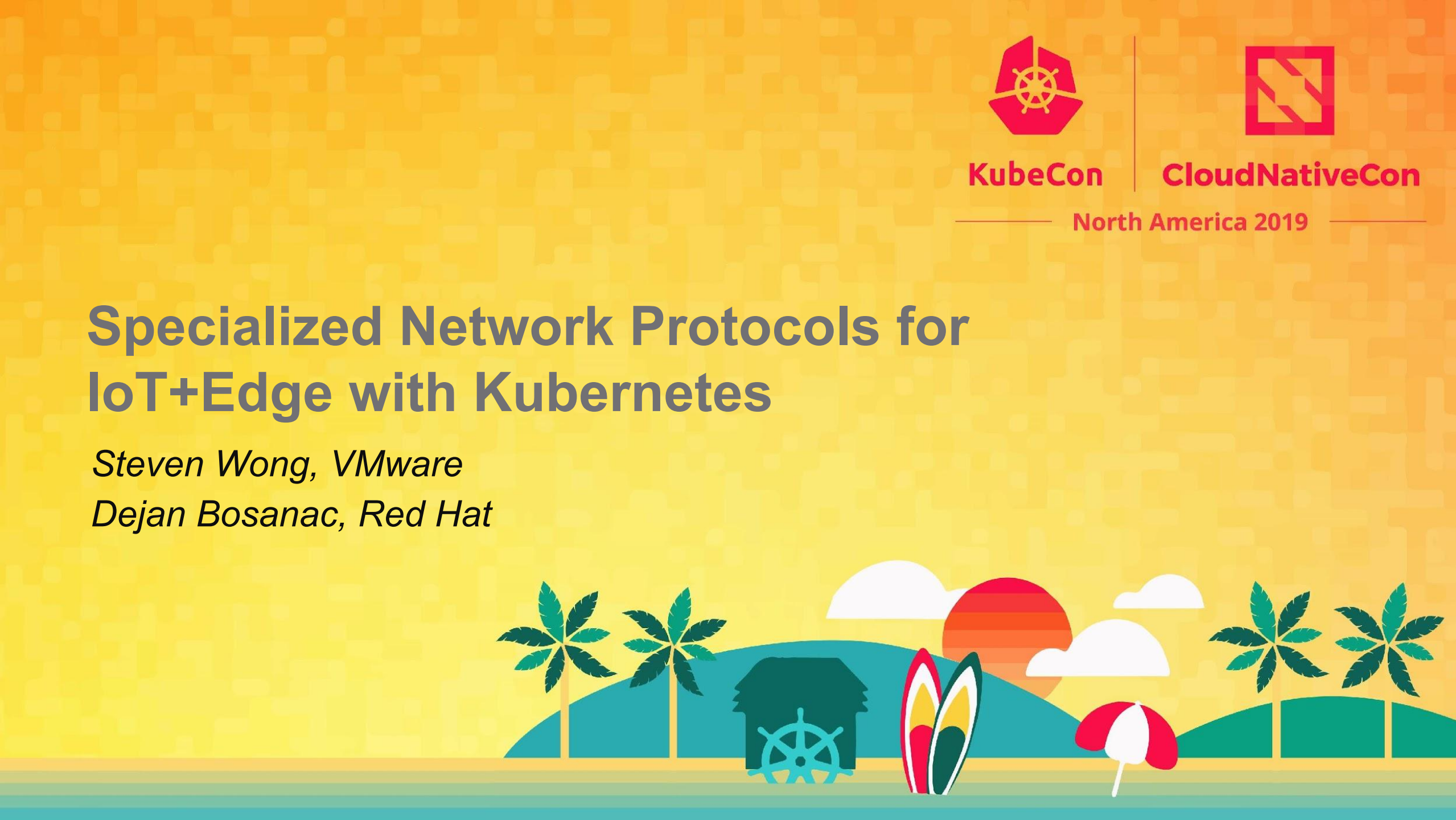



# Specialized Network Protocols for IoT+Edge with Kubernetes

*Steven Wong, VMware*

*Dejan Bosanac, Red Hat*

# Abstract



This session will survey communication protocols and technologies used in the edge and IoT space.

These use cases can call for specialized protocols and transports:
publish subscribe, multicast
protocols tolerant of intermittent connectivity
Protocols popular in industry verticals (vehicle bus, industrial automation, building automation)

In some cases, support exists now for use with Kubernetes. If not, device gateways and protocol converters might be an option.

Agenda (Intro):

- survey of protocols and transport standards for IoT and edge
- Intro to how a device gateway or protocol converter works
- Intro to extending Kubernetes with CRDs to manage new device types

Agenda (Deep Dive):

- Futures: Could the service mesh concept be extended beyond TCP, HTTP(s)?
- Demonstration: Kubernetes management of an edge application using a specialized protocol
- Demonstration: Use a device gateway with Kubernetes

# Speakers



Dejan Bosanac
Red Hat
@dejanb

Steve Wong
VMware
@cantbewong

# Agenda

Part 1: Intro

- survey of protocols and transport standards for IoT and edge
- device gateways & protocol converter
- Intro to extending Kubernetes with CRDs to manage new device types

Intermission (5 minutes): Meet others, “birds of a feather”

Part 2: Deep Dive

- Futures: Could the service mesh concept be extended beyond TCP, HTTP(s)?
- Demonstration: Kubernetes management of an edge application using a specialized protocol
- Demonstration: Use a device gateway with Kubernetes

How to get involved with the IoT Edge Working Group

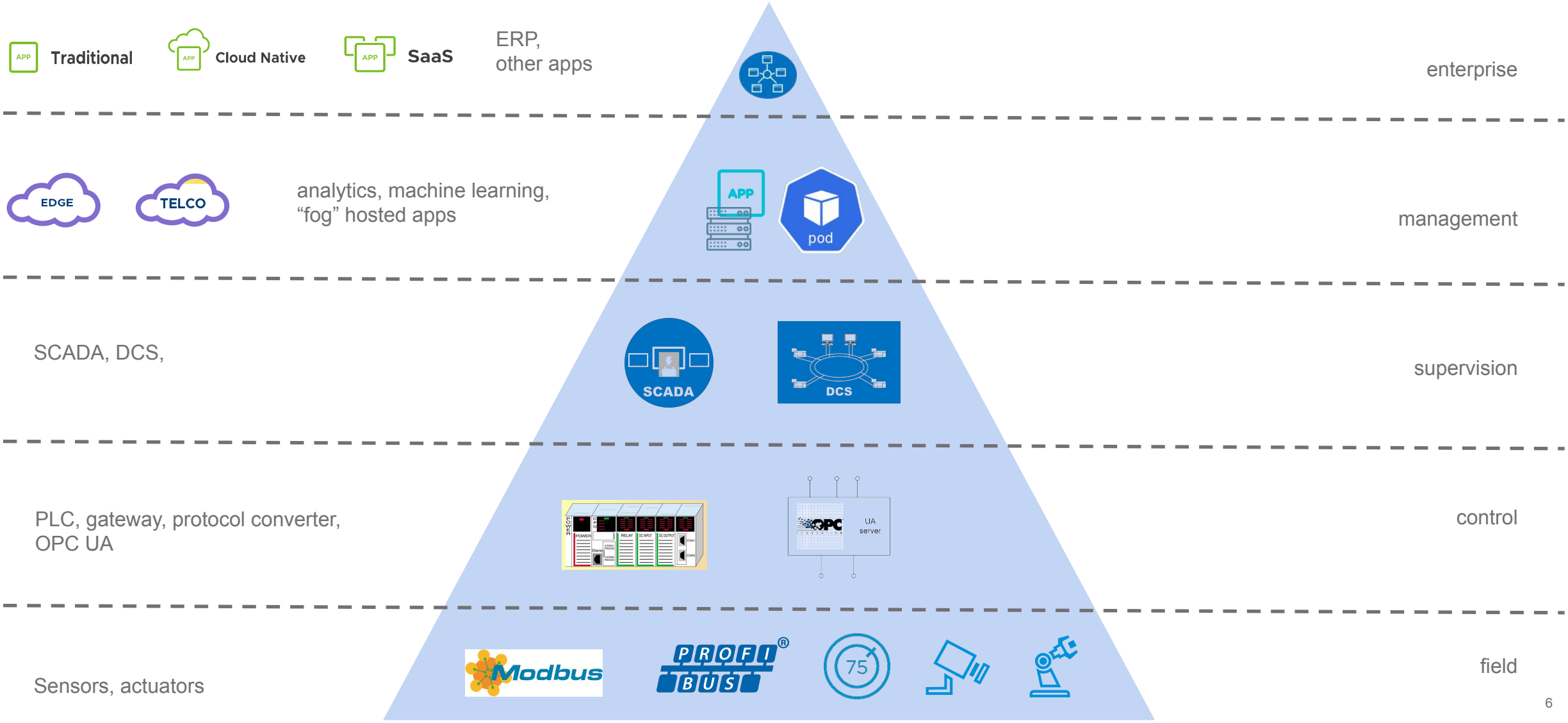
# Intro

Survey of protocols and transport standards for IoT and edge

# Applications are multi-level - maybe at multiple locations

## Tiers of interconnects support this, often using specialized protocols





# Data communication operates at multiple levels

## Not unusual to have all these going on simultaneously



May be multiple layers of protocols (OSI model)

**Hardware / physical media**: examples: ethernet, EIA-485 multidrop, EIA-232, etc.

**Data Link, Network, Transport**– example MAC, IP, TCP, UDP

**Application**– example http, domain specific APIs on http

data and control may be separated, data may be republished in a transformed context

**Data Plane – low level "raw" data flows**

**Control Plane**

- **Onboard devices or services**
- **Monitor**
- **Manage**
- **Secure**

**Content Plane – data transformed to a different context (ETL)**

# Protocols

## Evaluation considerations



**Type**: built in support for specialized applications

Sync req-response, pub-sub, both, or higher level data exchange agnostic that abstracts low level transport. Low level transport may be connection based or RPC style

**Implementation resource demands** – suitability for constrained environments? is a broken/router needed?

**Behavior with lossy/unreliable networks** – latency limits, QoS support, order & delivery guarantees

**Security**

**Topology**: point to point, bus, routable

**Support for discovery of nodes and data content**

**Health of community** – standards and certification org, scope and openness of software and hardware platforms, popularity within an application domain

**Stability and maturity**

# Protocols

## General Purpose



| protocol | standard | info | req-reply | pub-sub | QoS | |
|---|---|---|---|---|---|---|
| HTTP | IETF | WikipediA | ✔ | | no | |
| HTTP/2 | IETF | WikipediA | ✔ | ✔ | no | |
| AMQP | OASIS | WikipediA | ✔ | ✔ | yes | amqp.org |
| MQTT | OASIS | WikipediA | | ✔ | yes | mqtt.org |
| COAP | IETF | WikipediA | ✔ | ✔ | yes | coap.technology/ |
| DDS | OMG | WikipediA | | ✔ | yes | |
| Kafka | | WikipediA | | ✔ | no | kafka.apache.org |

# Protocols

## Higher Level



| protocol | standard | info | transport |
|---|---|---|---|
| LWM2M | OMA | WikipediA | COAP, on UDP or SMS |
| OGC SensorThings API | OGC | WikipediA | CoAP, MQTT, HTTP, 6LowPAN |
| PPMP | eclipse | WikipediA | |
| One2M | one2m.org | WikipediA | |

# Specialized Protocols

## Industrial, Building Automation



| protocol | standard | info |
|---|---|---|
| Modbus | Modbus Organization | WikipediA |
| BACnet | bacnet.org | WikipediA |
| OPC UA | OPC Foundation | WikipediA |

# Specialized Protocols

## Transportation (Automotive, Avionics, Rail)



| protocol | standard | info |
|---|---|---|
| A²B | proprietary | link |
| AFDX | proprietary | WikipediA |
| ARINC 429 | ARINC | WikipediA |
| Byteflight | byteflight | WikipediA |
| CAN | ISO | WikipediA |
| D2B | IEC 61030 | WikipediA |
| IDB-1394 | IEEE, 1394 trade assoc | WikipediA |
| IEBus | proprietray | Wikipedia |
| I²C | proprietary | Wikipedia |
| ISO 9141-1/-2 | ISO | Wikipedia |

| protocol | standard | info |
|---|---|---|
| J1708,J1587 | SAE | WikipediA |
| J1850 | SAE | WikipediA |
| J1939, ISO 11783 | ISO | WikipediA |
| Keyword protocol 2000 | ISO | WikipediA |
| LIN | ISO | WikipediA |
| MOST | proprietary | WikipediA |
| Multifunction Vehicle Bus | IEC | WikipediA |
| SPI | defacto | WikipediA |
| VAN | proprietrary | WikipediA |

# Device Gateways and Protocols Converters

# Why use a Gateway / Protocol convertor

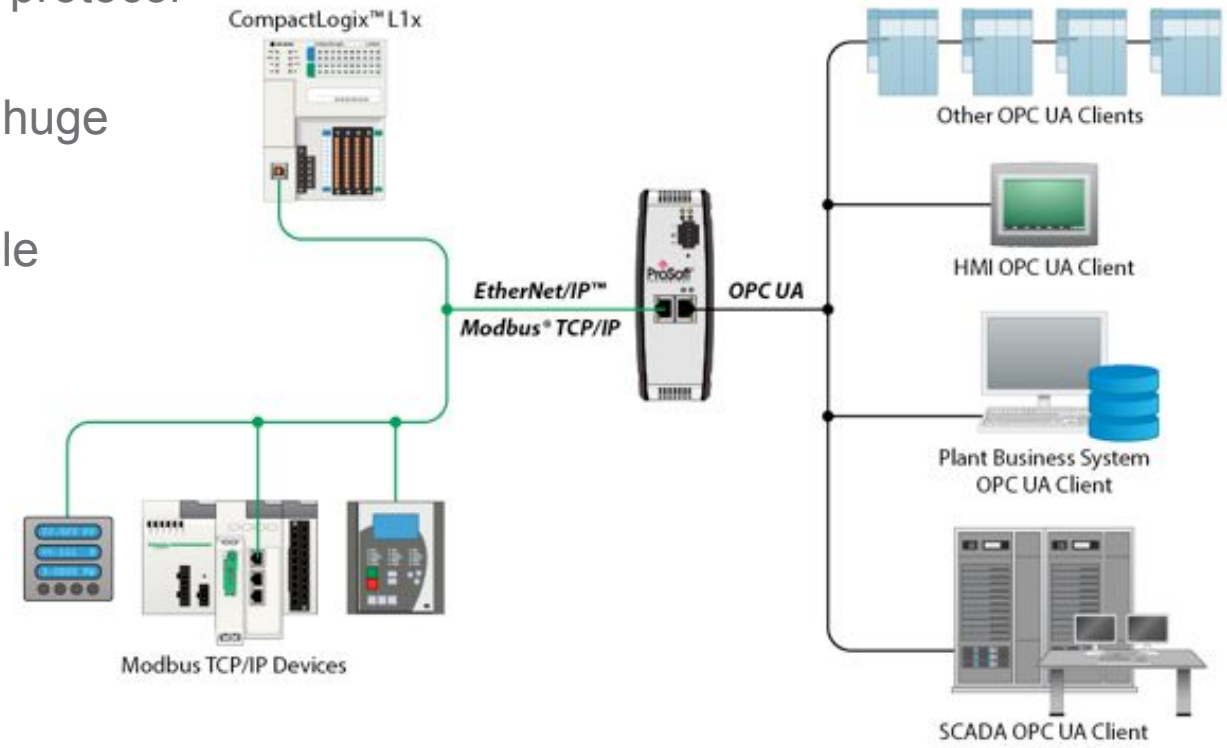


Convert point to point, and local bus protocols to a protocol that is routable

Allow efficient sharing of data sources across multiple consumers

- Low level devices often lack bandwidth and compute to respond to multiple consumers
- Higher tier consumers can be written to utilize just one protocol –
  - better than alternative of building and maintaining huge libraries of device drivers in *each* consumer.
  - Better that putting physical media NICs into multiple consumers

Picture source opcfoundation.org



# Why use a Gateway / Protocol convertor

## continued

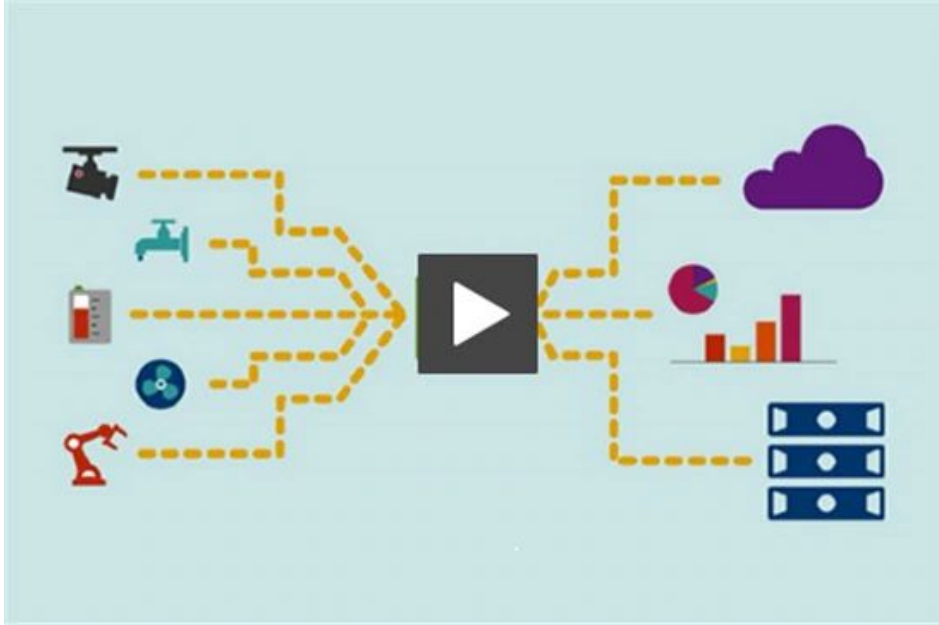


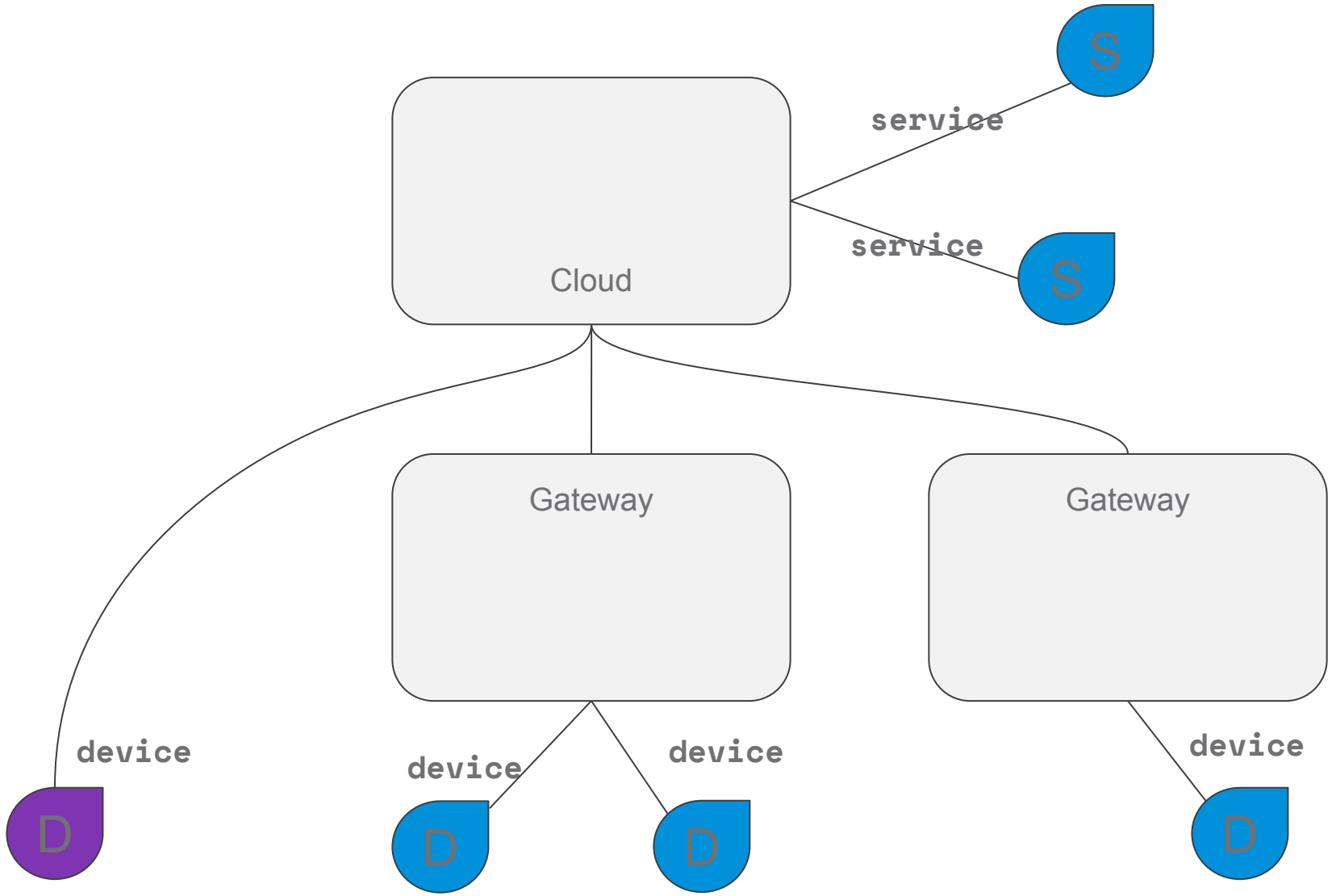
Enable load balancing and monitoring

Can be an injection point to add features to legacy or low level protocols

- Attach timestamps, location, other metadata
- Impose Security

Potentially enable base tier devices to interact with each other, while hiding details and operations from higher tiers.




S
service
Cloud
service
S
Gateway
Gateway
device
device
device
device
D
D
D
D

# Rethinking IoT gateways



- Contenarzation
- Adopt Cloud-native development practices
  - CI/CD
  - Gitops
  - ...

# Enabling new use cases



- More resources on the Edge = new use cases
- Machine learning
- Store and forward
- Caching
- ...

# Tools for field protocols



- Hardware abstraction layer
- Bluetooth REST API (REST-BLUE)

# IoT cloud connectivity



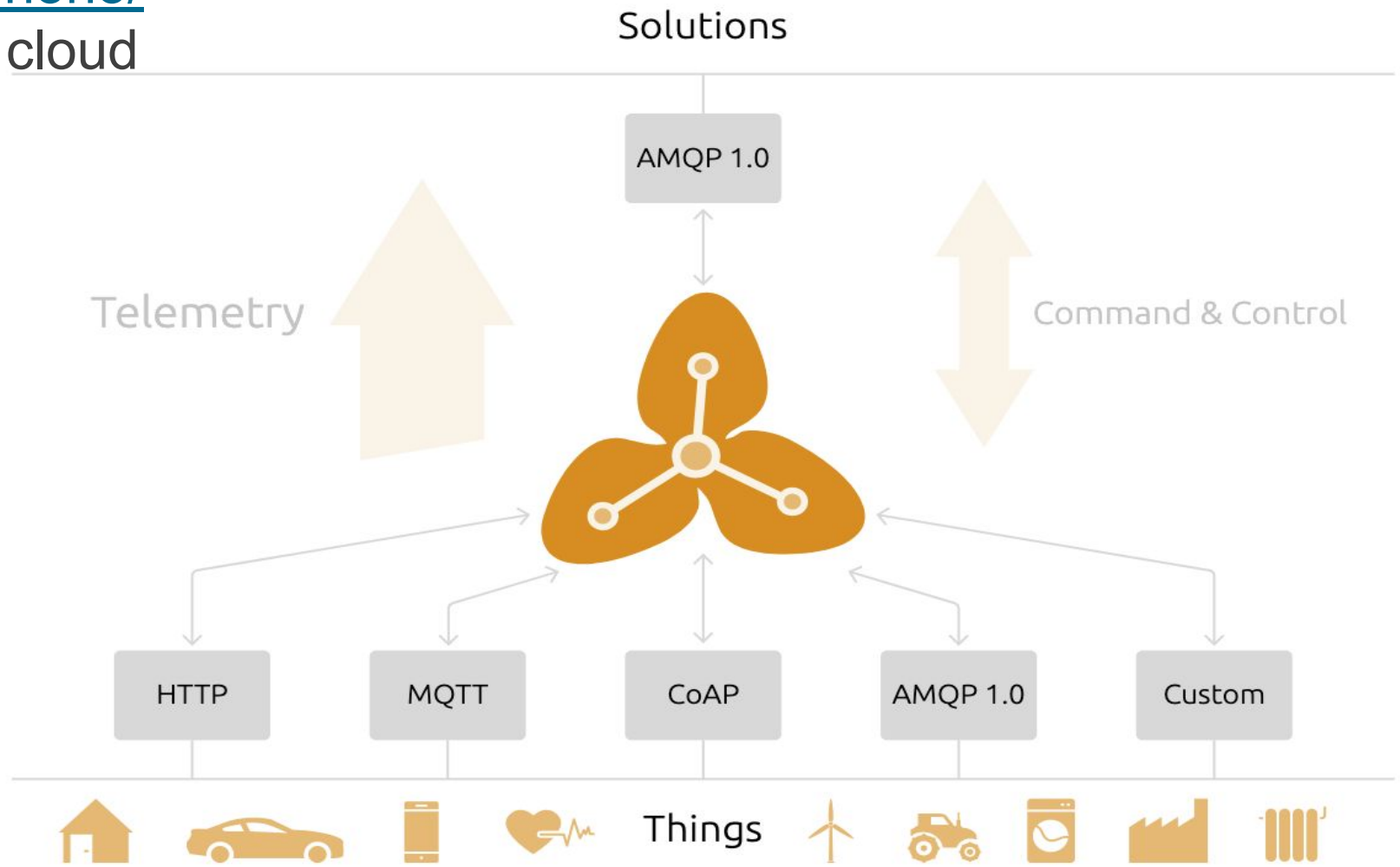
- Connection oriented
  - MQTT
- RPC style
  - HTTP

- Telemetry
- Command and Control

# Eclipse Hono

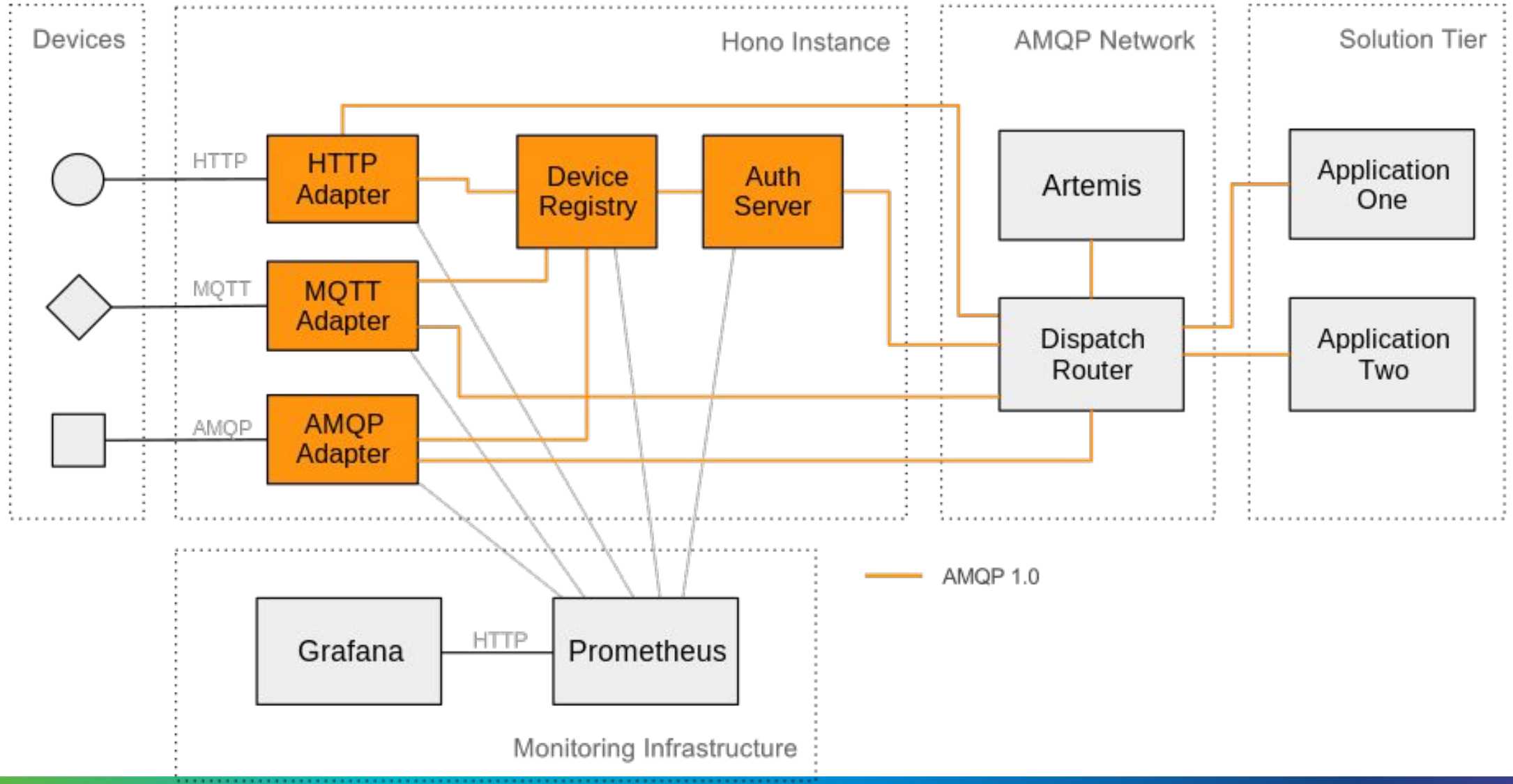


- https://www.eclipse.org/hono/
- IoT Connectivity for the cloud
- Scalable
- Secure
- Multi-protocol
- K8s based



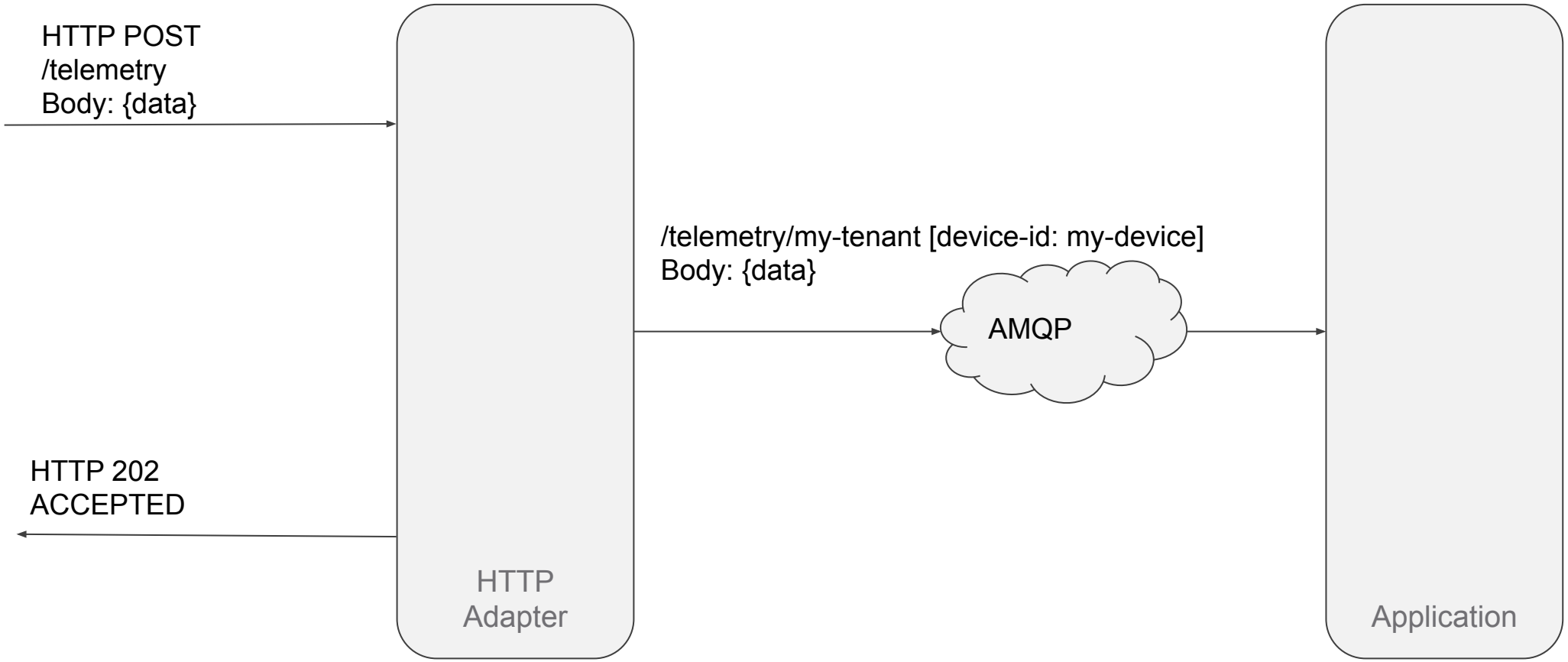
# Eclipse Hono





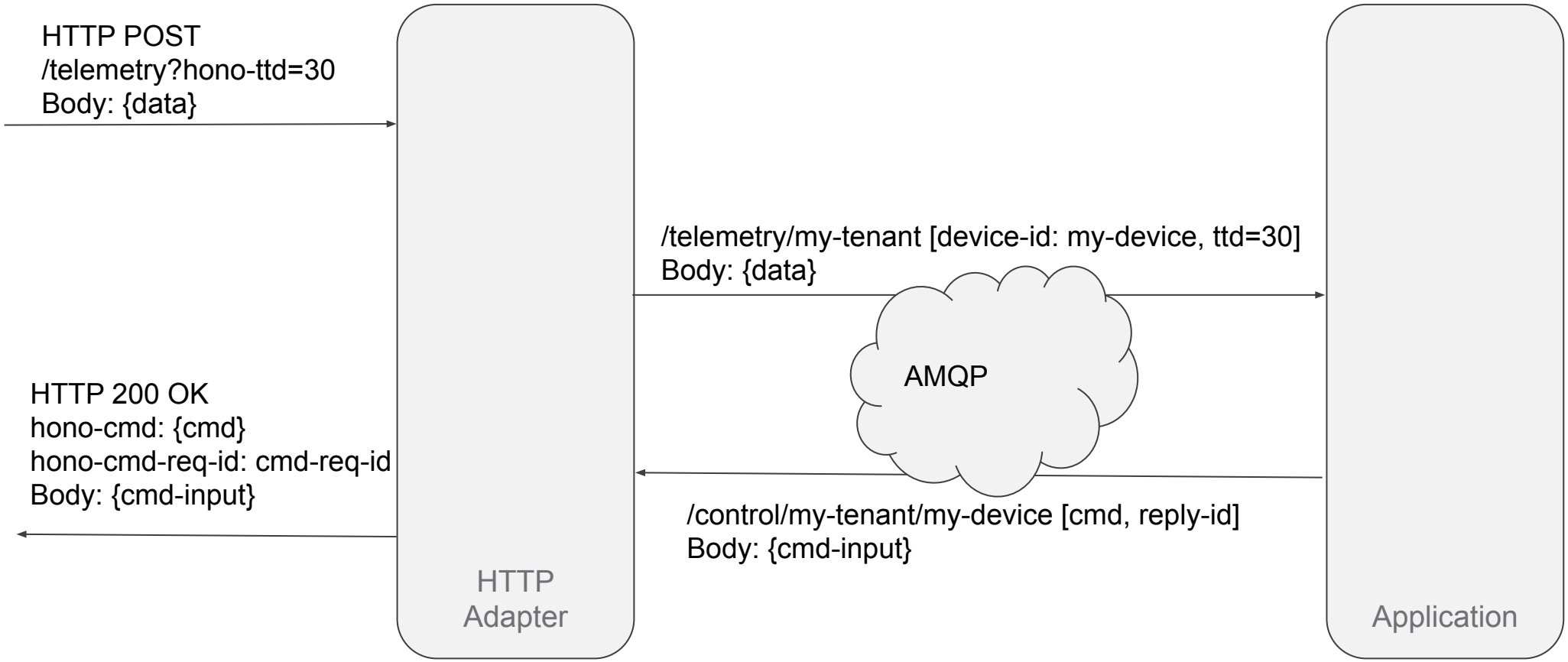
# Telemetry




HTTP POST
/telemetry
Body: {data}
/telemetry/my-tenant [device-id: my-device]
Body: {data}
AMQP
HTTP 202
ACCEPTED
HTTP
Adapter
Application

# Command and control





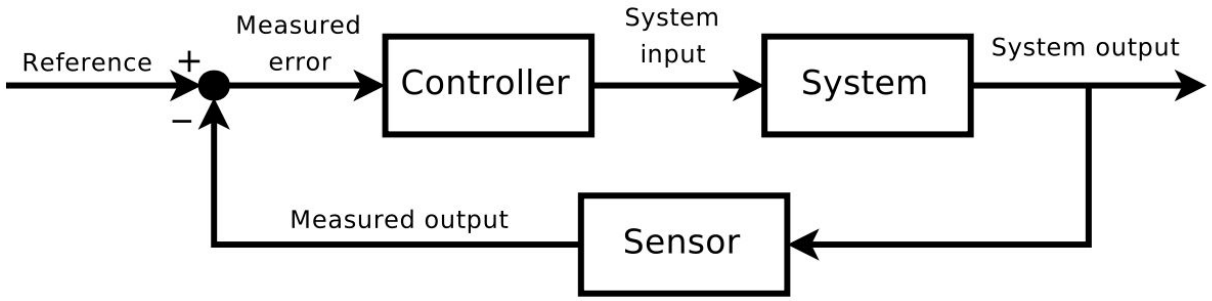
# CRDs to manage devices?

# Kubernetes

## Based on control loops



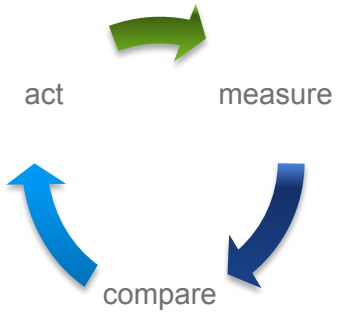
You tell Kubernetes the desired state




Kubernetes relentlessly:

- measures current state
- drives **current state -> desired state**

Recurring pattern of aspects in the system: measure - compare - act



# Kubernetes API Server

## A REST interface to the etcd database



The API server manages CRUD operations on *resources* like Pods/Deployments/Services

Object properties:

- API version
- Kind
- Metadata
- Specs

The API server itself doesn't actually understand the build-in objects, they might as well be apples, oranges and bananas

Custom Resource Definitions (CRDs) allow you to add *new* resources - allowing management via the Kubernetes API and the kubectl CLI

# IoT Sensor

## What if we managed via a CRD?



Custom Resource
hamburg-sensor.yaml

```yaml
apiVersion: containerdays.io/v1
kind: Sensor
metadata:
  name: hamburg
  namespace: germany
spec:
  unit: Celcius
status:
  lastUpdated: 2010-03-26T15:13:42.05Z
  temperature: 28
```

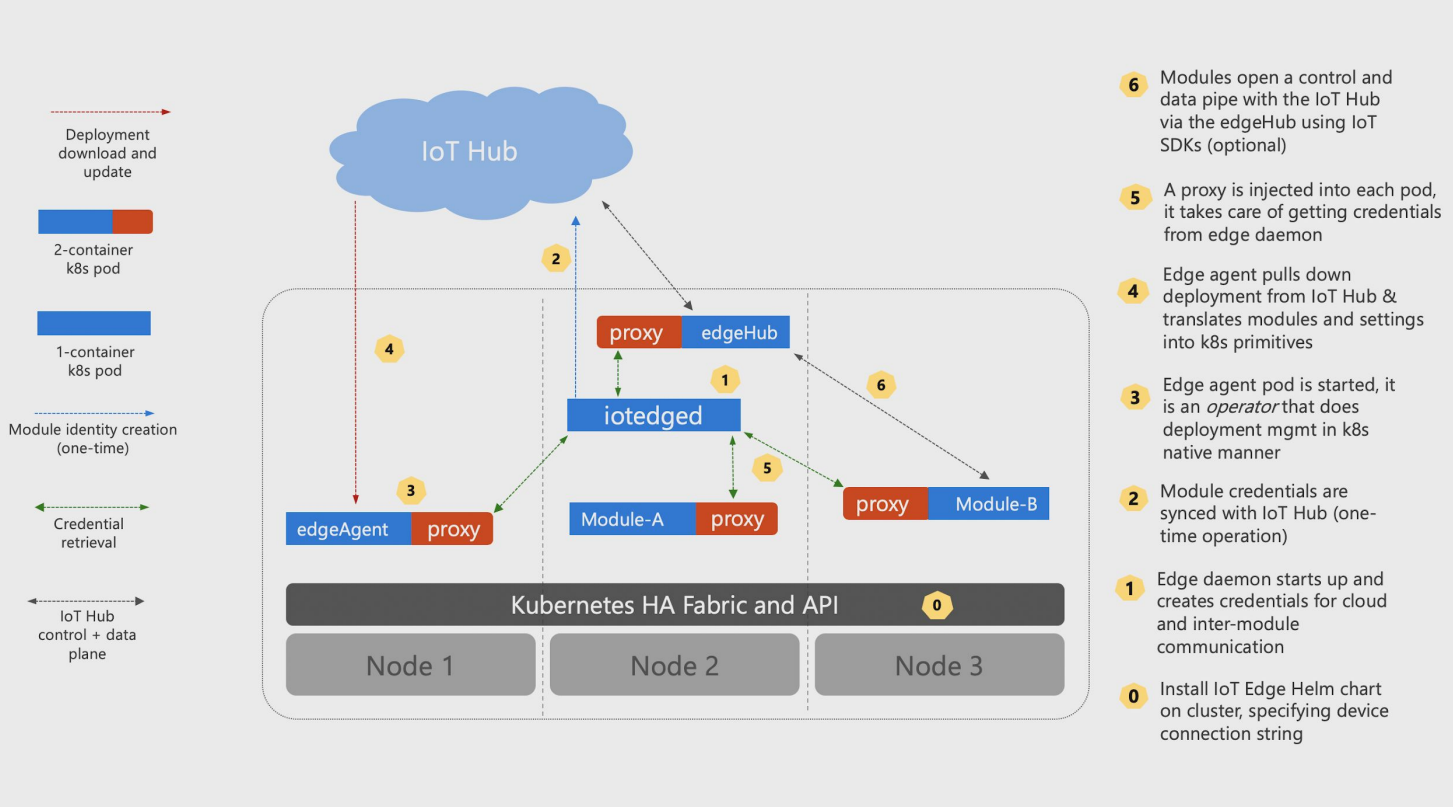
Source: Stefan Schimanski presentation on Extending Kubernetes
https://www.slideshare.net/sttts/extending-kubernetes-with-customresourcedefinitions

# Another example of a CRD for IoT

## Azure IoT Edge



Edge workloads deployed to on premise Kubernetes clusters. Uses Custom Resource Definitions (CRDs), with a Controller (IoT Edge Agent) that reconciles cloud managed desired state with the local device state




See details here:
github.com/Azure-Samples/iotedge-gateway-on-kubernetes

picture source: microsoft.com

# Emerging Use case for Kubernetes

## What if we view it as a customizable general purpose platform for managing “things”



Originally developed for specific use case of managing conterized apps in a data center

Kubernetes really is a resource management platform

Provides an API that accepts descriptions of desired states to drive controllers/operators

What already invented “wheels” could we re-use?

Popular, well architected API

Tremendous and growing collateral toolchain.. metrics, monitoring, more...

# Intermission

5 minutes - birds of a feather

# Deep Dive

Futures: Could the service mesh concept be extended beyond TCP, HTTP(s)?

Whether a location has devices, apps or both:

configuration and software update management is needed

This applies to devices, but also device connectivity

Photo: Rampion Offshore Wind Farm, United Kingdom
Nicholas Doherty on Unsplash

Probably not the most efficient

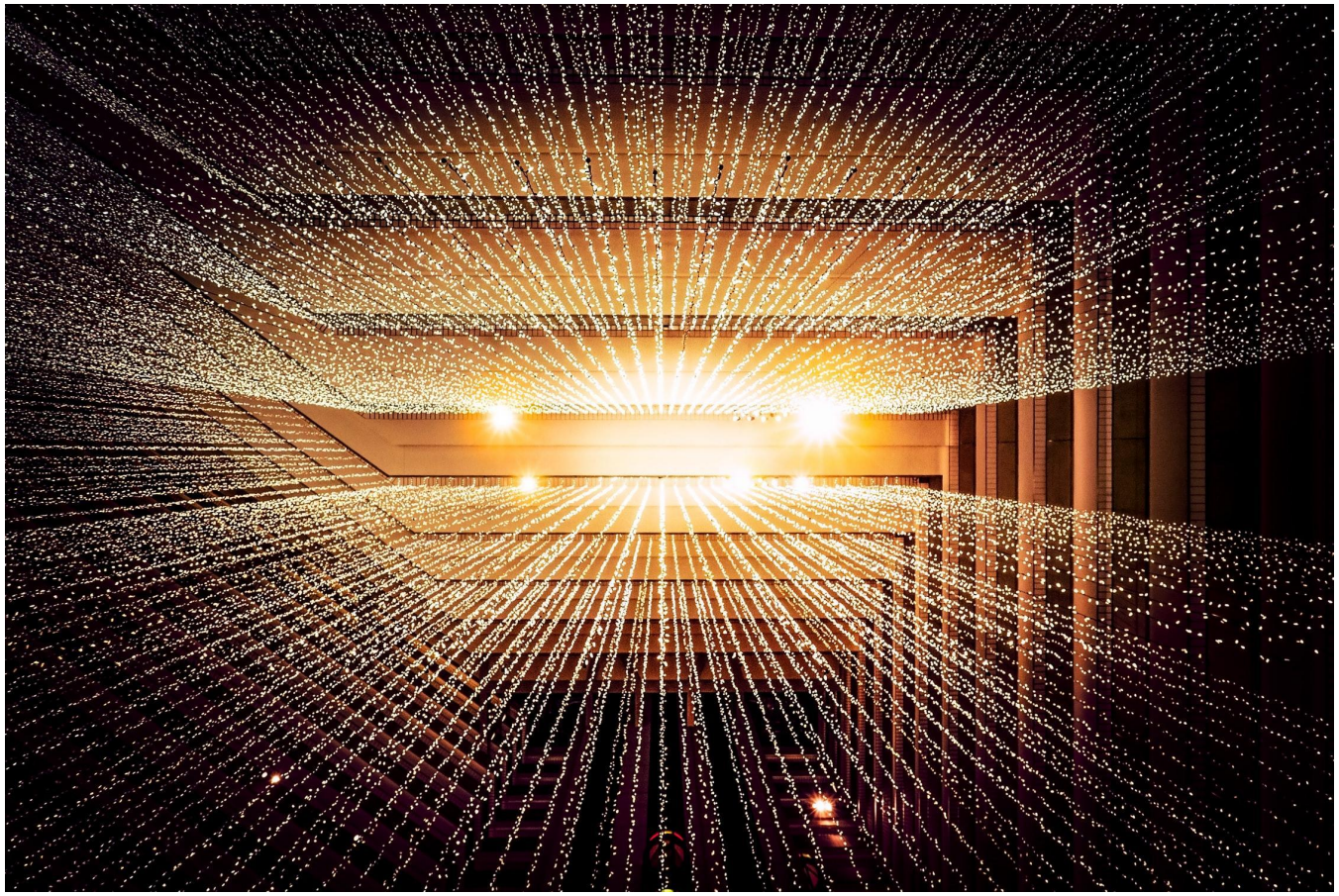
“over the air” update process

Photo: Wikipedia / Tobias Klenze / CC-BY-SA 4.0.

# Edge compute

## LTE 5 impact





“By 2022, more than half of enterprises-generated data will be created and processed outside of data centers, and outside of cloud. “ Gartner Dec 2018

https://www.gartner.com/en/newsroom/press-releases/2018-12-03-gartner-says-the-future-of-it-infrastructure-is-always-on-always-available-everywhere



Photo by Joshua Sortino on Unsplash

# Service Mesh

## What is it?



Service Mesh =
a data plane +
a control plane used to configure rules that control the data plane

Features:
load balancing, traffic management, routing, health monitoring (maybe not just network health), security policies, service and user authentication, and protection against intrusion and DDoS attacks + offload code for all this from the app developer. Maybe, service discovery add traceability

Sits between application and transport, manage from an application connection perspective, not from the low level infrastructure perspective

github.com/networkservicemesh/networkservicemesh

# Network Service Mesh

- Service mesh for L2/L3 payloads
- On-demand, dynamic, negotiated connections
- gRPC API to publish and consume Network Services
- Without changes to Kubernetes
- Works with any CNI
- Workload-To-Workload granular level of connectivity
- Loosely coupled heterogeneous network configurations

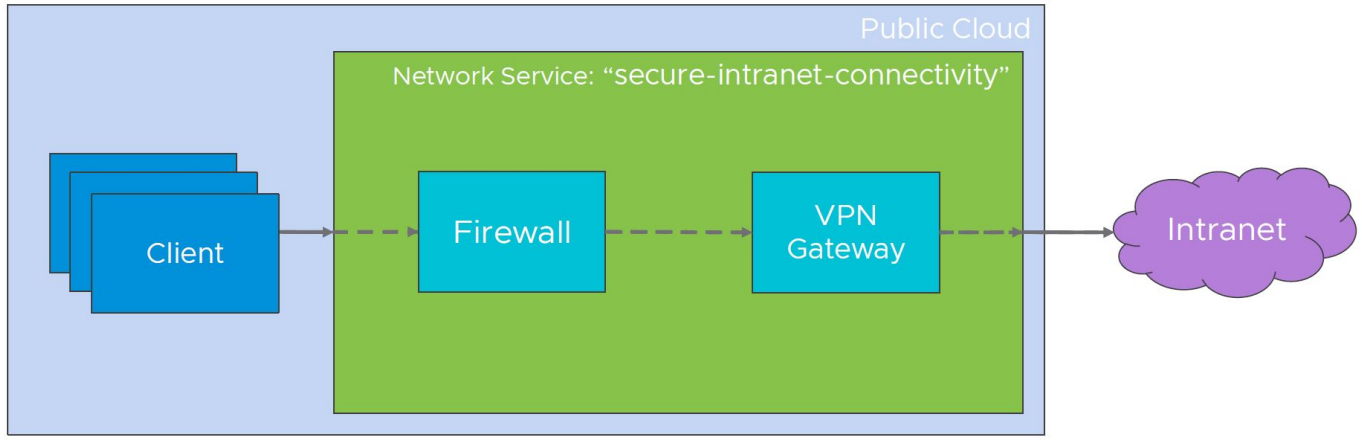
# Network Service Mesh

## Define a Network Service



- Specify type of payload
- Source and destination selection
- Service composition

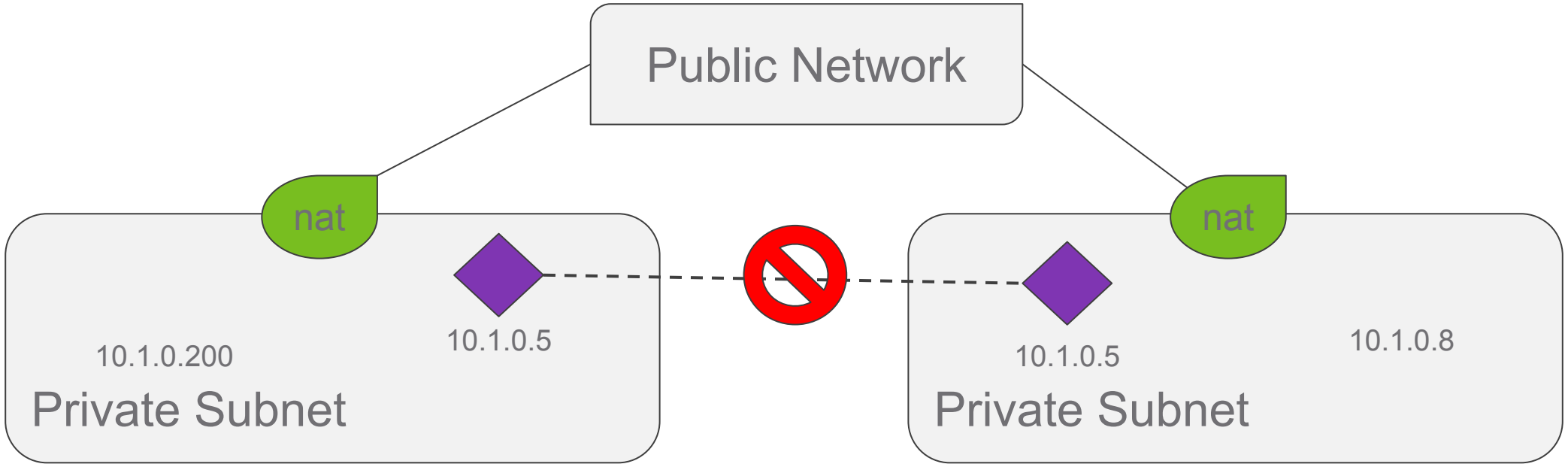
```yaml
apiVersion: networkservicemesh.io/v1
kind: NetworkService
metadata:
  name: secure-intranet-connectivity
spec:
  payload: IP
  matches:
    - match:
      sourceSelector:
        app: firewall
      route:
        - destination:
          destinationSelector:
            app: vpn-gateway
    - match:
      route:
        - destination:
          destinationSelector:
            app: firewall
```



# Edge networking

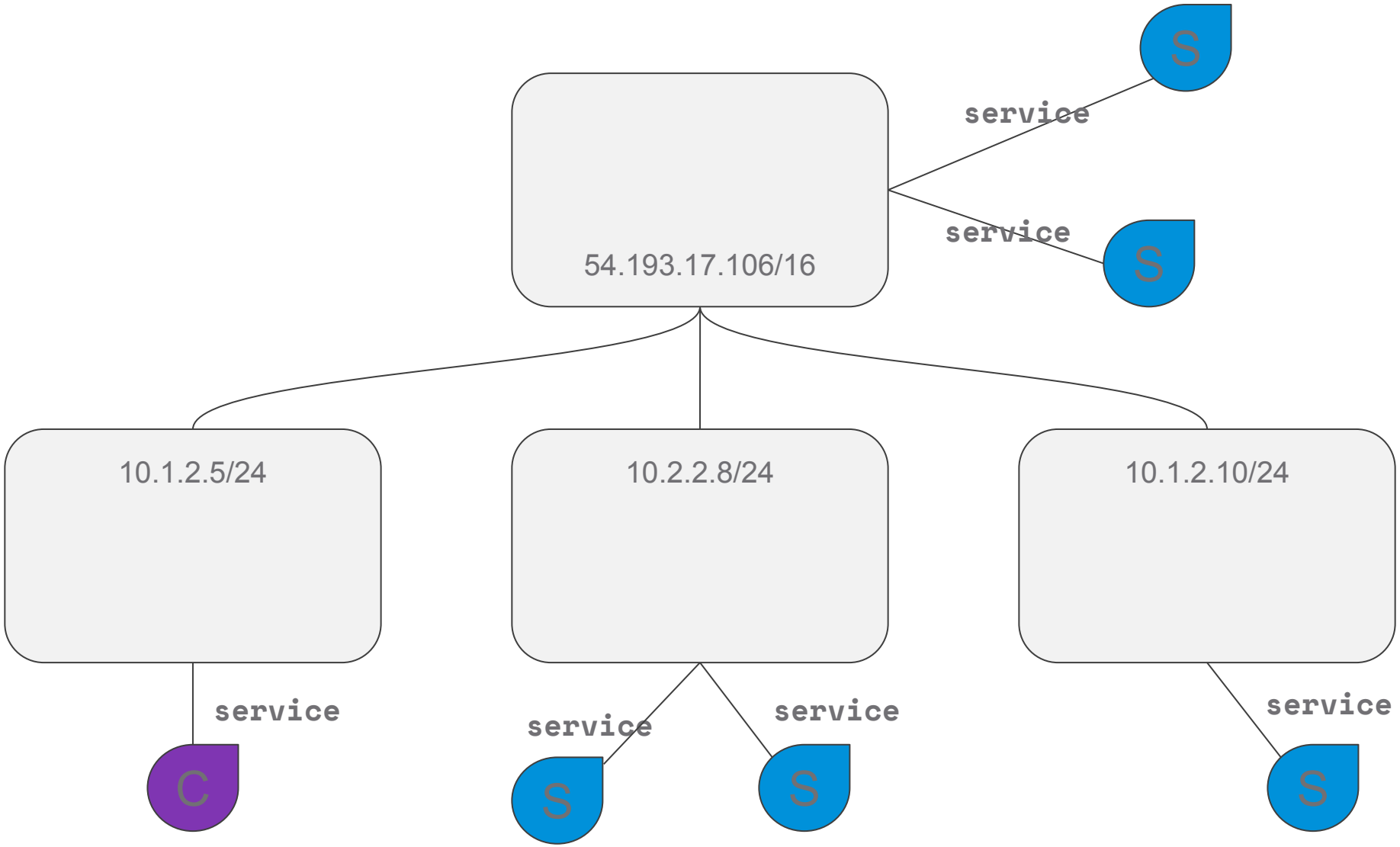


- Hybrid cloud, microservice architecture, agile integration, etc.
  - Not client/server
  - Services/processes want to be deployable and addressable everywhere (north/south/east/west)
- Edge computing - Lots of private subnetworks



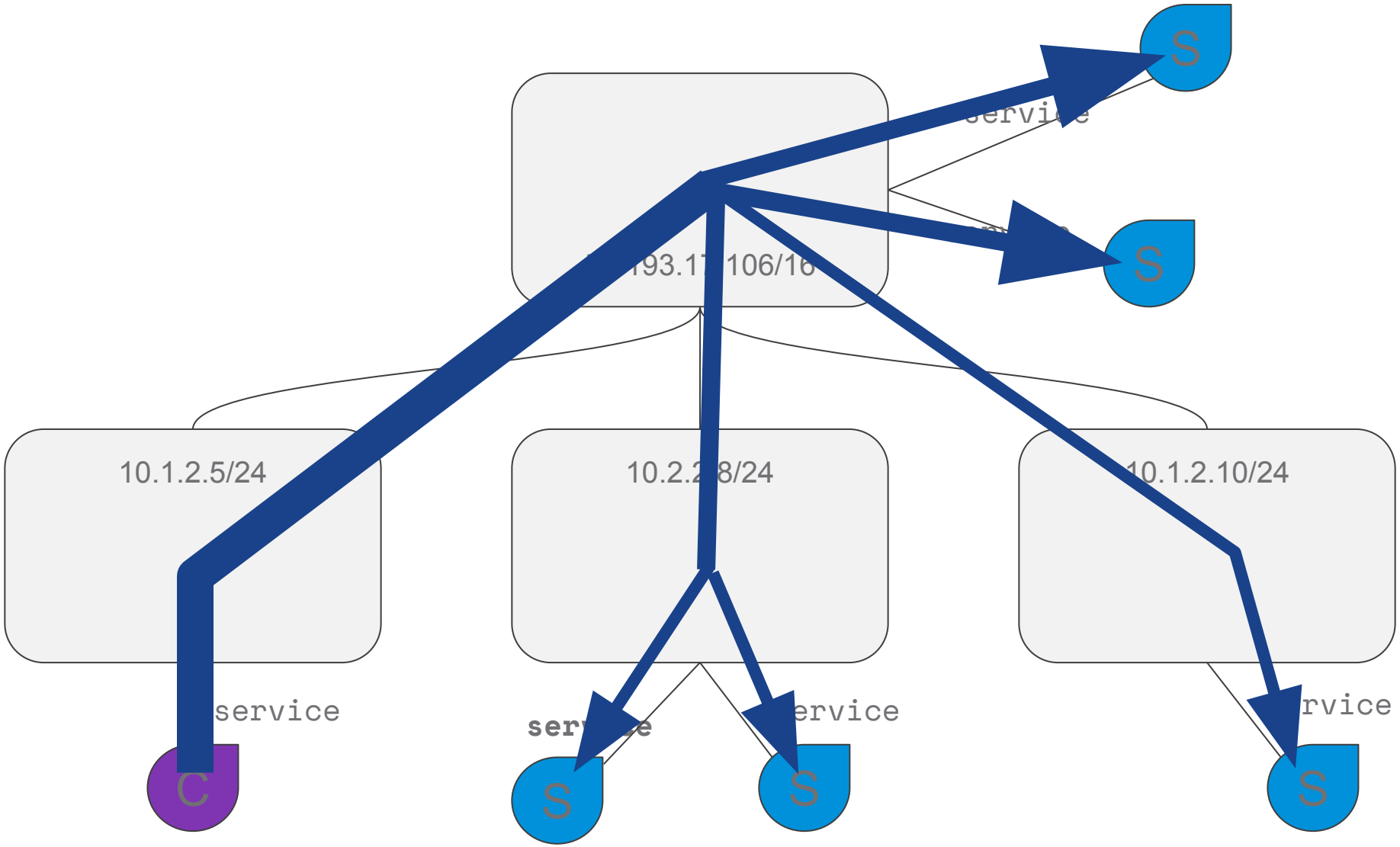
# Application Layer Addressing





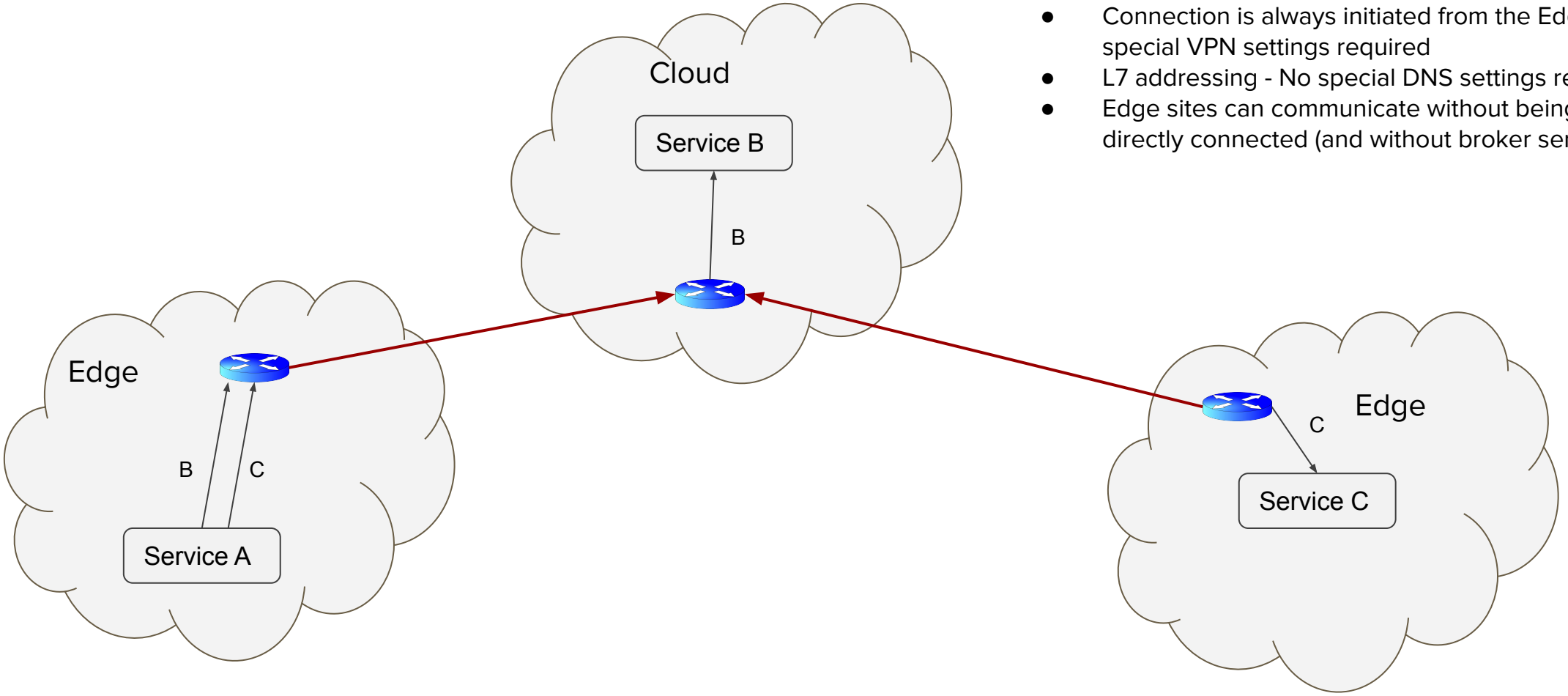
# Application Layer Addressing





# Secure multi-site communication






- Connection is always initiated from the Edge - No special VPN settings required
- L7 addressing - No special DNS settings required
- Edge sites can communicate without being directly connected (and without broker service)

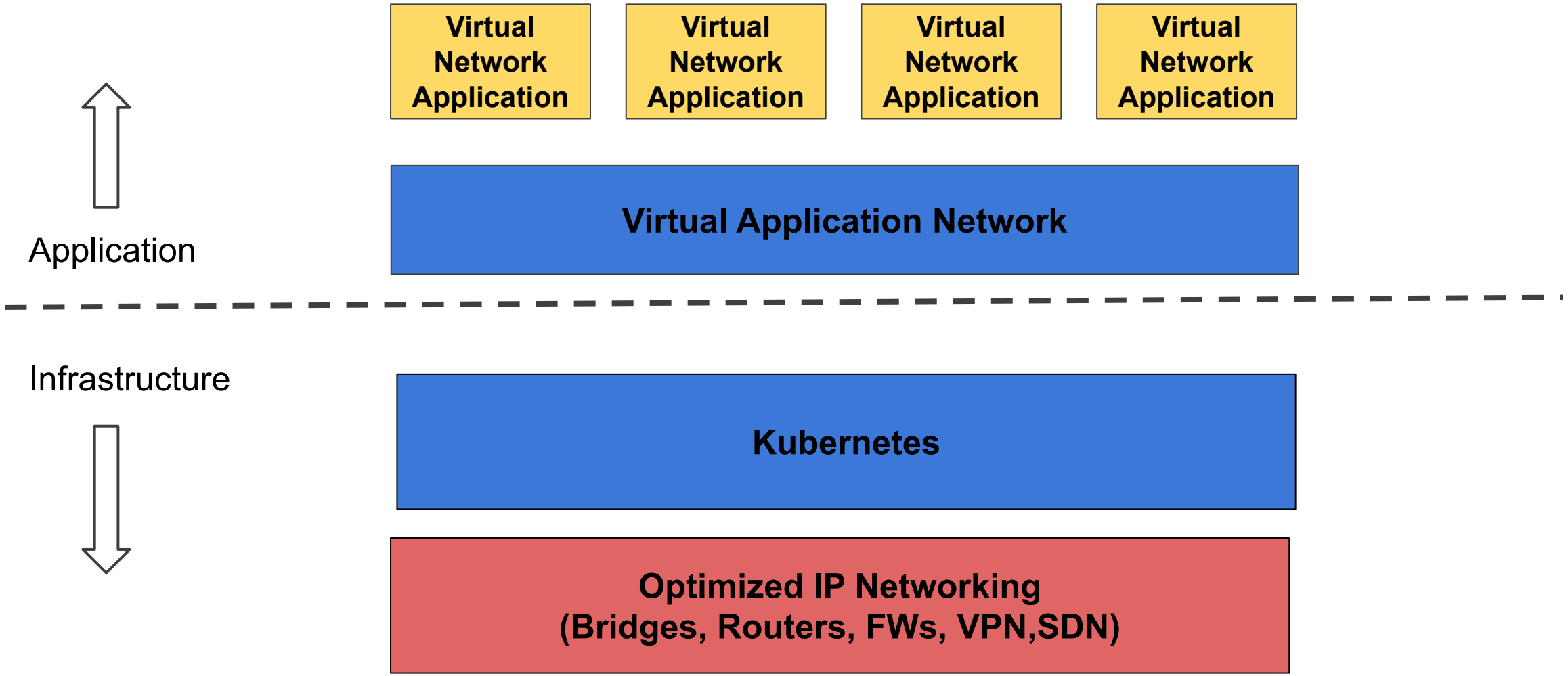
# Virtual Application Networks



- Extends a private L7 network across a public network
- Enables users to send and receive data across a shared network as if their computing devices were directly connected to the private network
- The network association is defined by the application's L7 addressing
- E.g. Applications can communicate with other applications through their addressing endpoints independently of the LAN or Subnet the computing device is operating on
- VAN's are built on top of Application Routers

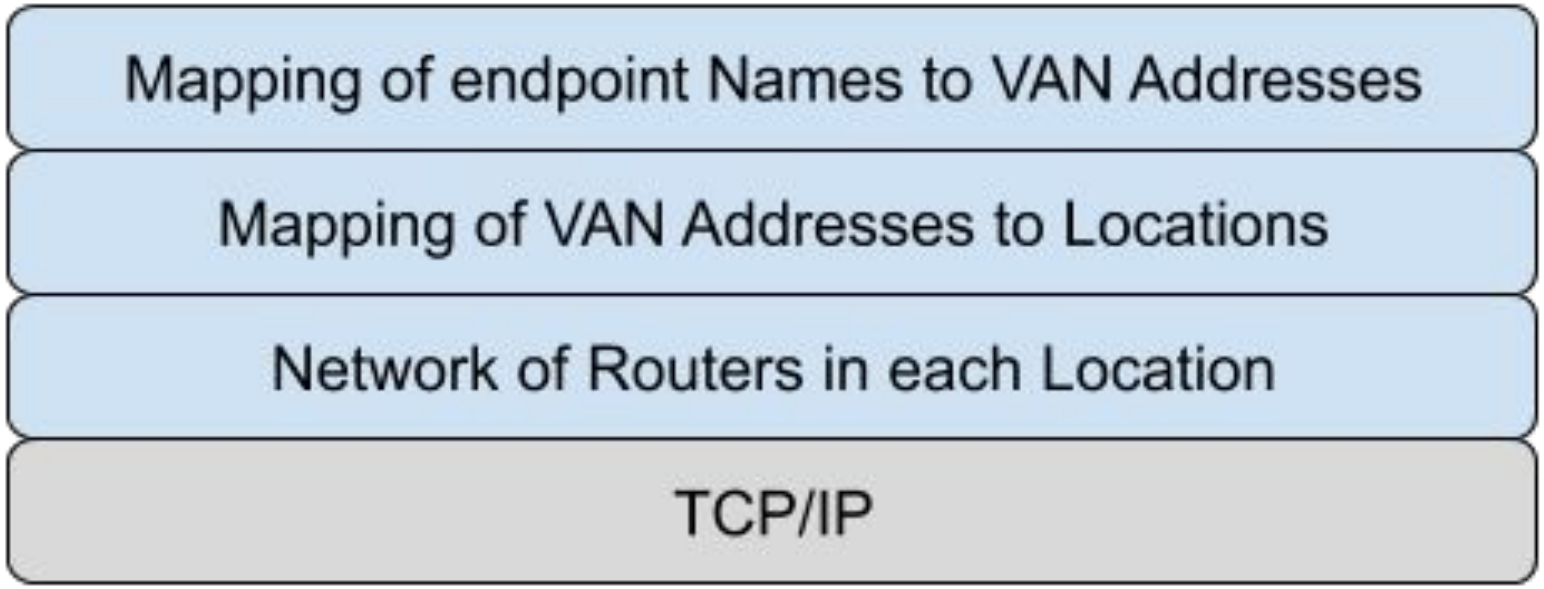
# Virtual Application Network “Layer”





# Virtual Application Network Layers





# Implications of Application Addressing



- Security
  - Access control for addresses - at the service/process/business resolution
  - Locked-down network membership - Mutual TLS for inter-site connections
  - Cross-cluster applications not exposed via Kube networking
    - Public exposure limited to ingress
  - Trusted and untrusted edges
- Management
  - Metrics collected at business resolution

# Skupper.io

AMQP
Advanced Message Queuing Protocol

KubeCon | CloudNativeCon
North America 2019

- Operational Ease
  - Easy to deploy in a multi-cluster network
  - No advanced networking (SDN, VPNs, Tunnels, Firewall rules, etc.)
  - No need for elevated or admin privileges
  - No problem with overlapping CIDR subnets or mixes of IPv4 and IPv6
  - No single point of failure - use redundant topology
- Not just for messaging
  - Proxy maps HTTP, TCP, UDP, etc. to AMQP
- http://skupper.io
  - Examples, demo-videos, etc.
  - New, emerging project

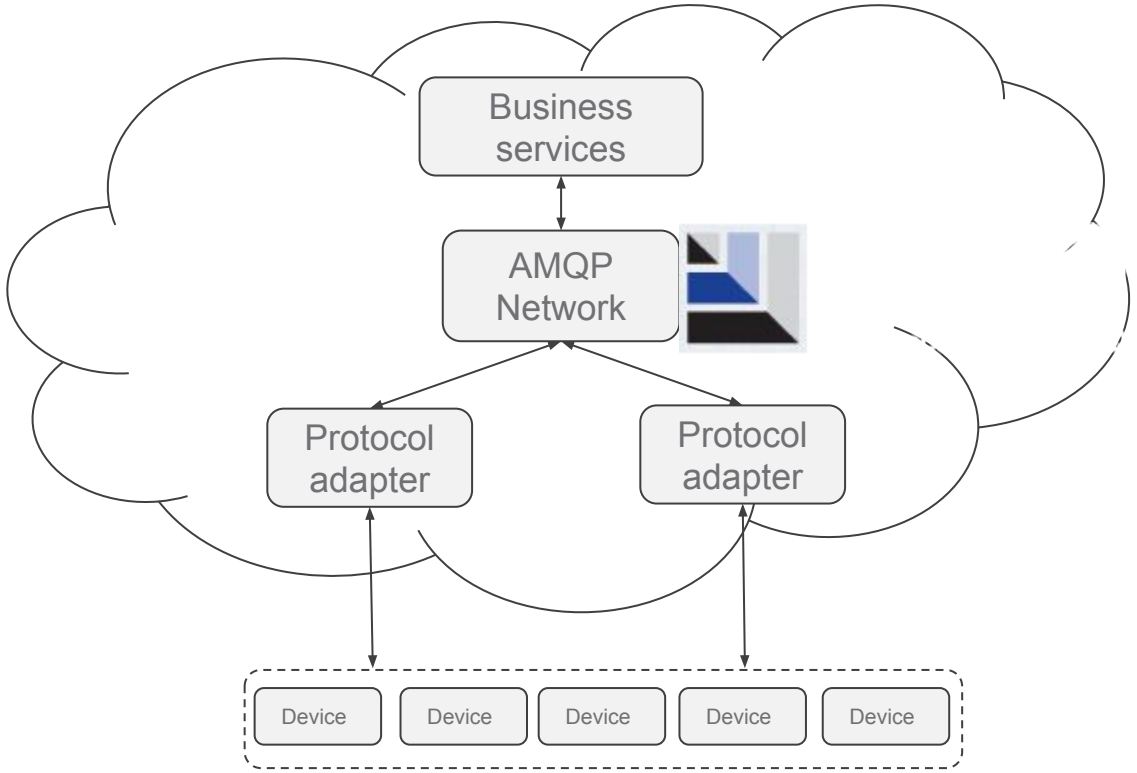
# Deep Dive

Futures: Upcoming development, bringing it all together!

# Eclipse Hono in the cloud



HONO

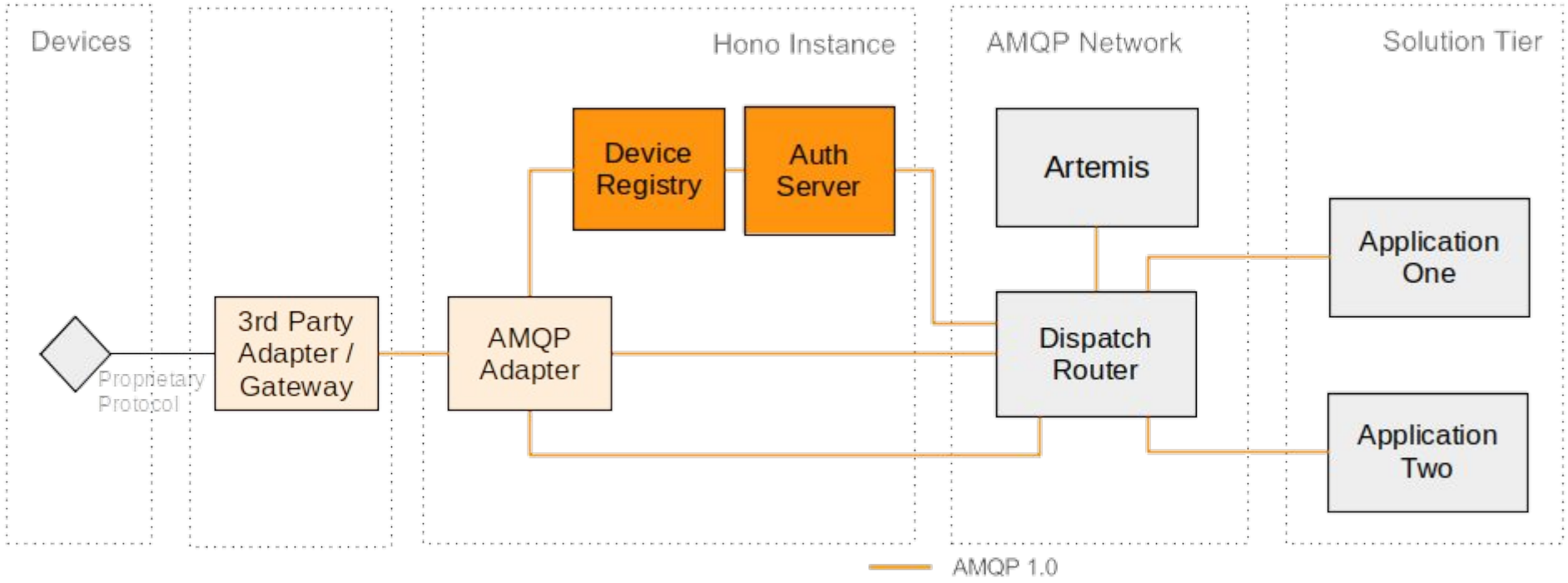


# Is cloud obsolete?

# Way forward

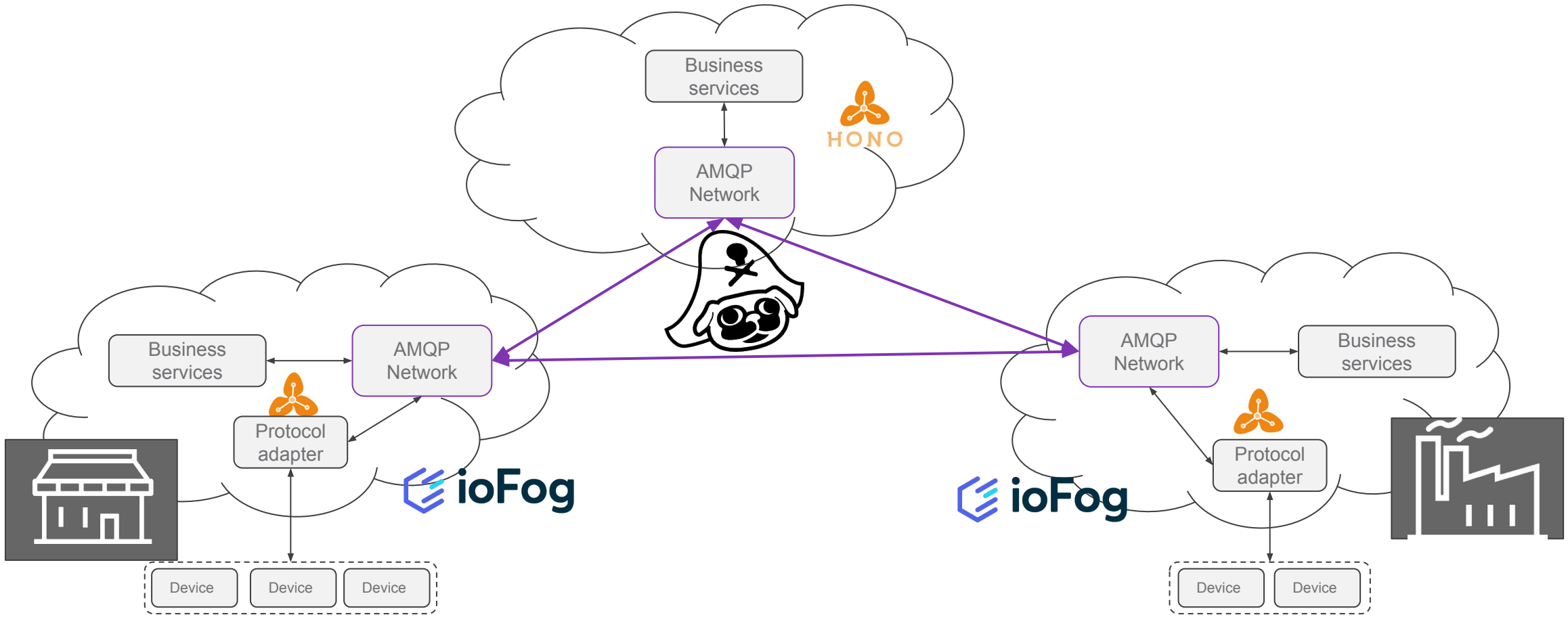


- Cloud is not obsolete
- Cloud IoT platforms still needed
  - Business applications
  - Long term data storage
- Work on distributed Edge deployments for IoT services


Devices
Hono Instance
AMQP Network
Solution Tier
Device Registry
Auth Server
Artemis
Application One
3rd Party Adapter / Gateway
AMQP Adapter
Dispatch Router
Proprietary Protocol
Application Two
AMQP 1.0

# Eclipse Hono on the Edge





# Full integration



- Provide development continuum from field to cloud via edge
- Better integration
  - Platforms
    - KubeEdge
    - ioFog
  - Communication
    - Hono
    - Skupper

# Demonstrations

- Demonstration: Kubernetes management of an edge application using a specialized protocol
- Demonstration: Use a device gateway with Kubernetes

# How to get involved with the IoT Edge Working Group

## Learn more…..



Regular Work Group Meeting:

USA WG Meeting Wednesday 9am PT, every 4 weeks, next on December 4

APAC WG meeting Wednesday 5 UTC every 4 weeks, next on December 17

- Meeting notes and agenda

Link to join the group

- groups.google.com/forum/#!forum/kubernetes-wg-iot-edge

Link to join Slack

- https://kubernetes.slack.com/messages/wg-iot-edge

White Paper

- http://bit.ly/iot-edge-whitepaper

# Thank You