eKuiper

Status

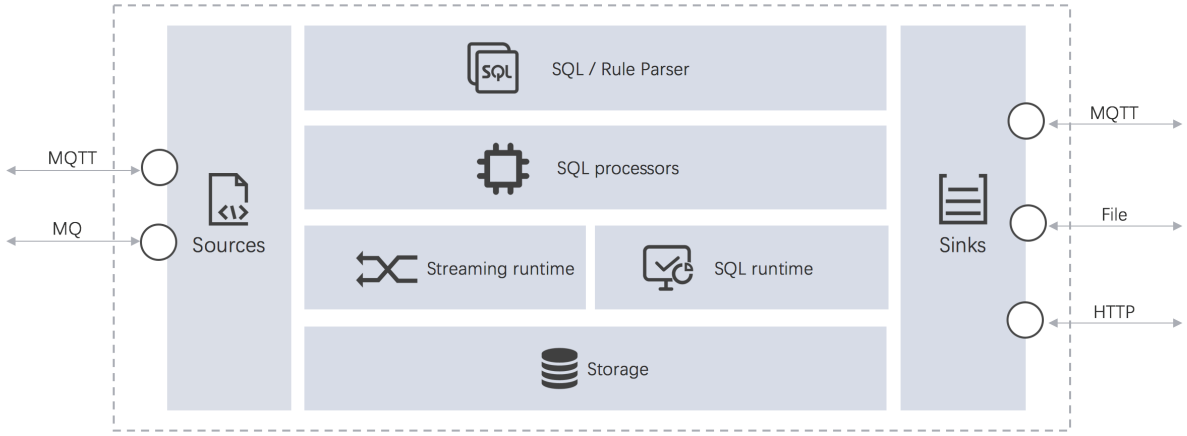
- **Current Project Stage:** Stage 1 - At Large
- **Website:** Coming Soon
- **Wiki:** Coming Soon
- **TAC Sponsors:** Leding Li (Baidu), jianguo zeng (Huawei)
- **Project Overview**: TAC Presentation (April 21)

- Originally applied under the name Kuiper. Final Open Source Project Name: eKuiper.
- Presented during the Wednesday, April 21 TAC call: Meeting Recording (https://zoom.us/rec/share/JKTpoOax0sntLIujoRdoaP-qcfFkcRwlvvoDGo-PYB7JKRn9txqTq-wJEJfpLw_7.MkQNsH_6bYEzJosY?startTime=1619013013000)
- TAC two-thirds vote approval reached on May 18, 2021.
- Governing Board Strategic Planning Committee approval reached on May 24, 2021.

1. Project Description

What it does

EMQ X Kuiper is an edge lightweight IoT data analytics/streaming software implemented by Golang, and it can be run at all kinds of resource-constrained edge devices. One goal of Kuiper is to migrate the cloud streaming software frameworks (such as Apache Spark, Apache Flink) to edge side. Kuiper references these cloud streaming frameworks, and also considered a special requirement of edge analytics, and introduced **rule engine**, which is based on Source, SQL (business logic) and Sinks, the rule engine is used for developing streaming applications at edge side.



Why it is valuable

- Lightweight
  - The core server package is only about 4.5M, the initial memory footprint is about 10MB
- Cross-platform
  - CPU ArchX86 AMD * 32, X86 AMD * 64; ARM * 32, ARM * 64; PPC
  - The popular Linux distributions, OpenWrt Linux, MacOS and Docker
  - Industrial PC, Raspberry Pi, industrial gateway, home gateway, MEC edge cloud server
- Data analysis support
  - Support data extract, transform and filter through SQL
  - Data order, group, aggregation, and join
  - 60+ functions include mathematical, string, aggregate and hash, etc
  - 4-time windows & count window
- Highly extensible

  Plugin system is provided, and it supports to extend at Source, SQL functions and Sink.

    - Source: embedded support for MQTT, and provide extension points for sources
    - Sink: embedded support for MQTT and HTTP, and provide extension points for sinks
    - UDF functions: embedded support for 60+ functions, and provide extension points for SQL functions
- Management
    - A web-based management dashboard (opens new window)for nodes, plugins, streams & rules management
    - Plugins, streams and rules management through CLI & REST API
    - Easily be integrate with KubeEdge (opens new window), K3s (opens new window) and Baetyl (opens new window), which bases Kubernetes
- Integration with EMQ X Edge

  Seamless integration with EMQ X Neuron & EMQ X Edge, and provided an end-to-end solution from messaging to analytics.

Origin and history

The project is originated from a fully-functional streaming software created and donated by EMQ. It is developed with feedback and contributions from partners, customers and community users. Below is the milestones of the project.

- 22, Oct 2020: 1.0.0 - The 1st stable version.
- 21, Feb 2020: 0.3.2 - The 1st version integrated with the LF EdgeX Foundry project.
- 23, Oct 2019: 0.2.0 - The 1st released version.

2. Statement of mission

The mission of Kuiper is aligned with LF Edge:

- Build Kuiper as a lightweight IoT data analytics / streaming software in edge side
- Enable organizations to accelerate the adoption of edge computing by providing handy tools and frameworks
- Integrate with relevant edge projects by providing built-in support
- Promote edge computing ecosystem by providing mechanisms for extension and integration

3. Project synergy

1. EdgeX Foundry rules engine, it's reference implementation edge analytics service that performs if-then conditional actuation at the edge based on sensor data collected by the EdgeX instance. Kuiper has partnered with EdgeX Foundry to provide the default rule engine service.
2. Baetyl project integration. Baetyl project also integrated Kuiper, which can help users to perform data analytics within Baetyle framework.
3. Fledge: Kuiper has a flexible and powerful data analytic feature that compliments Fledge. One possible way to harmonize both products is that Fledge collects data from all kinds of sensors and then deliver to Kuiper for analysis.

4. Leadership team and decision making process

5. Architecture

The Kuiper core runtime architecture consist a pipeline of sources (built-in or extended), SQL layer and sinks (built-in or extended) as shown below.

The SQL layer components include:

- SQL/Rule Parser: Parse commands to orchestrate the pipeline of processors
- SQL Processors: The working operators to do the SQL interpreted data transformation
- Streaming/SQL runtime: The runtime to deal with streaming like window, event time/watermark and effective once semantic.
- Storage: Components to persist rule state and system data.

There are other supportive components include:

- Plugin management: Manage plugin creation, loading and deletion for source, function and sink.
- Configuration management: Manage system configuration.
- Monitoring: provide metrics for rules.
- REST API and CLI: expose management API

6. Project Template

Name of Project	EMQ X Kuiper
Project Description (what it does, why it is valuable, origin and history)	EMQ X Kuiper is an edge lightweight IoT data analytics / streaming software implemented by Golang, and it can be run at all kinds of resource constrained edge devices. Considered special requirement of edge analytics, and introduced **rule engine**, which is based on *Source*, *SQL (business logic)* and *Sink*, rule engine is used for developing streaming applications at edge side. Kuiper is created by EMQ X and open sourced since July 2019. It is then developed with feedback and contributions from partners, customers and community users.

Statement on alignment with Foundation Mission Statement	The mission of Kuiper is aligned with LF Edge: • Build Kuiper as a general lightweight IoT data analytics / streaming software in edge side; • Enable organizations to accelerate adoption of edge computing by providing handy tools and frameworks that can help user to process IoT data closing to device; • Integrate with relevant edge projects by providing built-in support; • Promote edge computing ecosystem by providing mechanisms for extension and integration.
High level assessment of project synergy with existing projects under LF Edge, including how the project compliments/overlaps with existing projects, and potential ways to harmonize over time. Responses may be included both here and/or in accompanying documentation.	Kuiper is a neutral edge streaming analytics framework, and it can be easily integrated with IoT edge frameworks that require streaming analytics. 1. EdgeX Foundry rules engine, it's reference implementation edge analytics service that performs if-then conditional actuation at the edge based on sensor data collected by the EdgeX instance. Kuiper has partnered with EdgeX Foundry to provide the default rule engine service. 2. Baetyl project integration. Baetyl project also integrated Kuiper, which can help users to perform data analytics within Baetyle framework. 3. Fledge: Kuiper has a flexible and powerful data analytic feature which compliments Fledge. One possible way to harmonize both products is that Fledge collects data from all kinds of sensors and then deliver to Kuiper for analysis.
Link to *current* Code of Conduct	https://github.com/emqx/kuiper/wiki/Code-of-Conduct
2 TAC Sponsors, if identified (Sponsors help mentor projects) - See full definition on Project Stages: Definitions and Expectations	Leding Li, Jianguo Zeng
Project license	Apache 2.0
Source control (GitHub by default)	https://github.com/emqx/kuiper
Issue tracker (GitHub by default)	https://github.com/emqx/kuiper
External dependencies (including licenses)	https://github.com/PaesslerAG/gval (BSD 3-Clause "New" or "Revised" License) https://github.com/PaesslerAG/jsonpath (BSD 3-Clause "New" or "Revised" License) https://github.com/benbjohnson/clock (MIT License) https://github.com/buger/jsonparser (MIT License) http://github.com/eclipse/paho.mqtt.golang (the Eclipse Public License 1.0 and the Eclipse Distribution License 1.0 as described in the epl-v10 and edl-v10 files) https://github.com/edgexfoundry/go-mod-core-contracts (Apache License 2.0) https://github.com/edgexfoundry/go-mod-messaging (Apache License 2.0) https://github.com/go-yaml/yaml (Apache License 2.0) https://github.com/golang-collections/collections (MIT License) https://github.com/google/uuid (BSD 3-Clause "New" or "Revised" License) https://github.com/gorilla/handlers (BSD 2-Clause "Simplified" License) https://github.com/gorilla/mux (BSD 3-Clause "New" or "Revised" License) https://github.com/keepeye/logrus-filename (MIT License) https://github.com/lestrrat-go/file-rotatelogs (MIT License) https://github.com/lestrrat-go/strftime (MIT License) https://github.com/mattn/go-sqlite3 (MIT License) https://github.com/pebbe/zmq4 (BSD 2-Clause "Simplified" License) https://github.com/prometheus/client_golang (Apache License 2.0) https://github.com/sirupsen/logrus (MIT License) https://github.com/urfave/cli (MIT License) https://github.com/golang.org/x/net (BSD 3-Clause) https://github.com/gopkg.in/ini.v1 (Apache License 2.0)
Release methodology and mechanics	Use the Github actions to automatically building new versions.
Names of initial committers, if different from those submitting proposal	• Fahua Jin • Jiyong Huang • Yuedong Ma • Rory Zhang • Shifan Yu • Yongxing Ma
Current number of code contributors to proposed project	16
Current number of organizations contributing to proposed project	5
Briefly describe the project's leadership team and decision-making process	The Technical Steering Committee (TSC) is a committee composed of technical leaders from the open source project responsible for oversight of the technical codebase, the technical community and release process. Members: • Fahua Jin / zh-cn: • Jiyong Huang / zh-cn: • Yuedong Ma / zh-cn: • Rory Zhang / zh-cn: • Shifan Yu / zh-cn: • Yongxing Ma / zh-cn: TSC team will discuss any proposal, and make decision based on the discussion.
List of project's official communication channels (slack, irc, mailing lists)	slack kuiper channel
Link to project's website	https://www.emqx.io/products/kuiper, and the website https://kuiper.emqx.io is under construction.
Links to social media accounts	blocked URLEMQTech
Existing financial sponsorship	EMQ Technologies Co., Ltd.
Infrastructure needs or requests (to include GitHub/Gerrit, CI/CD, Jenkins, Nexus, JIRA, other ...)	GitHub

Currently Supported Architecture	x86, x86-64, ARM, MIPS
Planned Architecture Support	N/A
Project logo in svg format (see https://github.com/lf-edge/lfedge-landscape#logos for guidelines)	Preview unavailable 
Trademark status	Kuiper trademark is applied in P.R China.
Does the project have a Core Infrastructure Initiative security best practices badge? (See: https://bestpractices.coreinfrastructure.org)	No
Any additional information the TAC and Board should take into consideration when reviewing your proposal?	No

Mapping Criteria and Data:

Stage 1: At Large Projects (formerly 'Sandbox')

Criteria	Data
2 TAC Sponsors, if identified (Sponsors help mentor projects) - See full definition on Project Stages: Definitions and Expectations	Leding Li, Jianguo Zeng
A presentation at an upcoming meeting of the TAC, in accordance with the project proposal requirements	TBD, can be arranged.
The typical IP Policy for Projects under the LF Edge Foundation is Apache 2.0 for Code Contributions, Developer Certificate of Origin (DCO) for new inbound contributions, and Creative Commons Attribution 4.0 International License for Documentation. Projects under outside licenses may still submit for consideration, subject to review/approval of the TAC and Board.	Apache 2.0
Upon acceptance, At Large projects must list their status prominently on website/readme	https://www.emqx.io/products/kuiper * New stage statement will be added on this official website once the proposal is accepted.

Taxonomy Data:

Functions (Provide, Consume, Facilitate, or N/A; Add context as needed)

Functions	(Provide, Consume, Facilitate, or N/A; Add context as needed)
APIs	Provide
Cloud Connectivity	Provide
Container Runtime & Orchestration	Provide
Data Governance	N/A

Data Models	Provide
Device Connectivity	Consume
Filters/Pre-processing	Provide
Logging	Provide
Management UI	Provide, not open sourced
Messaging & Events	Consume
Notifications & Alerts	Consume
Security	Consume
Storage	Facilitate

Deployment & Industry Verticals (Support, Possible, N/A; Add context as needed)

Deployment Type	(Support, Possible, N/A; Add context as needed)
Customer Devices (Edge Nodes)	Support
Customer Premises (DC and Edge Gateways)	Support
Telco Network Edge (MEC and Far-MEC)	Support
Telco CO & Regional	Support
Cloud Edge & CDNs	Support
Public Cloud	Support
Private Cloud	Support

Deployment & Industry Verticals (or X; Add context as needed)

Directly applicable Industry/Verticals use cases	(or X; Add context as needed)
Automotive / Connected Car	
Chemicals	
Facilities / Building automation	
Consumer	
Manufacturing	
Metal & Mining	
Oil & Gas	
Pharma	
Health Care	

Power & Utilities	
Pulp & Paper	
Telco Operators	
Telco/Communications Service Provider (Network Equipment Provider)	
Transportation (asset tracking)	
Supply Chain	
Preventative Maintenance	
Water Utilities	
Security / Surveillance	
Retail / Commerce (physical point of sale with customers)	
Other - Please add if not listed above (please notify TAC-subgroup@lists.lfedge.org when you add one)	N/A

Deployments (static v dynamic, connectivity, physical placement) - (or X; Add context as needed)

Use Cases	**(or X; Add context as needed)**
Gateways (to Cloud, to other placements)	
NFV Infrastructure	
Stationary during their entire usable life / Fixed placement edge constellations / Assume you always have connectivity and you don't need to store & forward.	
Stationary during active periods, but nomadic between activations (e.g., fixed access) / Not always assumed to have connectivity. Don't expect to store & forward.	
Mobile within a constrained and well-defined space (e.g., in a factory) / Expect to have intermittent connectivity and store & forward.	
Fully mobile (To include: Wearables and Connected Vehicles) / Bursts of connectivity and always store & forward.	

Compute Stack Layers (architecture classification) - (Provide, Require, or N/A; Add context as needed)

Compute Stack Layers	**(Provide, Require, or N/A; Add context as needed)**
APIs	Provide
Applications	Provide
Firmware	N/A
Hardware	N/A
Orchestration	N/A
OS	N/A

VM/Containers	Provide

Cloud Stack Layers (architecture classification) - (Provide, Require, or N/A; Add context as needed)

Cloud Stack Layers	**(Provide, Require, or N/A; Add context as needed)**
Applications	Provide
Configuration (drive)	Provide
Content (management system)	Provide
IaaS	N/A
PaaS	N/A
Physical Infrastructure	N/A
SaaS	N/A