IoT at the Edge with K3s and Akri

Andrés Valero
SUSE

KCD SPAIN

Sobre mí

Andrés Valero

Posición actual:
- Technical Marketing Manager, Edge

Experiencia:
- Marketing Técnico
- Arquitecto de Soluciones
- Consultor
- Ventas

Contacto:
- andres.valero@suse.com
- https://www.linkedin.com/in/avaleror/

Intro al Edge Computing

Edge Computing es un paradigma de computación distribuida que acerca la computación y el almacenamiento de datos a la fuente de generación de datos, permitiendo el procesamiento y análisis en tiempo real.

Principios fundamentales:

1. **Proximidad:** Al procesar datos cerca de su origen, la Computación en el Borde minimiza la latencia y reduce la necesidad de que los datos recorran largas distancias para llegar a un centro de datos centralizado.
2. **Optimización del Ancho de Banda:** Al procesar y filtrar datos en el borde, solo se transmite a la nube o al centro de datos la información relevante, optimizando el uso del ancho de banda.
3. **Respuesta en Tiempo Real:** El "Edge Computing" trata de procesar datos lo más cerca posible de su origen, lo que la hace ideal para aplicaciones que requieren baja latencia y toma de decisiones en tiempo real.
4. **Operación sin Conexión:** Los dispositivos en el "Edge" pueden funcionar de manera autónoma incluso sin una conexión de red, asegurando un funcionamiento ininterrumpido en entornos remotos o desconectados.

Internet of Things (IoT)

Definición:
El IoT es una red de dispositivos interconectados que se comunican e intercambian datos. Estos dispositivos van desde los dispositivos Alexa y termostatos de tu hogar hasta maquinaria industrial. Conectados a través de una red, estos dispositivos integran los datos físicos recopilados con la computación, haciendo que estos datos estén fácilmente disponibles para su procesamiento.

IoT y Edge Computing:
El Internet de las Cosas (IoT) y el "Edge Computing" están facilitando un salto tecnológico no visto desde la Revolución Industrial en el siglo XVIII. Sectores como la agricultura, la manufactura, la logística, la automoción y el comercio minorista están aprovechando el poder del "Edge Computing" y el IoT para volverse más eficientes, sostenibles y adaptables.

La problemática actual

- Los dispositivos IoT son difíciles de integrar o gestionar de una forma sencilla.
- Los diferentes protocolos, complican extremadamente la gestión.
- El mercado tiende a usar K8s en el “Edge” pero la integración con los dispositivos IoT y su gestión es compleja.
- La separación “IT/OT” es todavía un problema para el mundo Industrial.
- La gestión como código de este tipo de dispositivos es bastante compleja y en la mayoría de los casos no es posible.

¿Cómo solucionar el problema?

- Necesitamos una manera para poder simplificar la brecha “IT/OT” unificando la forma de trabajo para estos dos mundos.
- Es necesario unificar la gestión de los diferentes protocolos de los dispositivos IoT.
- La integración con soluciones utilizadas en el “Edge Computing” como Kubernetes.
- La gestión como código de los dispositivos IoT integrados en Kubernetes proporciona una flexibilidad desconocida en el mundo del IoT.

Proyecto Akri

¿Qué es Akri?

Akri es una Interfaz de Recursos de Kubernetes que te permite exponer fácilmente dispositivos heterogéneos periféricos (como cámaras IP y dispositivos USB) como recursos en un clúster de Kubernetes, al mismo tiempo que admite la exposición de recursos de hardware integrados, como GPUs y FPGAs. Akri detecta continuamente nodos que tienen acceso a estos dispositivos y programa cargas de trabajo basadas en ellos.

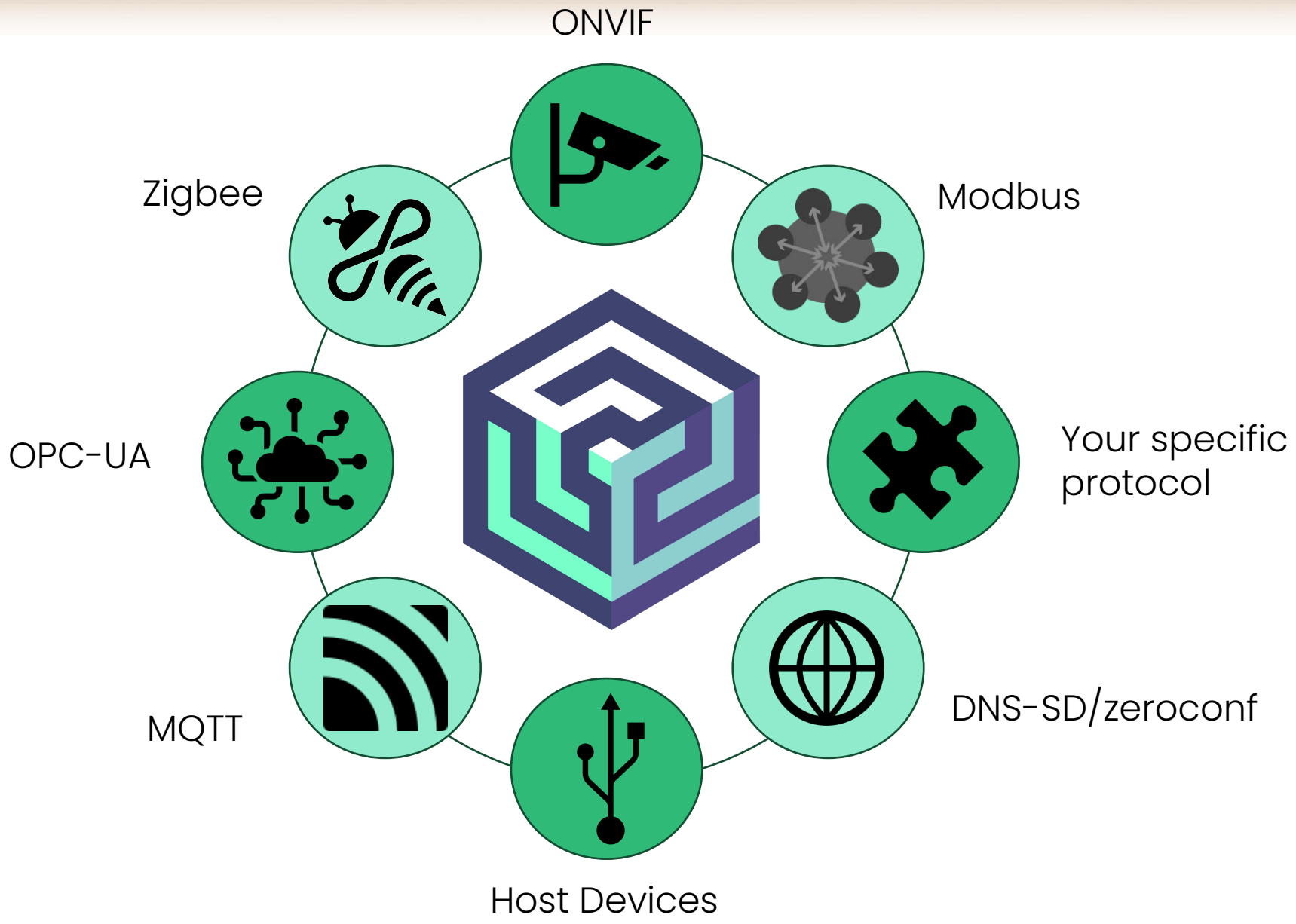
Akri es parte de la Cloud Native Computing Foundation (CNCF) como Sandbox project.

https://github.com/project-akri/akri

https://docs.akri.sh/

Akri

Peripheral Discovery

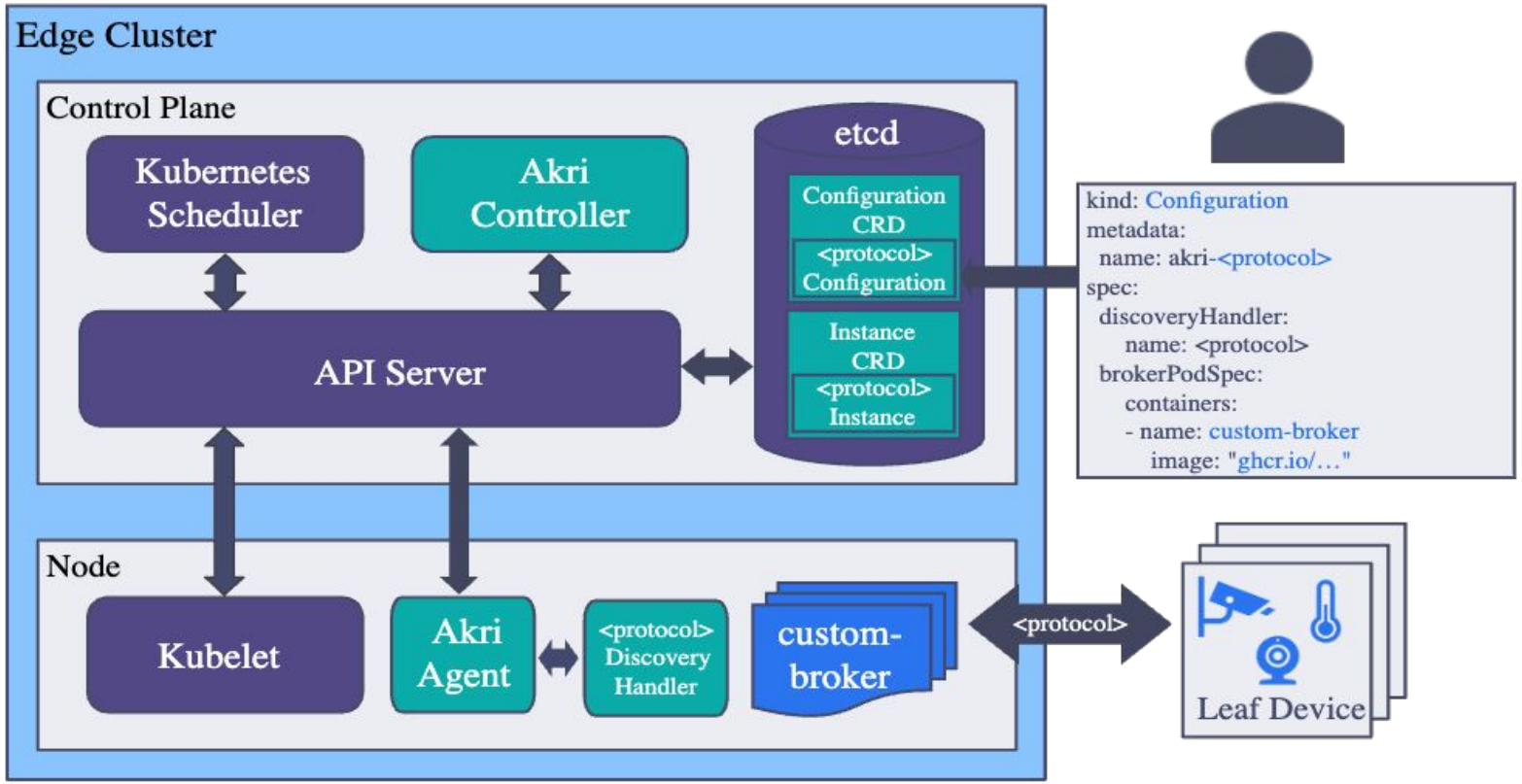


Arquitectura de Akri

Componentes:

1. CRD's
2. Agent
3. Controller
4. Discovery Handlers

```yaml
apiVersion: akri.sh/v0
kind: Configuration
metadata:
  name: akri-udev-video
spec:
  discoveryHandler:
    name: udev
    discoveryDetails: |+
        udevRules:
        - 'KERNEL=="video[0-9]*"'
  brokerSpec:
    brokerPodSpec:
      containers:
      - name: akri-udev-video-broker
```



https://docs.akri.sh/architecture/architecture-overview

Arquitectura de Akri

Agent

El agente Akri implementa los Plugins de Kubernetes para los dispositivos descubiertos. El flujo básico del Agente Akri es el siguiente:

1. Observar cambios en la configuración para determinar qué recursos buscar.
2. Monitorear la disponibilidad de recursos (ya que los dispositivos periféricos pueden aparecer y desaparecer) para determinar qué recursos publicar.
3. Informar a Kubernetes sobre la salud/disponibilidad de los recursos a medida que cambia.

Controller

El controlador Akri cumple dos propósitos:

1. Gestionar (crear y/o eliminar) los Pods y Servicios que permiten la disponibilidad de recursos.
2. Asegurar que las Instancias estén alineadas con el estado del clúster en cualquier momento dado.

Para lograr estos objetivos, el flujo básico del controlador es el siguiente:

1. Observar cambios en las Instancias para determinar qué Pods y Servicios deberían existir.
2. Observar los Nodos que están contenidos en instancias que ya no existen.

Discovery Handlers

Los Discovery Handlers descubren dispositivos alrededor del clúster, ya sea conectados a Nodos (por ejemplo, sensores USB), integrados en Nodos (por ejemplo, GPUs) o en la red (por ejemplo, cámaras IP), y los presentan al Agente.

Configuraciones y Templates en Akri

Templates

Helm nos permite parametrizar los campos comúnmente modificados en nuestras plantillas de configuración (templates), y podemos proporcionar muchos de ellos (para verlos, ejecuta helm inspect values akri-helm-charts/akri). Para cambios de configuración más avanzados que no son facilitados por un Helm chart.

Akri templates: https://github.com/project-akri/akri/tree/main/deployment/helm/templates

Ejemplo de uso:

```
helm template akri akri-helm-charts/akri \
    --set onvif.configuration.enabled=true \
    --set onvif.configuration.brokerPod.image.repository=nginx \
    --set rbac.enabled=false \
    --set controller.enabled=false \
    --set agent.enabled=false > configuration.yaml
```

K3S - Kubernetes para IoT y Edge

K3S es la distribución de Kubernetes más utilizada en el “Edge”

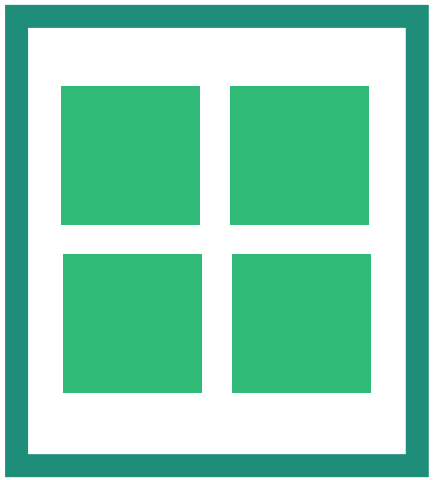
- Distribución de Kubernetes diseñada para IoT y Edge computing
- Lista para Producción – testeada y ampliamente utilizada para “Edge” y “Data Centre”
- Ligera, escalable y fácilmente operable. Lista para desplegar miles de clústeres en el Edge.
- Solo un binario, ligera, optimizada para x86 y Arm.
- Sencillo de instalar y actualizar

K3S
CNCF Project
Created by Rancher

20,000+ GitHub stars for K3s.

Clasificación del Edge según la distancia al DC

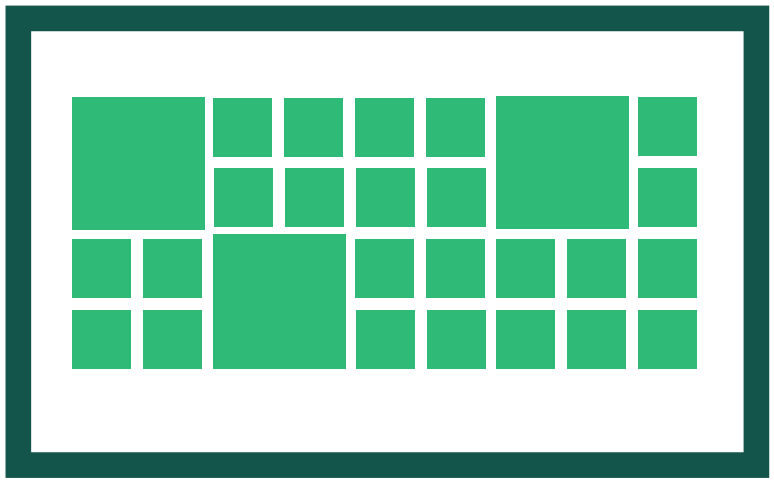
NEAR Edge



10s to 100s devices

Closer to data center

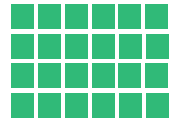
FAR Edge



100s to 1000s devices

On-site, Farthest from data center, Closer to users

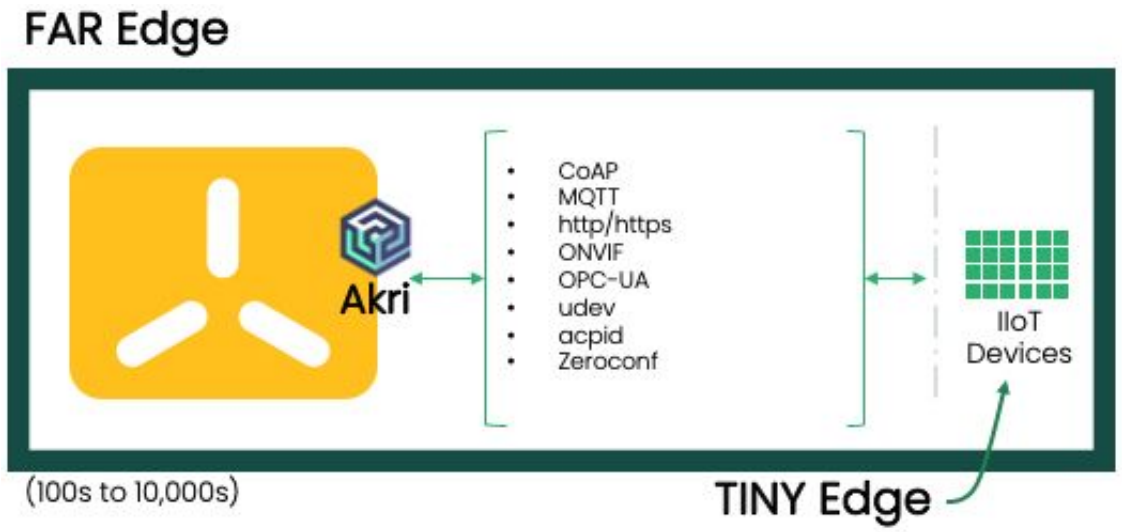
Tiny Edge



> 1000s devices

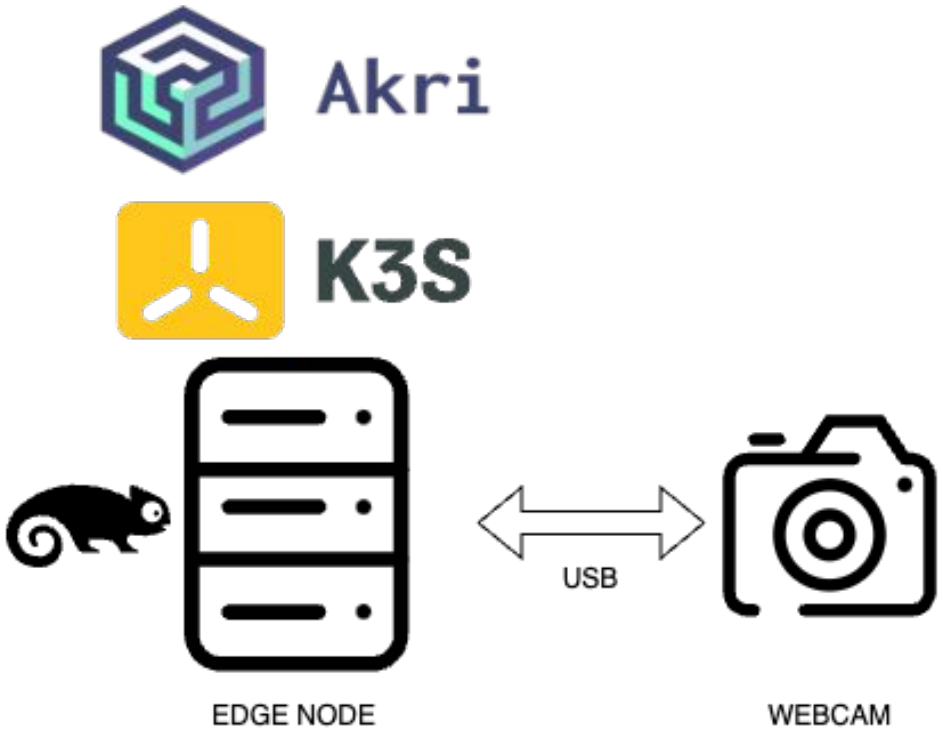
End-point itself
E.g., **Industrial IoT**

K3s y Akri como solución



- K3s ofrece la base Kubernetes dónde corre Akri.
- Akri es el “traductor” entre los diferentes protocolos de los dispositivos IoT y Kubernetes.
- Cuando un nuevo dispositivo se conecta es detectado por Akri y es gestionado como un objeto Kubernetes.
- Una vez el nuevo dispositivo IoT es gestionado por Akri. En ese momento, otros contenedores corriendo en el clúster pueden conectar con el dispositivo y usarlo.

Laboratorio



1 Edge node:
- N95 Processor
- 16Gb
- SSD disk

1 Webcam

Software:
- OpenSUSE
- K3s
- Akri
- Demo Application

Demo

 # SSH en el nodo Edge e instalar K3s
ssh user@<edge node IP address>
curl -sfL https://get.k3s.io | sh -

 # Configurar Kubeconfig
export KUBECONFIG=/etc/rancher/k3s/k3s.yaml

 # Instalar Akri Helm repo
helm repo add akri-helm-charts https://project-akri.github.io/akri/

 # Actualizar helm
helm repo update

 # Configurar crictl para K3s
export AKRI_HELM_CRICTL_CONFIGURATION="--set kubernetesDistro=k3s"
echo $AKRI_HELM_CRICTL_CONFIGURATION

Demo

 # Instalar & configurar Akri

```
helm install akri akri-helm-charts/akri $AKRI_HELM_CRICTL_CONFIGURATION \
--set udev.discovery.enabled=true \
--set udev.configuration.enabled=true \
--set udev.configuration.name=akri-udev-video \
--set udev.configuration.discoveryDetails.udevRules[0]='KERNEL=="video[0-9]*"' \
--set udev.configuration.brokerPod.image.repository="ghcr.io/project-akri/akri/udev-video-broker"
```

 # Control

```
kubectl get -o wide pods | grep controller
kubectl get -o wide pods | grep agent
kubectl get nodes,akric,akrii,pods -o wide
```

 # Desplegar la aplicación de video

```
kubectl apply -f
https://raw.githubusercontent.com/project-akri/akri/main/deployment/samples/akri-video-streaming-app.yaml
```

Demo

 # Espera hasta que los pods estén activos
watch kubectl get pods

 # Puerto de la streaming app
kubectl get service/akri-video-streaming-app
--output=jsonpath='{.spec.ports[?(@.name=="http")].nodePort}' && echo

 # SSH forwarding para acceder la aplicación
ssh user@<edge node IP address> -L 50000:localhost:<streaming-app-port>
navigate to http://localhost:50000/

 # Desinstalar la aplicación de vídeo
kubectl delete service akri-video-streaming-app
kubectl delete deployment akri-video-streaming-app
kubectl delete akric akri-udev-video
watch kubectl get pods,services,akric,akrii -o wide #wait until the pods are stopped

Demo

 # Desinstalar Akri

helm delete akri
kubectl delete crd instances.akri.sh
kubectl delete crd configurations.akri.sh

 # Grabación en asciinema de la demo

Cómo visualizar el vídeo.
https://github.com/avaleror/event-talks/blob/master/2023-KCD-Spain/demo-video/README.md

https://raw.githubusercontent.com/avaleror/event-talks/master/2023-KCD-Spain/demo-video/tmp7jwao48n-ascii.cast