Kubernetes for EPICS IOCs

Spring 2021 EPICS Collaboration Meeting

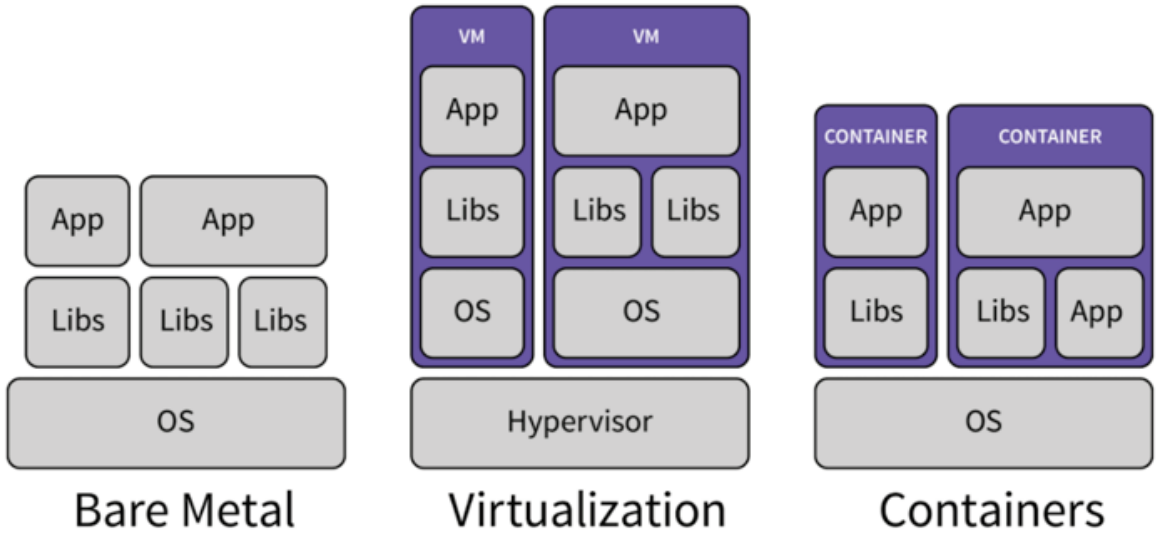
Giles Knap

08/07/2021

Kubernetes for EPICS IOCs

- Applying modern industry standards to manage EPICS IOCs
  - Containers
    - Package IOC software and execute it in a lightweight virtual environment
  - Kubernetes
    - Centrally orchestrate all IOCs at a facility
  - Helm Charts
    - Deploy IOCs into Kubernetes with version management
  - Repositories
    - Source, container and helm repositories manage all the above assets. No shared filesystems required.
  - Continuous Integration / Delivery
    - Source repositories automatically build assets from source when it is updated.

Introduction to Containers



Containers, like VMs, isolate an application and its dependencies into a self-contained unit that can run anywhere.

Lightweight

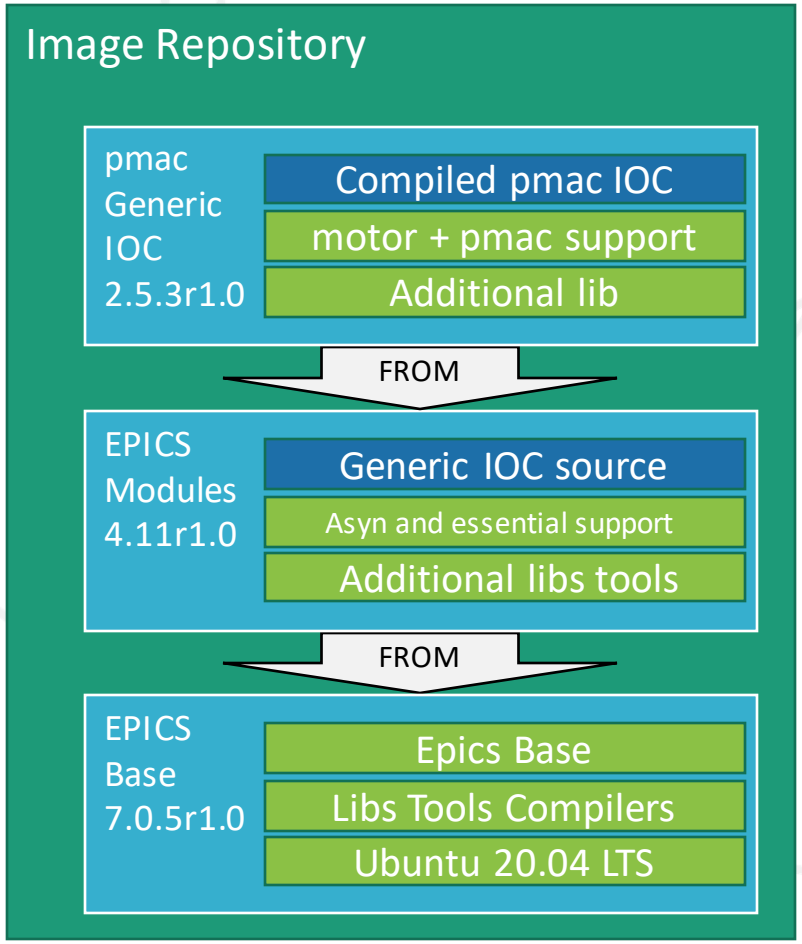
- starts with just one process
- shares OS between all containers

A container's virtual filesystem is initialized with an image.

Image registries like Docker Hub hold many useful predefined images.

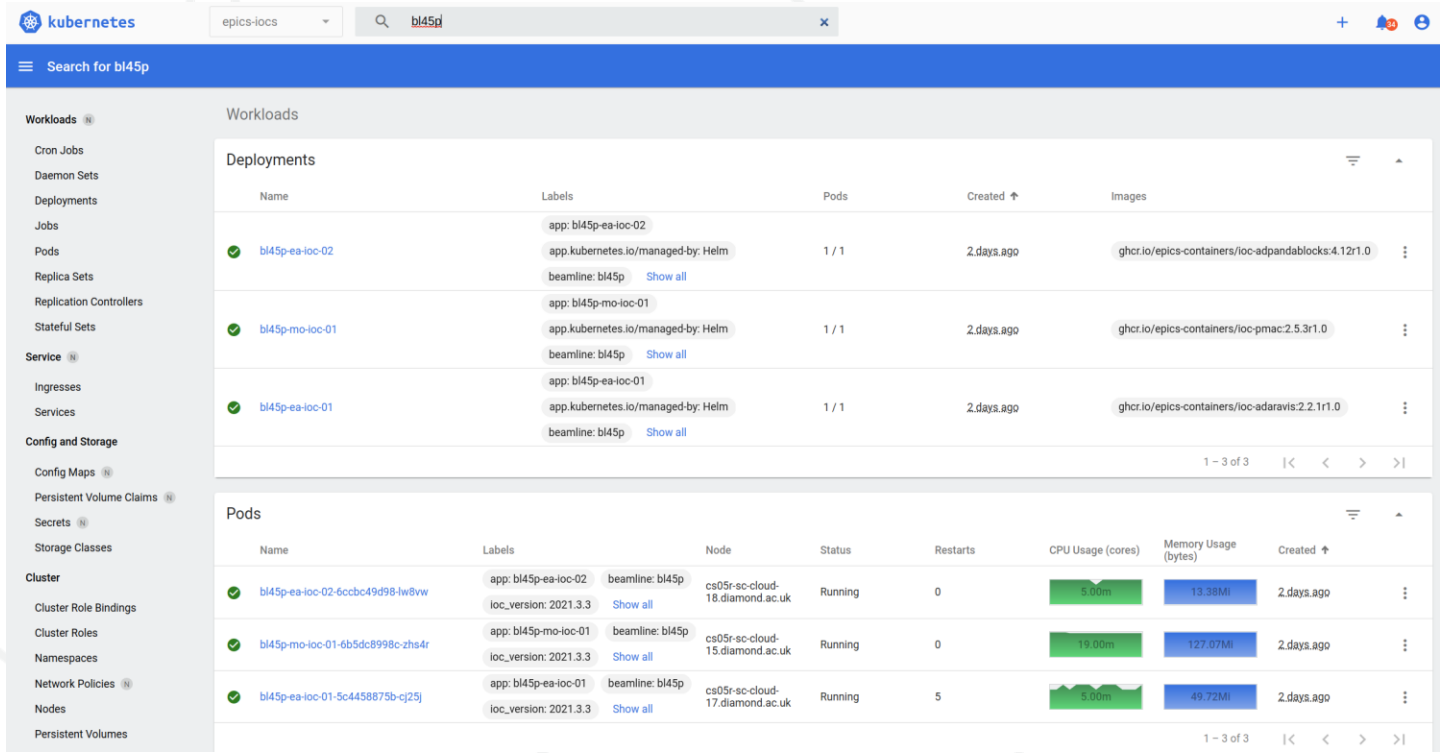
New custom images can be derived from these by adding extra layers of software.

IOCs in containers



- IOC images represent a **generic** IOC
- The same image is used for every IOC that controls a given class of device
- An IOC **instance** is a container based on a **generic** IOC image with an added startup script that makes it unique
- This example shows the filesystem layers in the **generic** IOC for the standard DLS Motion Controller

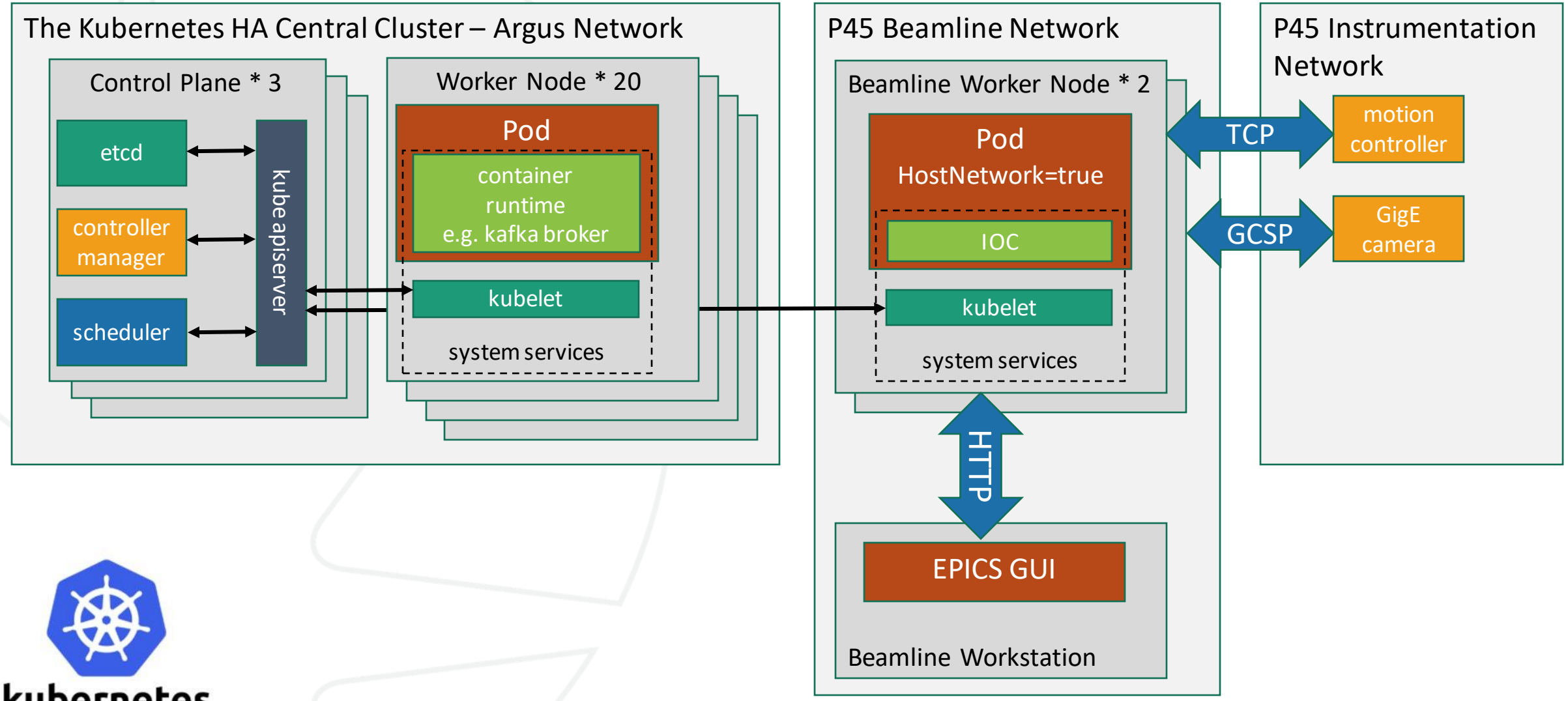
Introduction to Kubernetes https://kubernetes.io



- Kubernetes easily and efficiently manages containers across clusters of hosts.
- It builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community
- Today it is by far the dominant orchestration technology for containers
- Many household names have adopted it:
  - CERN
  - Spotify
  - Box
  - Many more .. https://kubernetes.io/case-studies/
- This screenshot shows the standard K8s dashboard managing P45 IOCs

DLS Kubernetes Cluster

Now extended to beamlines

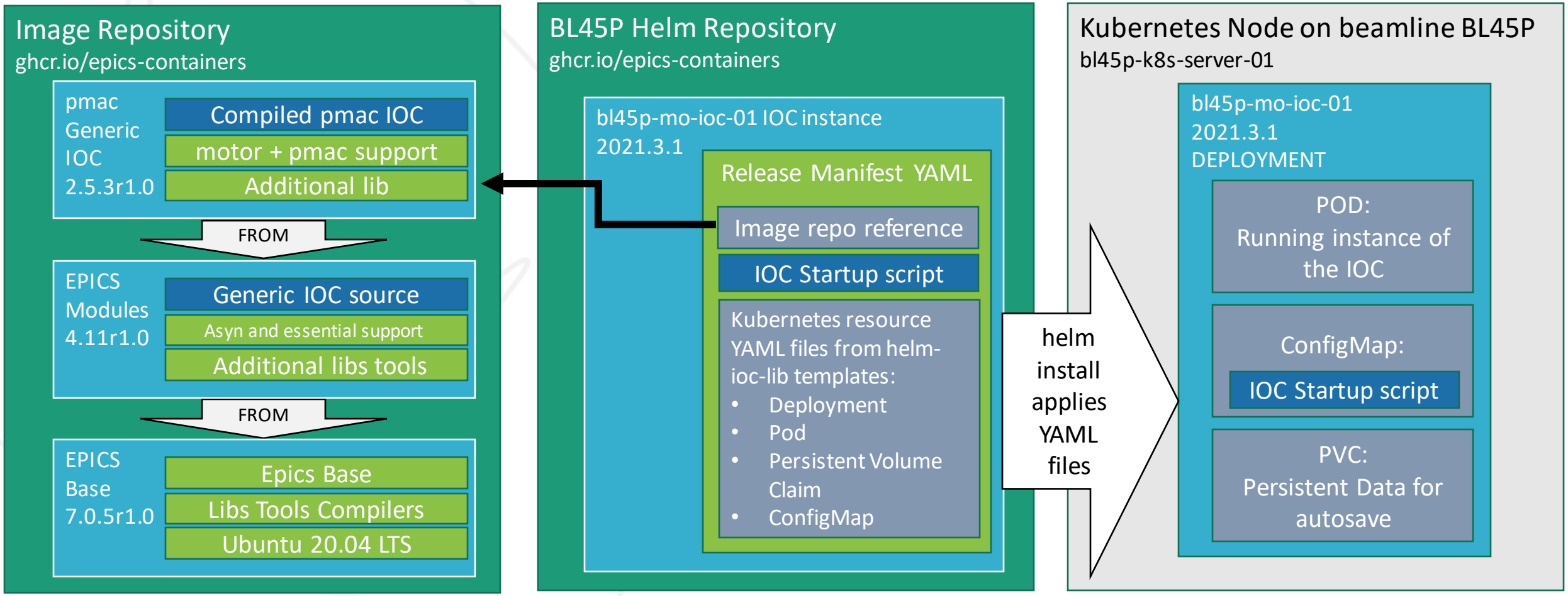


kubernetes

Introduction to Helm

- Helm is the most popular package manager for K8S applications
- The packages are called Helm Charts
- Charts contain templated YAML files to define a set of resources to apply to a Kubernetes cluster
- Helm has functions to deploy Charts to a cluster and manage multiple versions of the chart within the cluster
- It also supports registries for storing version history of charts, much like Docker Hub

An Example IOC bl45p-mo-ioc-01



- A Helm Chart defines an IOC instance: IMAGE + STARTUP SCRIPT + K8S DEPLOYMENT YAML
- The entire definition of the P45 beamline is held in https://github.com/orgs/epics-containers/packages

Repositories and Continuous Integration

- Kubernetes for EPICS IOCs uses these types of repository:
  - Beamline definition source repo
    - CI publishes a helm chart for each IOC instance to the beamline Helm Repo
  - Generic IOC images source repo
    - CI publishes a generic IOC image to the image repo
  - Images
    - 1 production image repository for all released generic IOCs
  - Helm Charts
    - 1 repository per beamline, with 1 chart per IOC instance
- Helm deploys to Kubernetes directly from the repository
- Kubernetes pulls generic IOC images directly from the image repository
- No intermediate shared filesystem is required.

Features Provided by Kubernetes and Helm

- Auto start IOCs when servers come up
- Restart crashed IOCs
- Manually Start and Stop IOCs
- Allocate the server which runs an IOC
- Move IOCs if a server fails
- Throttle IOCs that exceed CPU limit
- Restart IOCs that exceed Memory limit

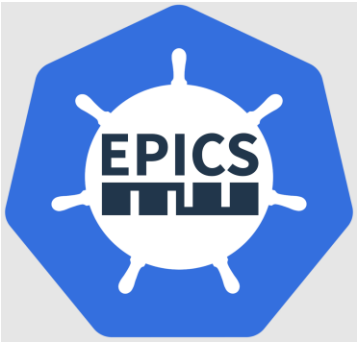
- Deploy versioned IOCs to the beamline
- Track historical IOC versions
- Rollback to a previous IOC version
- Monitor IOC status and versions
- View the current log
- View historical logs (via graylog)
- Connect to an IOC and interact with its shell

Current Status

- A test DLS beamline BL45P has most of its IOCs running on Beamline worker nodes managed by our central cluster.
- All source and assets for the BL45P POC work are published at
  - https://github.com/epics-containers
- The organization includes enough documentation for others to try this approach
  - https://epics-containers.github.io/
- Including a tutorial that walks through setting up a mini Kubernetes cluster and deploying an ADSimDetector IOC.
  - https://epics-containers.github.io/main/tutorials/setup_k8s.html
- Please Join the organization and contribute your own ideas!

Benefits

- Containers are decoupled from the host OS and each other.
  - This is VERY good news for collaboration:
    - All BL45P IOCs dependencies are vanilla EPICS Community versions
  - Isolation protects against most security vulnerabilities
  - Run anywhere: develop, test, demo on a laptop or home machine.
- Kubernetes provides economy of scale through centralized:
  - Software deployment and management
  - Logging and Monitoring
  - Resource management: Disk, CPU, Memory
- Remove maintenance of internal management tools
- Remove need for shared filesystem
- Remove the need to build a binary for every IOC



But isn't it hard to set up Kubernetes?

A quick tutorial review