Kubernetes for IoT: Seamless Management, Scaling and Application Deployment in IoT-based Environments

George Kornaros, Othon Tomoutzoglou, Dimitris Bakoyiannis, Miltos Grammatikakis and Marcello Coppola

Intelligent Systems and Computer Architecture Lab – ISCA

Hellenic Mediterranean University, Heraklion, Crete, GR

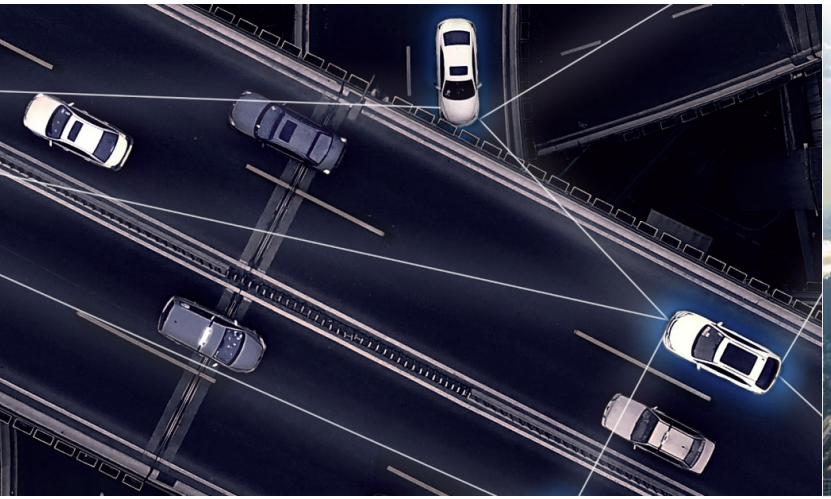
2025 International Symposium on Ambient Intelligence and Embedded Systems, AmiES 2025,

Hamburg, September 24-27, 2025

Outline

- Challenges and objectives
- Services security challenges
- Cloud-Edge-IoT technology overview
- FLUIDOS – KubeEdge integration
- Demonstration
- Conclusion

Billion of Embedded Devices



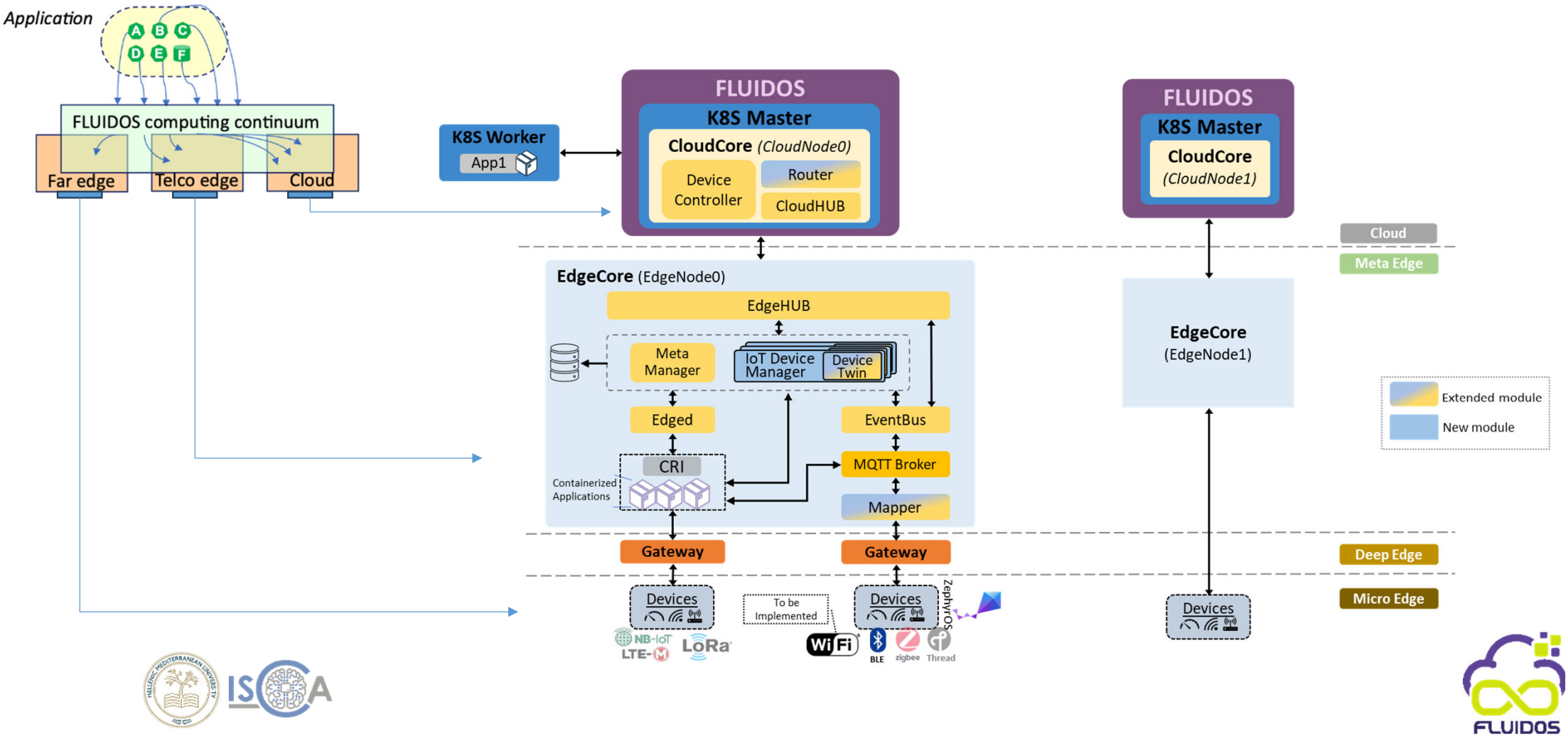

ADAS & Autonomous Vehicles, Cellular-V2X, connected cars

Secure anchors for Mobiles, Industrial, Communications, IoT & Edge nodes

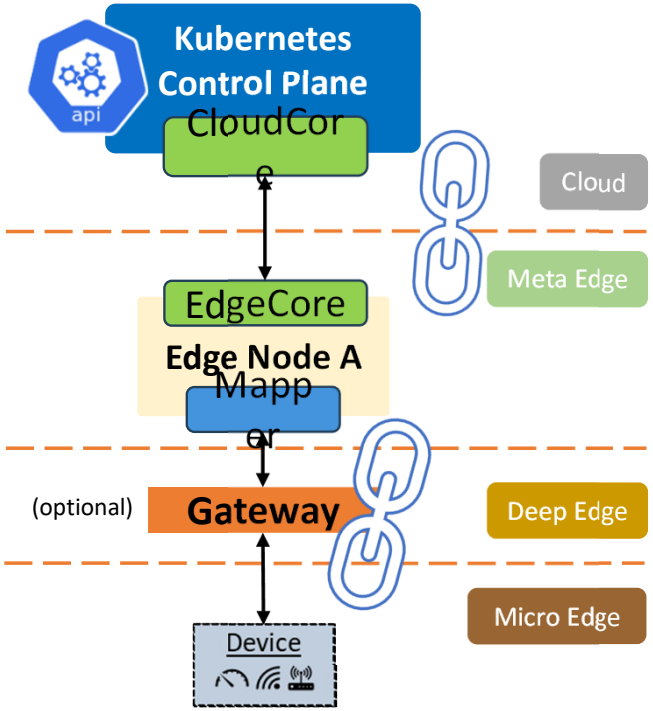
Industrial & IoT Communication with IoT devices
- Trust provisioning
- Secure element communication

Digital signatures
- Secure boot, Industrial & IoT Firmware integrity for IoT devices
- Over-the-air updates, Firmware authentication, smart car access

FLUIDOS: Edge Support



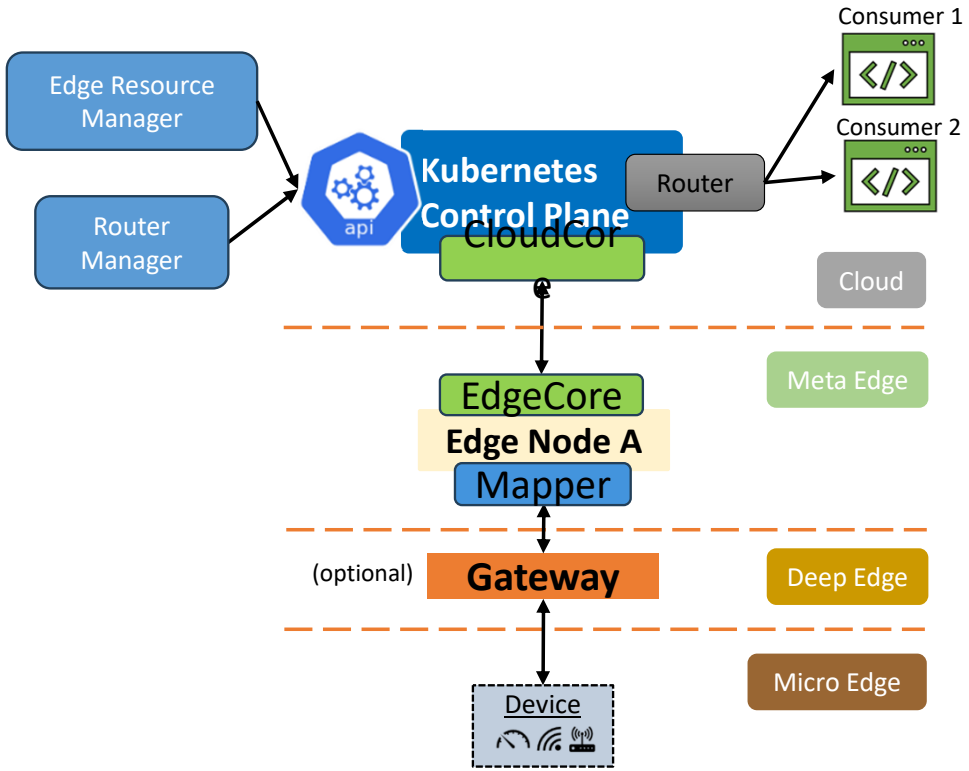
FLUIDOS Edge: Prerequisites



1. Kubernetes Control Plane
2. Edge Worker Node
3. Fluidos-Edge Components
4. Register Leaf Edge **Device/s**
5. Deploy the Mapper/s

Digital Twin

FLUIDOS Edge: Node modules



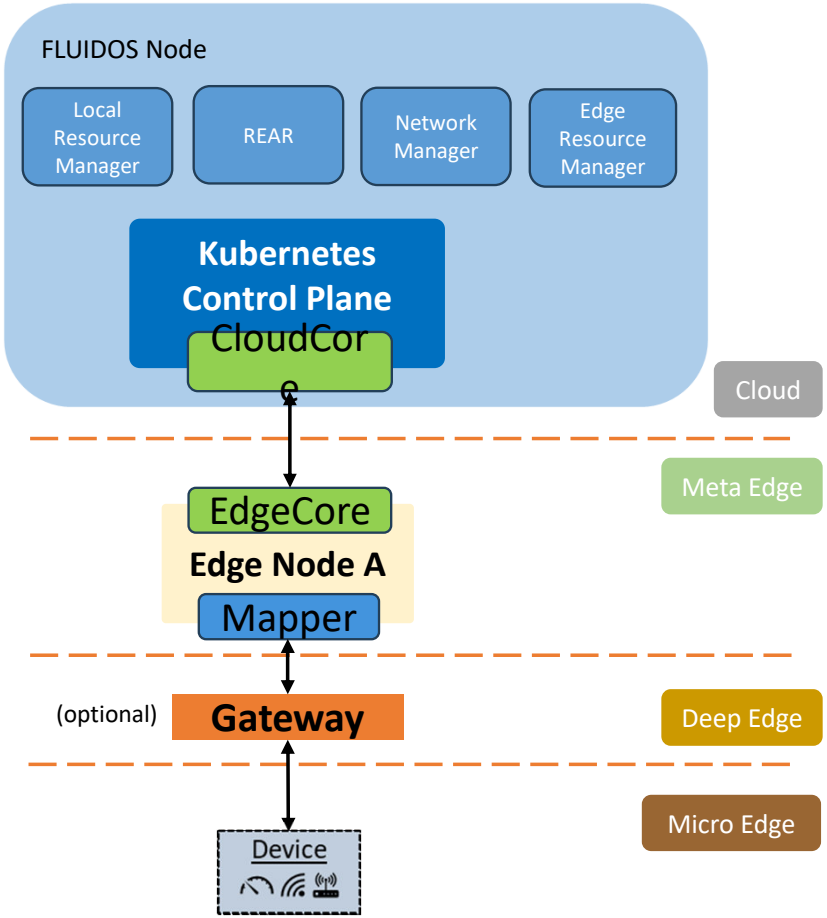
Edge Resource Manager

- Discovers Leaf Edge Devices (LEDs) and Sersors
- Creates Sensor Flavors
- Is as a K8S controller

Router Manager

- Configures the router via the API server, i.e., define sensor data data-paths
- Will be integrated to an existing REAR component?
  OR
  Will be implemented as a separate module?

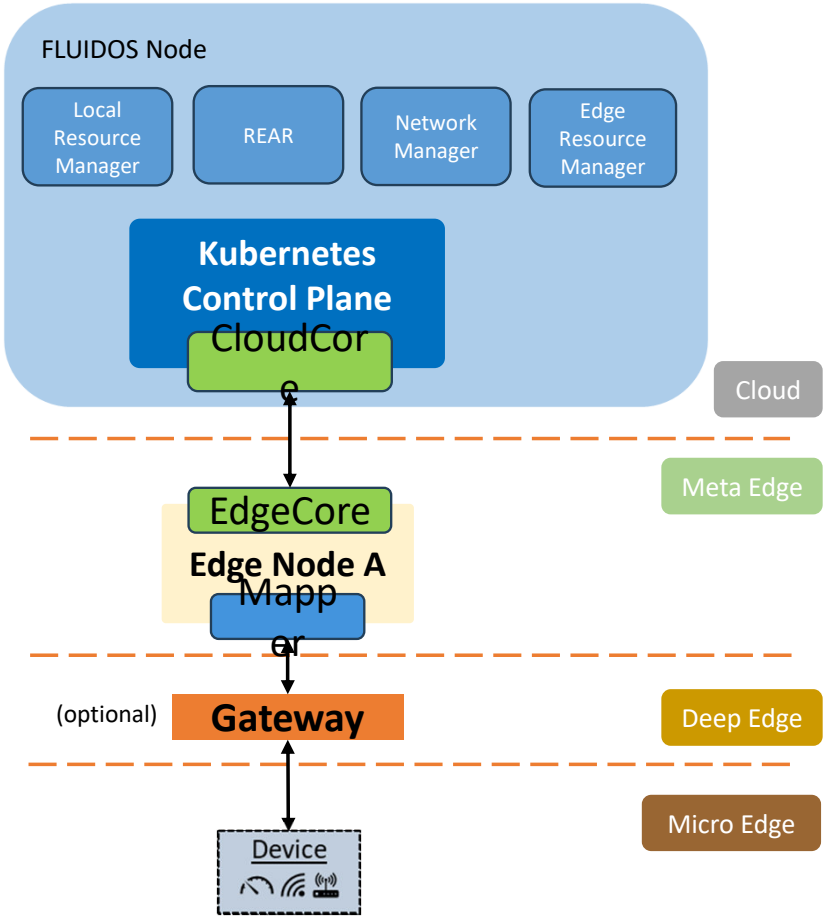
FLUIDOS Edge: Edge Resource Discovery



Edge Resource Manager (ERM)

1. Discover **Edge Worker Nodes** labeled with: *node-role.fluidos.eu/sensors=true*
2. Discover Leaf Edge **Device/s** attached to each Edge Worker Node
3. Get LED information
4. Decode LED sensors information
5. Create Sensor type of Flavor

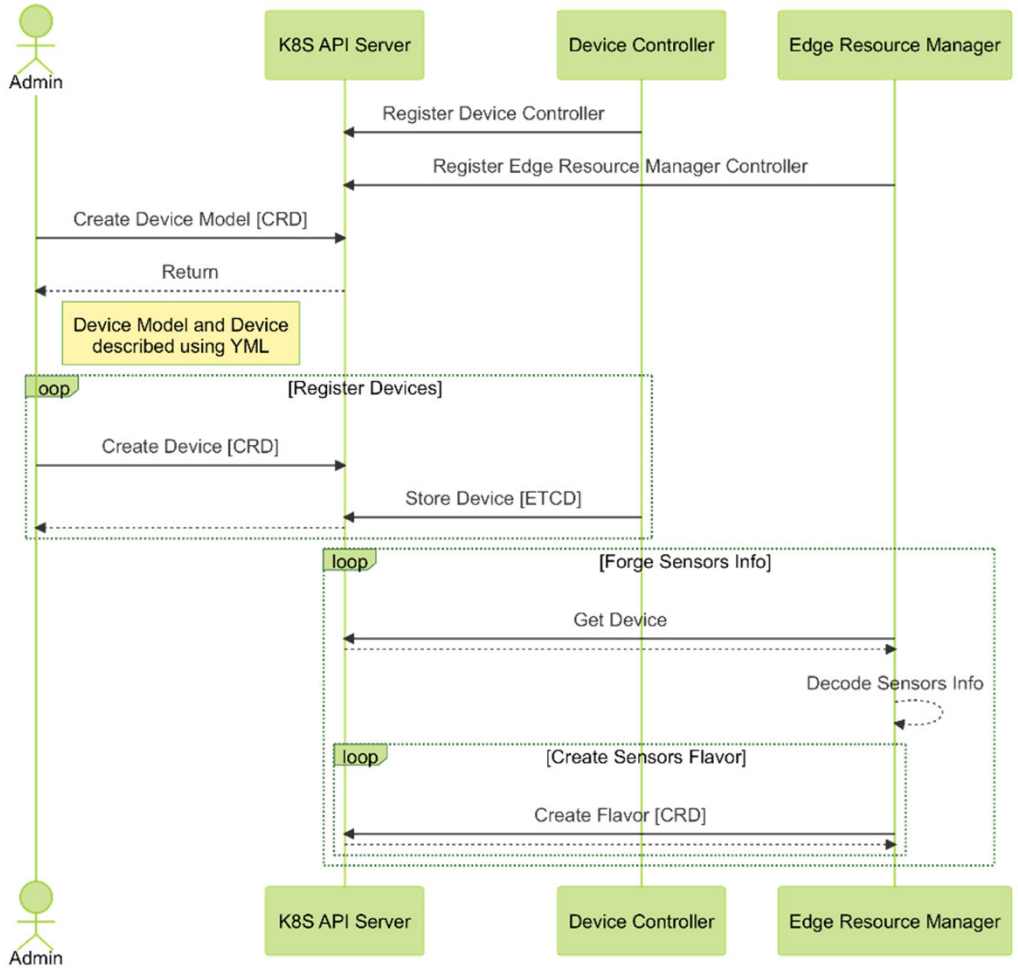
FLUIDOS Edge: Edge Resource Discovery (cont.)



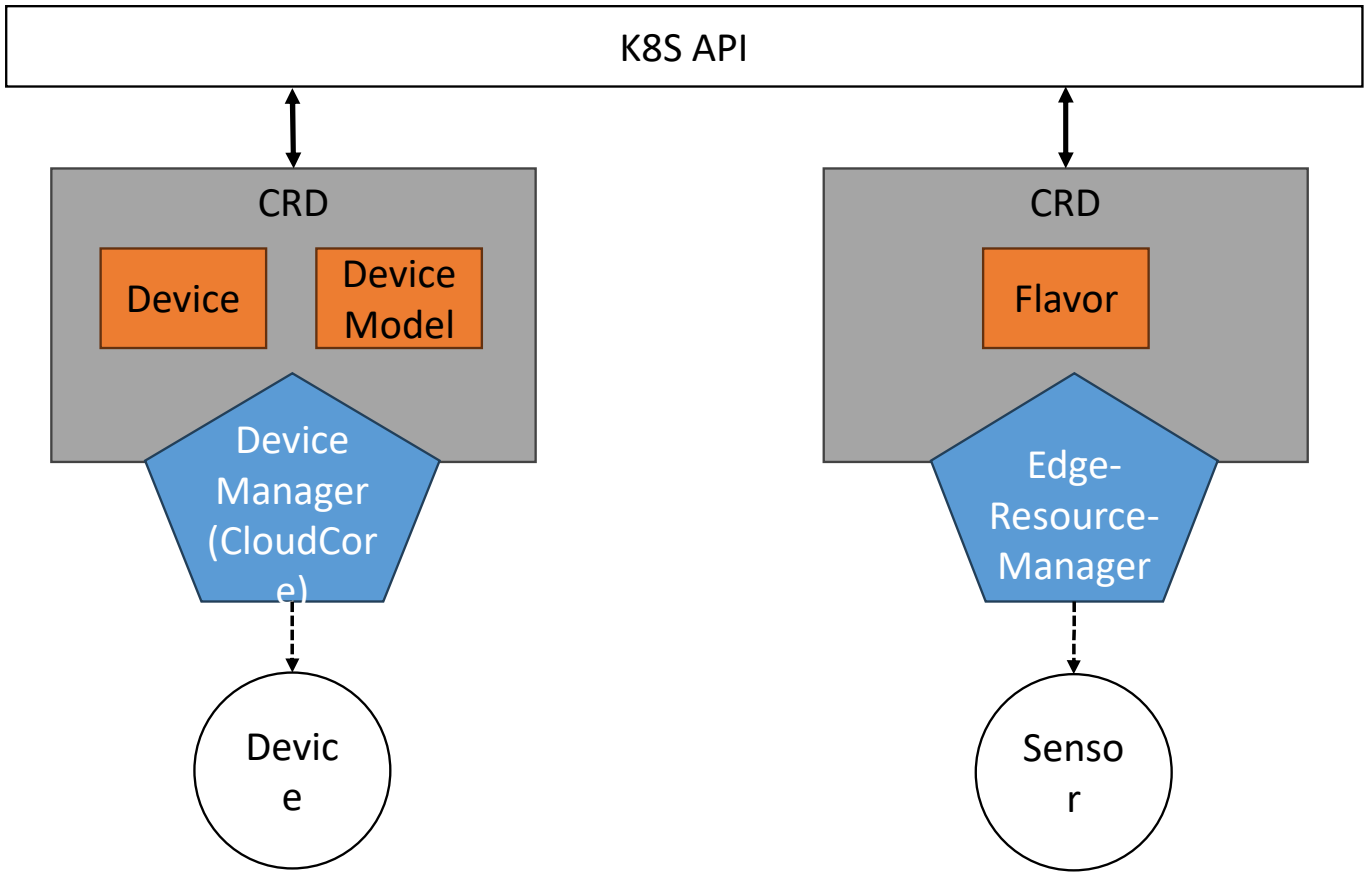
Edge Resource Manager (ERM)

1. Construct Flavor based on:
   - Edge Worker Node
   - LED (@Device CRD)
   - LED sensors (base64 encoded JSON format description @Device CRD annotation field)
2. FlavorType -> Sensor, as described at REAR v0.2 document (some adaptations had to be done): https://docs.google.com/document/d/1Qb2goqjyhBvwnewKMnZQSn3ZUVWurAXZ9vbhJ8wPwrQ

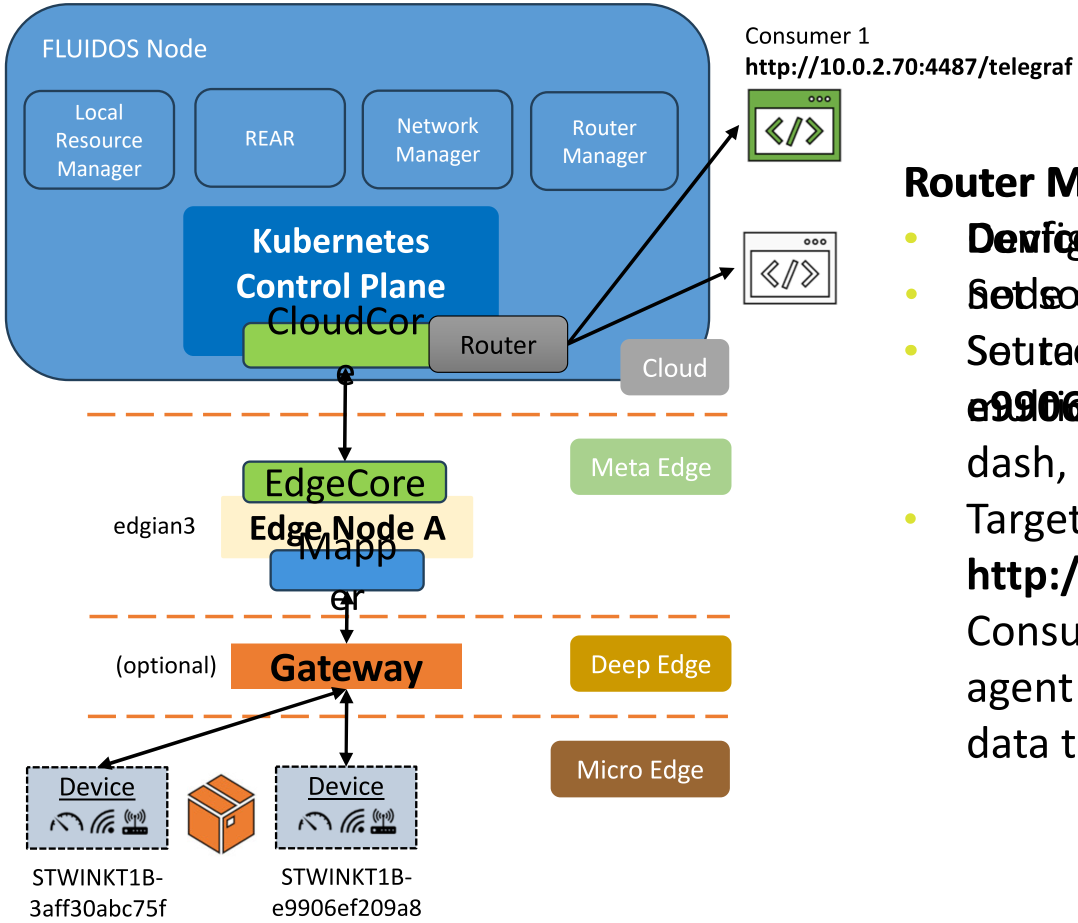
Sensor flavor creation: Sequence diagram



FLUIDOS Edge: CRDs



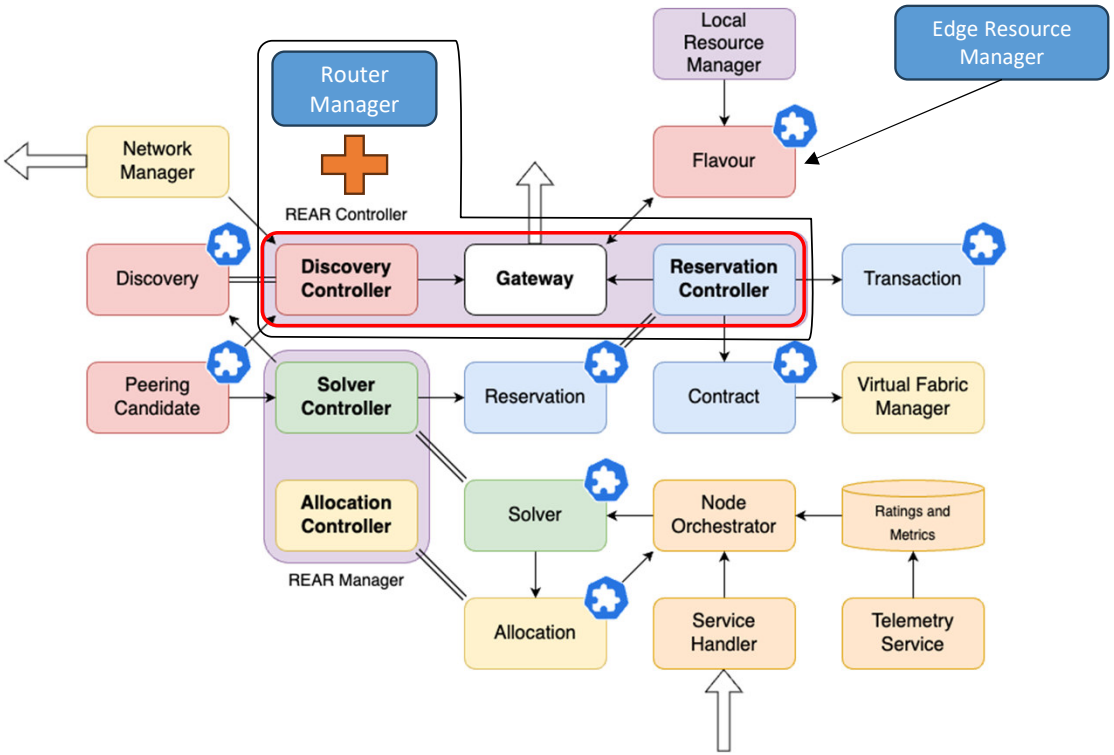
FLUIDOS Edge: Data paths



Router Manager (RM) / **Example**

- **Device** is attached to a Meta Edge worker node called "**edgian3**"
- Source resource is "**STWINKT1B-e9906ef209a8**" (implemented as), composed by device name, dash, and the 96-bit ST unique identifier
- Target resource is **http://10.0.2.70:4487/telegraf**, i.e., Consumer's 1 endpoint where InfuxDB Telegraf agent serves HTTP requests by storing received data to the database

FLUIDOS Edge: Integration Progress



Tested alongside a fully deployed FLUIDOS Node v0.1.1

- Edge Resource Manager as a K8S controller that:
  - Discovers Edge Worker Nodes
  - Fetches LED Information
- Router Manager
  - As a standalone application that writes or deletes Route rules based on command-line arguments

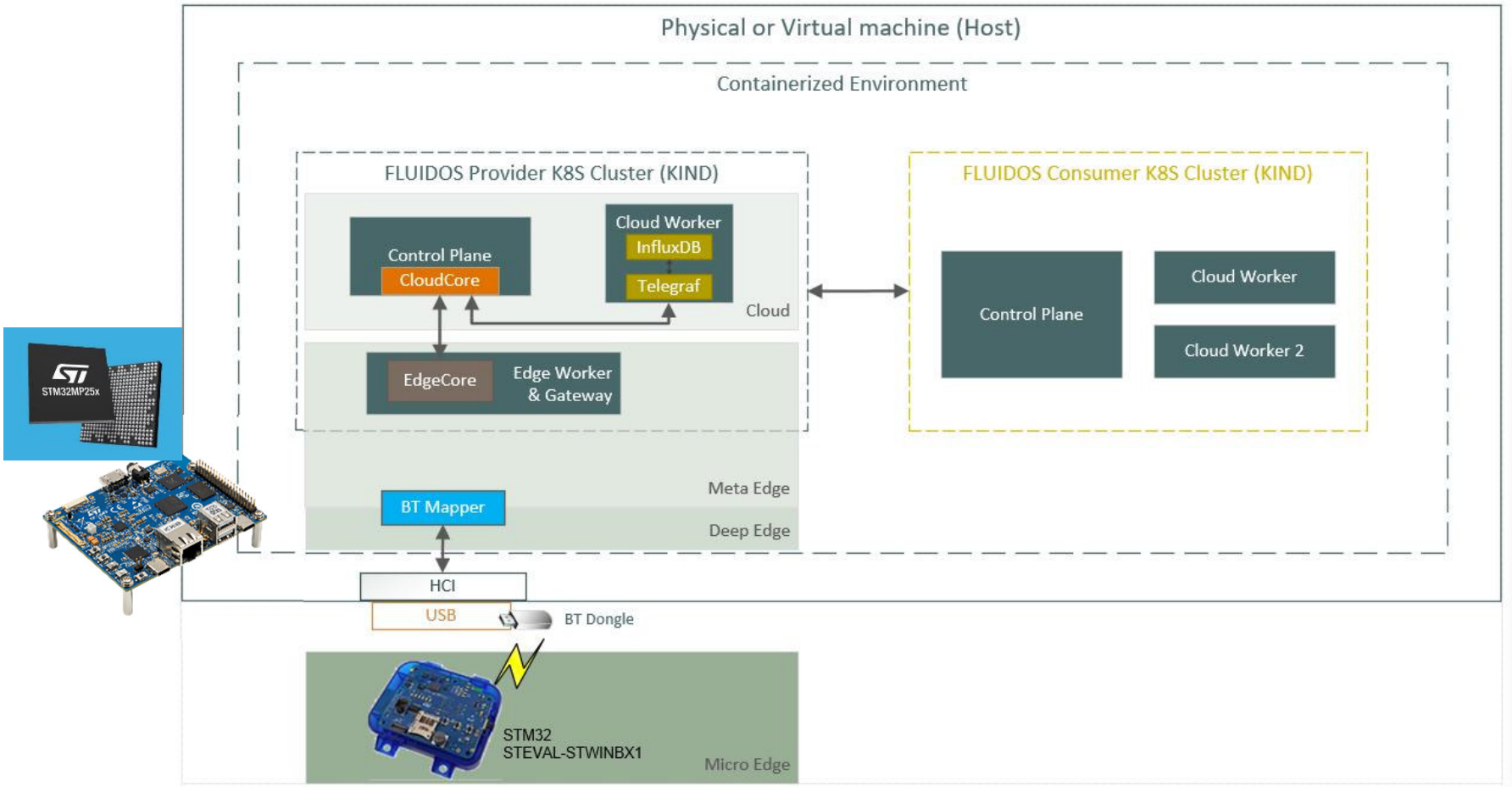
Work in progress

- Edge Resource Manager create ‘Sensor’ type Flavors

TBD

- Router Manager will properly configure the Router based on information provided by:
  - REAR Controller ?
  - REAR Manager ?
  - A Custom Resource ?

FLUIDOS Edge Testbed



A fully automated process creates a FLUIDOS Nodes testbed, enabled for Edge-IoT, using KIND as the underlying platform.

Based on the latest available scripts (tools/scripts).

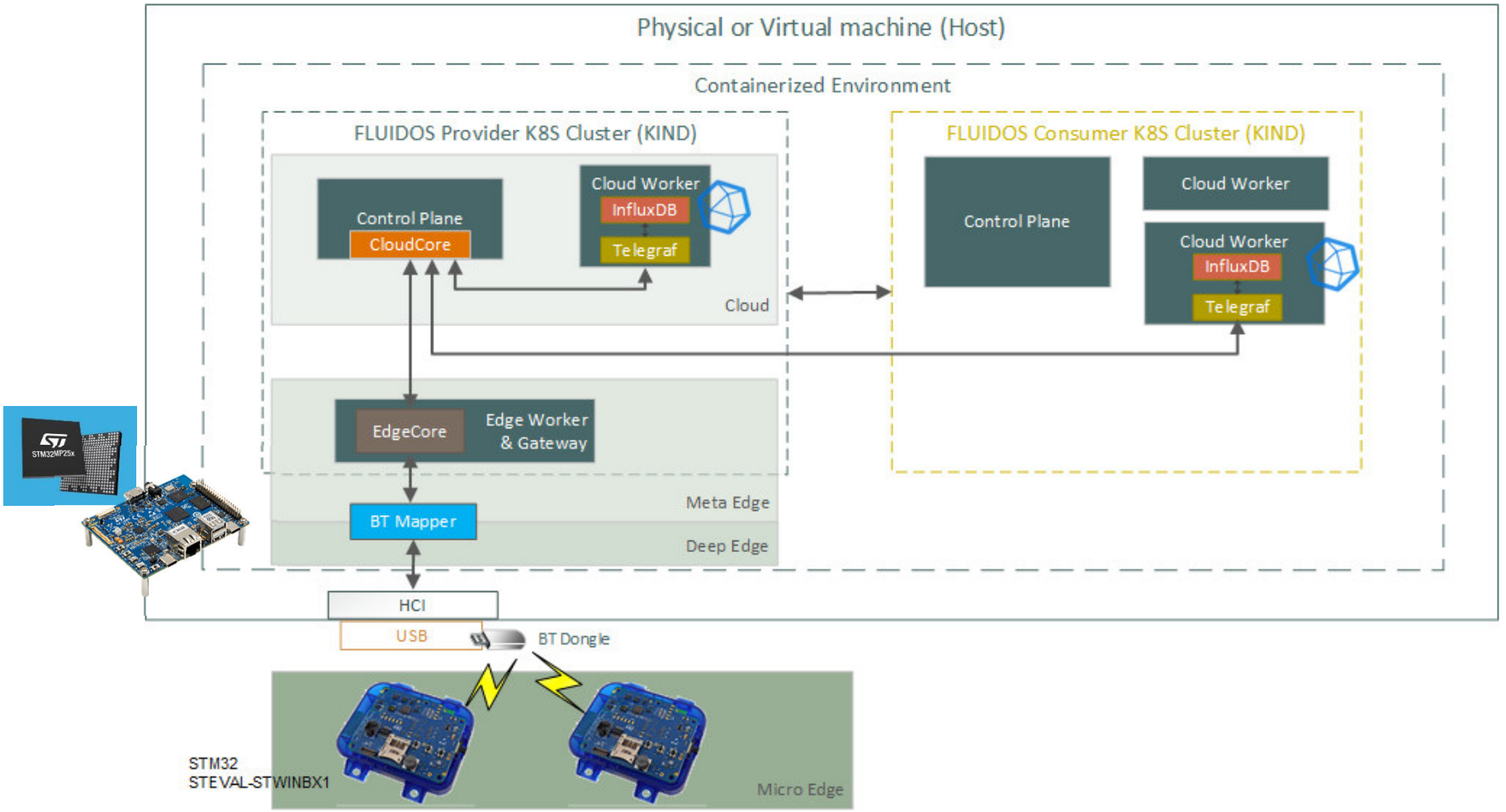
FLUIDOS Edge: System Availability Overview

- The code is currently available as a Fluidos Node **fork**: https://github.com/otto-tom/fluidos-node-edge
  Branch: features/edge-support
- To be merged with the official Fluidos Node
- Key points:
  - “Edge enabled” testbed
  - yed as

```
$kubectl get pods -n fluidos
NAME                                          READY   STATUS
node-edge-resource-manager-f4cdf57bd-5xwrk    1/1     Running
node-local-resource-manager-74687dd8f-tqksb   1/1     Running
node-network-manager-p6wrk                    1/1     Running
node-rear-controller-7f64449b6-n5wpx          1/1     Running
node-rear-manager-5cf797bdb6-zwblx            1/1     Running
```

```
$fluidos-node-edge/tools/scripts/setup.sh
-------------------------------------------------
      WELCOME TO THE FLUIDOS NODE INSTALLER
-------------------------------------------------
We'll now run the installation process for the FLUIDOS node.
What type of environment do you want to use? /
1. Use demo KIND environment (one consumer and one provider) /
2. Use a custom KIND environment with n consumer and m provides /
Please enter the number of the option you want to use:
 1
Do you want to use local repositories? [y/n] y
Do you want to enable resource auto discovery? [y/n] n
Do you want to enable LAN node discovery? [y/n] n
Do you want to enable edge worker nodes support? [y/N]: y_
```

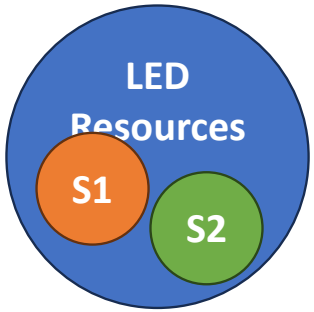
FLUIDOS Edge: Final Demo Testbed Overview

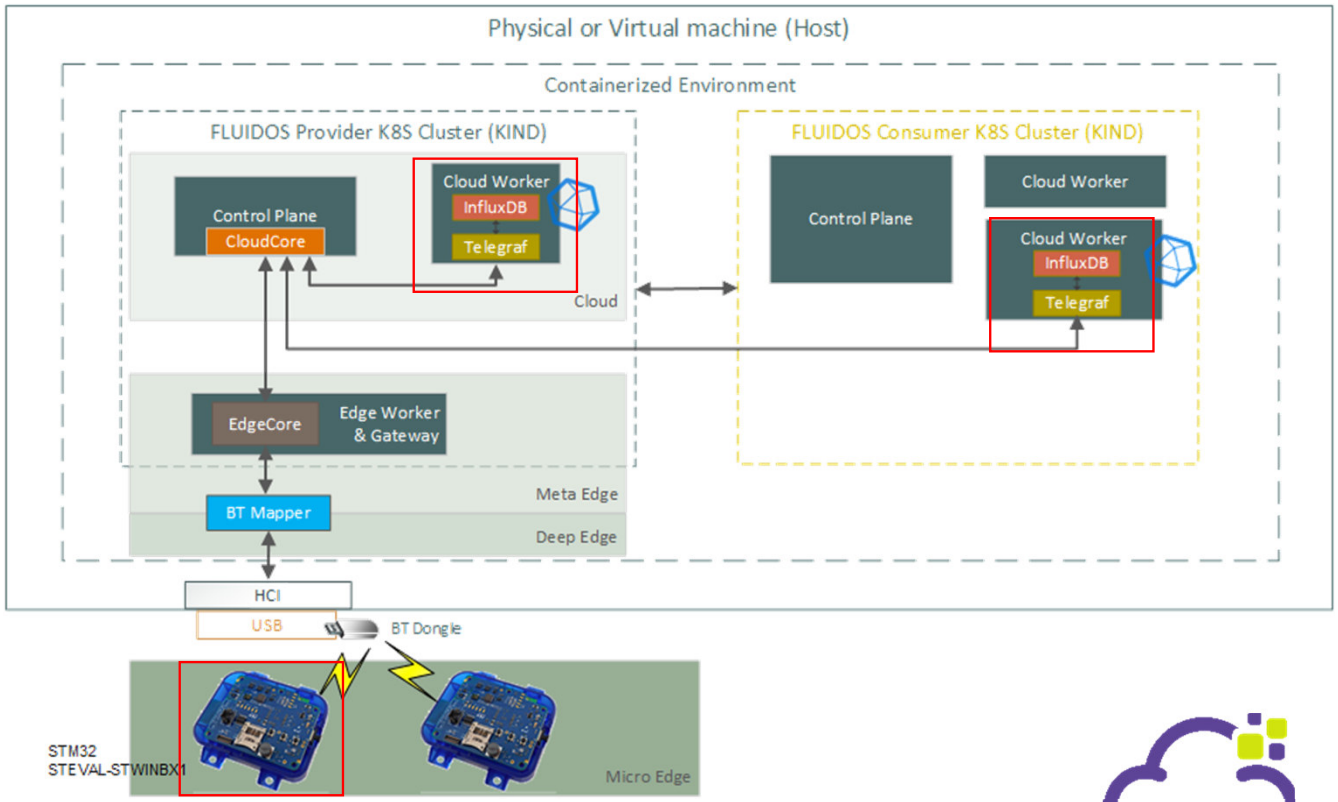


FLUIDOS Edge: Final Demo Scenario 1

Resource sharing/Data Flow Partitioning

- One consumer receives sensor data at many endpoints.
- Many consumers receive data from the whole set or subset of the available sensors (resources) coming from LEDs.
  - E.g., Consumer1 and Consumer2 receives the same subset S1 (**multicast**). Consumer2 also receives another subset S3.

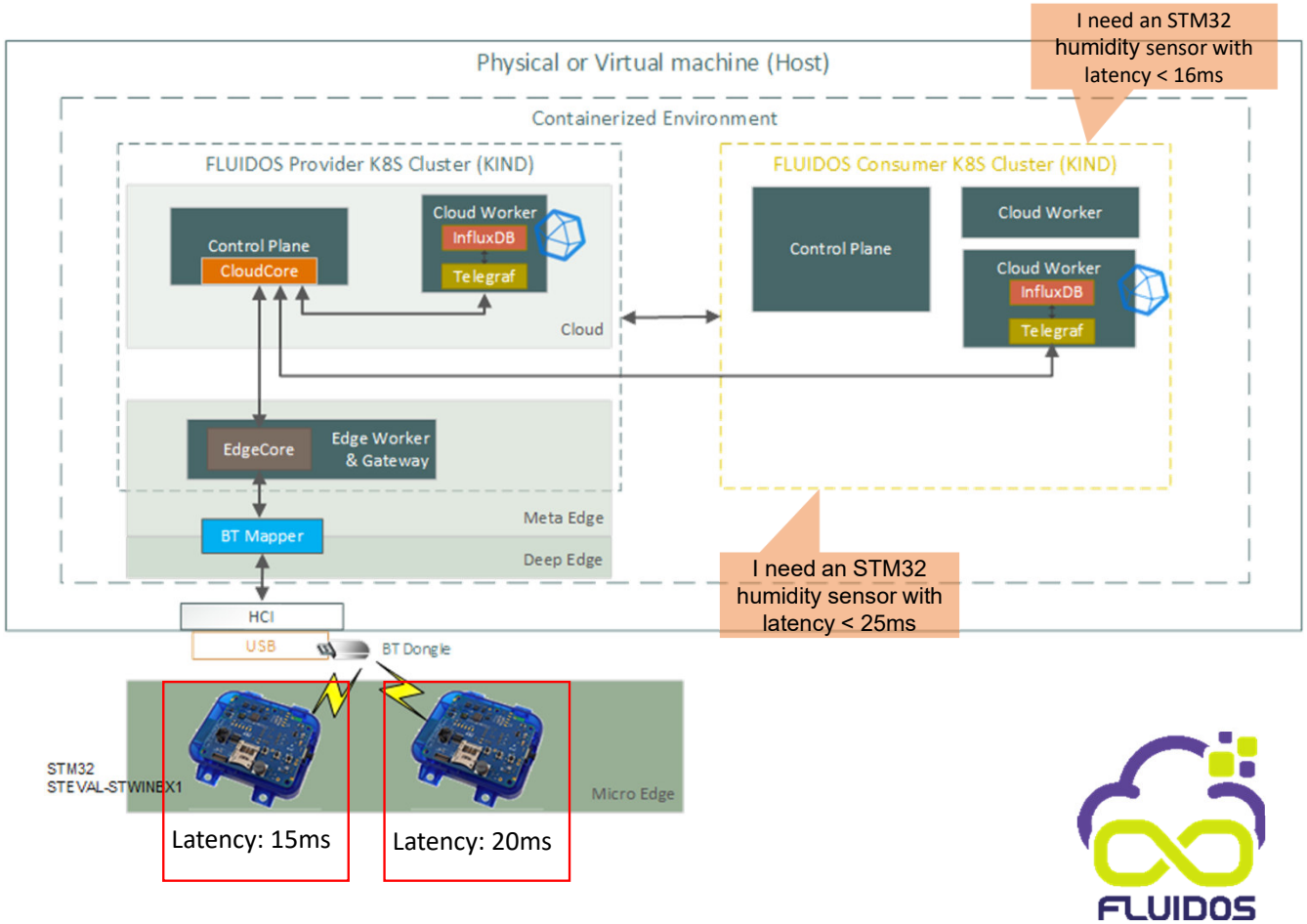




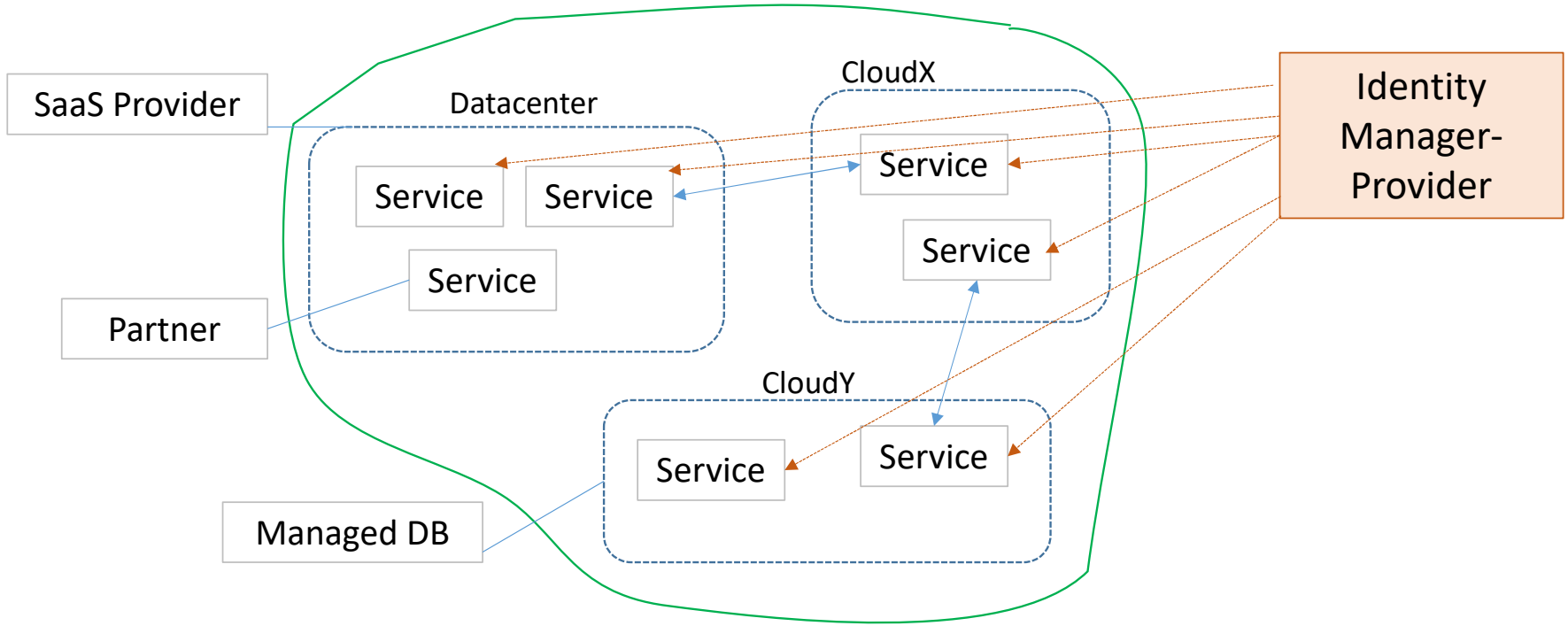
FLUIDOS Edge: Final Demo Scenario 2

Resource filtering

- Filter sensor based on predefined filters, e.g., :
  - Latency
  - Consumption
  - Accuracy
  - Manufacturer
  - Etc.
- Example:
  - A consumer requests an STM32 humidity sensor with latency < **16ms**
  - A consumer requests an STM32 humidity sensor with latency < **25ms**



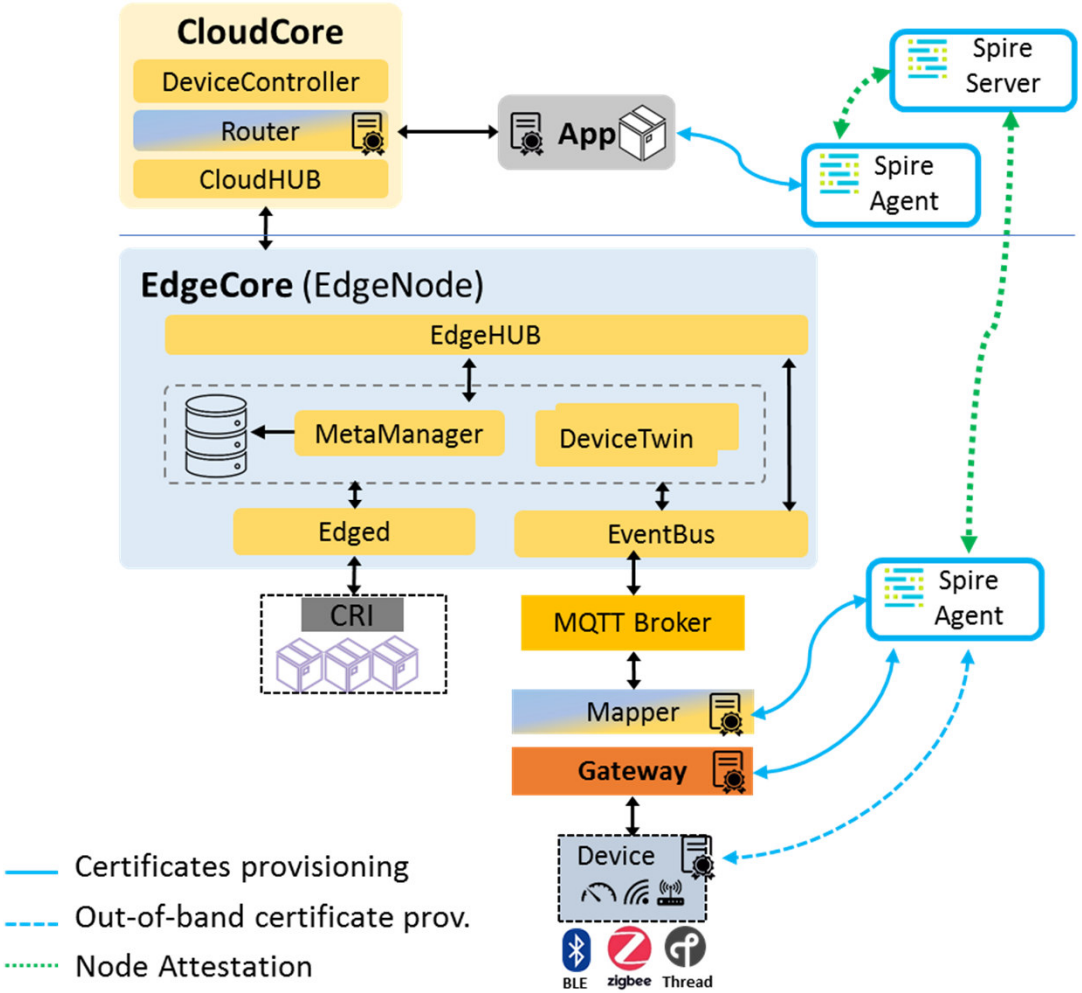
SPIFFE Ensures Zero-Trust Security



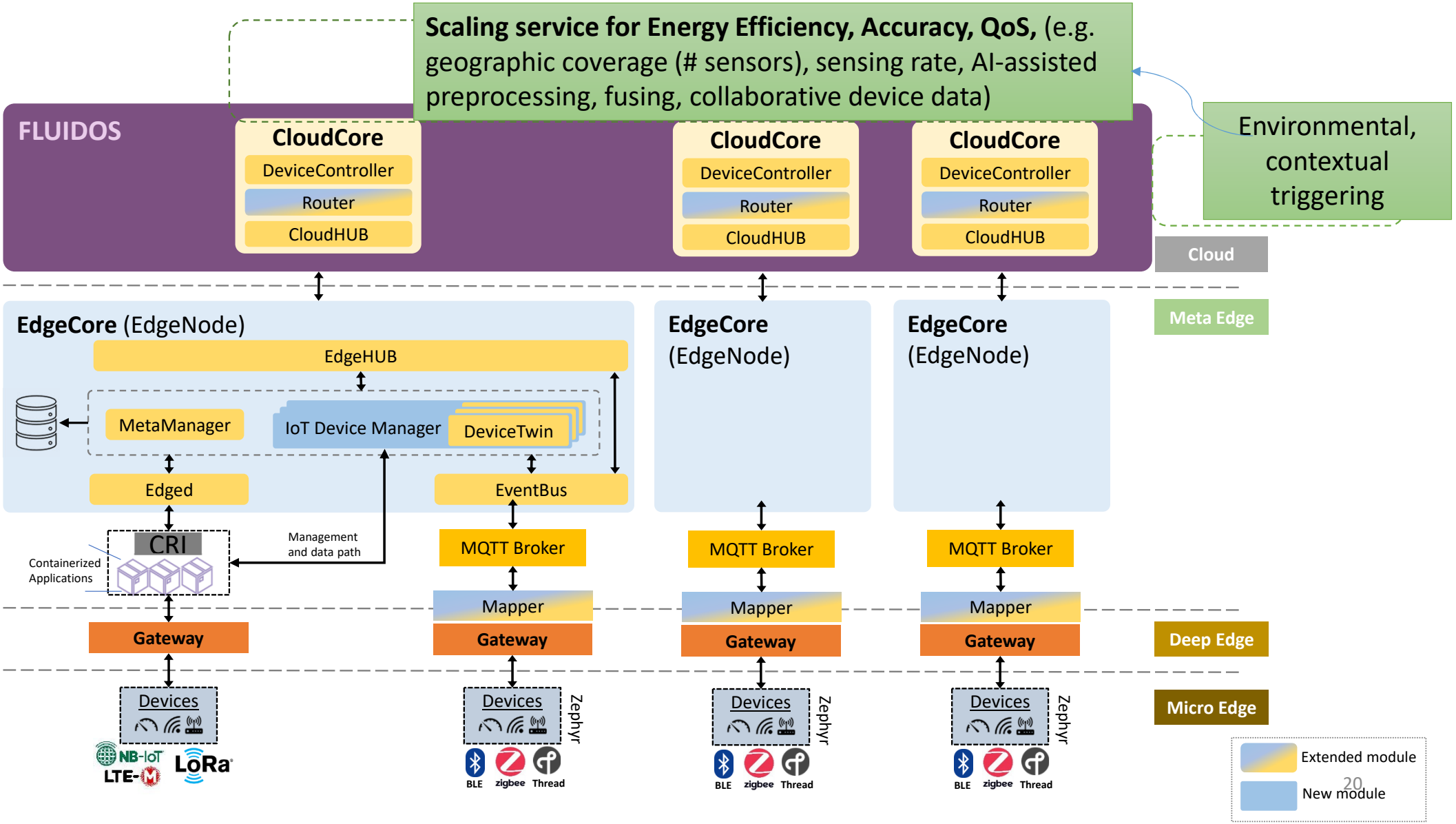
Guarantee each workload/service will get its own SERVICE identity

- unique
- secure
- provable

SPIFFE in IoT-Edge Architecture (FLUIDOS)



- **Cloudcore-EdgeCore** monitoring resource changes
  - Metadata traffic includes edge node status and application status
- Extended Kubernetes **Custom Resource Definitions** (CRDs) to manage sensor data traffic


Scaling service for Energy Efficiency, Accuracy, QoS, (e.g. geographic coverage (# sensors), sensing rate, AI-assisted preprocessing, fusing, collaborative device data)
FLUIDOS
CloudCore
DeviceController
Router
CloudHUB
CloudCore
DeviceController
Router
CloudHUB
CloudCore
DeviceController
Router
CloudHUB
Environmental, contextual triggering
Cloud
Meta Edge
EdgeCore (EdgeNode)
EdgeHUB
MetaManager
IoT Device Manager
DeviceTwin
Edged
EventBus
EdgeCore (EdgeNode)
EdgeCore (EdgeNode)
CRI
Containerized Applications
Management and data path
MQTT Broker
MQTT Broker
MQTT Broker
Mapper
Mapper
Mapper
Gateway
Gateway
Gateway
Gateway
Deep Edge
Micro Edge
Devices
Devices
Devices
Devices
Zephyr
Zephyr
Zephyr
NB-IoT
LTE-M
LoRa
BLE
zigbee
Thread
BLE
zigbee
Thread
BLE
zigbee
Thread
Extended module
New module

Conclusions

- Sensor-generated traffic flows in a Kubernetes-managed cloud-edge setup within and across clusters
- Managing a large fleet of IoT devices can be challenging - Enhance management, scaling and efficient application deployment in IoT-based environment
- Kubernetes supports horizontal scaling: it can add more devices to a pool of resources, and vertical scaling: it increases the capacity of an existing setup, and high availability through its built-in failover mechanism

Thank you for your attention!

EU-H2020 FLUIDOS
https://www.fluidos.eu/

Intelligent Systems and Computer Architecture Lab
https://isca.hmu.gr

George Kornaros
[kornaros@hmu.gr]

Secure BLE-connected STWINBX1 device via SPIRE Certificates