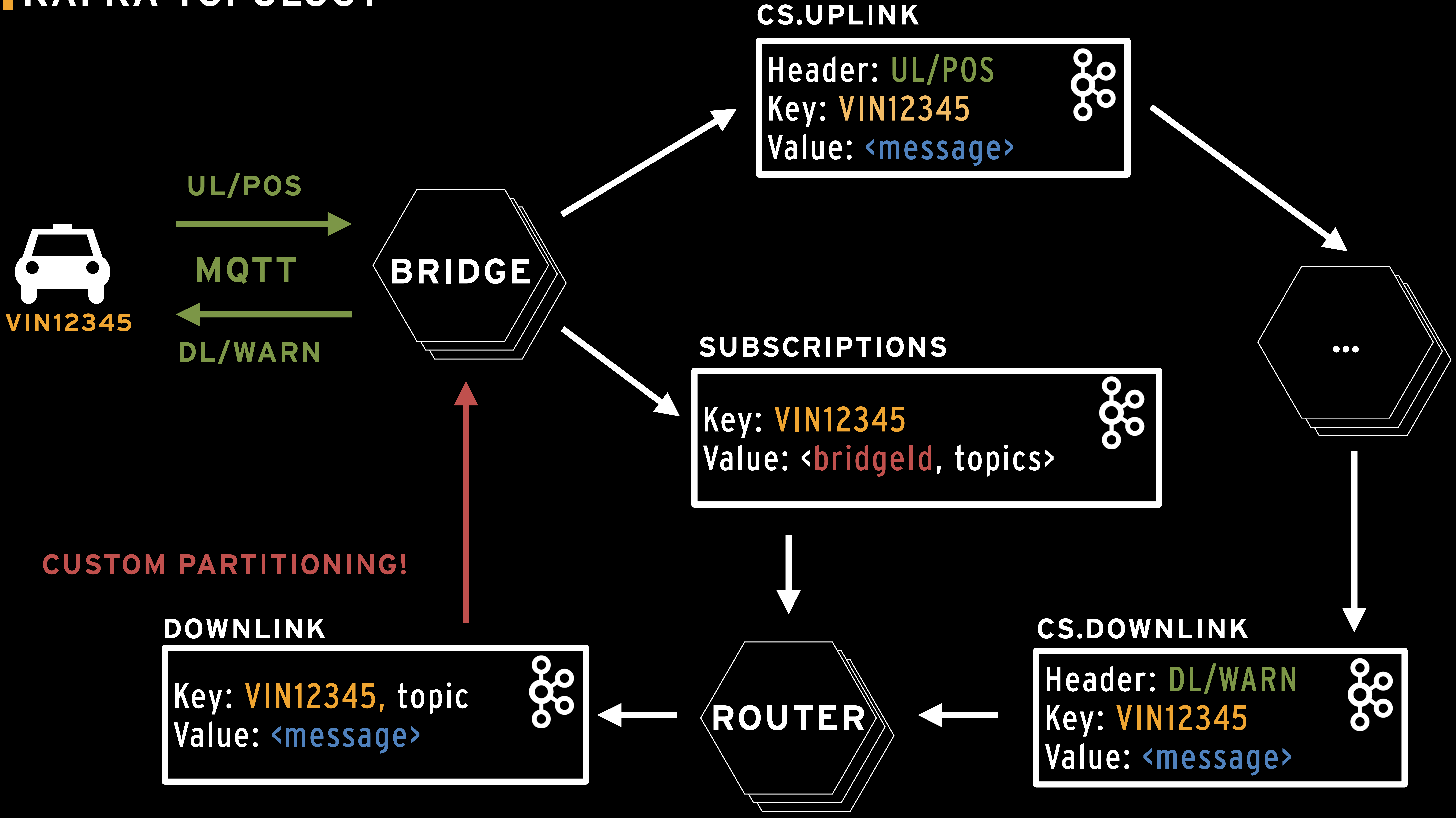
- What we have implemented so far is the same functionality as mqtt-proxy provides in the Confluent Platform product (commercial license)
- It does not support subscription and downlink messages either
- Kafka Connect supports both directions but requires a broker.
- It’s possible to implement custom partitioning with Kafka
- Don’t want to do this in every micro service that sends MQTT. Build another component instead: Router

CALLISTA

CUSTOM PARTITIONING

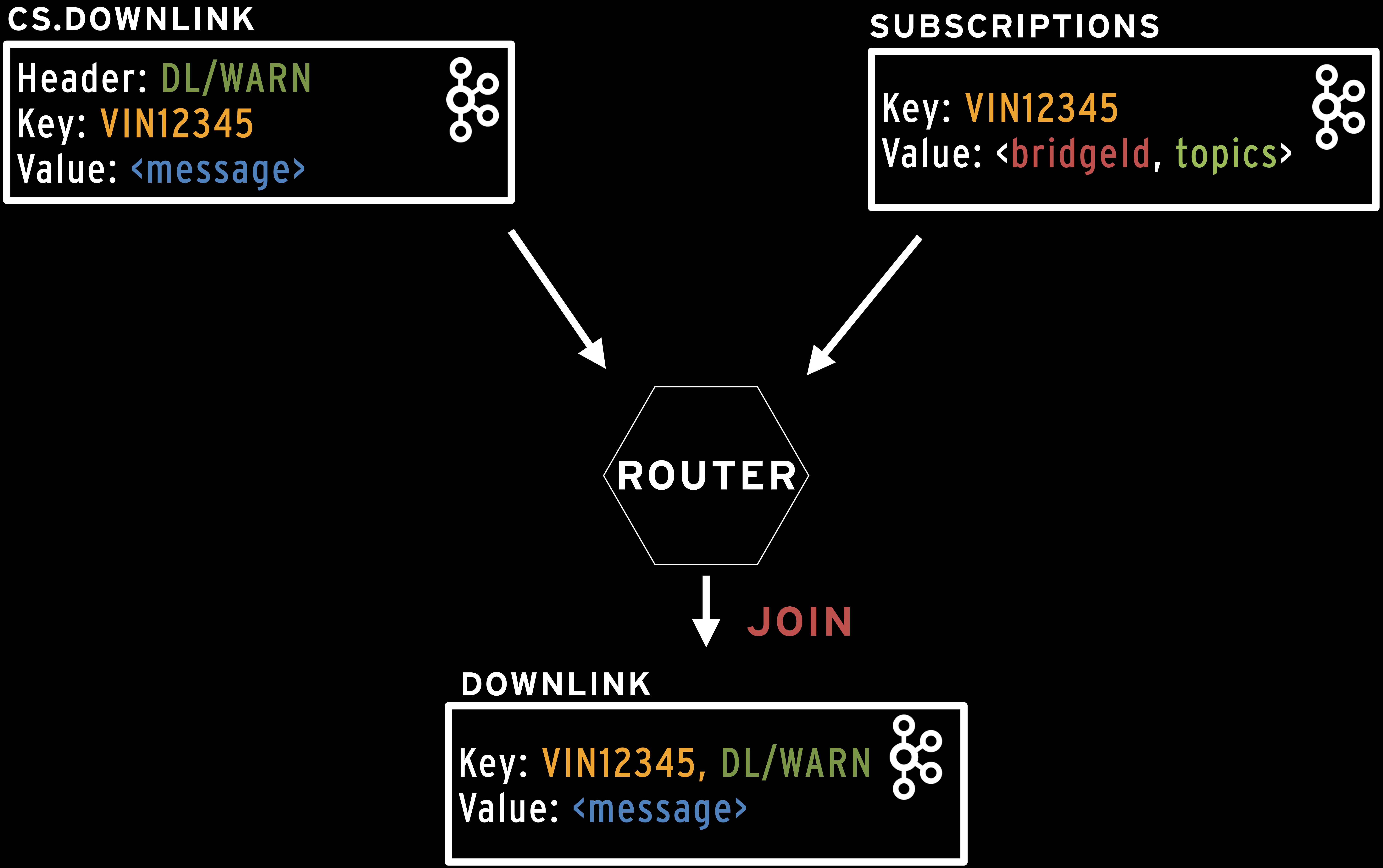


KAFKA TOPOLOGY



CALLISTA

KAFKA STREAMS IMPLEMENTATION OF ROUTER



CALLISTA

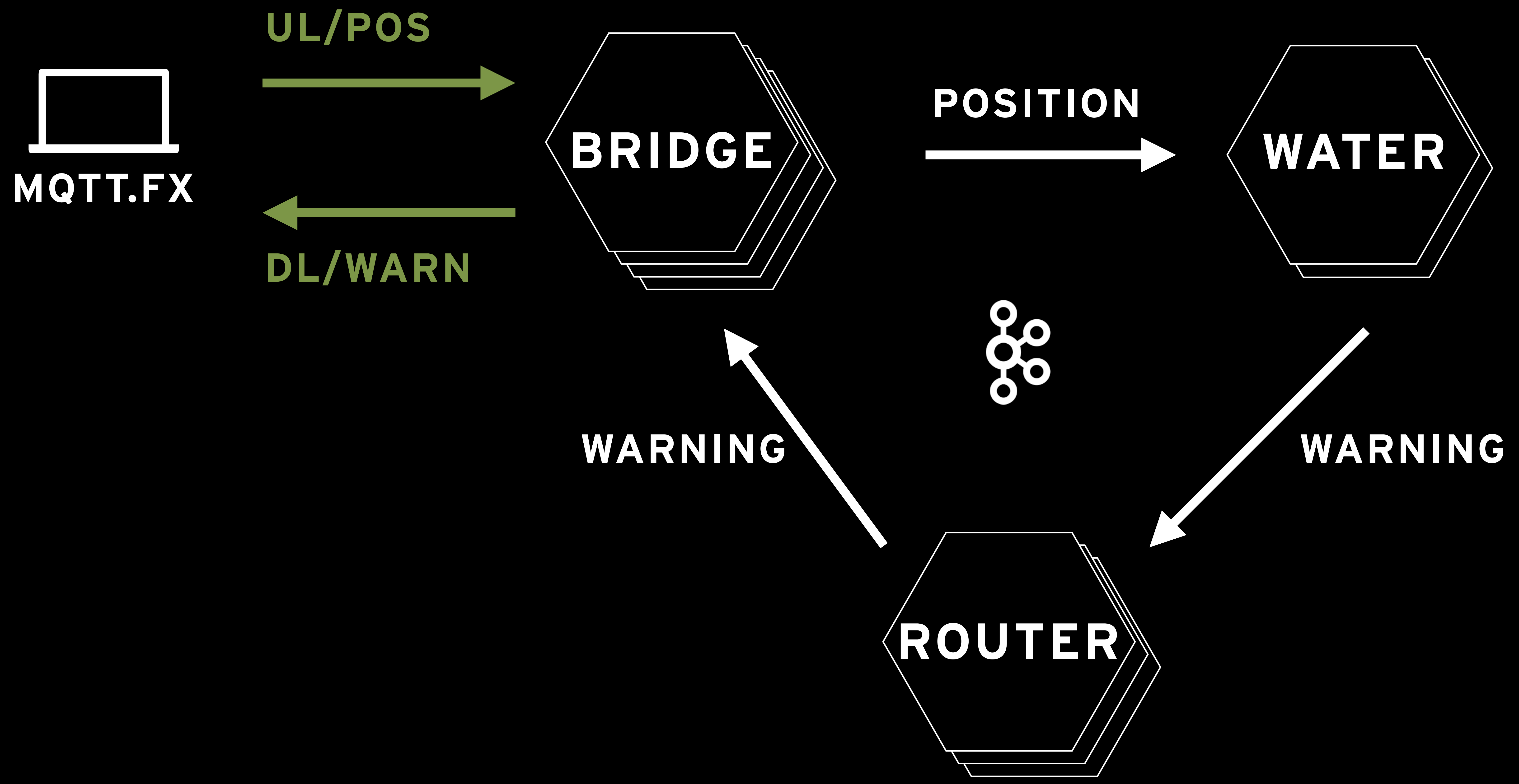
SECURITY

- TLS, client is identified before reaching bridge
- Kafka ACL
  - Access control lists is a feature in Kafka
  - It is possible to configure the components that should have access to each Kafka topic
  - Because of this only selected Kafka topics are accessible over MQTT
- The message payload could possibly be malicious, must be decoded with care.

DEMO

CALLISTA

WATER WARNING SYSTEM



CONCLUSIONS

- Possible to build an advanced, horizontally scalable bridge solution connecting MQTT and Kafka with only two microservices of totally 1300 lines of code
- Possible since we leverage on the scalability of Kafka and Kubernetes
- Combining parts of open source software can be incredibly powerful
- We are about to deploy MQTT Kafka Bridge in production

THANK YOU

ANDREAS.MOSSLJUNG@CALLISTAENTERPRISE.SE

CALLISTA