

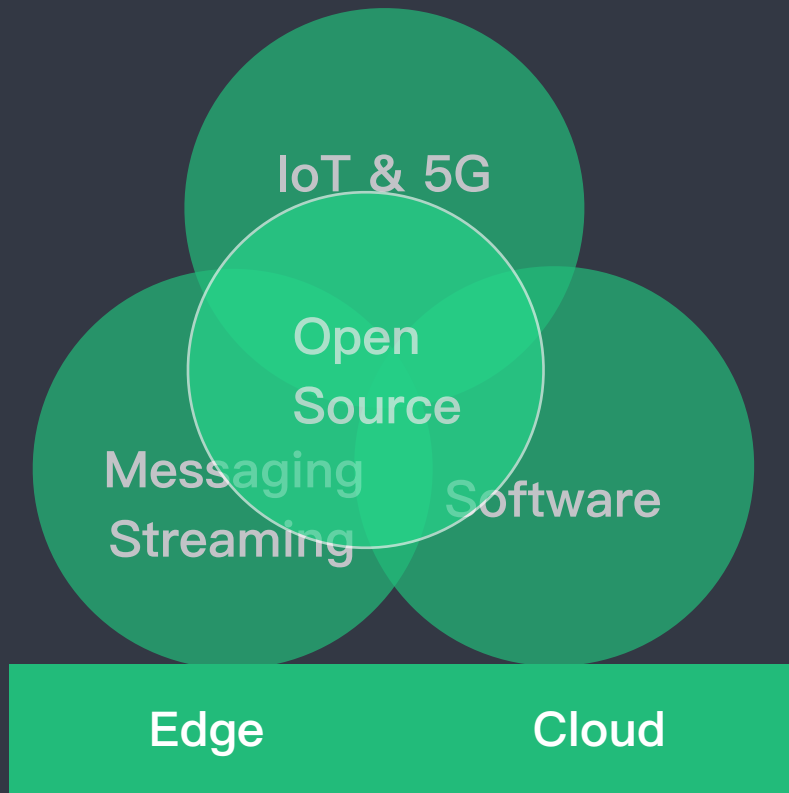
EMQ X Kuiper introduction

A lightweight streaming open source project for edge computing

2019/11



EMQ Technologies – The Global Leader of IoT Messaging



- 1 Commercial Open Source Software
- 2 Serving IoT Industry in 5G Era
- 3 Messaging Middleware Software
- 4 Over 5000+ Enterprises Users Globally
- 5 Global Operations: China, NA, Europe

IoT edge streaming processing challenges

- Running at resource constrained devices
 - No enough resource as @cloud
- Application development
 - Quickly respond to agile biz changes
- Maintenance efforts for large # of deployments
 - Install
 - Upgrade applications
 - Monitor

EMQ X Kuiper

A SQL based IoT rule engine running at resource constrained edge devices.

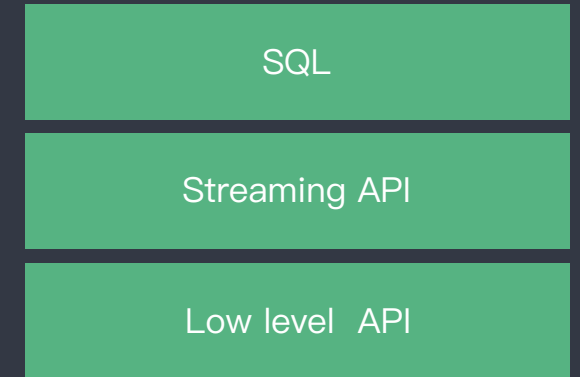
- Native run with small overhead (~7MB package), support Linux/Mac OS
- SQL based, easy to use
- Built-in support for MQTT source
- Extension – user can customize the rule engine
- RESTful APIs for rules management

Project Github address

- <https://github.com/emqx/kuiper>

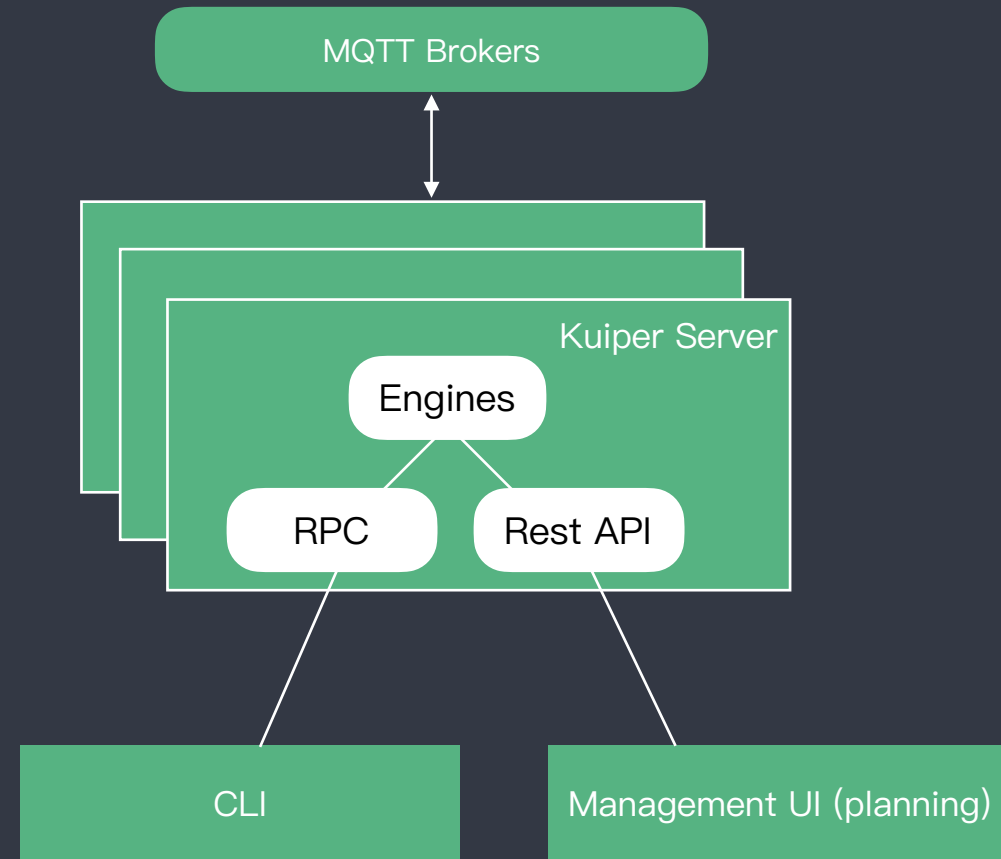
Levels of streaming process abstraction

- Low level API
 - Use low-level API provided by different SDKs, such as MQTT language SDKs
 - Protocol specific, flexible, but not easy to develop streaming applications
- Streaming API
 - Provides streaming abstraction API, easy to develop stream orient applications
- SQL
 - Embedded streaming support, agile development

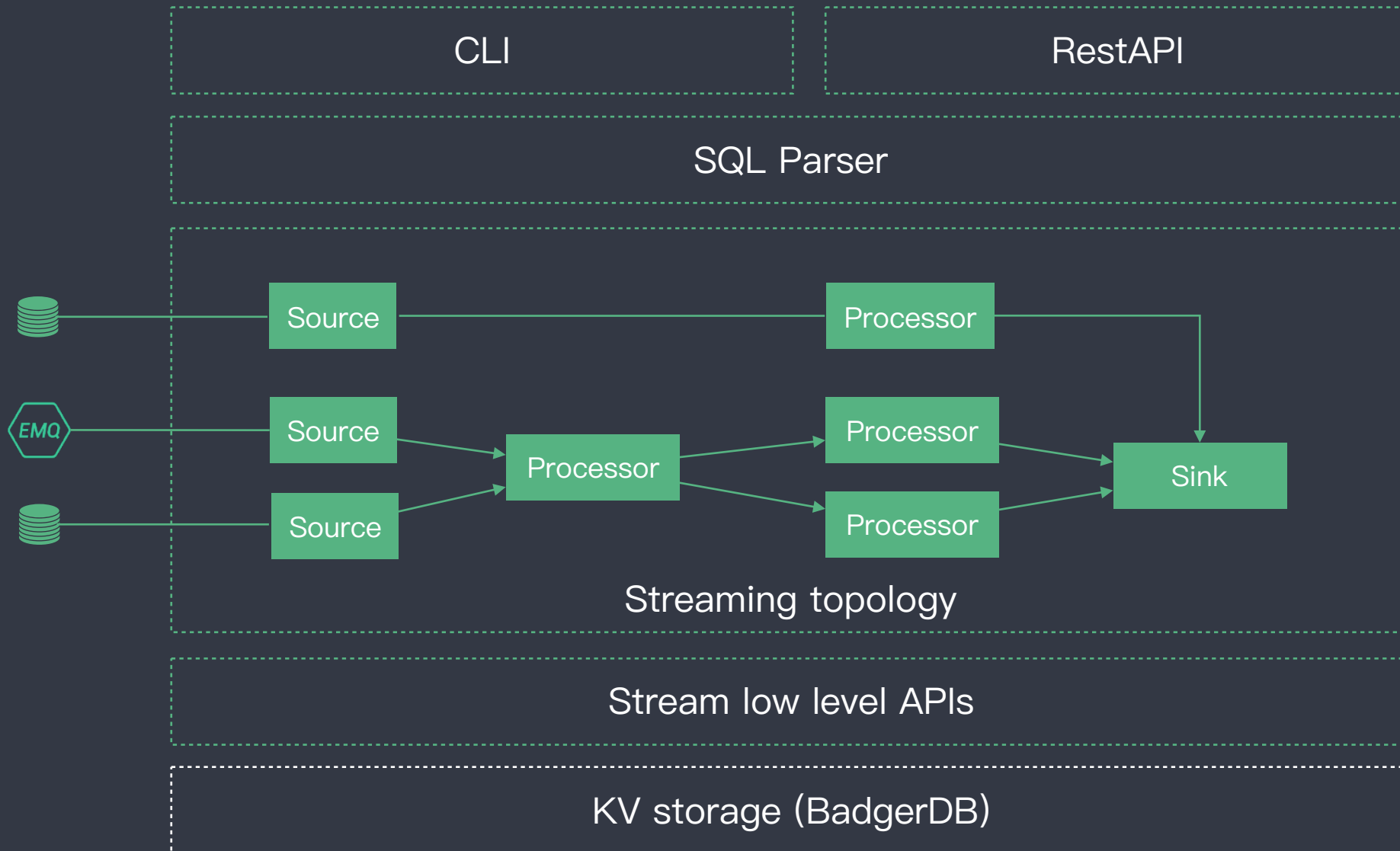


Components

- Kuiper server
 - Engine
 - SQL parser
 - Stream infrastructure
 - RPC: RPC interface for remote CLI tools
 - REST API: APIs for management UI (In planning)
- CLI
 - Command line tools
 - Stream manager
 - Rules manager
 - Query tools
- Management UI (planning)
 - Web interface for Kuiper management



Architecture



Streams

- Stream definition
 - Data types: bigint, float, string, datetime, boolean, array, struct
- Stream management
 - create stream
 - drop stream
 - show streams
 - describe stream

```
CREATE STREAM
  stream_name
  ( column_name <data_type> [ ,...n ] )
  WITH ( property_name = expression [, ...] );
```

```
demo (
  USERID BIGINT,
  FIRST_NAME STRING,
  LAST_NAME STRING,
  NICKNAMES ARRAY(STRING),
  Gender BOOLEAN,
  ADDRESS STRUCT(STREET_NAME STRING, NUMBER BIGINT),
) WITH (datasource="test/", FORMAT="JSON", KEY="USERID", CONF_KEY="demo");
```


Rules

- Rules definition
 - id, sql & actions
- Rules management
 - create rule
 - drop rule
 - show rules
 - start/stop/restart rule
 - getstatus

Parameter name	Optional	Description
id	false	The id of the rule
sql	false	The sql query to run for the rule
actions	false	An array of sink actions

```
{
  "sql": "SELECT avg(temperature) AS t_av, max(temperature) AS t_max, min(temperature) AS
t_min, COUNT(*) AS t_count, split_value(mqtt(topic), '\\\\', 1) AS device_id FROM demo
GROUP BY device_id, TUMBLINGWINDOW(ss, 10)",
  "actions": [
    {
      "log": {}
    },
    {
      "mqtt": {
        "server": "ssl://xyz-ats.iot.us-east-1.amazonaws.com:8883",
        "topic": "devices/result",
        "qos": 1,
        "clientId": "demo_001",
        "certificationPath": "/var/aws/d3807d9fa5-certificate.pem",
        "privateKeyPath": "/var/aws/d3807d9fa5-private.pem.key"
      }
    }
  ]
}
```

SQL

SELECT

```
SELECT
  *
  | [source_stream.]column_name [AS column_alias]
  | expression
```

FROM

```
FROM source_stream | source_stream AS source_stream_alias
```

JOIN

```
LEFT | RIGHT | FULL | CROSS
JOIN
source_stream | source_stream AS source_stream_alias
ON <source_stream|source_stream_alias>.column_name =<source_stream|source_stream_alias>.column_name
```

GROUP

```
GROUP BY <group by spec>

<group by spec> ::=
  <group by item> [ ,...n ]
  | <window_type>

<group by item> ::=
  <column_expression>
```

ORDER

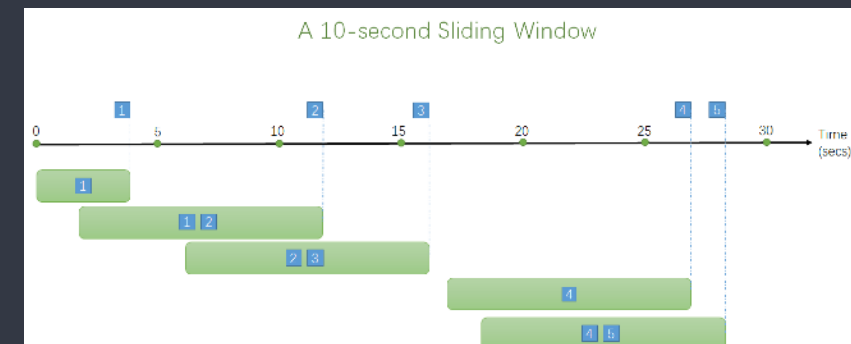
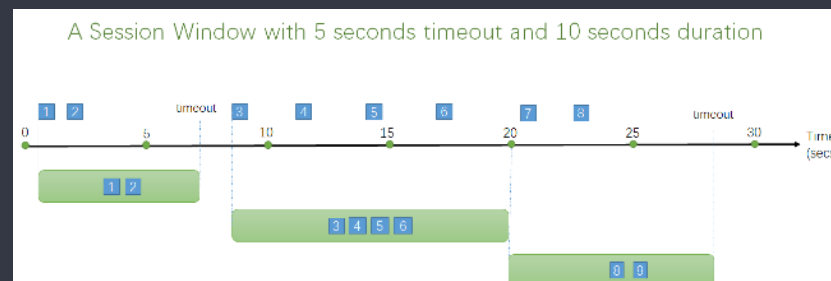
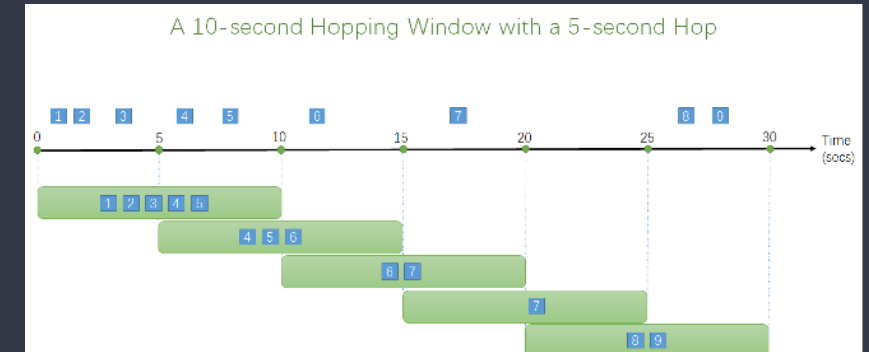
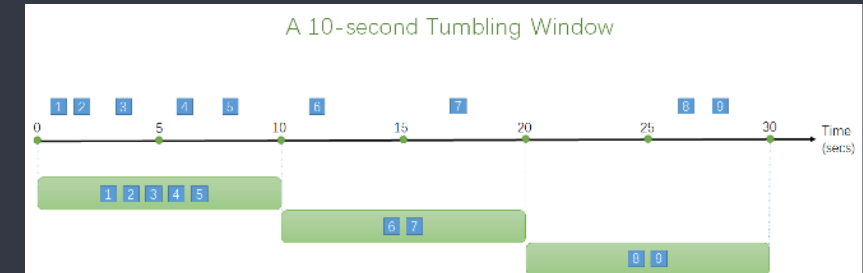
```
ORDER BY column1, column2, ... ASC|DESC
```

HAVING

```
HAVING <search condition>
```

Windows support

- **Tumbling window**
 - Segment a data stream into distinct time segments and perform a function against them
- **Hopping window**
 - Hop forward in time by a fixed period. It may be easy to think of them as Tumbling windows that can overlap
- **Sliding window**
 - Produce an output **ONLY** when an event occurs
- **Session window**
 - Group events that arrive at similar times, filtering out periods of time where there is no data. It has two main parameters: timeout and maximum duration



Functions (~60)

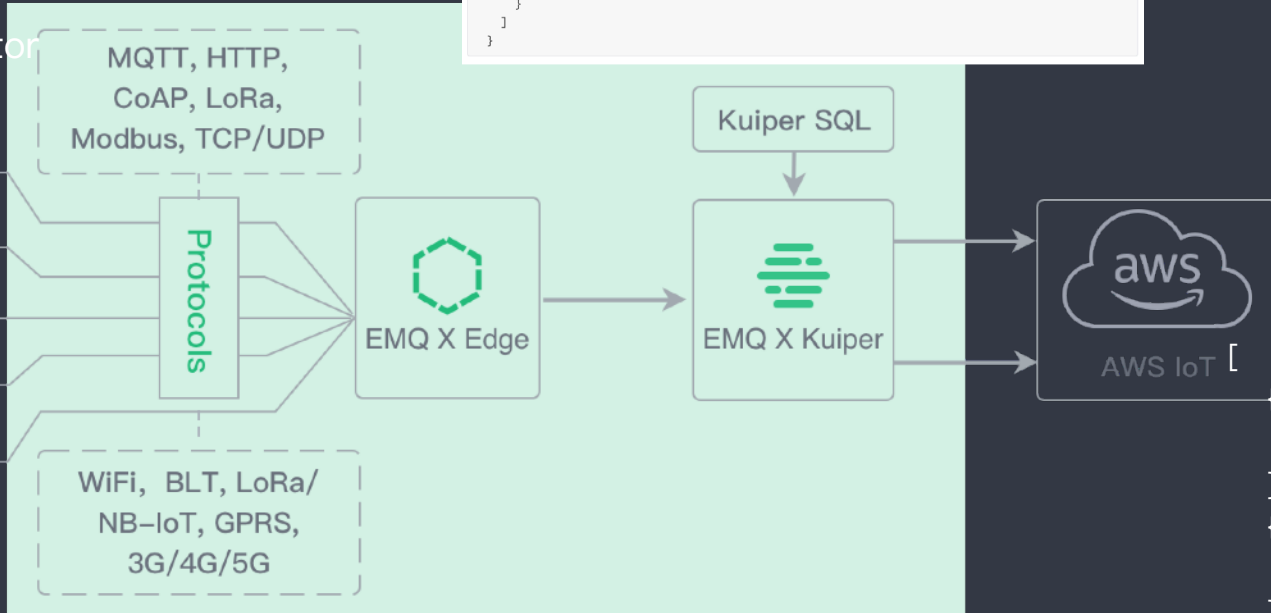
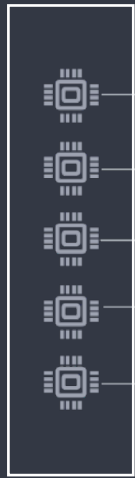
- Aggregate Functions
 - avg/count/max/min/sum
- Mathematical Functions
 - abs/acos/asin.../sqrt/tan/tanh
- String Functions
 - concat/endswith/indexof/length/.../trim/upper
- Conversion Functions
 - cast/chr/encode
- Hashing Functions
 - md5/sha1/sha256/sha384/sha512
- Other Functions
 - isNull/nanvl/newuuid

Extensions – under development

- Sources
 - Consume message from different source
- Actions/Sinks
 - Extension point for supporting different kinds of actions
- Functions
 - Extension point for developing customized functions

Demo

JMeter
Device simulator



```
{
  "sql": "SELECT avg(temperature) AS t_av, max(temperature) AS t_max, min(temperature) AS t_min, COUNT(*) AS t_count, split_value(mqtt(topic), '\\/', 1) AS device_id FROM demo GROUP BY device_id, TUMBLINGWINDOW(ss, 10)",
  "actions": [
    {
      "log": {}
    },
    {
      "mqtt": {
        "server": "ssl://xyz-ats.iot.us-east-1.amazonaws.com:8883",
        "topic": "devices/result",
        "qos": 1,
        "clientId": "demo_001",
        "certificationPath": "/var/aws/d3807d9fa5-certificate.pem",
        "privateKeyPath": "/var/aws/d3807d9fa5-private.pem.key"
      }
    }
  ]
}
```

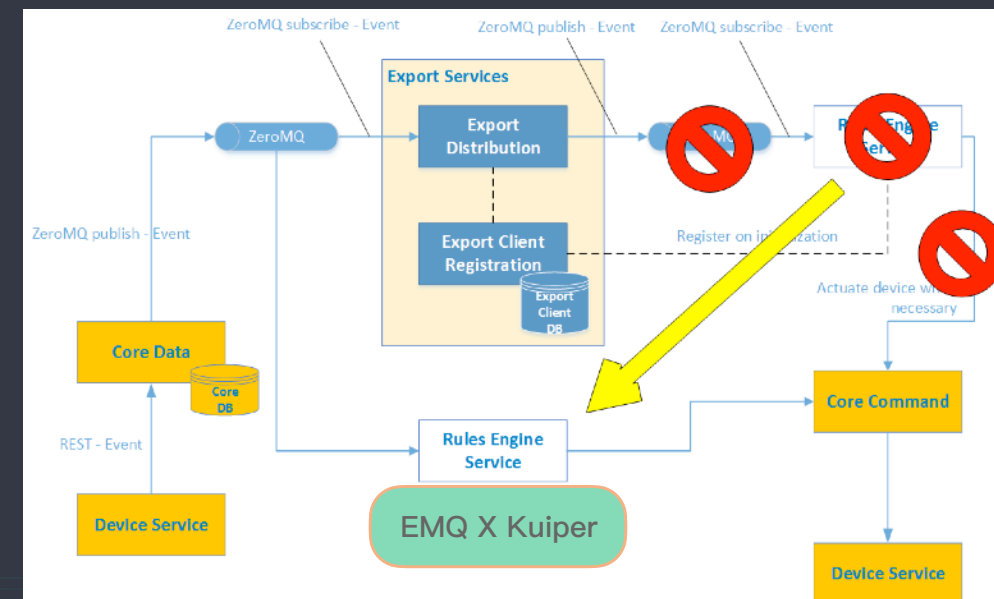
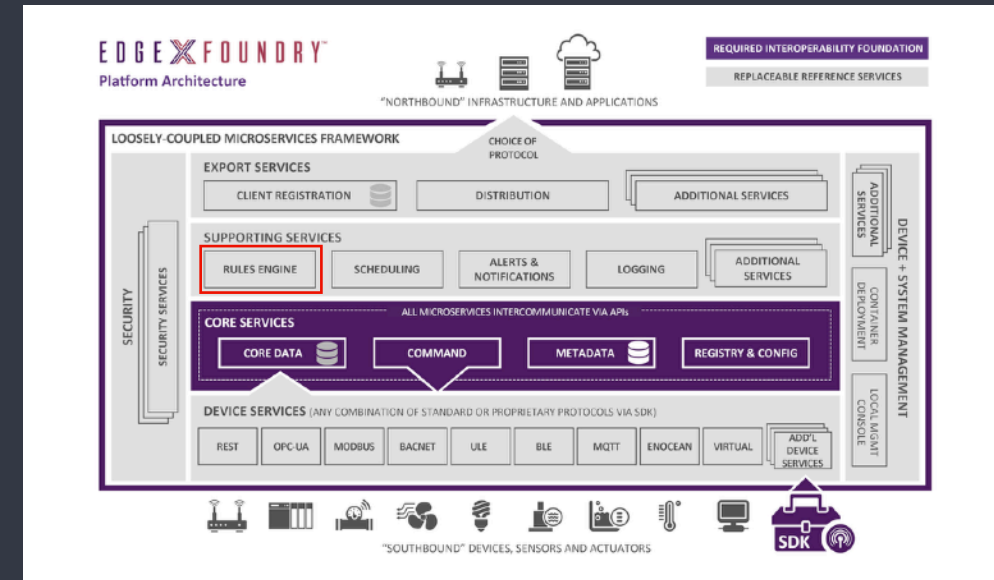
```
Topic: devices/{device_id}/messages
{
  "temperature": 30,
  "humidity" : 20
}
```

```
["device_id" : "1", "t_av" : 25, "t_max" : 45, "t_min" : 5, "t_count" : 2
},
{
"device_id" : "2", "t_av" : 25, "t_max" : 45, "t_min" : 5, "t_count" : 2
},
...
]
```



Integration with Edge X Foundry

- Extend Kuiper source & sink to integrate with Edge X Foundry, and contribute the extension to Edge X Foundry
- EMQ X Kuiper AS a separated open source project (similar current relationship with Drools).
- Technical solution
 - Extend source & sink
 - ZeroMQ source & sink
 - Services
 - Support the Edge X component services
 - Management tools
 - CLI / Restful API



Next step

- Kuiper management UI
- Stream state management support

A decorative graphic consisting of multiple overlapping, wavy lines in a light green color, creating a sense of motion and depth across the middle of the slide.

Thank You

contact@emqx.io

