KubeCon | CloudNativeCon

Europe 2022

WELCOME TO VALENCIA

KubeCon CloudNativeCon Europe 2022

Kubernetes IoT Edge Working Group

Edge Device Onboarding and Management

Kate Goldenring
Microsoft

Steven Wong
VMware

Thursday, May 19 • 11:00 - 11:35 room 4f https://sched.co/ytoP

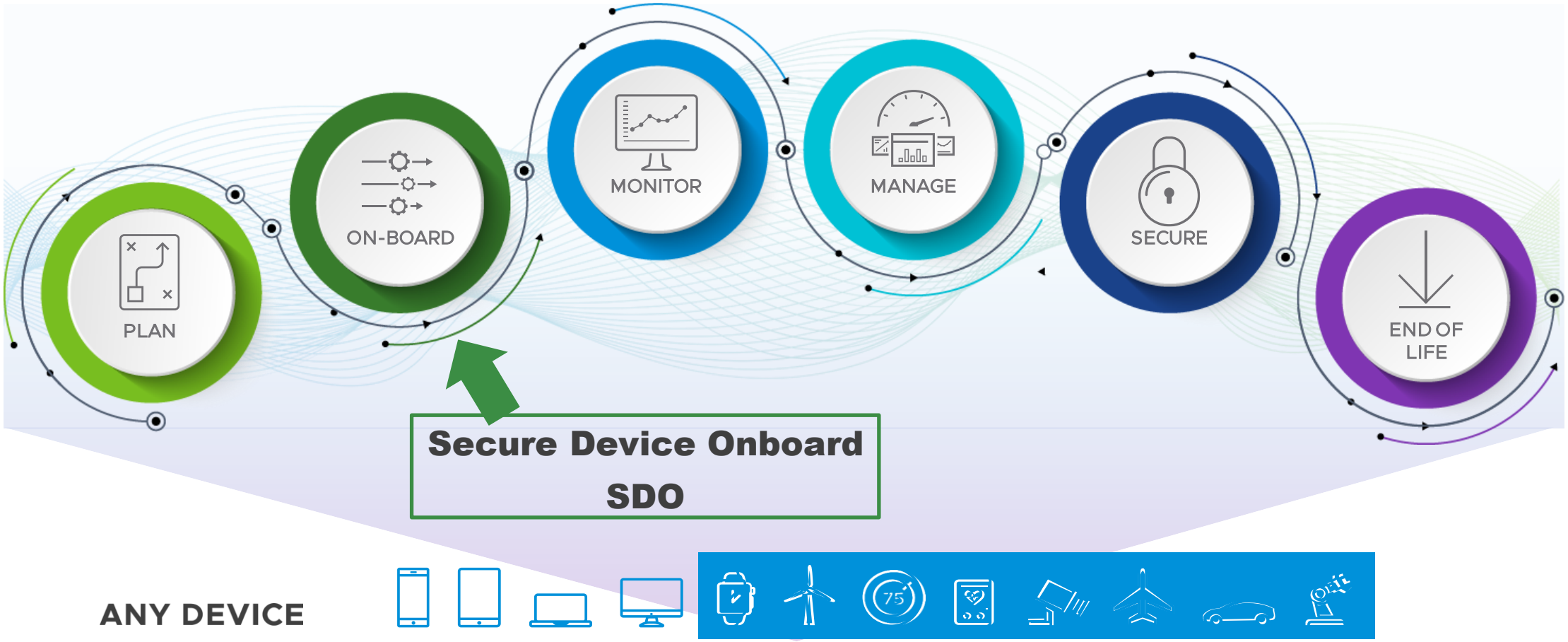
Agenda

Overview of a few OSS projects that enable device onboarding and management across a range of edge devices

- Compute device onboarding with SDO
- Constrained device use and management with Akri
- The IoT Edge Working Group – Get involved!

Device operation is a multi-dimensional challenge

Device lifecycle – a time dimension



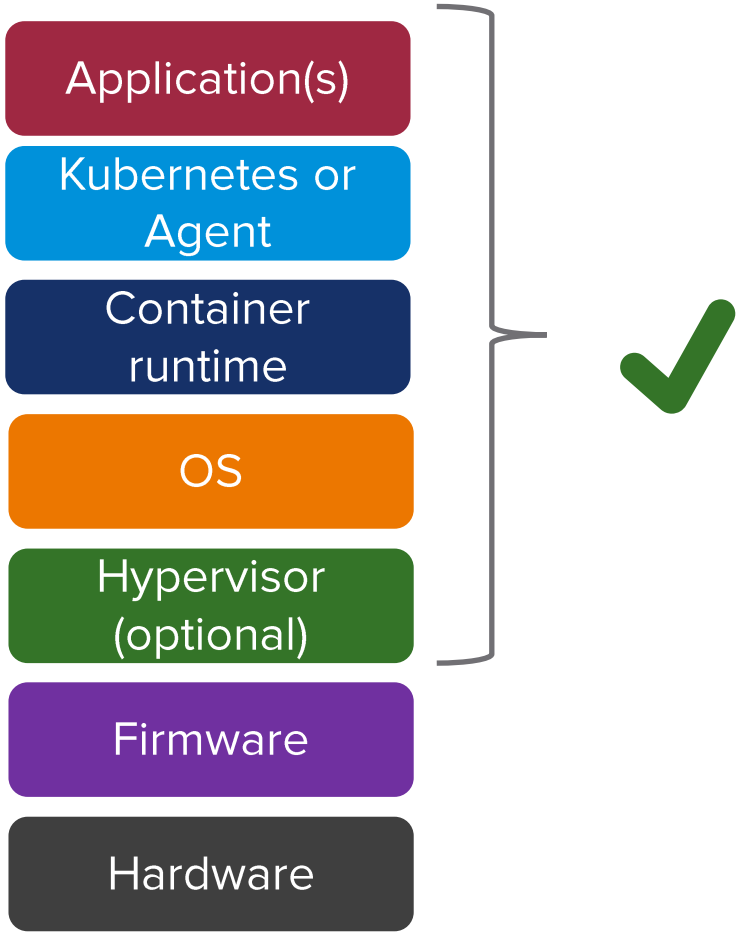
Helicopter visit to offshore rig

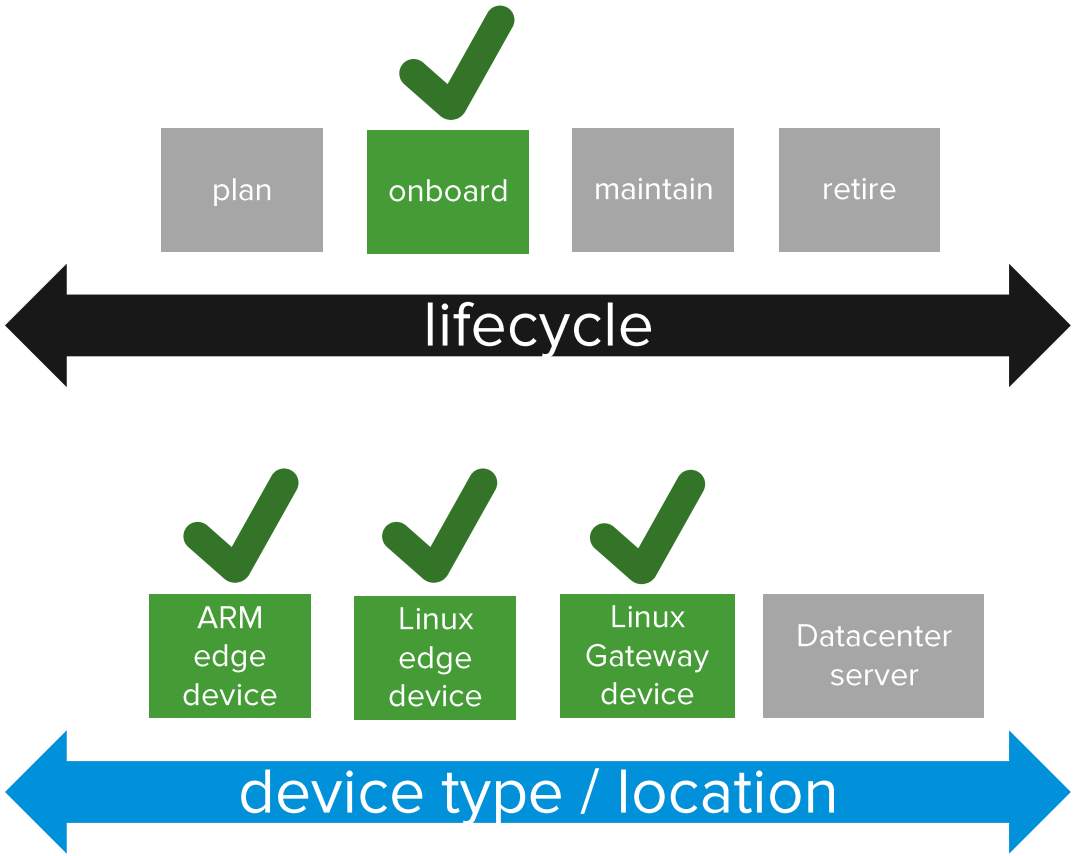
Maybe not an efficient way to perform
“over the air updates”
at scale

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

Secure Device Onboard

Address these areas of concern





FDO Secure Onboard

An automatic onboarding protocol for IoT devices

https://github.com/secure-device-onboard

The FIDO Device Onboard Specification is an open spec being shepherded by the FIDO Aliance

The LF Edge umbrella organization is fostering an open source implementation of the FIDO Device Onboard Specification

Origin is the Secure Device Onboard Project published in open source by Intel in 2020

FDO – Pre-req

Manufacturer hosts two API services

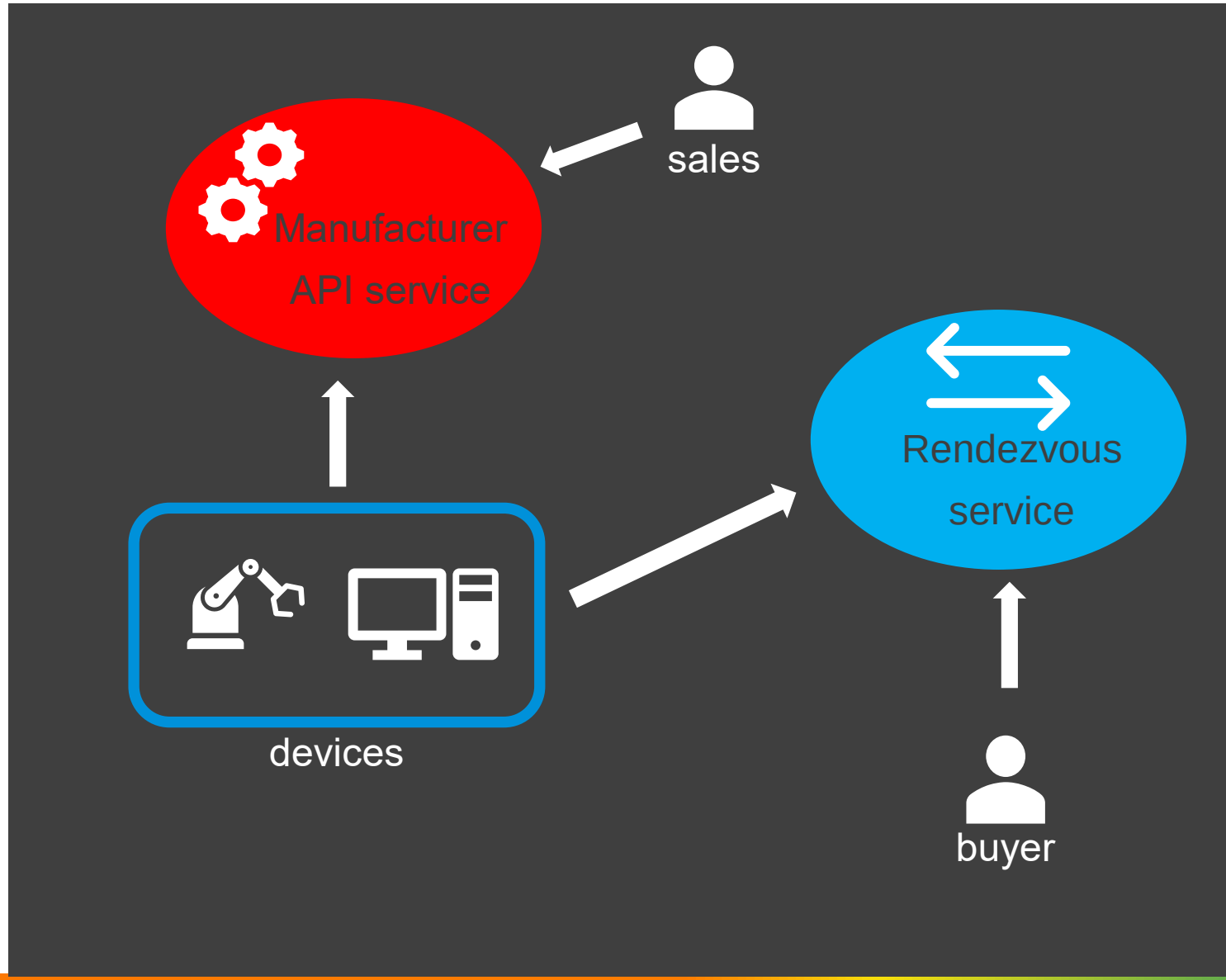
Device manufacturer makes arrangements to operate a Manufacturer Server and a Rendezvous Server.

Manufacturer server will be used interact with:

1. Device upon creation
2. Sales operation business logic upon first sale

Rendezvous Server will be used after sale by:

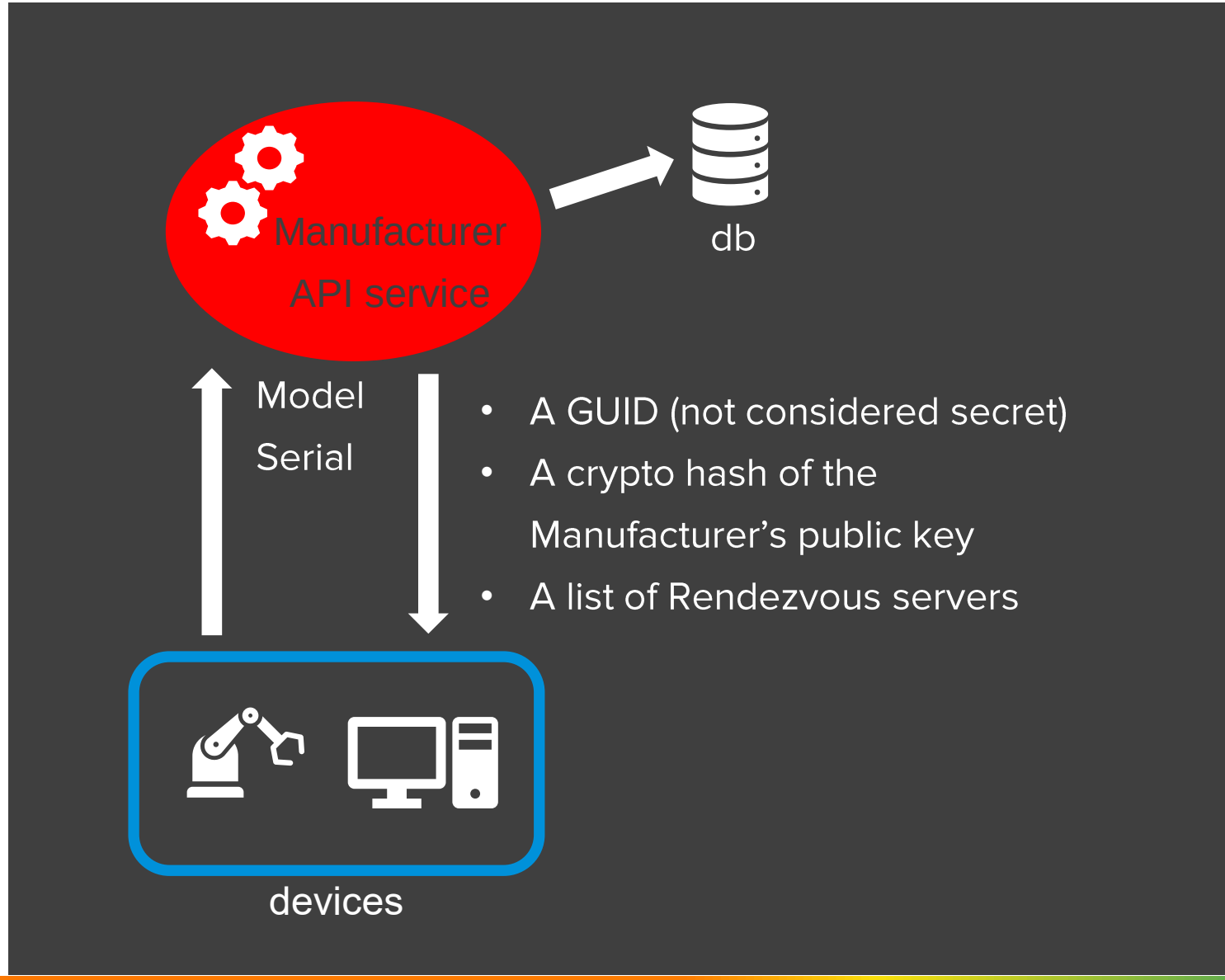
1. Device Owner after sale
2. Device on first power up

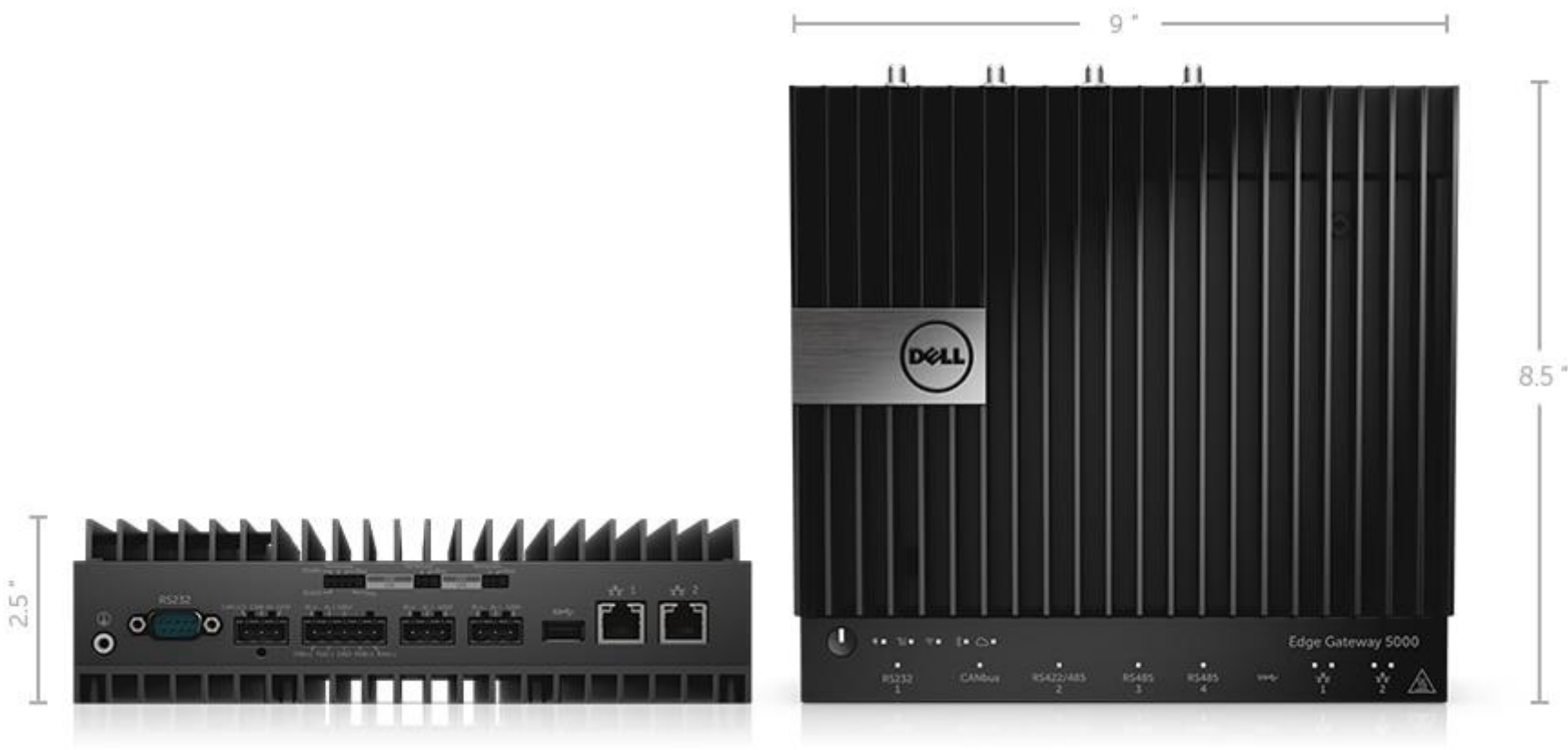


Step 1 - DI

Device Initialization

1. Device is created. Should have a TPM or PUF for best security
2. An Executable (built from client SDK) is installed in device and run.
   It makes a call to the Manufacturer API, reporting device model and serial number.
3. Manufacturer ID Hash & Rendezvous server address(es) are returned and stored securely
4. The Manufacturer server maintains a record that this occurred, indexed by the device serial number




9 "
8.5 "
2.5 "
Edge Gateway 5000


Edge Gateway 5000
RS232
1
CANbus
RS422/485
2
RS485
3
RS485
4
1
2
DELL
KTS
LITHIUM BATTERY
JAPAN STD
CR2032
Made in CHINA
Water Detection Tape
WICK
CP-2515
WIDTH
0.1 inch

Step 2 device is sold

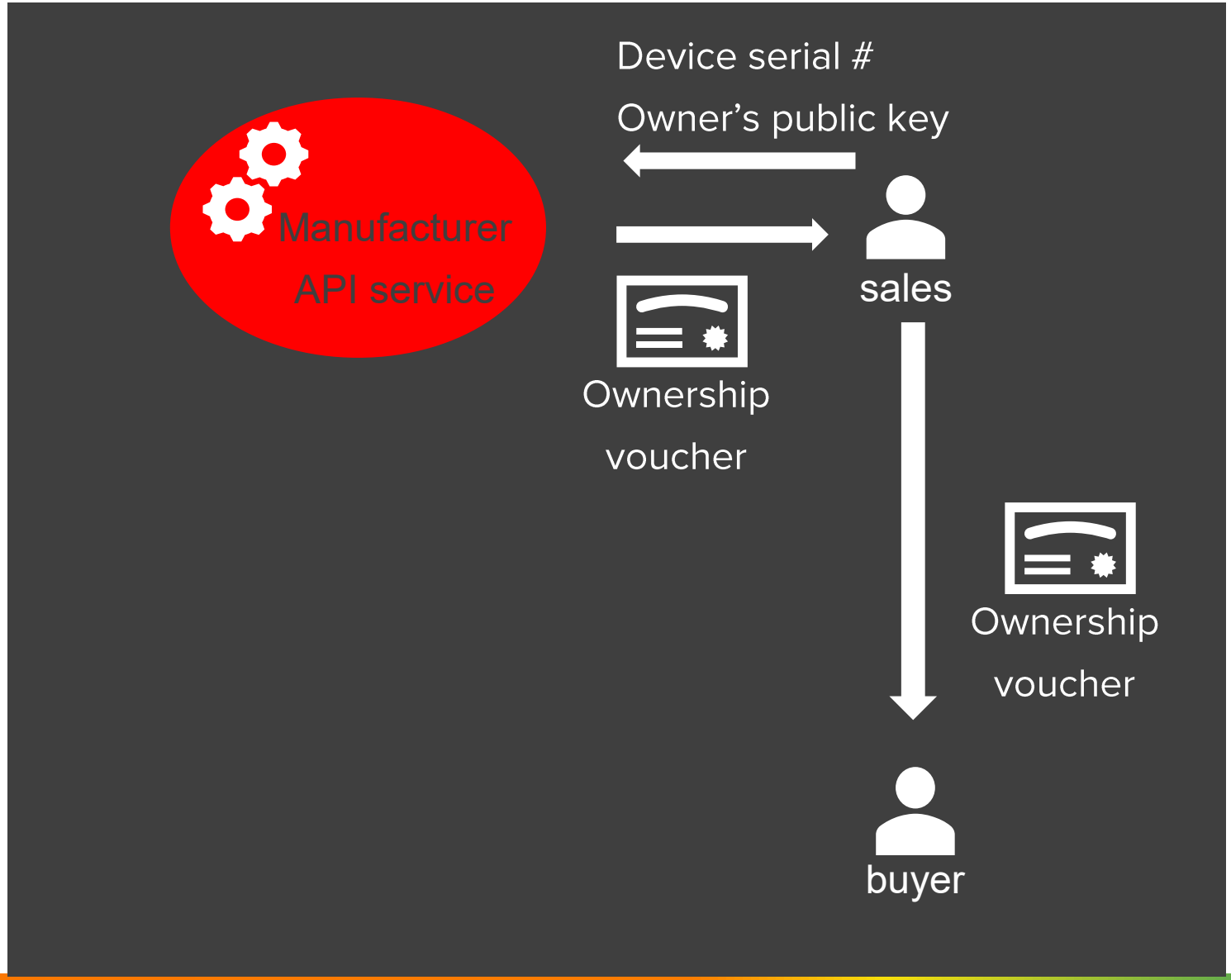
An ownership voucher is generated

Manufacturer's business logic requests a voucher generation

The voucher goes to the new owner (could be a distributor/reseller).

The voucher has:

- A voucher ID
- The GUID issued to the device
- Chain
  - Manufacturer's public key
  - Owner(s) public key
- Hash of chain of all former owners including Manufacturer



Step 3 –TO0, transfer of ownership phase zero

Owner asks Rendezvous server for service for a particular device for a limited time

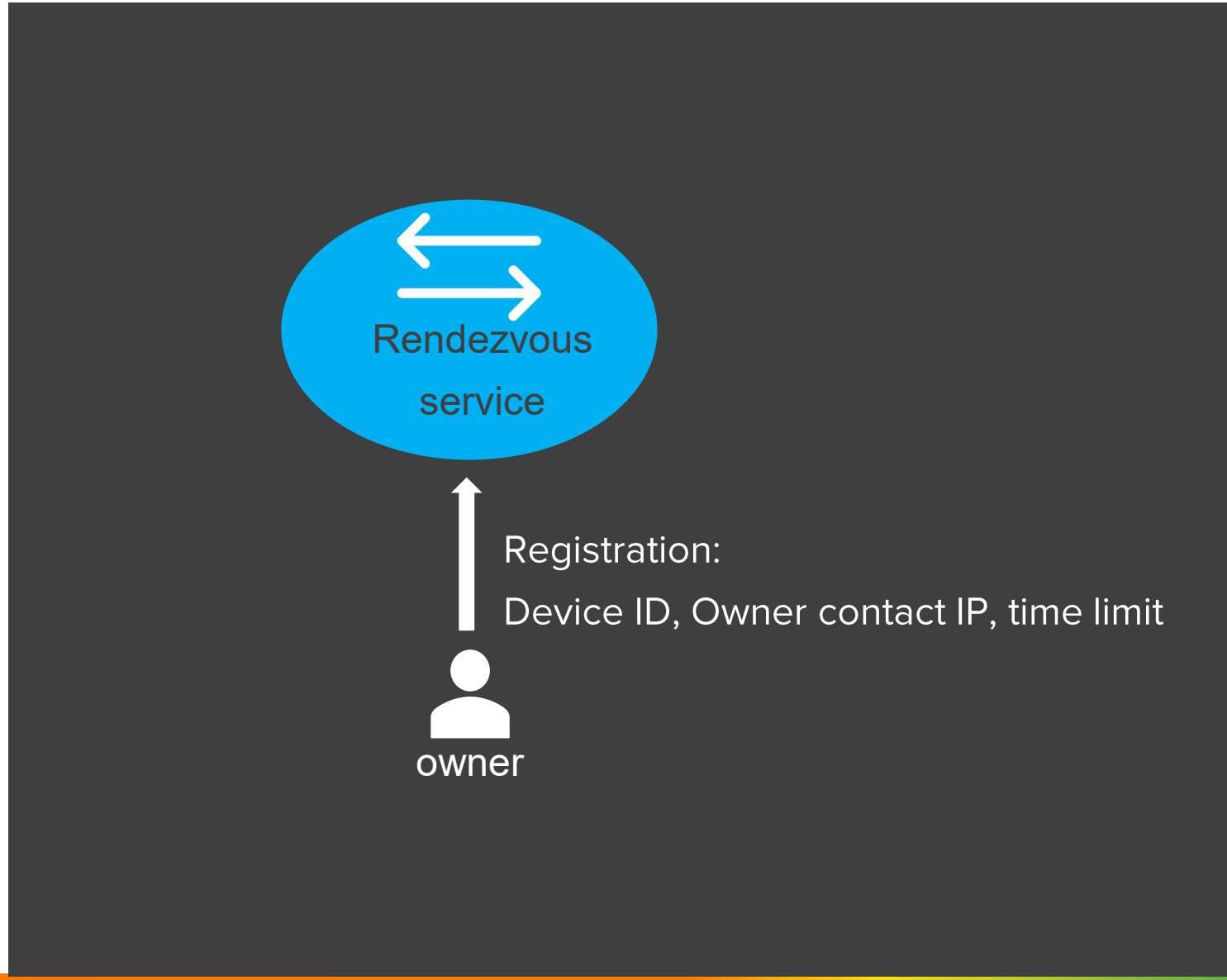
Transfer Ownership 0 (TO0) - Owner

Establishes the mapping of device to the Owner's IP address

Pre-req

Owner Service is running:

- Listening on an IP reachable by the device
- configured with the private key counterpart to the public key in the device ownership voucher



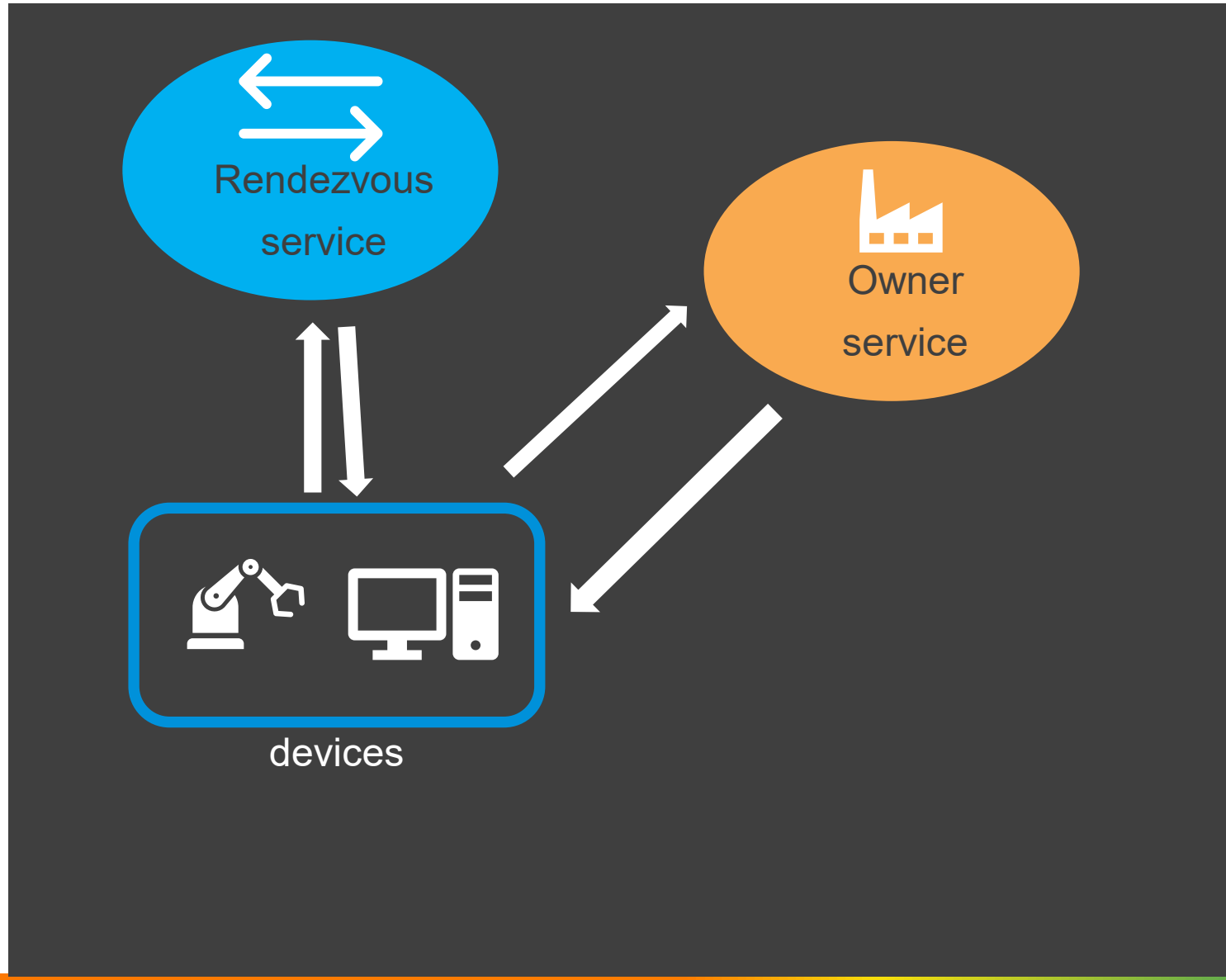
Step 4 – (TO1 and TO2)

Device is powered on and runs FDO client executable

Device learns how to contact owner

Device contacts Owner

- Mutual trust is established
- Ownership transfer is completed,
- Configuration key pairs are delivered and acted upon
  - Linux “reference example” runs a bash script

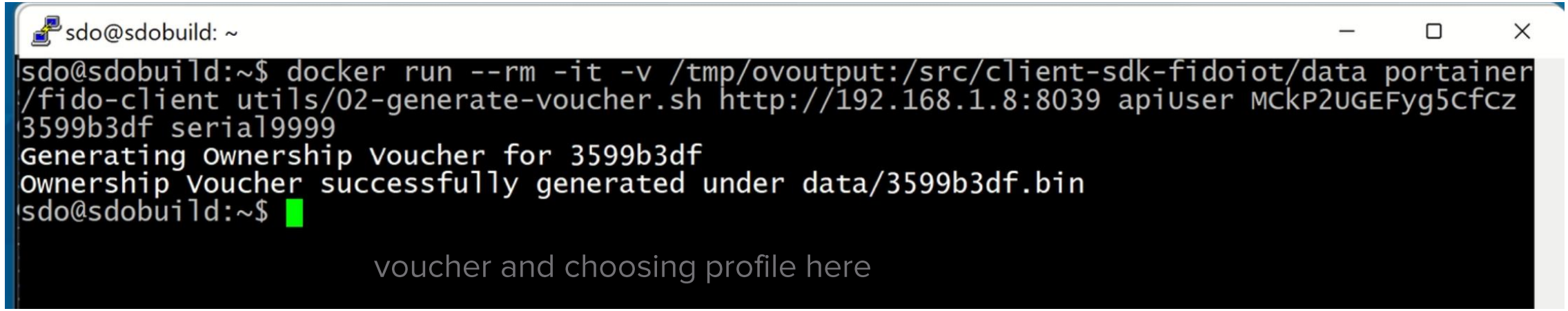


Recorded demo – STEP 1 device initialization

This extracts and installs a prebuilt version of an FDO client SDK from a container, then uses it to engage in API operations with the Manufacturer server

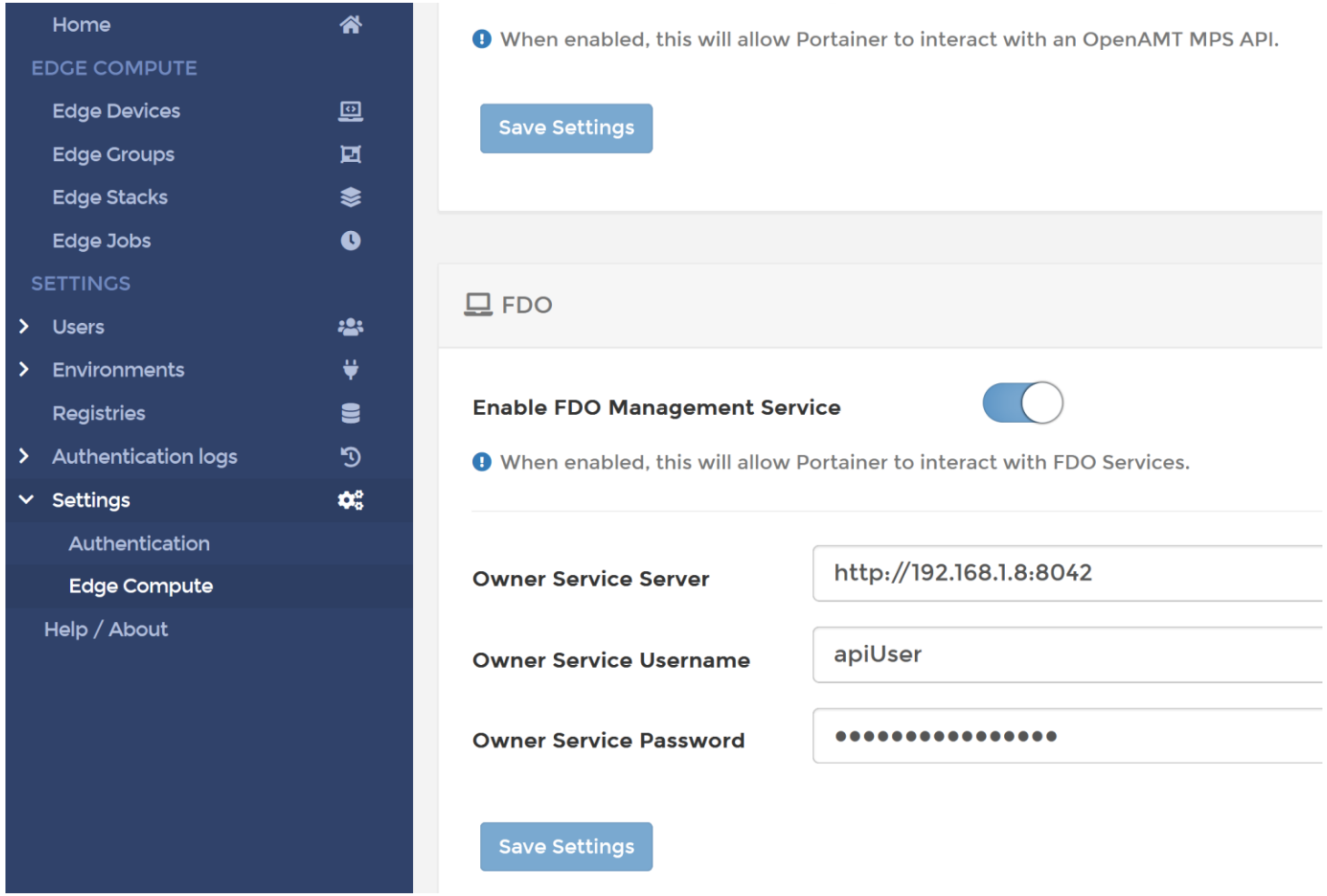
Step 2 get a voucher

Specify Manufacturer Server address, credentials and device serial number



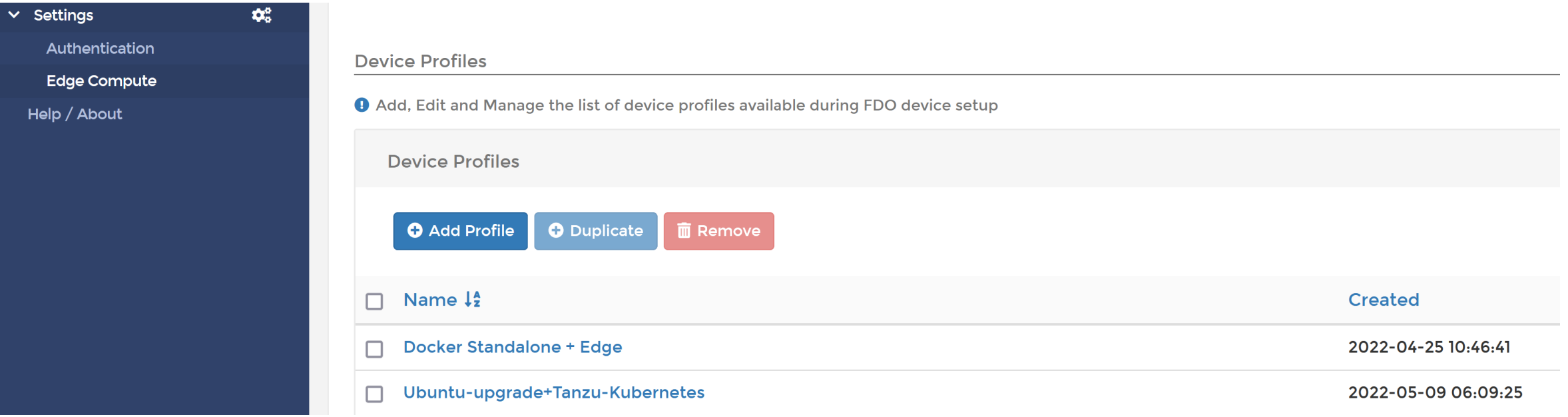
Configure Portainer to onboard device and supply it with a profile

You could write code to do this, or maybe use Postman to make REST calls



Create a device profile

This one updates Linux and Installs Kubernetes
profile is in deck

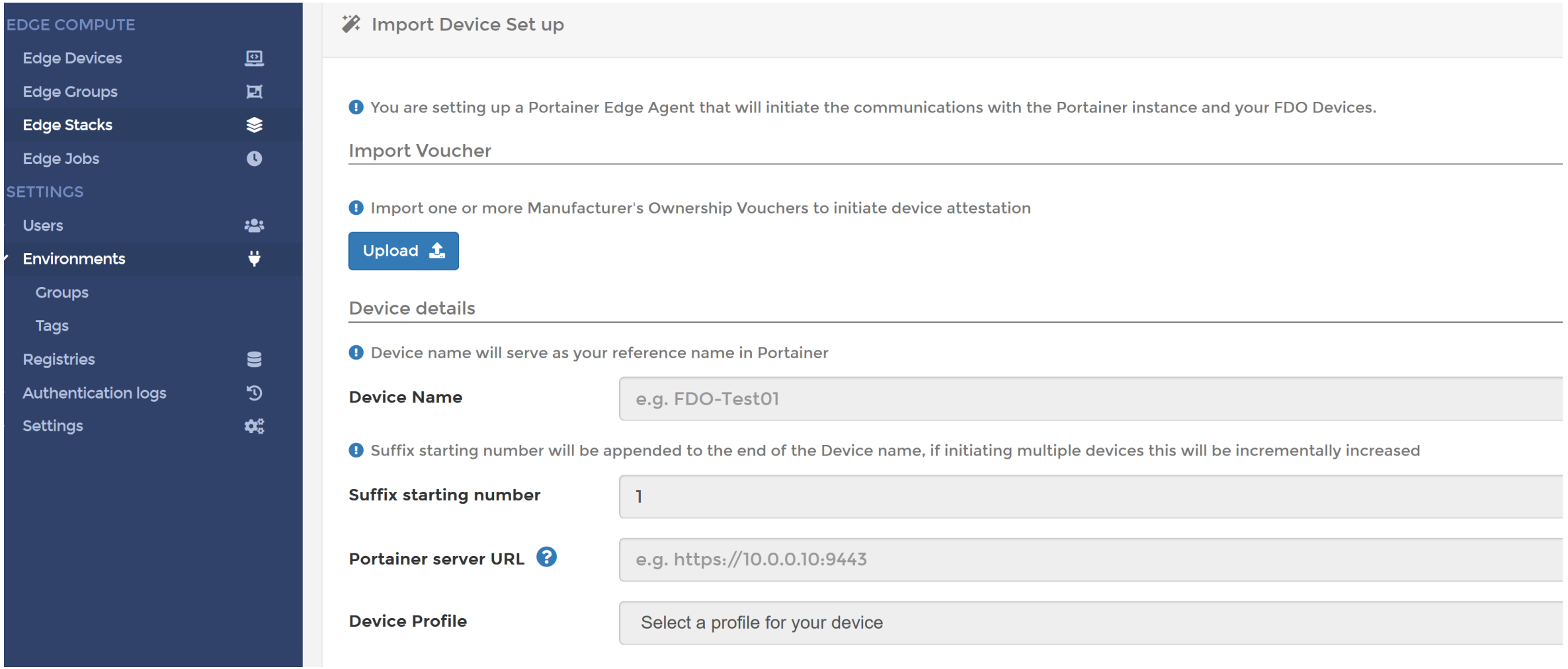


Profile content

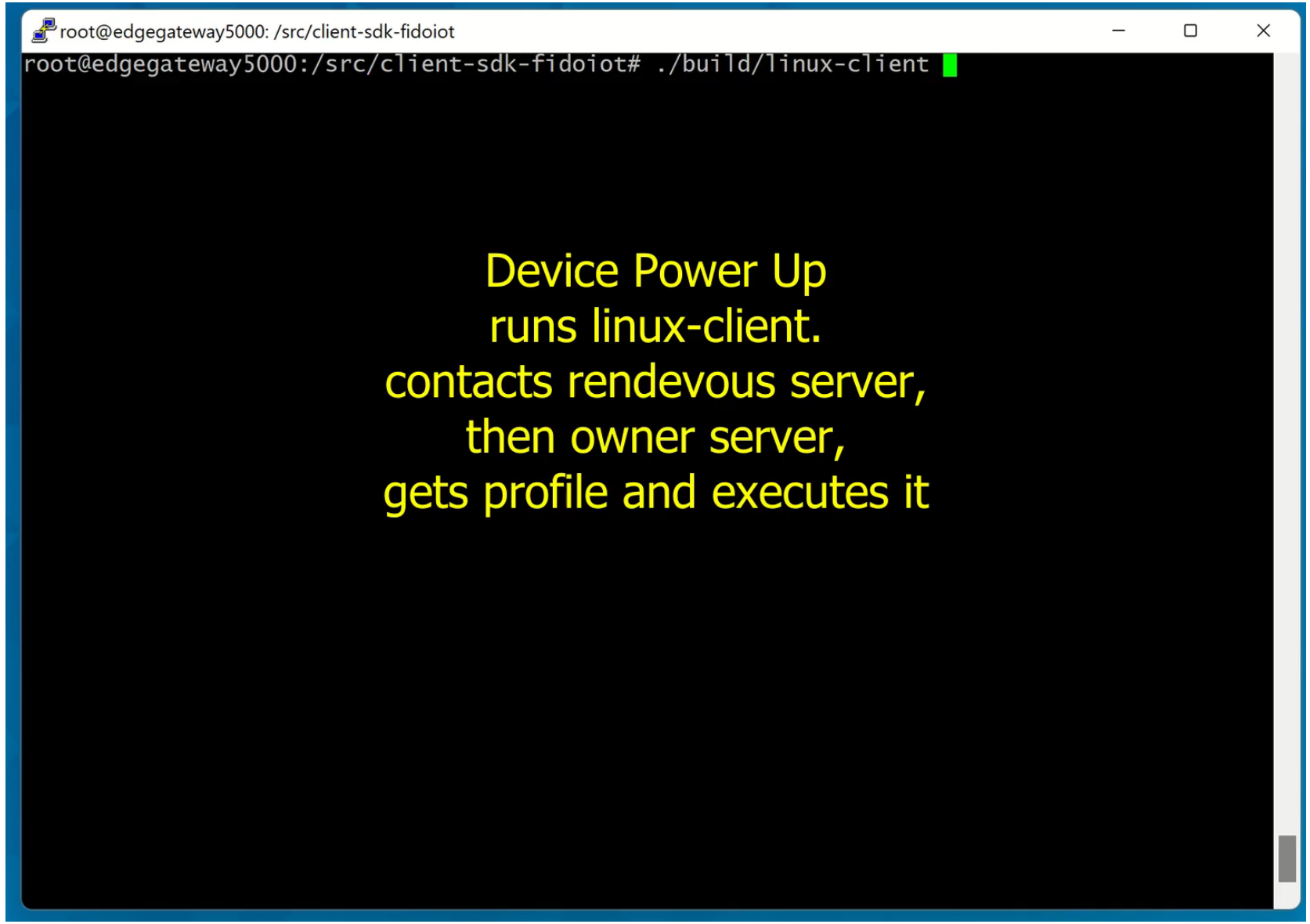
```
#!/bin/bash

env > env.log

export AGENT_IMAGE=portainer/agent:2.11.0
export GUID=$(cat /src/client-sdk-fidoiot/DEVICE_GUID.txt)
export DEVICE_NAME=$(cat /src/client-sdk-fidoiot/DEVICE_name.txt)
export EDGE_ID=$(cat /src/client-sdk-fidoiot/DEVICE_edgeid.txt)
export EDGE_KEY=$(cat /src/client-sdk-fidoiot/DEVICE_edgekey.txt)
export AGENTVOLUME=/src/client-sdk-fidoiot/data/portainer_agent_data/

# update OS since it may have stayed in the distribution channel for a long time
sudo apt update -y
sudo apt full-upgrade -y
sudo apt autoremove -y
sudo apt clean -y
sudo apt autoclean -y

# we are assuming curl and Docker are already installed

#Install kubectl
curl -LO https://dl.k8s.io/release/v1.22.8/bin/linux/amd64/kubectl
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

# create a user account for running Kubernetes
adduser sdouser --disabled-password --home "/home/sdouser" --gecos "Sdo User"
usermod -aG sudo,docker sdouser

# install tanzu cli
cd /home/sdouser
su -c "curl -LO https://github.com/vmware-tanzu/community-edition/releases/download/v0.12.0/tce-linux-amd64-v0.12.0.tar.gz" sdouser
su -c "tar xzvf tce-linux-amd64-v0.12.0.tar.gz" sdouser
cd tce-linux-amd64-v0.12.0

export ALLOW_INSTALL_AS_ROOT=true
./install.sh

# deploy a k8s cluster
tanzu uc create mycluster

curl https://downloads.portainer.io/portainer-ce211-edge-agent-setup.sh | bash -s -- ${EDGE_ID} ${EDGE_KEY} 1
```

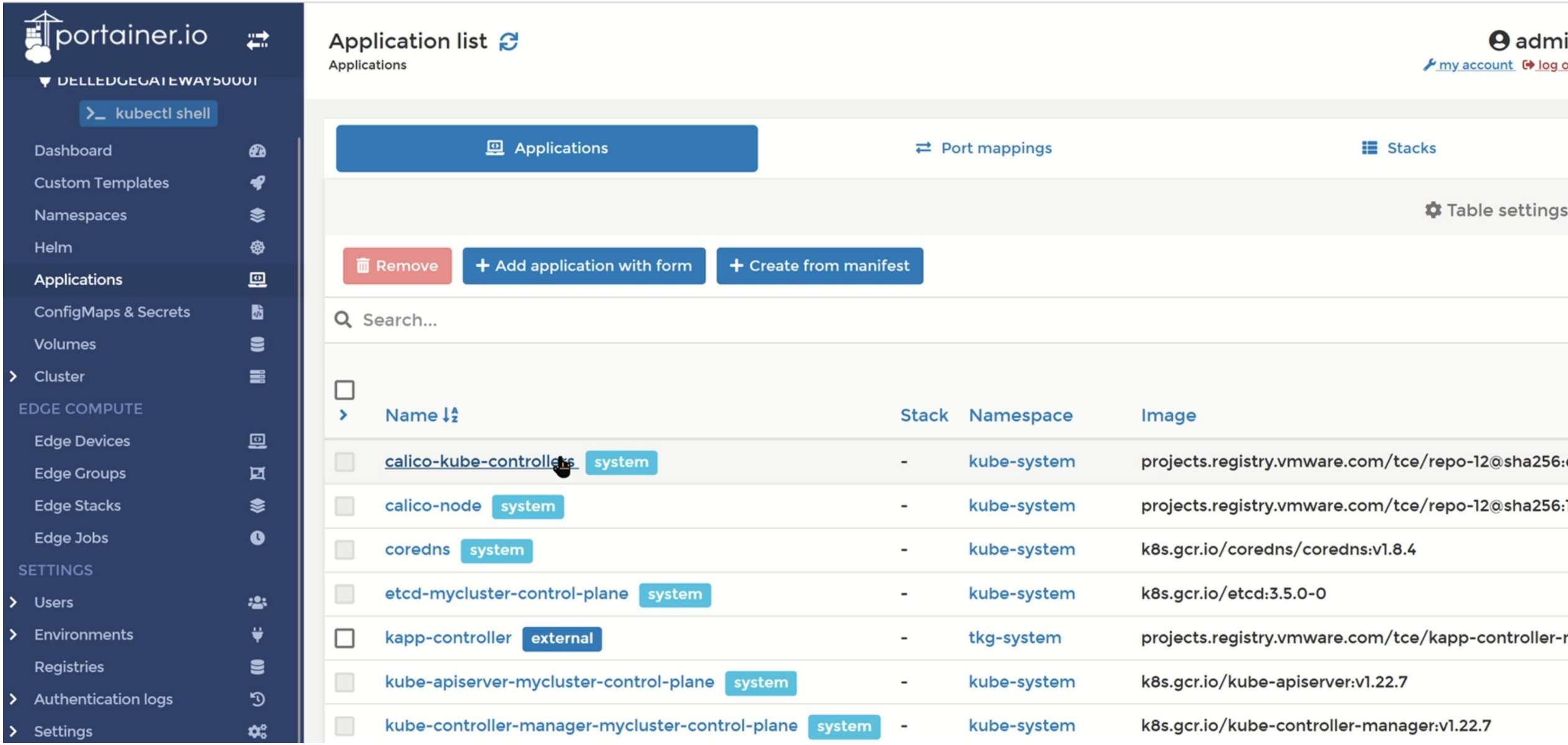
Get Rendezvous and Owner servers ready for the device

Upload the voucher, name the device, pick the profile we just created from a drop down menu

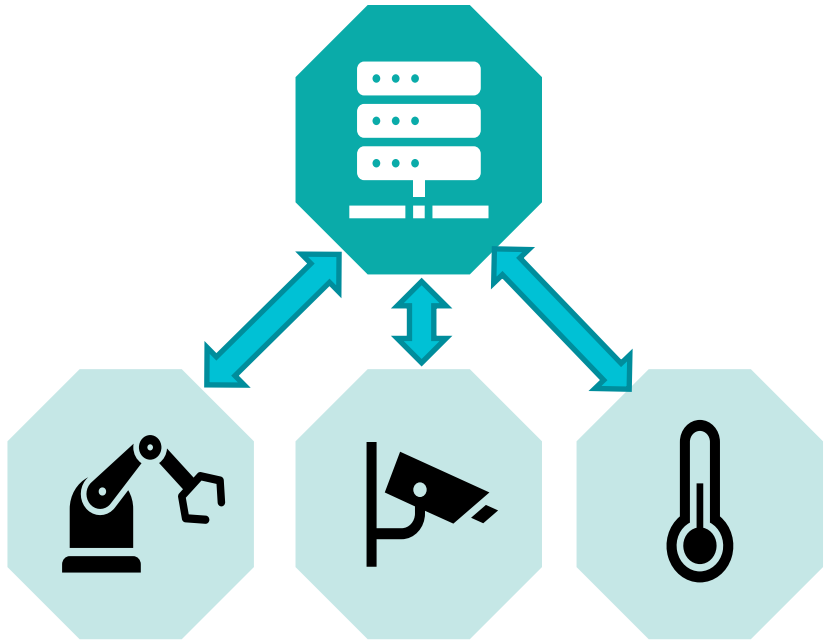


Turn on device (runs /src/client/client-sdk-fidoiot/build/linux-client)




portainer.io
kubectl shell
Dashboard
Custom Templates
Namespaces
Helm
Applications
ConfigMaps & Secrets
Volumes
Cluster
EDGE COMPUTE
Edge Devices
Edge Groups
Edge Stacks
Edge Jobs
SETTINGS
Users
Environments
Registries
Authentication logs
Settings
Application list
Applications
admi
my account
Applications
Port mappings
Stacks
Table settings
Remove
+ Add application with form
+ Create from manifest
Search...
Name
Stack
Namespace
Image
calico-kube-controllers system - kube-system projects.registry.vmware.com/tce/repo-12@sha256:
calico-node system - kube-system projects.registry.vmware.com/tce/repo-12@sha256:
coredns system - kube-system k8s.gcr.io/coredns/coredns:v1.8.4
etcd-mycluster-control-plane system - kube-system k8s.gcr.io/etcd:3.5.0-0
kapp-controller external - tkg-system projects.registry.vmware.com/tce/kapp-controller-
kube-apiserver-mycluster-control-plane system - kube-system k8s.gcr.io/kube-apiserver:v1.22.7
kube-controller-manager-mycluster-control-plane system - kube-system k8s.gcr.io/kube-controller-manager:v1.22.7

How can we easily give servers at the edge access to the variety of devices around them?



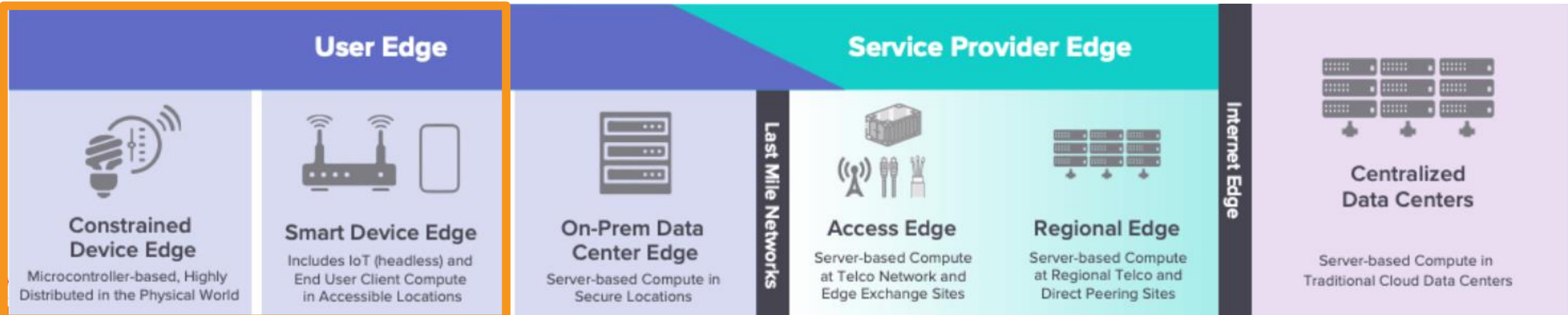
Tiny Edge

Edge nodes are surrounded by a variety of IoT devices

These IoT devices are often too small to run Kubernetes

How can these devices be dynamically leveraged and managed by Kubernetes workloads?

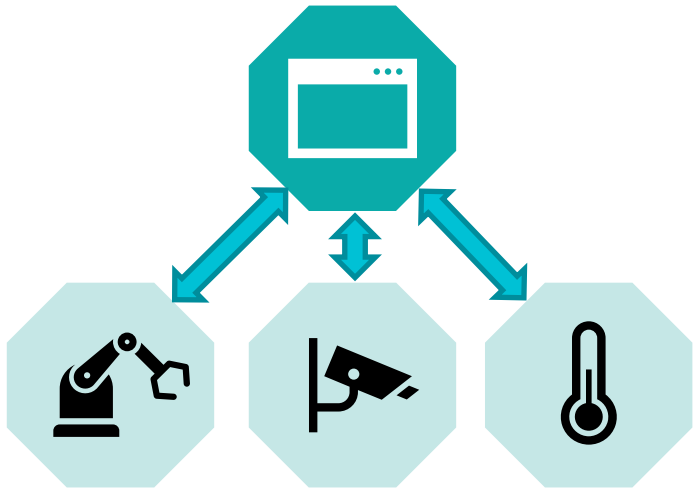
Source: LF Edge white paper *Sharpening the Edge: Overview of the LF Edge Taxonomy and Framework*



Akri: A Kubernetes Resource Interface for the edge

- Discovers IoT devices. Handles dynamic appearance and disappearance of devices
- Connects IoT devices to a Kubernetes cluster by representing them as a native Kubernetes resources
- Schedules workloads based on what devices are connected to the cluster
- Built with Rust to optimize for the edge and low footprint clusters (K3s, MicroK8s, etc.)
- Open-source CNCF Sandbox Project

Requesting Akri Kubernetes Resources

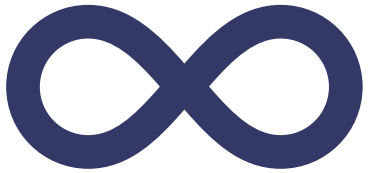


```yaml
apiVersion: v1
kind: Pod
metadata:
  name: user-monitoring-app
spec:
  containers:
  - name: user-monitoring-app
    image: user-monitoring-app:latest
    resources:
      requests:
        memory: "10Mi"
        akri.sh/akri-udev-therm-$ID1: "1"
        akri.sh/akri-onvif-video-$ID2: "1"
        akri.sh/akri-opcua-robot-$ID3: "1"
```

https://docs.akri.sh/user-guide/requesting-akri-resources

Akri for device use and management

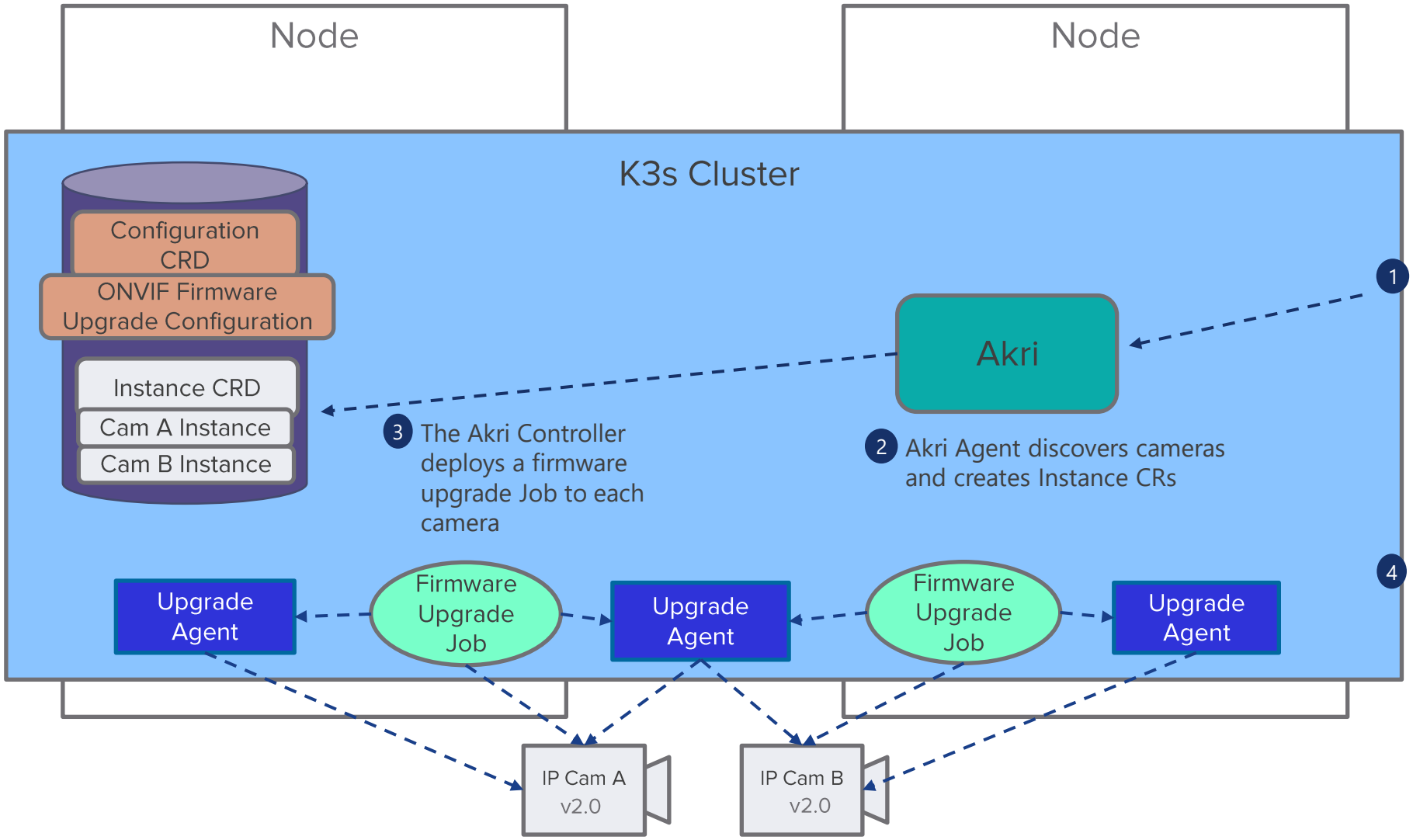
Configure Akri's Controller to automatically deploy Kubernetes Pods or Jobs to discovered devices.



Long-running Kubernetes Pods – Ideal for device use

Single-task Kubernetes Jobs – Ideal for device management (introduced in Akri v0.8.4)

IP Camera Firmware Upgrade with Akri



https://www.cncf.io/online-programs/cncf-live-webinar-discovering-and-managing-iot-devices-from-kubernetes-with-akri/

Integrating and Extending Akri

Goal: The open source, standard way to connect devices to a cluster and manage devices from a cluster

- A platform made to be pluggable and extensible
- Plug in your workloads: Enables you to declaratively state what devices your workloads need
- Discover your devices: Create new Discovery Handlers (i.e. proprietary, Modbus, CoAP, ZeroConf, MQTT)
- Integrate: existing upgrade solutions, device registry, certificate store solution
- Let us know where we can grow!

Akri Resources

Learn more

docs.akri.sh

CNCF Webinar: Discovering & Managing IoT Devices with Akri

https://youtu.be/9wCQCV0m5Kk

Join the community via Slack and monthly Zooms!

K8s #akri Slack

Try it out: End to end demo discovering mock USB cameras

https://docs.akri.sh/demos

A Transition: IoT Edge WG moving to CNCF

Goal: Build cohesion around heterogeneity of devices and solutions on the edge

Grown beyond Kubernetes specific solutions

Transitioning from Kubernetes to CNCF under Runtime TAG: https://github.com/cncf/tag-runtime#working-groups

Tune into existing Slack to stay updated and involved in new charter and WG resources

Kubernetes IoT Edge WG Resources

GitHub (current, future)

Previous working group meetings recordings

Join the community via Slack and bi-monthly Zooms!

K8s #wg-iot-edge Slack

Google group https://groups.google.com/forum/#!forum/kubernetes-wg-iot-edge

Speaker Contact Information

Steven Wong | VMware

@cantbewong

@cantbewong

Kate Goldenring | Microsoft

@KateGoldenring

@kate_goldenring

Q&A

This deck is available here:
https://via.vmw.com/EiAB

Thank You

IoT edge WG Slack

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