

Kubernetes for EPICS IOCs

Technical Seminar

giles knap

28/09/2022

<https://github.com/epics-containers/ec-talk>

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
 - Continuous Integration / Delivery
 - Source repositories automatically build assets from source when it is updated.
 - Developer Environment
 - Developers use the same container as Kubernetes, guaranteeing a matching environment.



kubernetes



GitHub



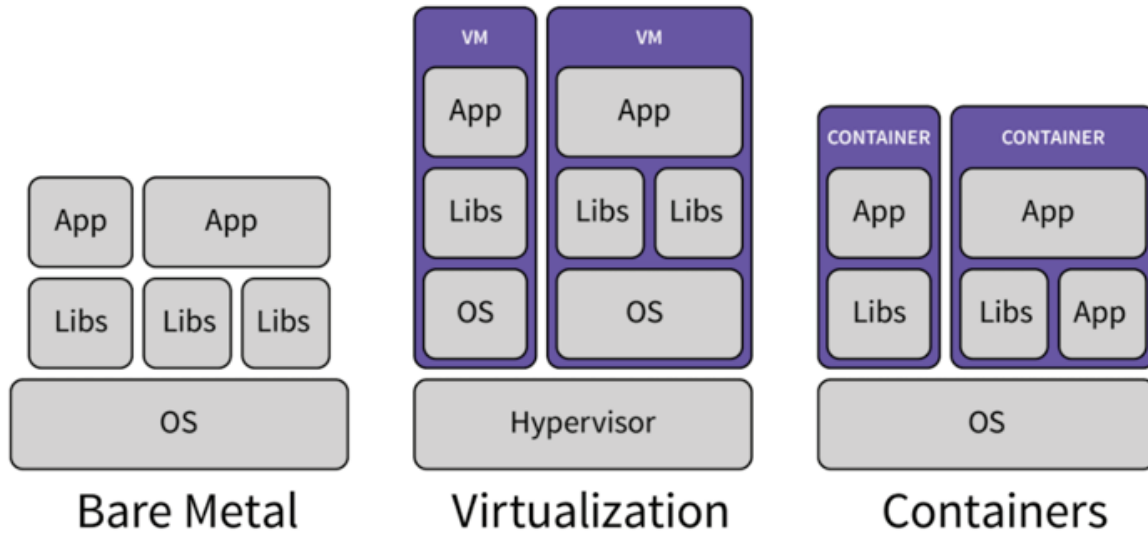
docker



podman



Introduction to Containers



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

Lightweight

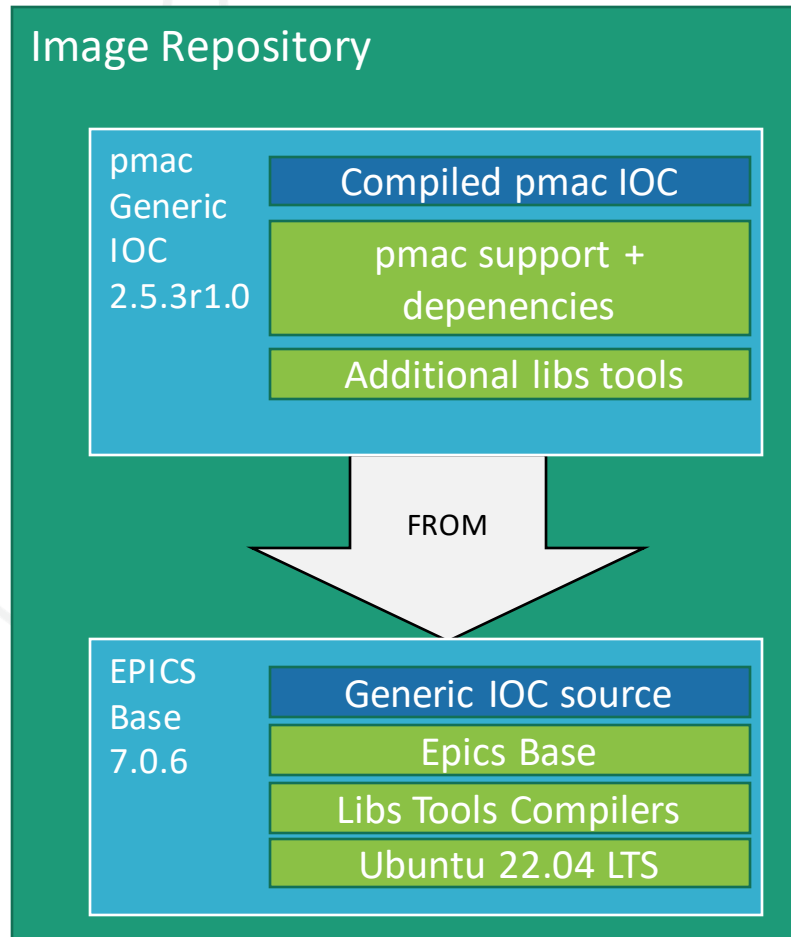
- starts with just one process
- shares the kernel with the host OS

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



Example Container Definition

01 Dockerfile

02 ioc.yaml

From the ioc-pmac project



Takeaways from ioc-pmac Dockerfile

- Repeatability
 - The file precisely defines the environment in which the IOC will run
- Version control
 - The file is text only so git can track the history of changes to the environment
- Layered
 - The container's filesystem adds a new hashed layer for each command in the Dockerfile.
 - Faster builds as layers are cached
 - Saved disk and memory – layers are shared by all container instances
- Staged Build
 - Separate targets provide a lightweight runtime image and fully loaded build /developerimage

Introduction to Kubernetes <https://kubernetes.io>

The screenshot shows the Kubernetes dashboard interface. At the top, there's a search bar with 'bl45p' entered. The left sidebar contains navigation links for Workloads, Service, Config and Storage, and Cluster. The main content area is divided into two sections: 'Workloads' and 'Pods'.

Workloads

Name	Labels	Pods
✓ bl45p-ea-ioc-02	app: bl45p-ea-ioc-02 app.kubernetes.io/managed-by: Helm beamline: bl45p Show all	1 / 1
✓ bl45p-mo-ioc-01	app: bl45p-mo-ioc-01 app.kubernetes.io/managed-by: Helm beamline: bl45p Show all	1 / 1
✓ bl45p-ea-ioc-01	app: bl45p-ea-ioc-01 app.kubernetes.io/managed-by: Helm beamline: bl45p Show all	1 / 1

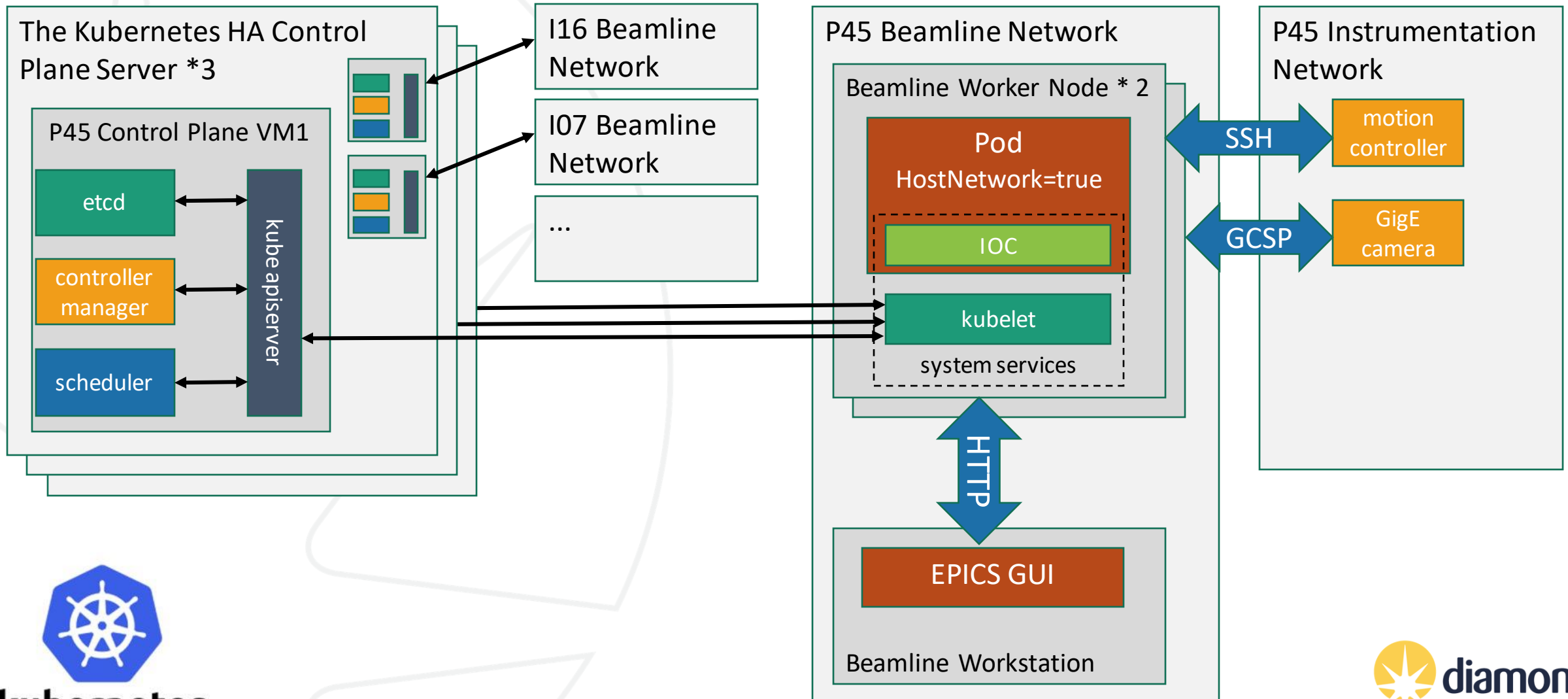
Pods


Name	Labels	Node	Status
✓ bl45p-ea-ioc-02-6ccbc49d98-lw8vw	app: bl45p-ea-ioc-02 ioc_version: 2021.3.3 Show all	beamline: bl45p cs05r-sc-cloud-18.diamond.ac.uk	Running
✓ bl45p-mo-ioc-01-6b5dc8998c-zhs4r	app: bl45p-mo-ioc-01 ioc_version: 2021.3.3 Show all	beamline: bl45p cs05r-sc-cloud-15.diamond.ac.uk	Running
✓ bl45p-ea-ioc-01-5c4458875b-cj25j	app: bl45p-ea-ioc-01 ioc_version: 2021.3.3 Show all	beamline: bl45p cs05r-sc-cloud-17.diamond.ac.uk	Running

- Kubernetes 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
 - Twitter
 - Many more .. <https://kubernetes.io/case-studies/>
- This screenshot shows the standard K8s dashboard managing P45 IOCs



Proposed DLS Kubernetes Cluster Topology





Example
Kubernetes
Manifest

03 An IOC Instance

BL45P motion IOC 01



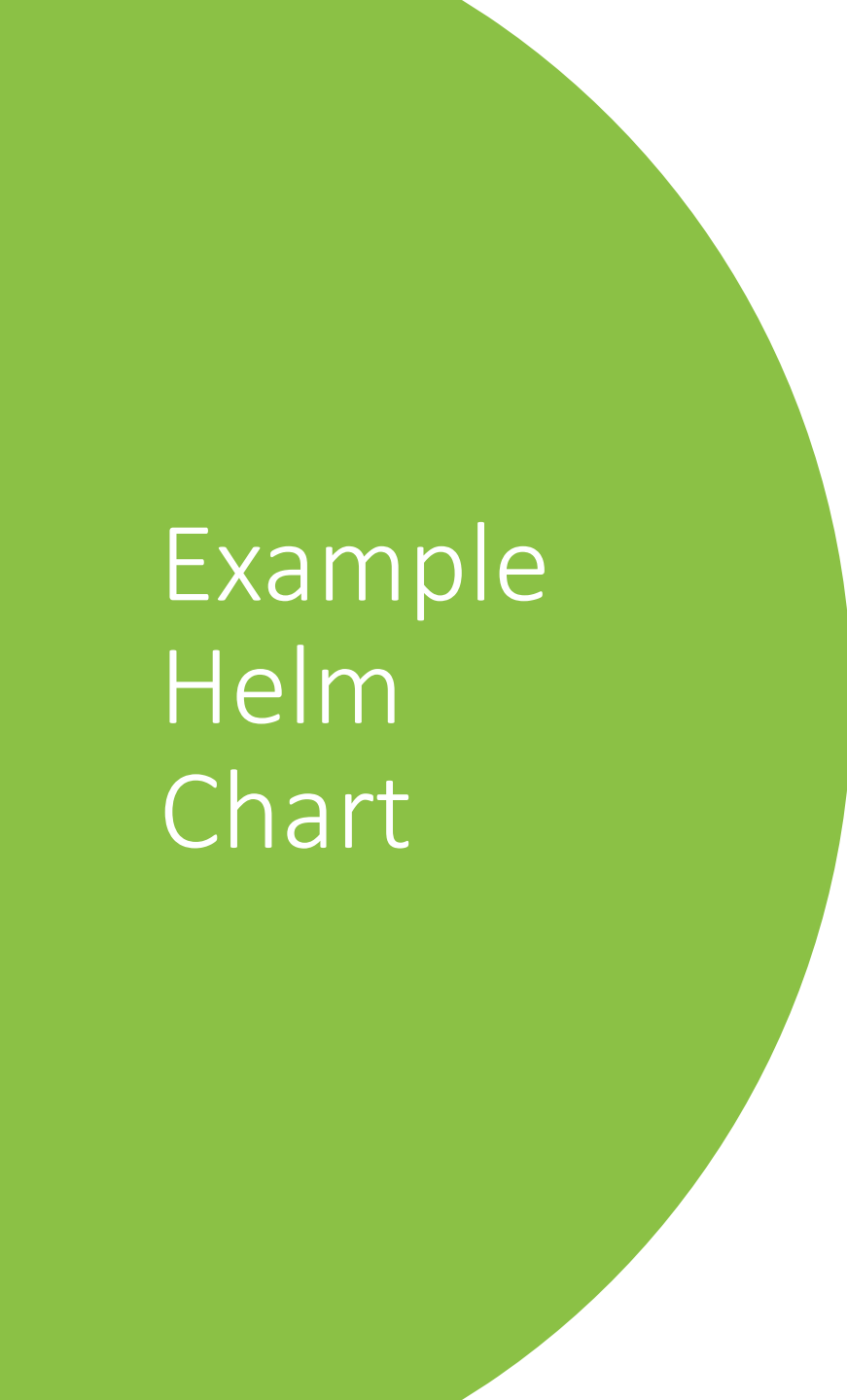
Takeaways from the Kubernetes Manifest

- Declarative
 - Describes the application requirements – Kubernetes allocates the best resources to meet those requirements
- Idempotent
 - Re-applying the same manifest has no affect. Modifying it and applying adds the changes only
- Lots of YAML to write
 - Therefore, we use helm to create IOC manifests



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



Example
Helm
Chart

04 An IOC Definition

BL45P motion IOC 01

TODO – what files to show ?

TODO – is this TMI ?



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.



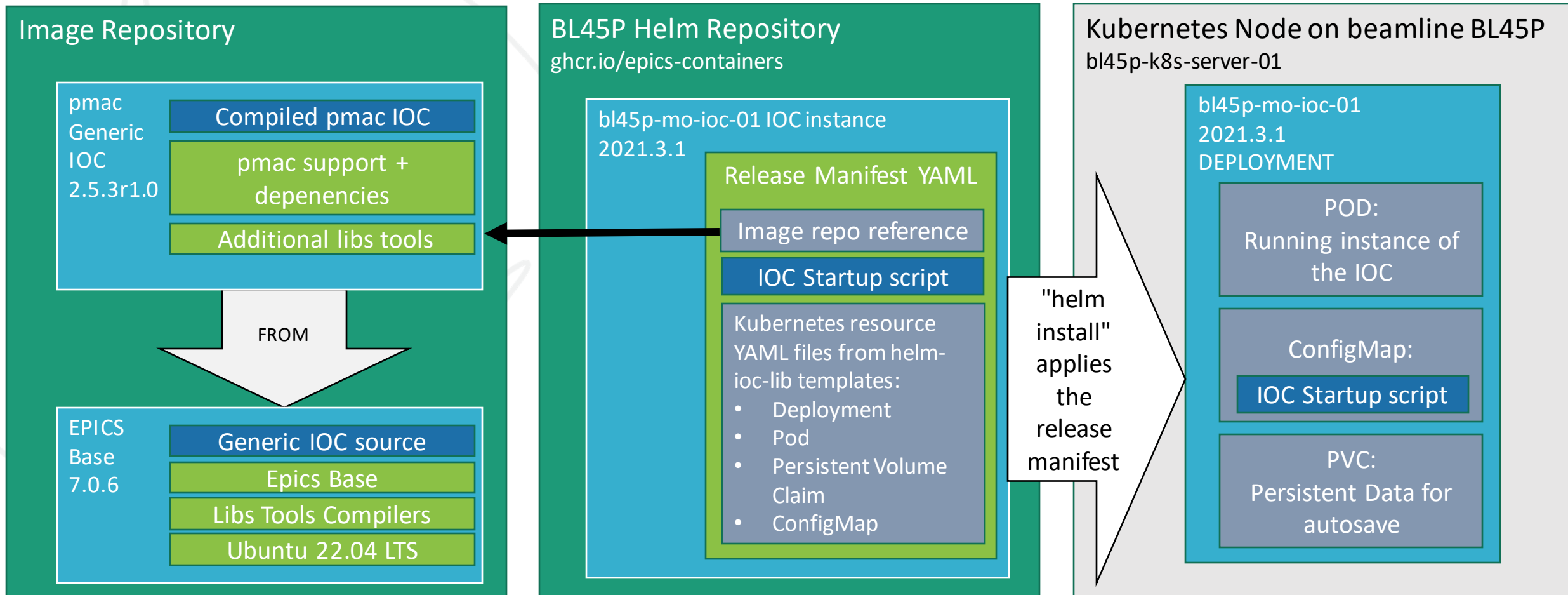
Example
Repositories

CI and Repositories for BL45P

<https://github.com/epics-containers>



Summary: Deploying 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>

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

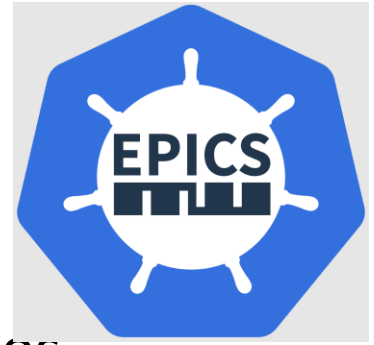
Developer Environment (TODO WIP)

- Locally run IOCs will execute in the same environment as they would on the Kubernetes cluster
- The containers are designed to provide a complete developer environment, with all work saved to the workstation filesystem
- These CLI tools will be essential to the developer
 - kubectl – send commands to the kubernetes cluster
 - helm – package manager for kubernetes
 - podman – container management for the local workstation
 - ec – the Epics Containers assistant CLI developed by DLS
- Documentation will provide prescriptive workflows for working with IOCs within this new framework

Epics Containers assistant CLI

- Written in Python
- Provides a very thin layer of assistance to save typing and assist with adoption.
- Only uses kubectl, helm and podman.
- Prints the system commands as a learning aid e.g.

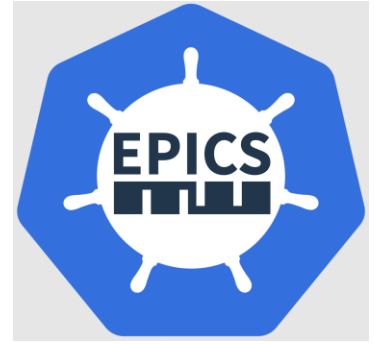
```
$ ec deploy bl45p-mo-ioc-01 0.0.1-b0  
+ helm upgrade --install bl45p-mo-ioc-01 oci://ghcr.io/epics-containers/bl45p-mo-ioc-01 --version 0.0.1-b0
```



Current Status

- A test DLS beamline BL45P 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/>
- The docs include 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 IOC dependencies are vanilla EPICS Community support modules
 - Upgrades to site RHEL version doesn't affect IOCs or EPICS developers!
 - Isolation protects against many 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



Thanks for listening

Any questions?