epics-containers

A workshop to look at containerizing EPICS IOCs

EPICS Collaboration Meeting @ ORNL
Sept 16th, 2024

giles knap & Marcell Nagy
Beamline Controls – Diamond Light Source

epics-containers workshop ORNL

- 13:40 – Presentation: an overview of epics-containers
- 14:20 – Questions
- 14:30 – Hands On Tutorials: Create an IOC in a container
- 15:00 – Break
- 15:30 – Demo: Kubernetes and ArgoCD on a DLS test beamline
- 16:10 – Questions and discussion
- 16:30 – Hands On Tutorials continued
- 17:10 - End

Preamble

- This presentation is about how DLS will be building and managing IOCs using containers.
- However, we have applied the following principals to make this work useful to other facilities:
  - **Open source first**. All source, re-usable containers, example beamlines and documentation is available at https://github.com/epics-containers.
  - **Modular.** All parts of the framework are as far as practical independent – you may adopt just the features you find useful.
  - **Standard EPICS only.** We use the default EPICS build system, upstream versions of EPICS base and support modules.
  - **Standard Tools only.** The tools used in the framework are widely adopted FOSS only.
- The SPARC beamline at INFN-LNF in Rome is already using epics-containers in production.

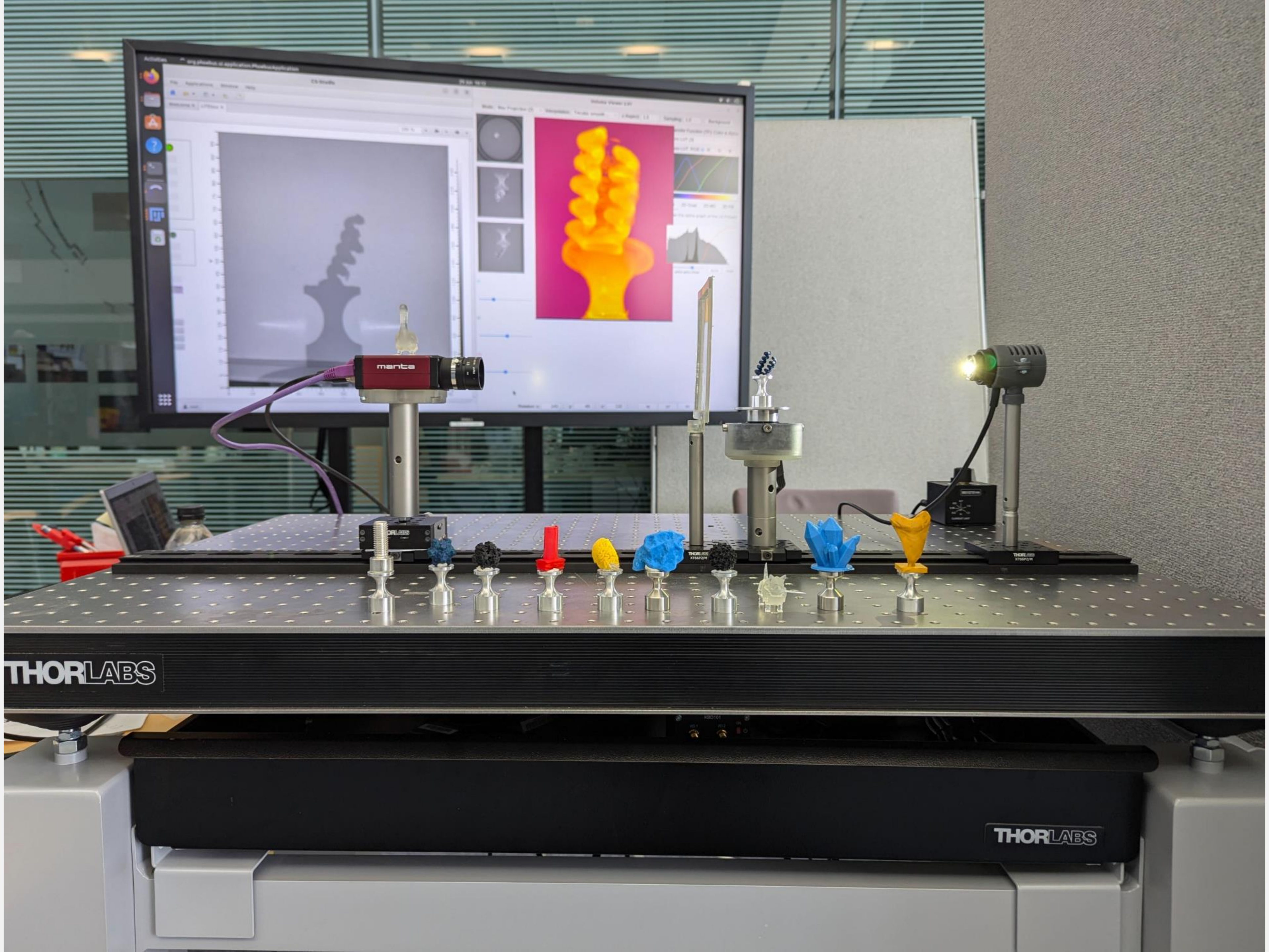
What?

Applying modern industry practices for software delivery to EPICS IOCs

Containers:	Package IOC software and execute it in a lightweight virtual environment
Kubernetes:	Orchestrates all IOCs at a facility
Helm Charts:	Deploy IOCs into Kubernetes with version management
Repositories:	Source and container repositories manage all the above assets
CI:	Source repositories automatically build assets from source when updated
CD:	Deployment repositories are automatically synced with the cluster

Why?

- IOCs are decoupled from the OS
    - No modifying support modules to suit facility infrastructure
    - Allows us to use upstream support modules with no need for local forks
    - Protection from many security vulnerabilities
- Simple server setup: any Linux + container runtime only.
    - Very easy server OS upgrades
- Remove maintenance of internal management tools
- Kubernetes provides (not just for IOCs!):
    - Shared software deployment and management
    - Shared Logging, monitoring, alerting
    - Shared resource management: Disk / CPU / Memory
    - Skills required are transferrable
    - A huge range of supporting tools and applications both FOSS and licensed
    - Just Google it when things go wrong



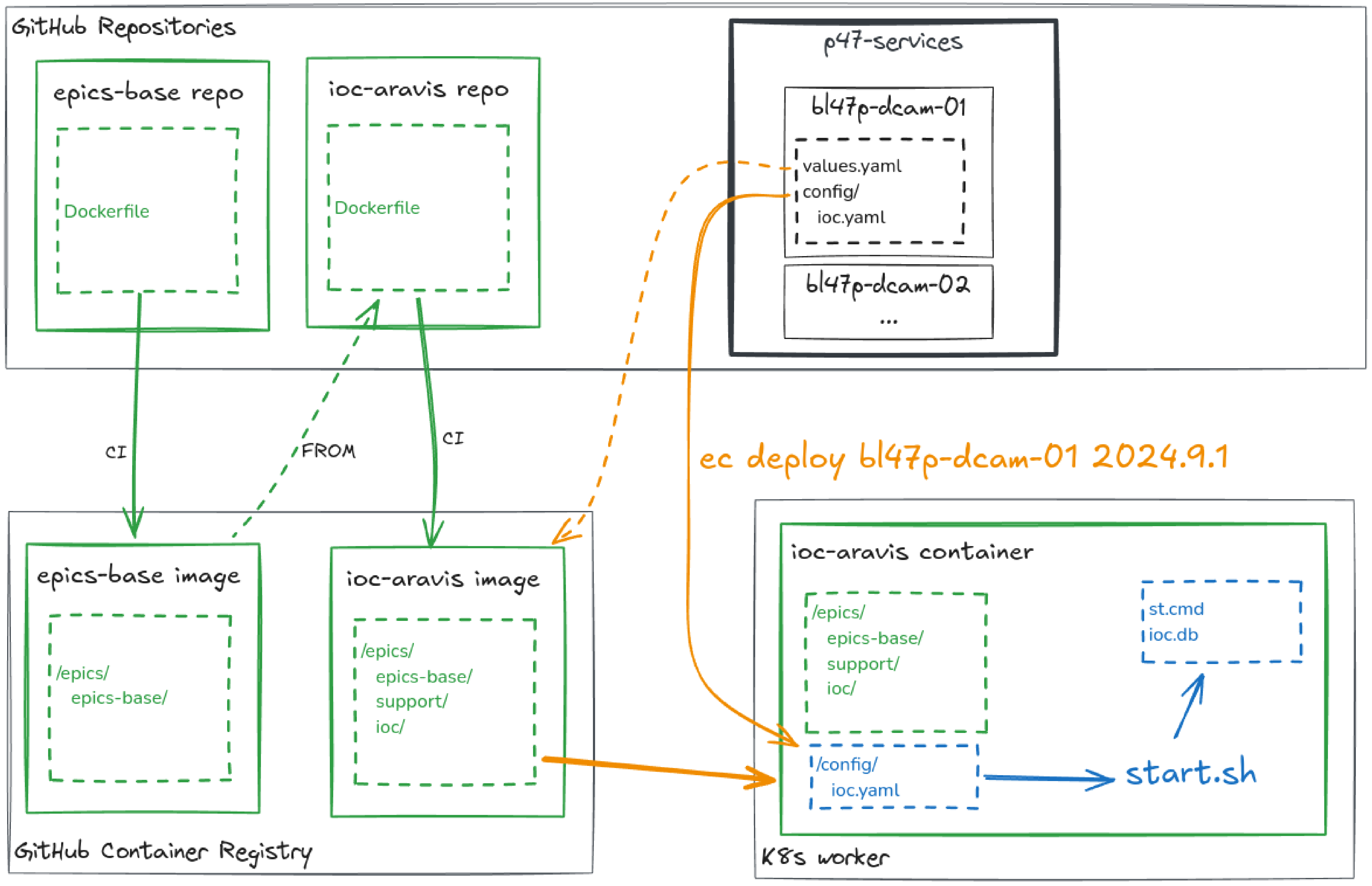
Why?

Portability:

Develop anywhere.
Execute anywhere.

e.g.
DIAD's portable tomography demo made for the RAL public open day.

How?


GitHub Repositories
epics-base repo
Dockerfile
ioc-aravis repo
Dockerfile
p47-services
bl47p-dcam-01
values.yaml
config/
ioc.yaml
bl47p-dcam-02
...
CI
FROM
CI
ec deploy bl47p-dcam-01 2024.9.1
epics-base image
/epics/
epics-base/
ioc-aravis image
/epics/
epics-base/
support/
ioc/
GitHub Container Registry
ioc-aravis container
/epics/
epics-base/
support/
ioc/
st.cmd
ioc.db
/config/
ioc.yaml
start.sh
K8s worker

Supporting Tools

- **ibek –** IOC builder for EPICS and Kubernetes
  - Runs inside the container at build time
    - Helpers for building Support and Generic IOC in the container environment
  - Runs inside the container at runtime startup
    - Makes IOC instance startup script and database from a YAML description
    - Extracts IOC instance engineering screens from the container

- **ec –** the epics containers CLI for developers
  - Runs outside the container
    - Helpers for building and deploy IOC Instances
    - Helpers for local debugging and testing of Generic IOCs
    - Thin wrapper around the tools git, helm, kubectl and argocd – can be used to learn these tools too

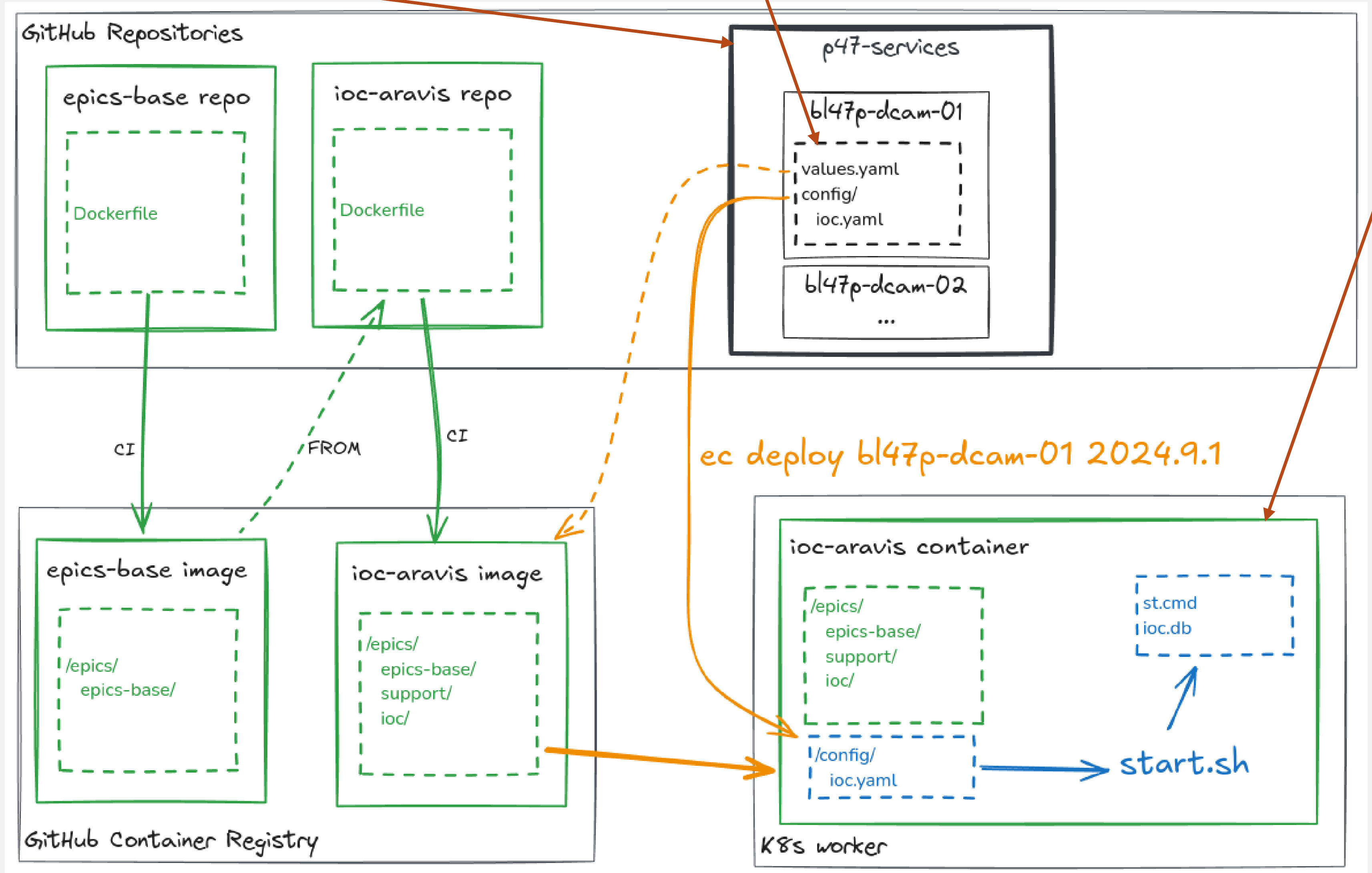
- **PVI -** Process Variables Interface
  - Provides structure for the PV interface to a device
  - Auto generates engineering screens for the device (bob/edm/adl)
  - Supplies DB metadata for use with Bluesky

How to make an IOC Instance?

1. Create a services repository

2. Edit the new IOC Instance config

3. Deploy the IOC instance



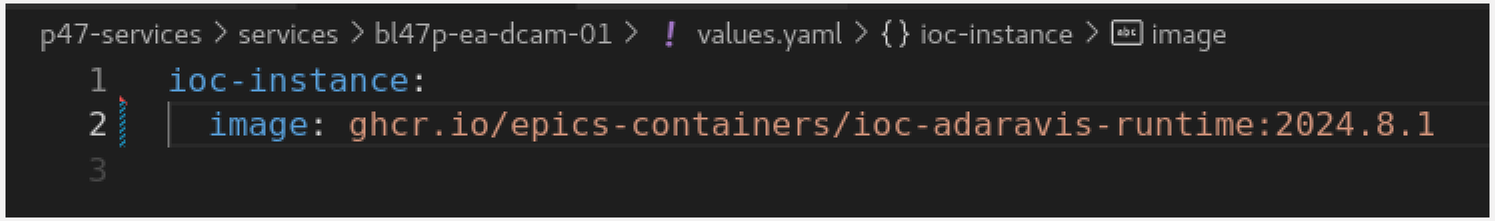
How to make an IOC Instance?

1. Create a services repository

- copier copy gh:epics-containers/services-template-helm .
- cp –r services/.ioc-template services/my-new-ioc

2. Edit the new IOC Instance config

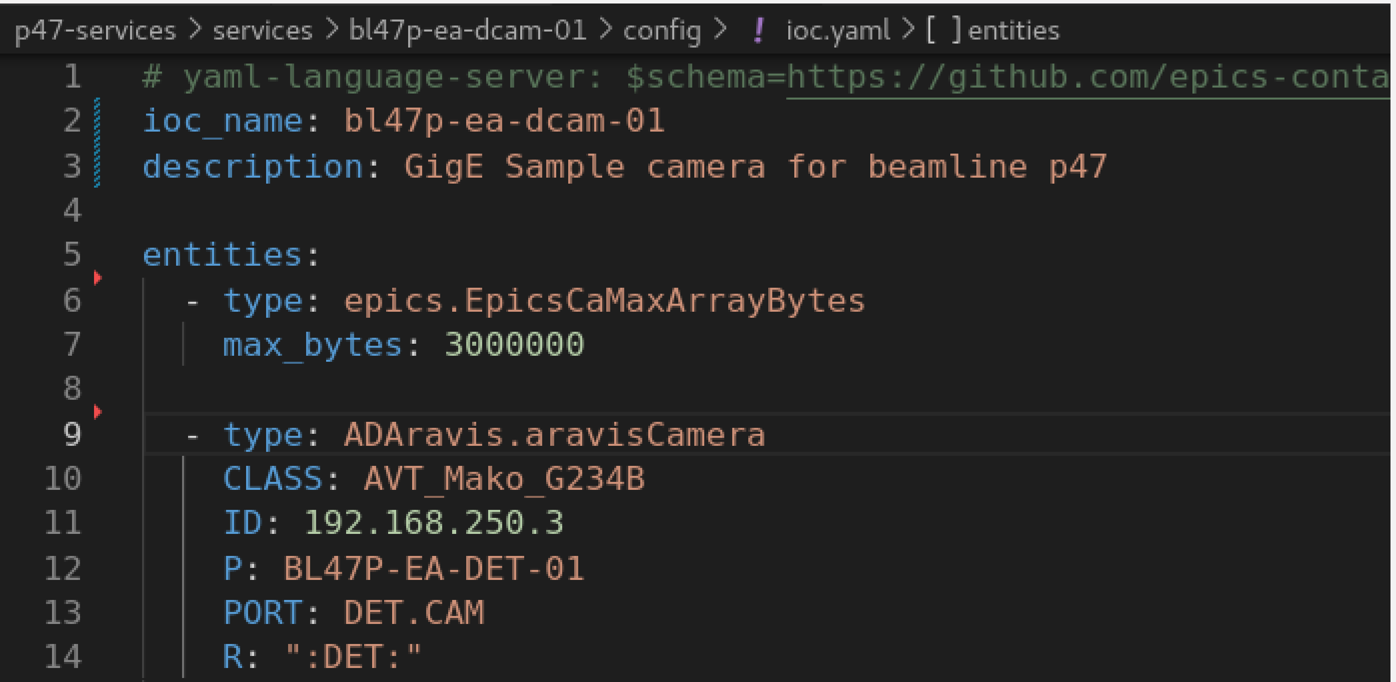
2a. *A helm chart values override file. The only required field is the URL of the Generic IOC to use.*



p47-services > services > bl47p-ea-dcam-01 > ! values.yaml > { } ioc-instance > image

```yaml
ioc-instance:
  image: ghcr.io/epics-containers/ioc-adaravis-runtime:2024.8.1
```

2b. *An **ibek** IOC YAML file listing the support 'entities' that we want to instantiate for this IOC instance.*



p47-services > services > bl47p-ea-dcam-01 > config > ! ioc.yaml > [ ] entities

```yaml
# yaml-language-server: $schema=https://github.com/epics-conta
ioc_name: bl47p-ea-dcam-01
description: GigE Sample camera for beamline p47

entities:
  - type: epics.EpicsCaMaxArrayBytes
    max_bytes: 3000000

  - type: ADAravis.aravisCamera
    CLASS: AVT_Mako_G234B
    ID: 192.168.250.3
    P: BL47P-EA-DET-01
    PORT: DET.CAM
    R: ":DET:"
```

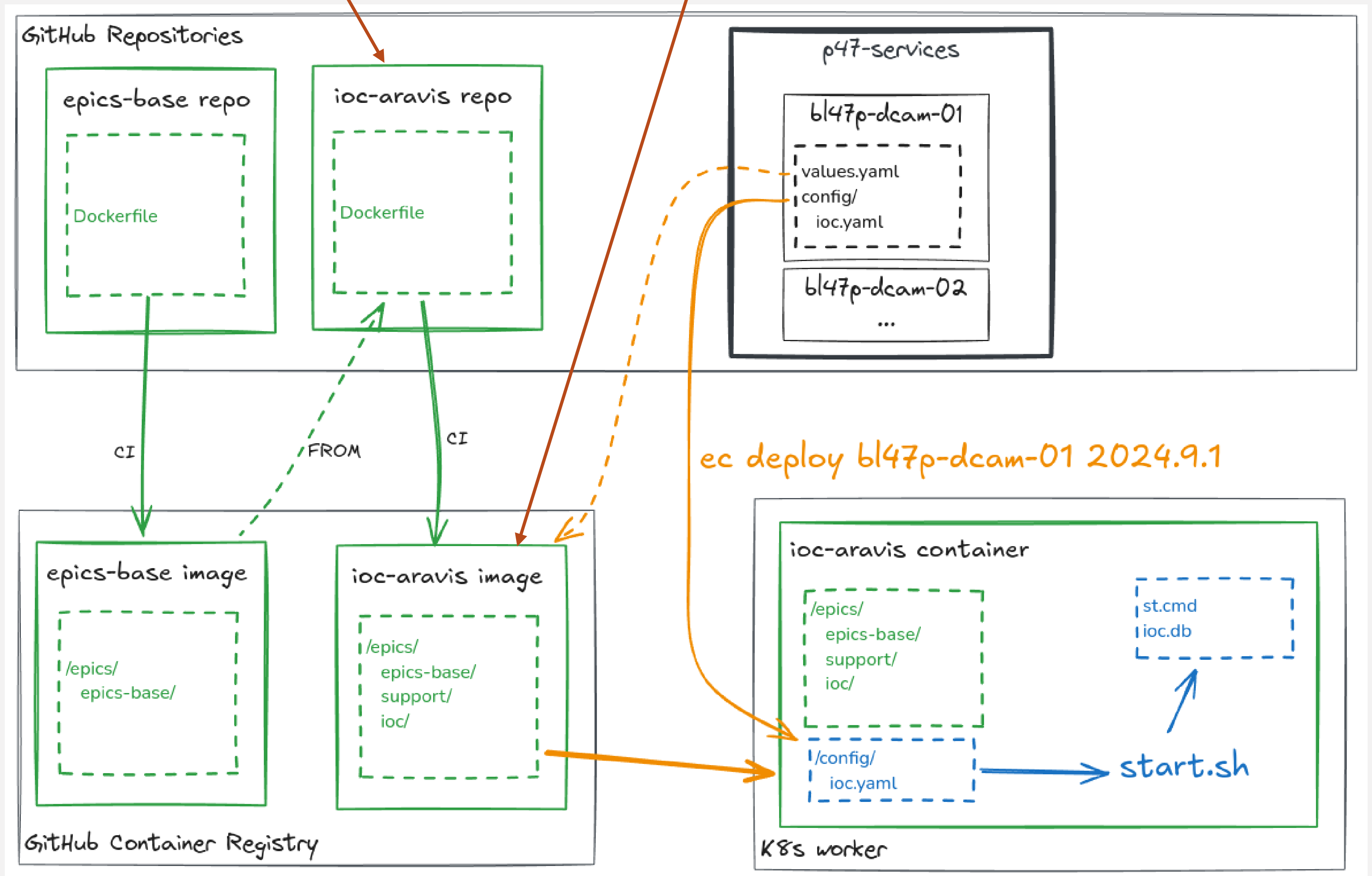
3. Deploy the IOC instance:

- Tag and push the beamline repo
- **ec** deploy bl45p-ea-ioc-01 2024.9.1

How to make a Generic IOC?

1.Create a Generic IOC repo

2. Deploy Generic IOC to your container registry



How to make a Generic IOC?

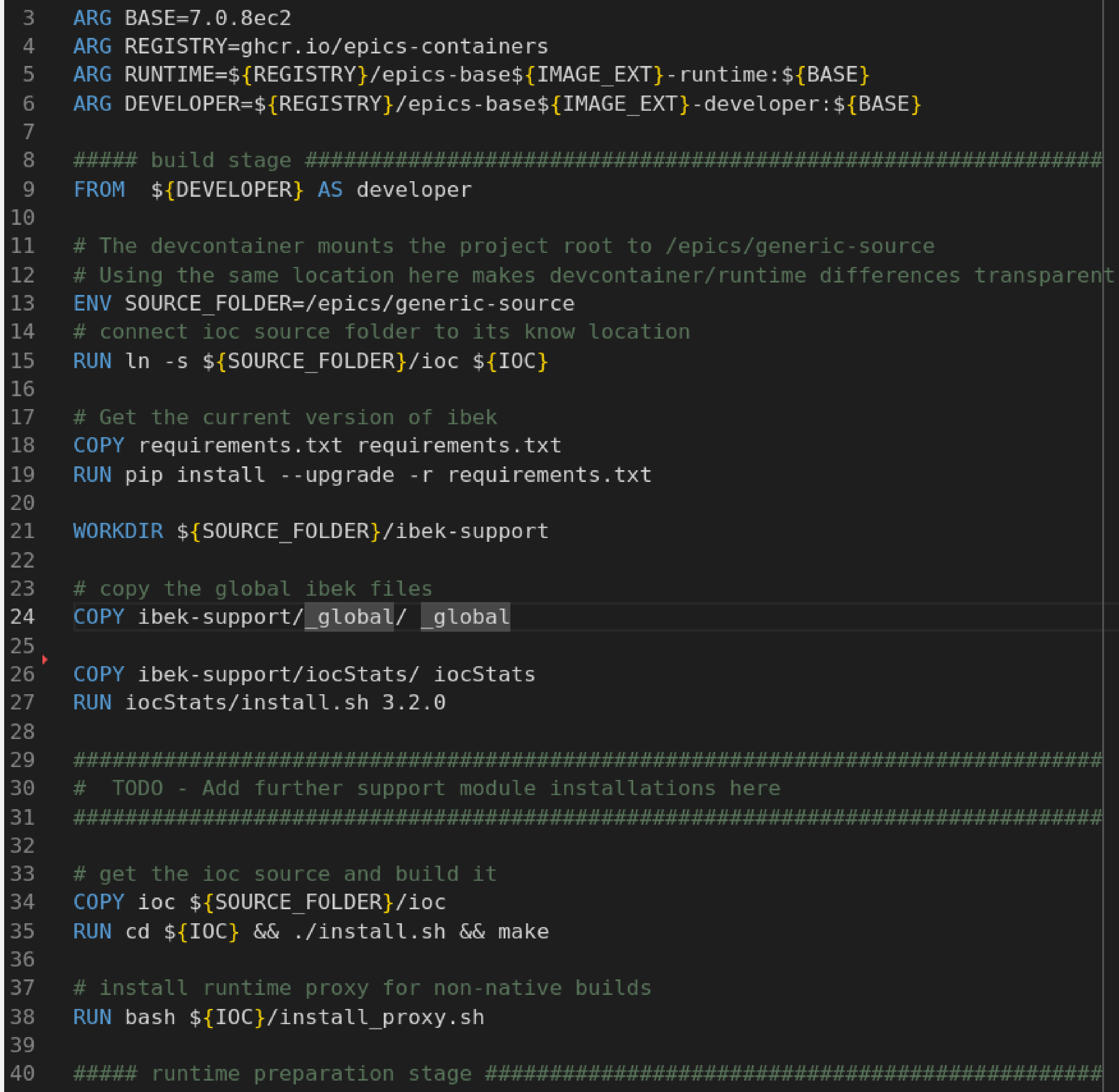
1. Create a Generic IOC repo:

- copier copy gh:epics-containers/ioc-template
- edit Readme.md
- edit Dockerfile: add COPY and RUN for each support module using ibek-support recipies

2. Deploy Generic IOC to your container registry:

- Tag and push the generic IOC repo
- CI then:
  - Builds the container image
  - Publishes it to GHCR
  - Publishes a JSON schema for the ibek 'entities' provided inside the container

Dockerfile: iocStats serves as an example for how to add additional support modules



```
ARG BASE=7.0.8ec2
ARG REGISTRY=ghcr.io/epics-containers
ARG RUNTIME=${REGISTRY}/epics-base${IMAGE_EXT}-runtime:${BASE}
ARG DEVELOPER=${REGISTRY}/epics-base${IMAGE_EXT}-developer:${BASE}

##### build stage ##############################################################
FROM  ${DEVELOPER} AS developer

# The devcontainer mounts the project root to /epics/generic-source
# Using the same location here makes devcontainer/runtime differences transparent
ENV SOURCE_FOLDER=/epics/generic-source
# connect ioc source folder to its know location
RUN ln -s ${SOURCE_FOLDER}/ioc ${IOC}

# Get the current version of ibek
COPY requirements.txt requirements.txt
RUN pip install --upgrade -r requirements.txt

WORKDIR ${SOURCE_FOLDER}/ibek-support

# copy the global ibek files
COPY ibek-support/_global/ _global


COPY ibek-support/iocStats/ iocStats
RUN iocStats/install.sh 3.2.0

################################################################################
#  TODO - Add further support module installations here
################################################################################

# get the ioc source and build it
COPY ioc ${SOURCE_FOLDER}/ioc
RUN cd ${IOC} && ./install.sh && make

# install runtime proxy for non-native builds
RUN bash ${IOC}/install_proxy.sh

##### runtime preparation stage ################################################
```

How to Add a new Support Module 1.

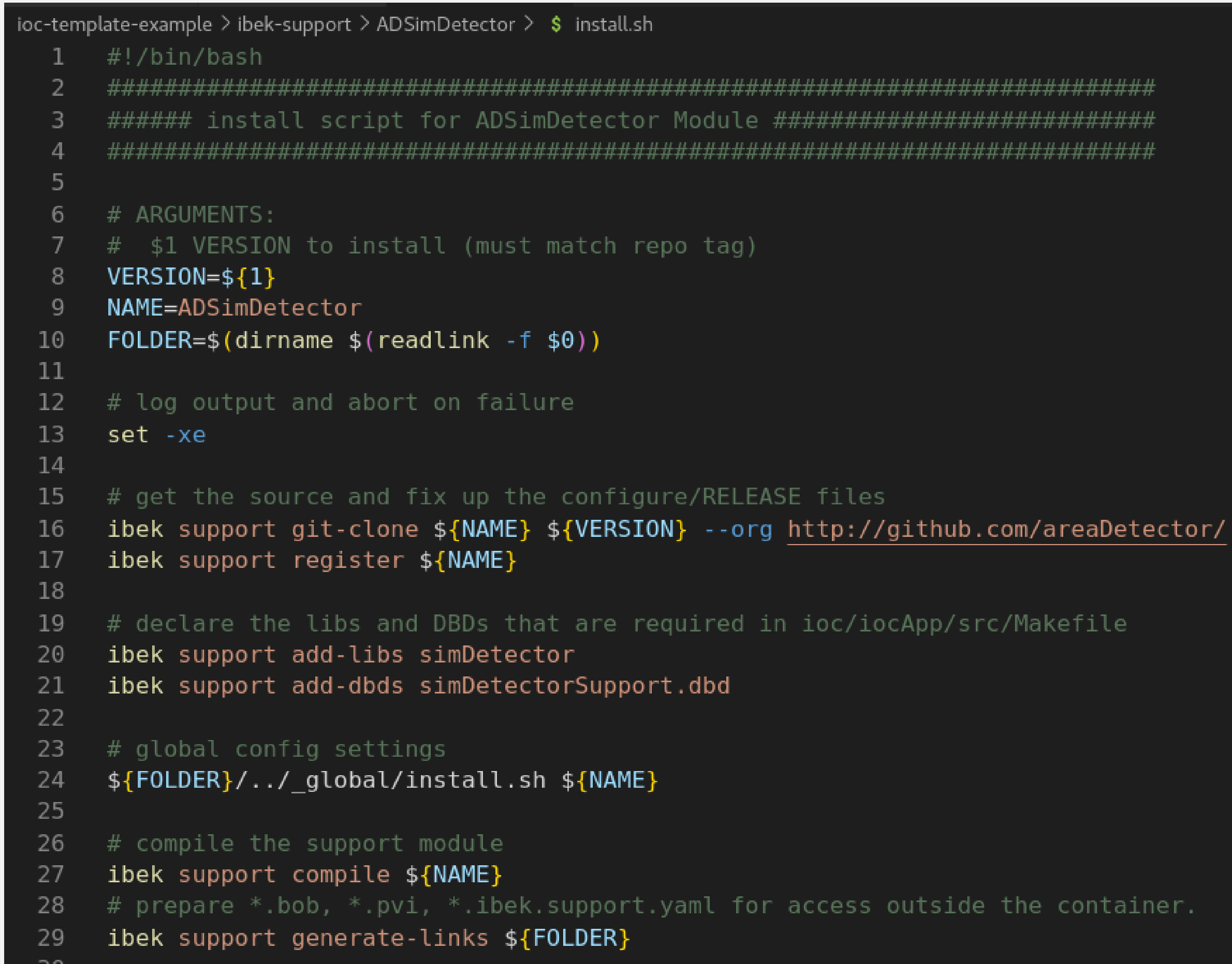
1. Add a folder in the ibek-support repo

- Shared ibek-support on GitHub
- Or private ibek-support per facility
- Public Generic IOCs should only use shared ibek-support
- Internal Generic IOCs may use both

2. Add a new install.sh file:

- Use the new support in your Generic IOC Dockerfile

Install.sh example for ADSimDetector. Most install.sh would look almost identical to this but You can add custom steps as needed –it's just bash.



ioc-template-example > ibek-support > ADSimDetector > $ install.sh

```bash
#!/bin/bash
##########################################################################
###### install script for ADSimDetector Module ###########################
##########################################################################

# ARGUMENTS:
#  $1 VERSION to install (must match repo tag)
VERSION=${1}
NAME=ADSimDetector
FOLDER=$(dirname $(readlink -f $0))

# log output and abort on failure
set -xe

# get the source and fix up the configure/RELEASE files
ibek support git-clone ${NAME} ${VERSION} --org http://github.com/areaDetector/
ibek support register ${NAME}

# declare the libs and DBDs that are required in ioc/iocApp/src/Makefile
ibek support add-libs simDetector
ibek support add-dbds simDetectorSupport.dbd

# global config settings
${FOLDER}/../_global/install.sh ${NAME}

# compile the support module
ibek support compile ${NAME}
# prepare *.bob, *.pvi, *.ibek.support.yaml for access outside the container.
ibek support generate-links ${FOLDER}
```

How to Add a new Support Module 2.

1. Add a folder in the ibek-support repo

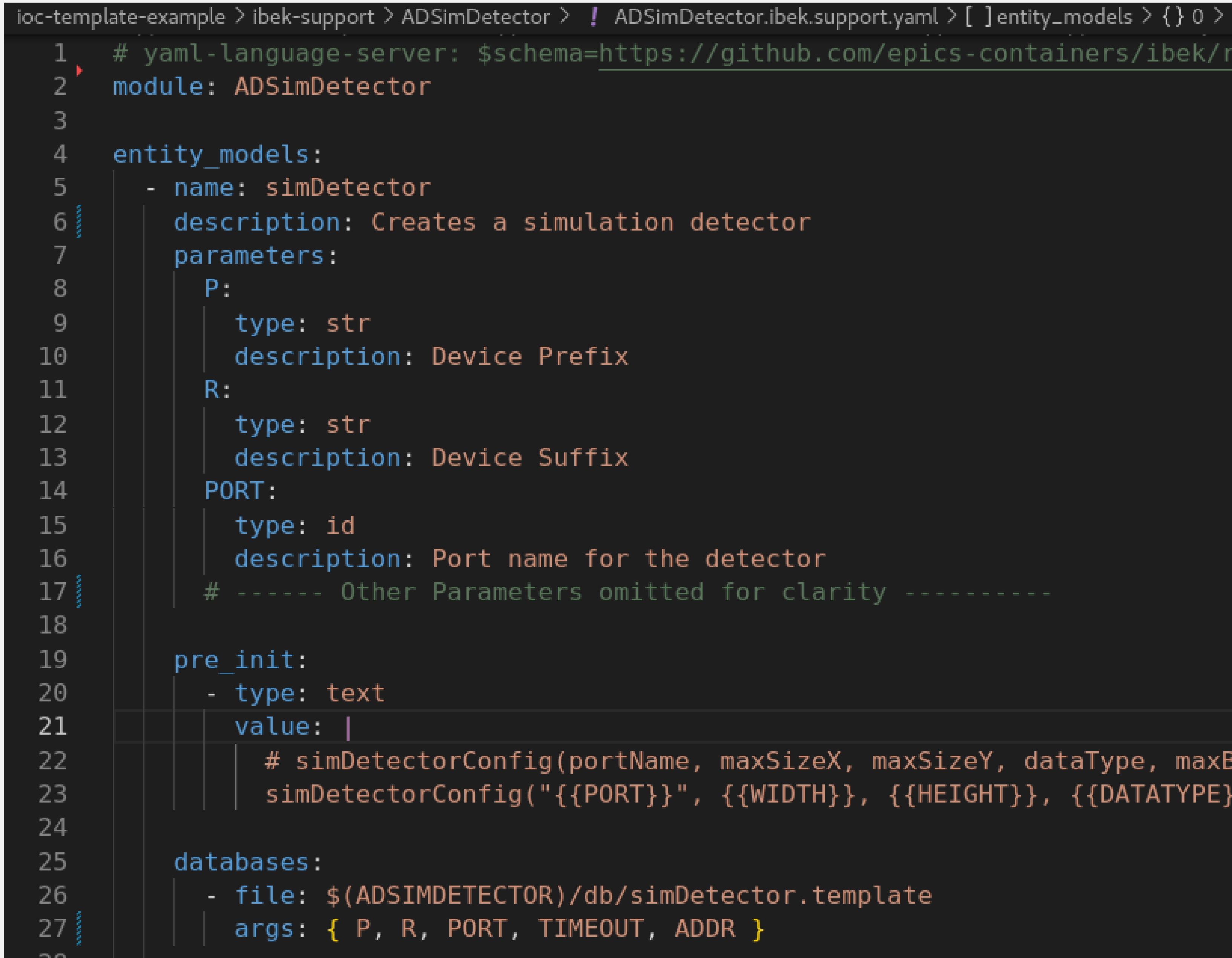
- Shared ibek-support on GitHub
- Or private ibek-support per facility
- Public Generic IOCs should only use shared ibek-support
- Internal Generic IOCs may use both

2. Add a new install.sh file:

- Use the new support in your Generic IOC Dockerfile

3. Add an ibek support YAML description:

- This allows us to describe our IOC instances as ibek IOC YAML.



ioc-template-example > ibek-support > ADSimDetector > ! ADSimDetector.ibek.support.yaml > [ ] entity_models > {} 0 >

```yaml
# yaml-language-server: $schema=https://github.com/epics-containers/ibek/r
module: ADSimDetector

entity_models:
  - name: simDetector
    description: Creates a simulation detector
    parameters:
      P:
        type: str
        description: Device Prefix
      R:
        type: str
        description: Device Suffix
      PORT:
        type: id
        description: Port name for the detector
      # ------ Other Parameters omitted for clarity ----------

    pre_init:
      - type: text
        value: |
          # simDetectorConfig(portName, maxSizeX, maxSizeY, dataType, maxB
          simDetectorConfig("{{PORT}}", {{WIDTH}}, {{HEIGHT}}, {{DATATYPE}

    databases:
      - file: $(ADSIMDETECTOR)/db/simDetector.template
        args: { P, R, PORT, TIMEOUT, ADDR }
```

Support YAML example for ADSimDetector. Declares instance arguments, startup script lines and database template substitutions.

Developer Containers

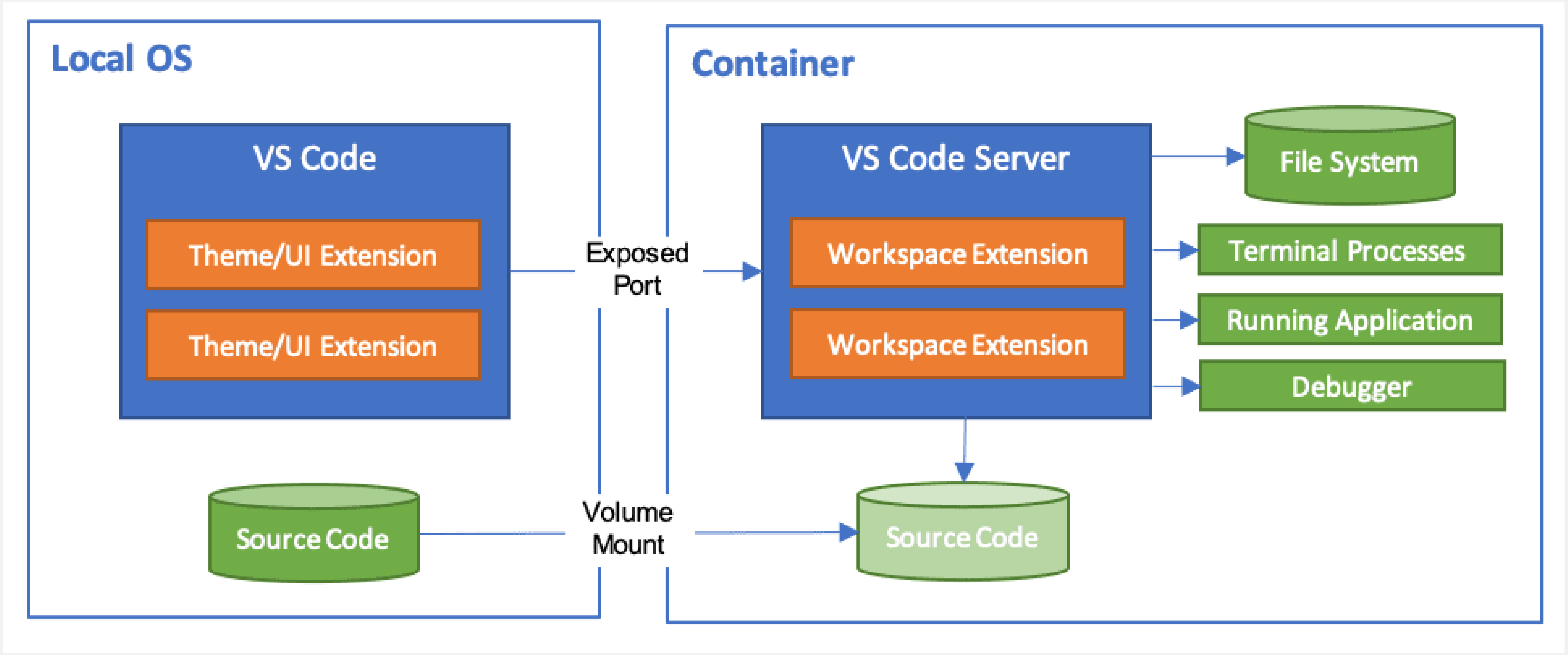
epics-containers development defines 3 levels of changes:

1. Changing IOC instance details only
   - Edit values.yaml or ioc.yaml in your beamline repository
   - Push and tag the changes, re-deploy the update instance
2. Changing a Generic IOC
   - Edit Dockerfile or ibek-support sub-module in a Generic IOC repository
   - Push changes to publish a new container image
   - Go to 1. to update affected instances
3. Changing Support Modules
   - Edit the support module, verify and push and tag source changes.
   - Go to 2. to update a Generic IOC to include the new support version

2 and 3 require rebuilding and deploying containers. For this reason, we use Developer Containers for a fast "inner loop".

For epics-containers the generic IOC container image is an ideal developer container.

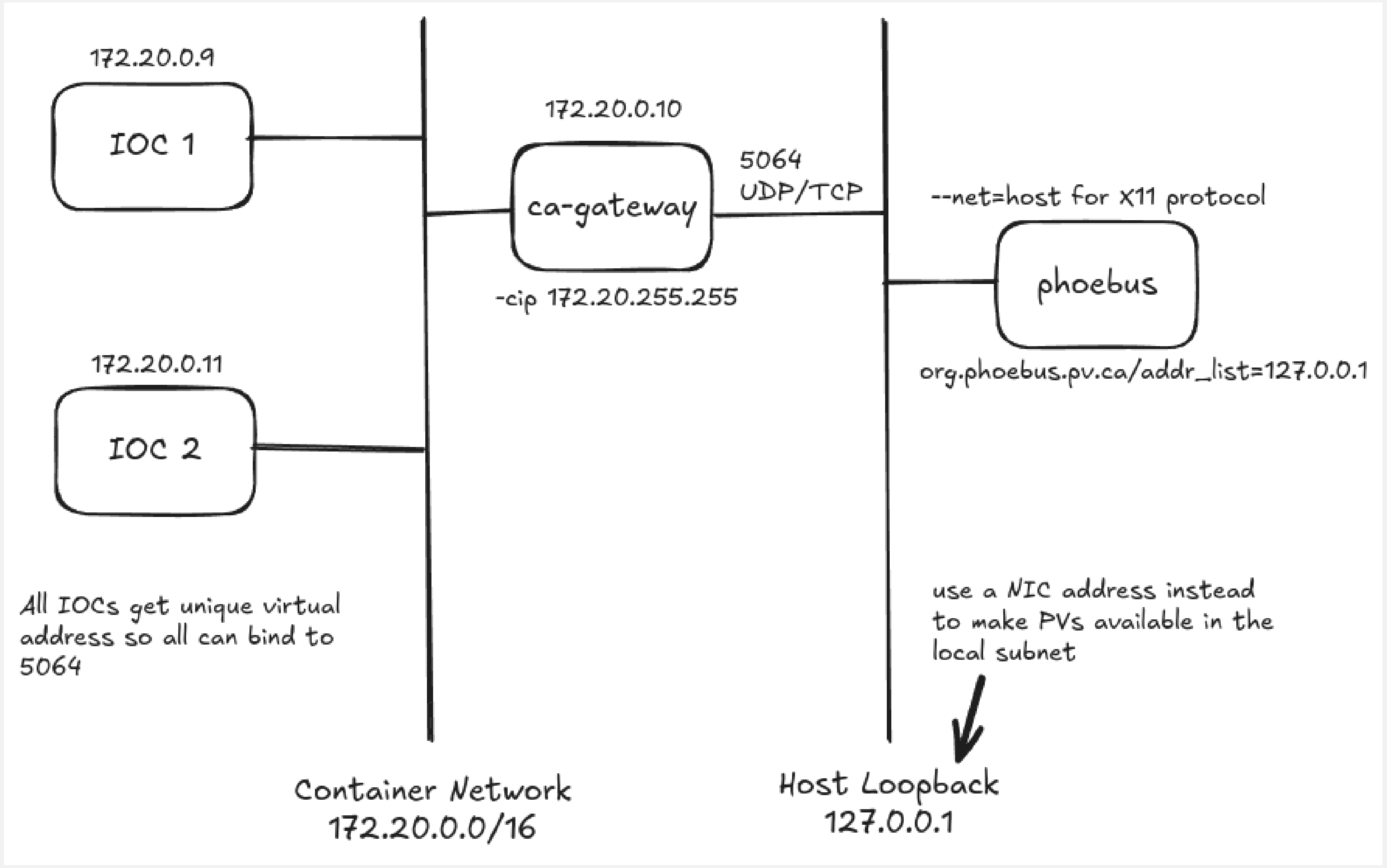
Developer Containers



See developer target in https://github.com/epics-containers/ioc-template/blob/main/template/Dockerfile

Networking

For local development and testing we configure the network like this:


172.20.0.9
IOC 1
172.20.0.10
ca-gateway
5064
UDP/TCP
-cip 172.20.255.255
--net=host for X11 protocol
phoebus
org.phoebus.pv.ca/addr_list=127.0.0.1
172.20.0.11
IOC 2
All IOCs get unique virtual address so all can bind to 5064
use a NIC address instead to make PVs available in the local subnet
Container Network
172.20.0.0/16
Host Loopback
127.0.0.1

Networking network=host

At DLS we use network=host for Kubernetes IOCs

- This means IOCs run without network isolation and look exactly like traditional IOCs from the client's perspective

Motivation

- Channel Access cannot route into the container network
- We did not want to pass all PVs through a ca-gateway
- Channel Access is not the only protocol that will not route into CNI
- For example, GigE streaming protocol will also fail
- Any protocol that does not like NAT will have this problem

Current Status - Sept 2024

- The epics-containers framework is ready for wider exposure and feedback
- There are tutorials to get people started using the framework
- There are a growing number of reference generic IOCs
- Example beamline repositories are also available
- DLS has several beamline clusters running a handful of containerized IOCs in production, plus some fully containerized test beamlines.
- DLS aims to have a representative set of production beamlines fully containerized before the DII dark period 12/2027.
- And all beamlines fully containerized at the end of the dark period 06/2029.

One Take Away

https://github.com/epics-containers

- Includes:
  - Tutorials
  - Documentation
  - Templates
  - Source code
  - A small but growing number of Generic IOC images
  - Example beamlines
  - A Simulation beamline

Questions ?

Hands On: Tutorials

- https://epics-containers.github.io/main/tutorials/intro.html
- Work through the tutorials at your own pace
  - Try out the simulation beamline
  - Create a services repository with docker compose
  - Create an IOC instance
  - Work with developer containers (stretch goal for today!)

Kubernetes

- Kubernetes is by far the dominant container orchestration platform today
- Open-sourced by Google in 2014
- Managed by the Cloud Native Computing Foundation, part of the Linux Foundation
- CNCF looks after a large list of open-source applications that run in Kubernetes
- At DLS we will have a Kubernetes Cluster per beamline, one for the accelerator and a large central cluster for centralized services.

Demo: a Kubernetes Beamline at DLS

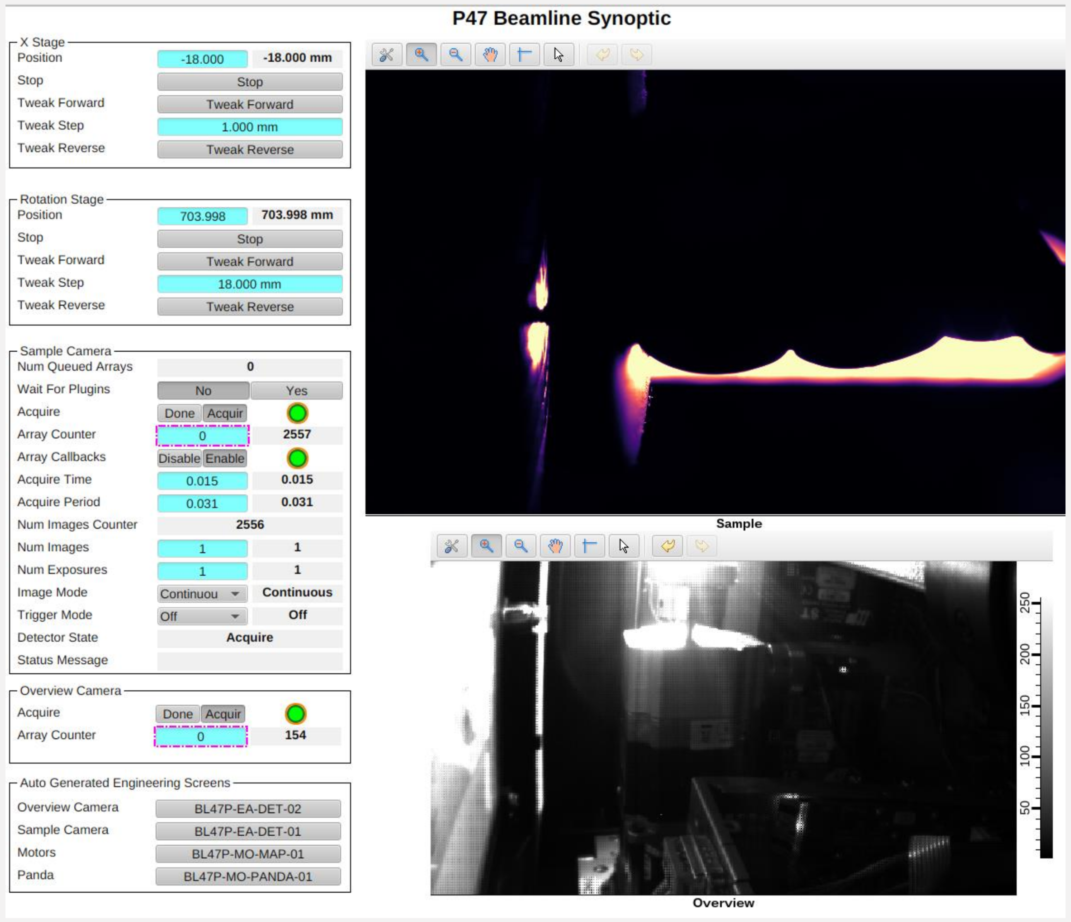
- p47 is a training beamline with 2 detectors, 2 motors, 1 pandabox
- Each Kubernetes Clusters at DLS runs standard services including:-
  - A landing page to access all user services
  - Kubernetes Dashboard – manage resources in the cluster
  - Alert Manager – sets thresholds and configures recipients of alerts
  - Prometheus – monitoring with time series Database
  - Grafana – rich visualization of the above data
  - StacksRox – monitor running containers for Common Vulnerabilities and Exploits
  - Keycloak – single sign on authorization service
  - Kynervo – policy engine
  - Argo CD – declarative GitOps continuous deployment

Questions ?

Remaining Slides

- Images of demo screens in case I can't connect to DLS
- Some overview diagrams for discussion if needed

Demo: a Kubernetes Beamline at DLS P47



P47 Beamline Synoptic

X Stage

Position	-18.000	-18.000 mm
Stop	Stop	
Tweak Forward	Tweak Forward	
Tweak Step	1.000 mm	
Tweak Reverse	Tweak Reverse	

Rotation Stage

Position	703.998	703.998 mm
Stop	Stop	
Tweak Forward	Tweak Forward	
Tweak Step	18.000 mm	
Tweak Reverse	Tweak Reverse	

Sample Camera

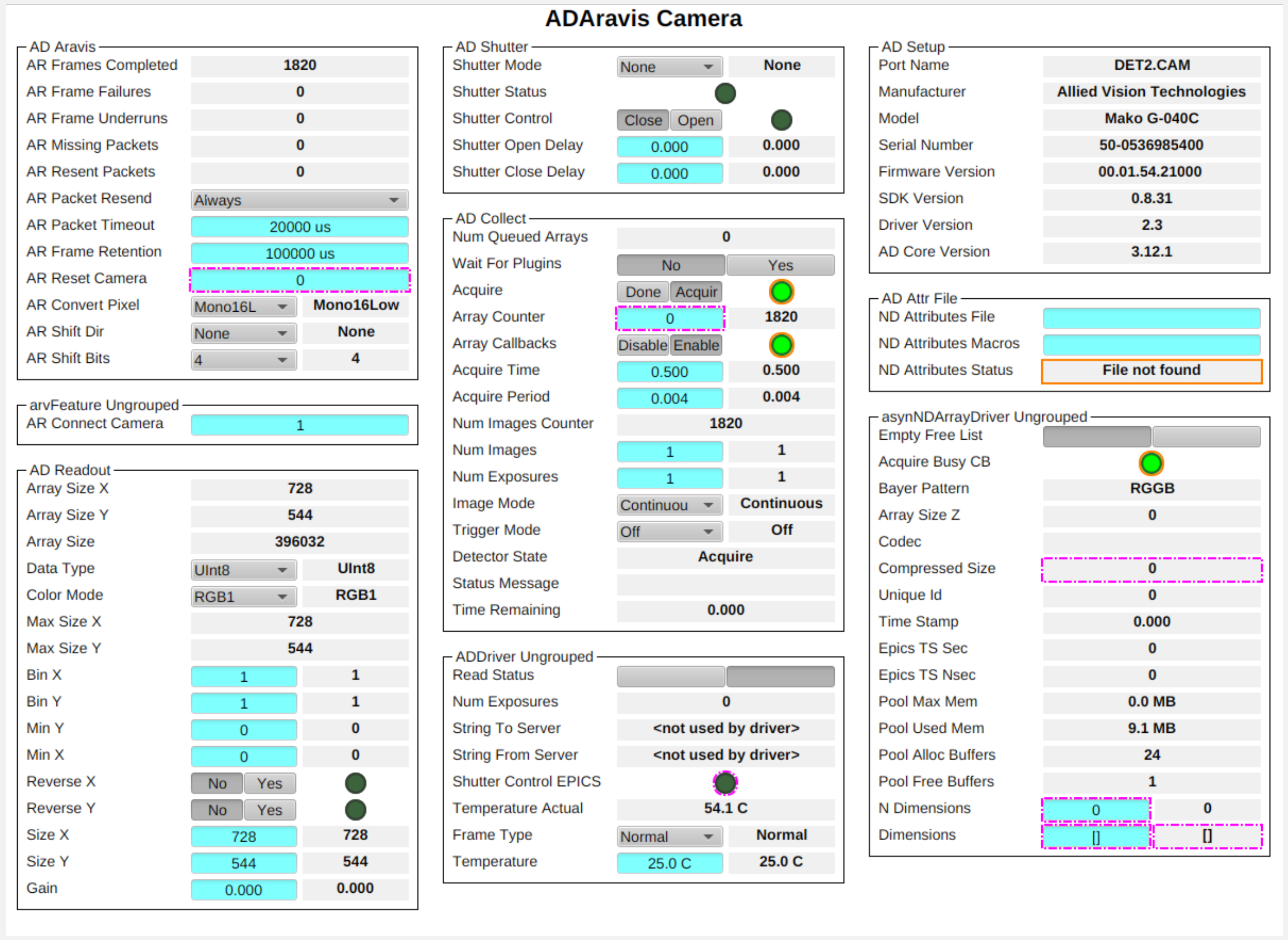
Num Queued Arrays	0	
Wait For Plugins	No	Yes
Acquire	Done Acquir	●
Array Counter	0	2557
Array Callbacks	Disable Enable	●
Acquire Time	0.015	0.015
Acquire Period	0.031	0.031
Num Images Counter	2556	
Num Images	1	1
Num Exposures	1	1
Image Mode	Continuou	Continuous
Trigger Mode	Off	Off
Detector State	Acquire	
Status Message		

Overview Camera

Acquire	Done Acquir	●
Array Counter	0	154

Auto Generated Engineering Screens

Overview Camera	BL47P-EA-DET-02
Sample Camera	BL47P-EA-DET-01
Motors	BL47P-MO-MAP-01
Panda	BL47P-MO-PANDA-01



ADAravis Camera

AD Aravis

AR Frames Completed	1820	
AR Frame Failures	0	
AR Frame Underruns	0	
AR Missing Packets	0	
AR Resent Packets	0	
AR Packet Resend	Always	
AR Packet Timeout	20000 us	
AR Frame Retention	100000 us	
AR Reset Camera	0	
AR Convert Pixel	Mono16L	Mono16Low
AR Shift Dir	None	None
AR Shift Bits	4	4

arvFeature Ungrouped

AR Connect Camera	1

AD Readout

Array Size X	728	
Array Size Y	544	
Array Size	396032	
Data Type	UInt8	UInt8
Color Mode	RGB1	RGB1
Max Size X	728	
Max Size Y	544	
Bin X	1	1
Bin Y	1	1
Min Y	0	0
Min X	0	0
Reverse X	No Yes	●
Reverse Y	No Yes	●
Size X	728	728
Size Y	544	544
Gain	0.000	0.000

AD Shutter

Shutter Mode	None	None
Shutter Status	●	
Shutter Control	Close Open	●
Shutter Open Delay	0.000	0.000
Shutter Close Delay	0.000	0.000

AD Collect

Num Queued Arrays	0	
Wait For Plugins	No	Yes
Acquire	Done Acquir	●
Array Counter	0	1820
Array Callbacks	Disable Enable	●
Acquire Time	0.500	0.500
Acquire Period	0.004	0.004
Num Images Counter	1820	
Num Images	1	1
Num Exposures	1	1
Image Mode	Continuou	Continuous
Trigger Mode	Off	Off
Detector State	Acquire	
Status Message		
Time Remaining	0.000	

ADDriver Ungrouped

Read Status		
Num Exposures	0	
String To Server	<not used by driver>	
String From Server	<not used by driver>	
Shutter Control EPICS	●	
Temperature Actual	54.1 C	
Frame Type	Normal	Normal
Temperature	25.0 C	25.0 C

AD Setup

Port Name	DET2.CAM
Manufacturer	Allied Vision Technologies
Model	Mako G-040C
Serial Number	50-0536985400
Firmware Version	00.01.54.21000
SDK Version	0.8.31
Driver Version	2.3
AD Core Version	3.12.1

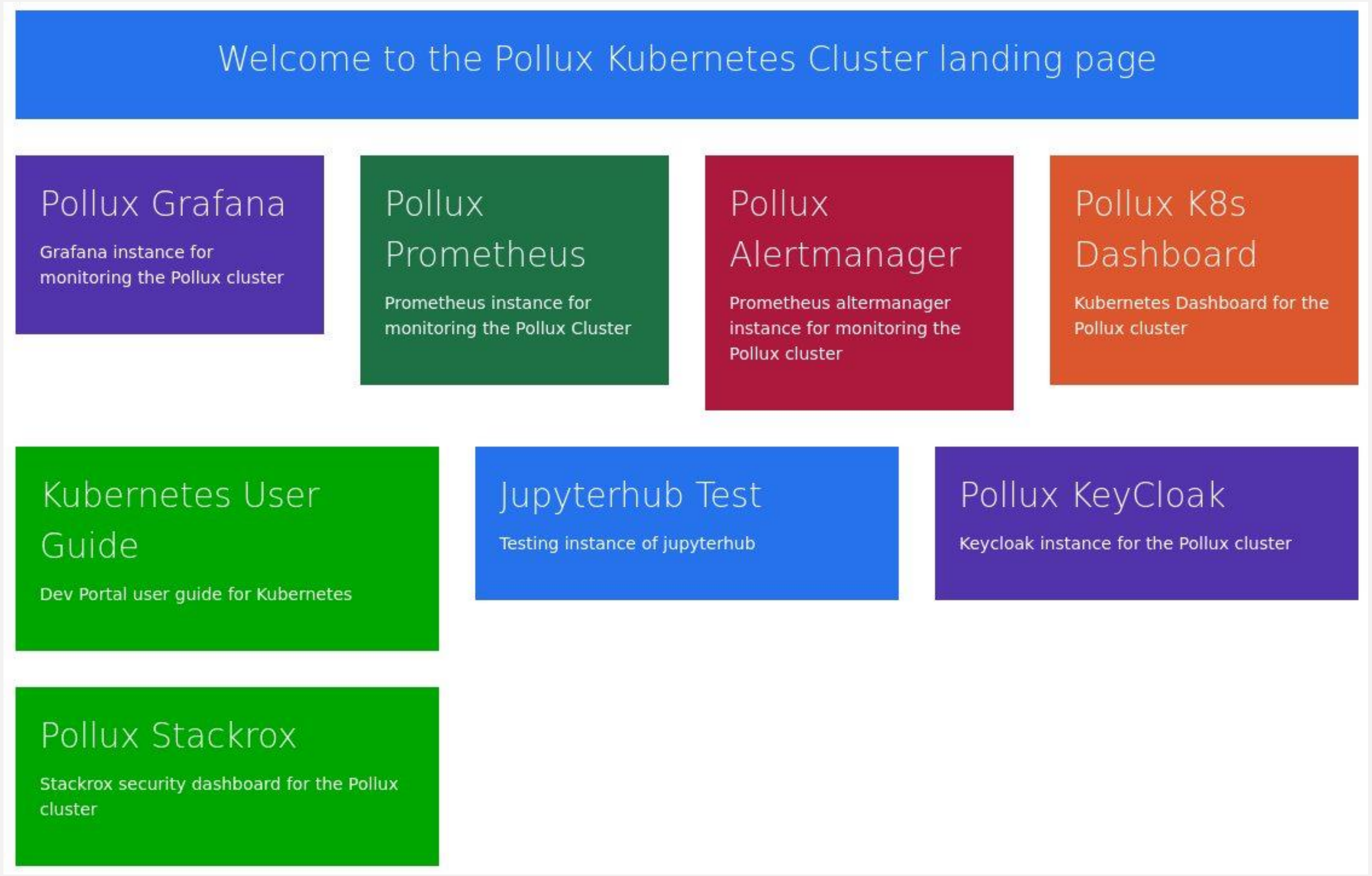
AD Attr File

ND Attributes File	
ND Attributes Macros	
ND Attributes Status	File not found

asynNDArrayDriver Ungrouped

Empty Free List		
Acquire Busy CB	●	
Bayer Pattern	RGGB	
Array Size Z	0	
Codec		
Compressed Size	0	
Unique Id	0	
Time Stamp	0.000	
Epics TS Sec	0	
Epics TS Nsec	0	
Pool Max Mem	0.0 MB	
Pool Used Mem	9.1 MB	
Pool Alloc Buffers	24	
Pool Free Buffers	1	
N Dimensions	0	0
Dimensions	[]	[]

Cluster landing page



Welcome to the Pollux Kubernetes Cluster landing page

Pollux Grafana

Grafana instance for monitoring the Pollux cluster

Pollux Prometheus

Prometheus instance for monitoring the Pollux Cluster

Pollux Alertmanager

Prometheus altermanager instance for monitoring the Pollux cluster

Pollux K8s Dashboard

Kubernetes Dashboard for the Pollux cluster

Kubernetes User Guide

Dev Portal user guide for Kubernetes

Jupyterhub Test

Testing instance of jupyterhub

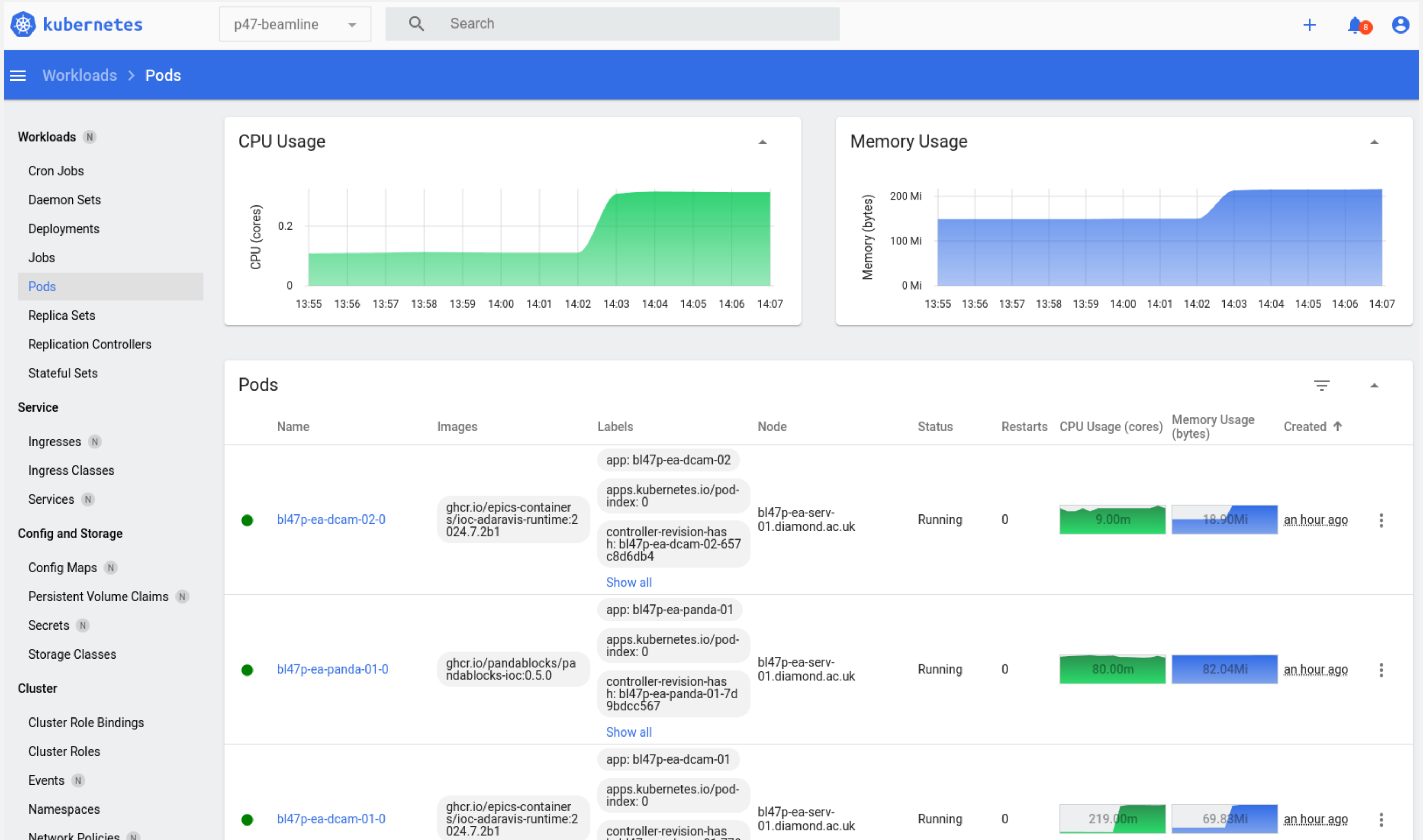
Pollux KeyCloak

Keycloak instance for the Pollux cluster

Pollux Stackrox

Stackrox security dashboard for the Pollux cluster

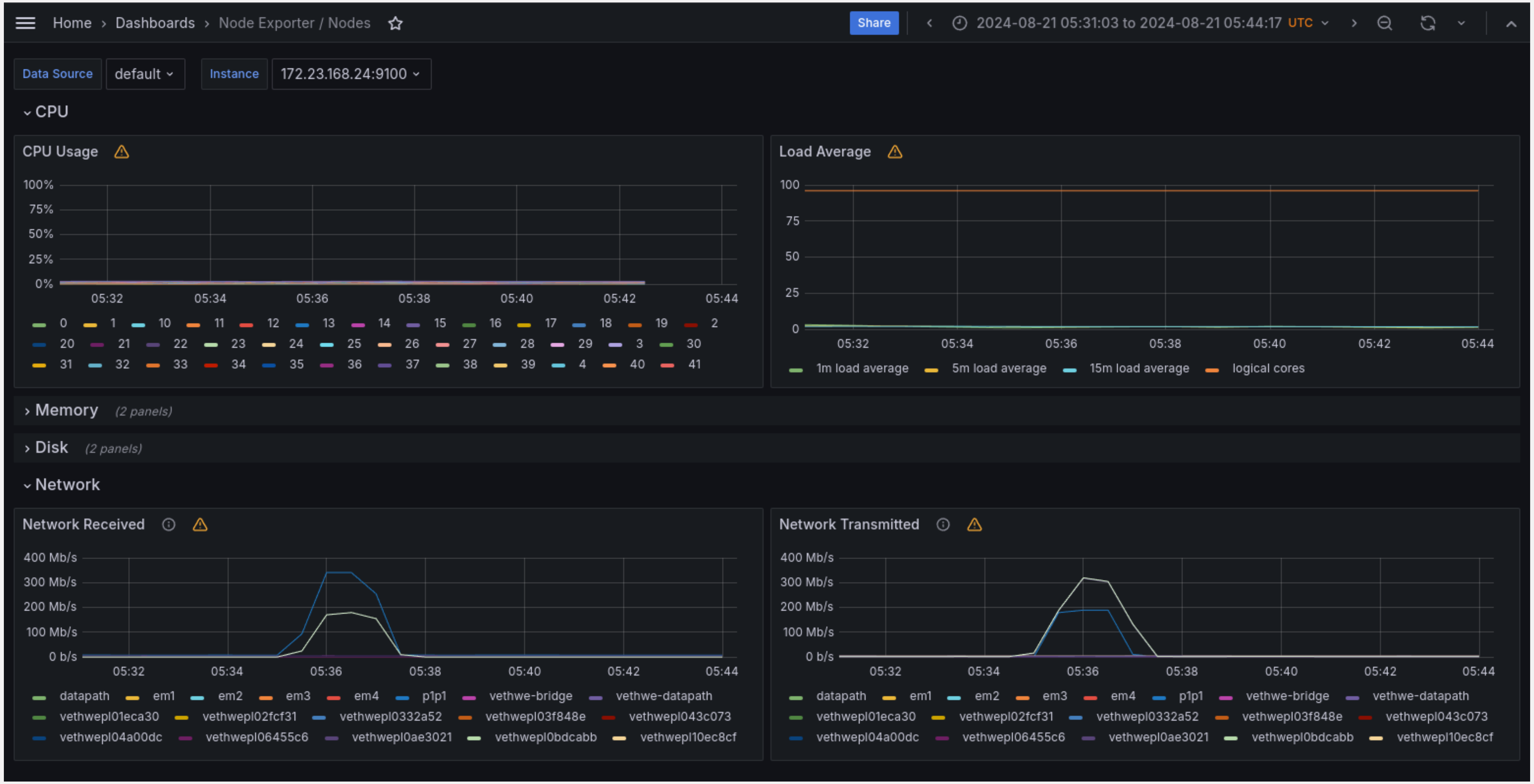
Kubernetes Dashboard



Pods

	Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
●	bl47p-ea-dcam-02-0	ghcr.io/epics-containers/ioc-adaravis-runtime:2024.7.2b1	app: bl47p-ea-dcam-02 apps.kubernetes.io/pod-index: 0 controller-revision-hash: bl47p-ea-dcam-02-657c8d6db4 Show all	bl47p-ea-serv-01.diamond.ac.uk	Running	0	9.00m	18.90Mi	an hour ago
●	bl47p-ea-panda-01-0	ghcr.io/pandablocks/pandablocks-ioc:0.5.0	app: bl47p-ea-panda-01 apps.kubernetes.io/pod-index: 0 controller-revision-hash: bl47p-ea-panda-01-7d9bdcc567 Show all	bl47p-ea-serv-01.diamond.ac.uk	Running	0	80.00m	82.04Mi	an hour ago
●	bl47p-ea-dcam-01-0	ghcr.io/epics-containers/ioc-adaravis-runtime:2024.7.2b1	app: bl47p-ea-dcam-01 apps.kubernetes.io/pod-index: 0 controller-revision-has	bl47p-ea-serv-01.diamond.ac.uk	Running	0	219.00m	69.83Mi	an hour ago

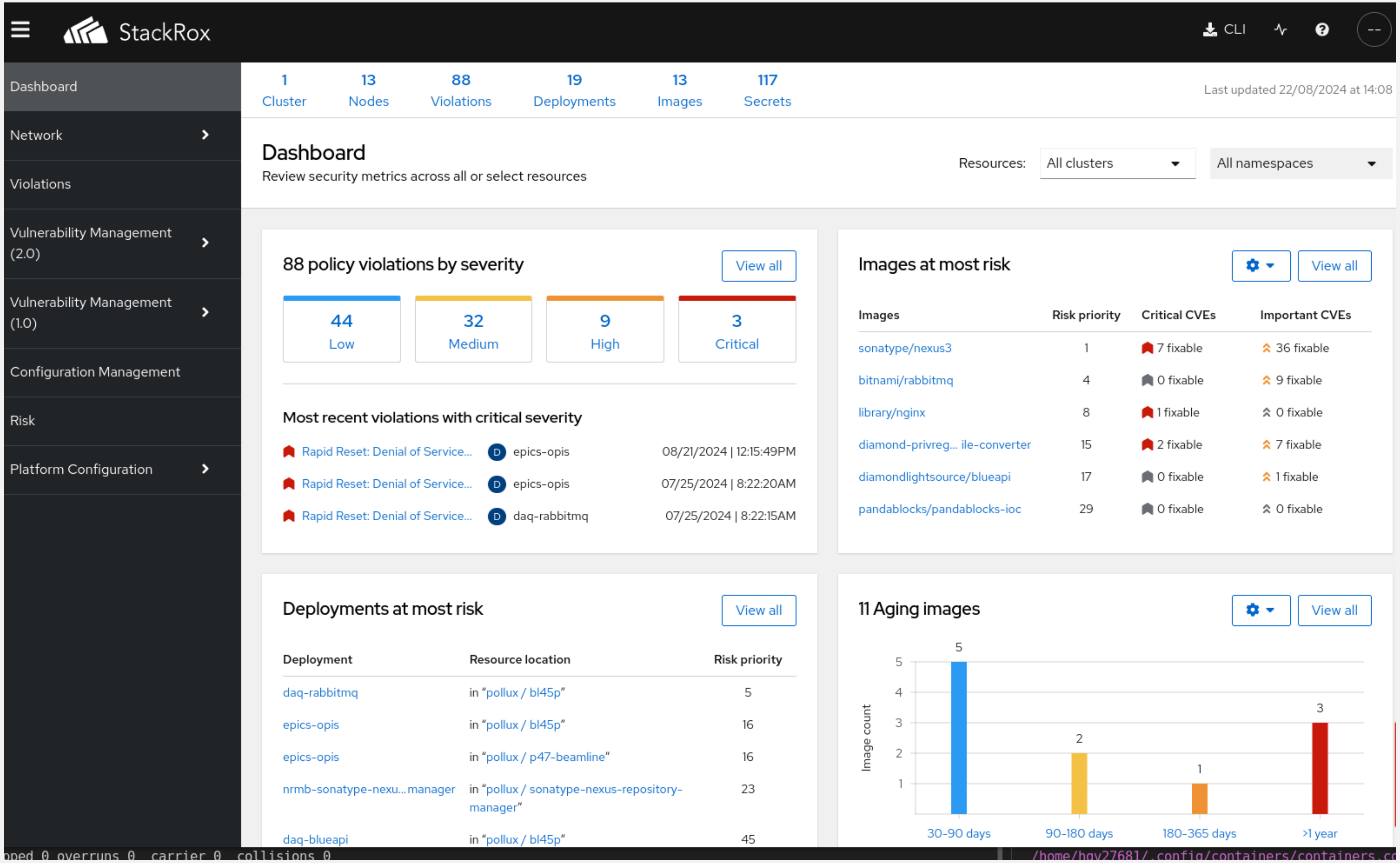
Grafana



Home › Dashboards › Node Exporter / Nodes
Share
2024-08-21 05:31:03 to 2024-08-21 05:44:17 UTC
Data Source default
Instance 172.23.168.24:9100
CPU
CPU Usage
Load Average
1m load average
5m load average
15m load average
logical cores
Memory (2 panels)
Disk (2 panels)
Network
Network Received
Network Transmitted
datapath
em1
em2
em3
em4
p1p1
vethwe-bridge
vethwe-datapath
vethwepl01eca30
vethwepl02fcf31
vethwepl0332a52
vethwepl03f848e
vethwepl043c073
vethwepl04a00dc
vethwepl06455c6
vethwepl0ae3021
vethwepl0bdcabb
vethwepl10ec8cf

StacksRox

Investigate CVE 2024-3094



StackRox

CLI

--

Dashboard

Network

Violations

Vulnerability Management (2.0)

Vulnerability Management (1.0)

Configuration Management

Risk

Platform Configuration

1	13	88	19	13	117
Cluster	Nodes	Violations	Deployments	Images	Secrets

Last updated 22/08/2024 at 14:08

Dashboard

Review security metrics across all or select resources

Resources: All clusters | All namespaces

88 policy violations by severity

View all

44	32	9	3
Low	Medium	High	Critical

Most recent violations with critical severity

Rapid Reset: Denial of Service...	D epics-opis	08/21/2024 | 12:15:49PM
Rapid Reset: Denial of Service...	D epics-opis	07/25/2024 | 8:22:20AM
Rapid Reset: Denial of Service...	D daq-rabbitmq	07/25/2024 | 8:22:15AM

Images at most risk

View all

Images	Risk priority	Critical CVEs	Important CVEs
sonatype/nexus3	1	7 fixable	36 fixable
bitnami/rabbitmq	4	0 fixable	9 fixable
library/nginx	8	1 fixable	0 fixable
diamond-privreg... ile-converter	15	2 fixable	7 fixable
diamondlightsource/blueapi	17	0 fixable	1 fixable
pandablocks/pandablocks-ioc	29	0 fixable	0 fixable

Deployments at most risk

View all

Deployment	Resource location	Risk priority
daq-rabbitmq	in "pollux / bl45p"	5
epics-opis	in "pollux / bl45p"	16
epics-opis	in "pollux / p47-beamline"	16
nrmb-sonatype-nexu...manager	in "pollux / sonatype-nexus-repository-manager"	23
daq-blueapi	in "pollux / bl45p"	45

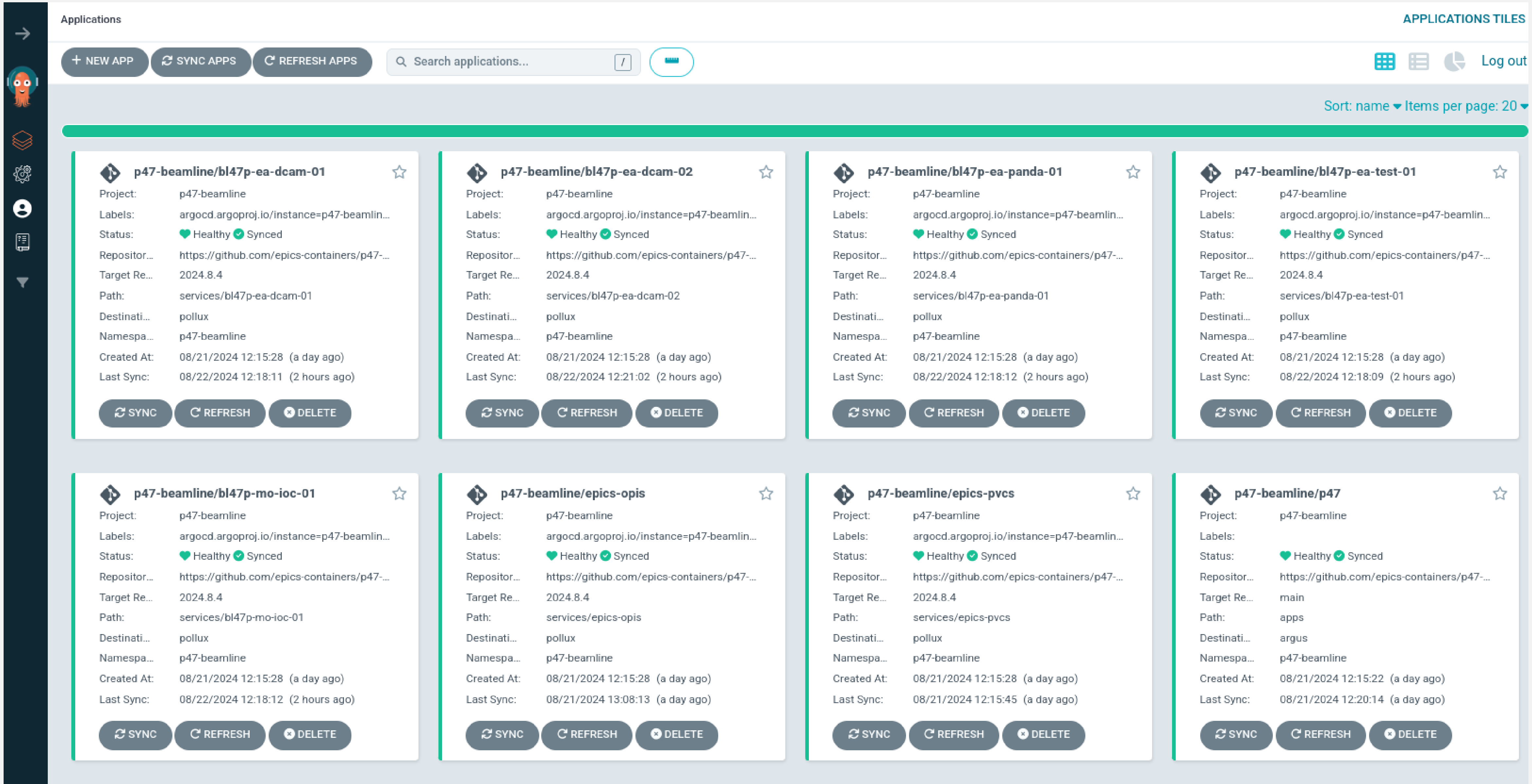
11 Aging images

View all

Age	Image count
30-90 days	5
90-180 days	2
180-365 days	1
>1 year	3

```
pped 0 overruns 0  carrier 0  collisions 0
/home/hgv27681/.config/containers/containers.co
```

Argo CD for p47-beamline



p47-beamline/bl47p-ea-dcam-01
Project: p47-beamline
Labels: argocd.argoproj.io/instance=p47-beamlin...
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... 2024.8.4
Path: services/bl47p-ea-dcam-01
Destinati... pollux
Namespa... p47-beamline
Created At: 08/21/2024 12:15:28 (a day ago)
Last Sync: 08/22/2024 12:18:11 (2 hours ago)

p47-beamline/bl47p-ea-dcam-02
Project: p47-beamline
Labels: argocd.argoproj.io/instance=p47-beamlin...
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... 2024.8.4
Path: services/bl47p-ea-dcam-02
Destinati... pollux
Namespa... p47-beamline
Created At: 08/21/2024 12:15:28 (a day ago)
Last Sync: 08/22/2024 12:21:02 (2 hours ago)

p47-beamline/bl47p-ea-panda-01
Project: p47-beamline
Labels: argocd.argoproj.io/instance=p47-beamlin...
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... 2024.8.4
Path: services/bl47p-ea-panda-01
Destinati... pollux
Namespa... p47-beamline
Created At: 08/21/2024 12:15:28 (a day ago)
Last Sync: 08/22/2024 12:18:12 (2 hours ago)

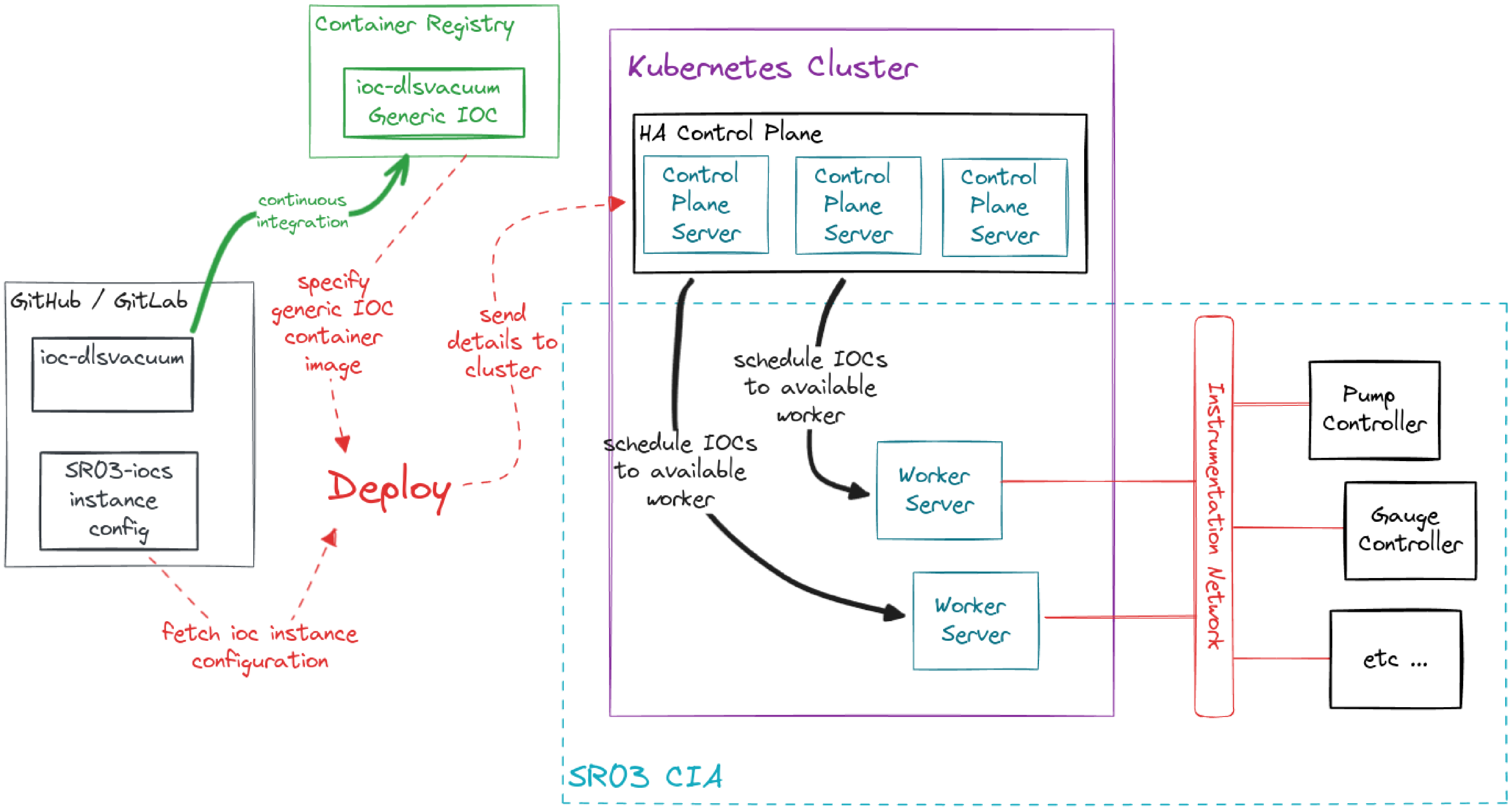
p47-beamline/bl47p-ea-test-01
Project: p47-beamline
Labels: argocd.argoproj.io/instance=p47-beamlin...
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... 2024.8.4
Path: services/bl47p-ea-test-01
Destinati... pollux
Namespa... p47-beamline
Created At: 08/21/2024 12:15:28 (a day ago)
Last Sync: 08/22/2024 12:18:09 (2 hours ago)

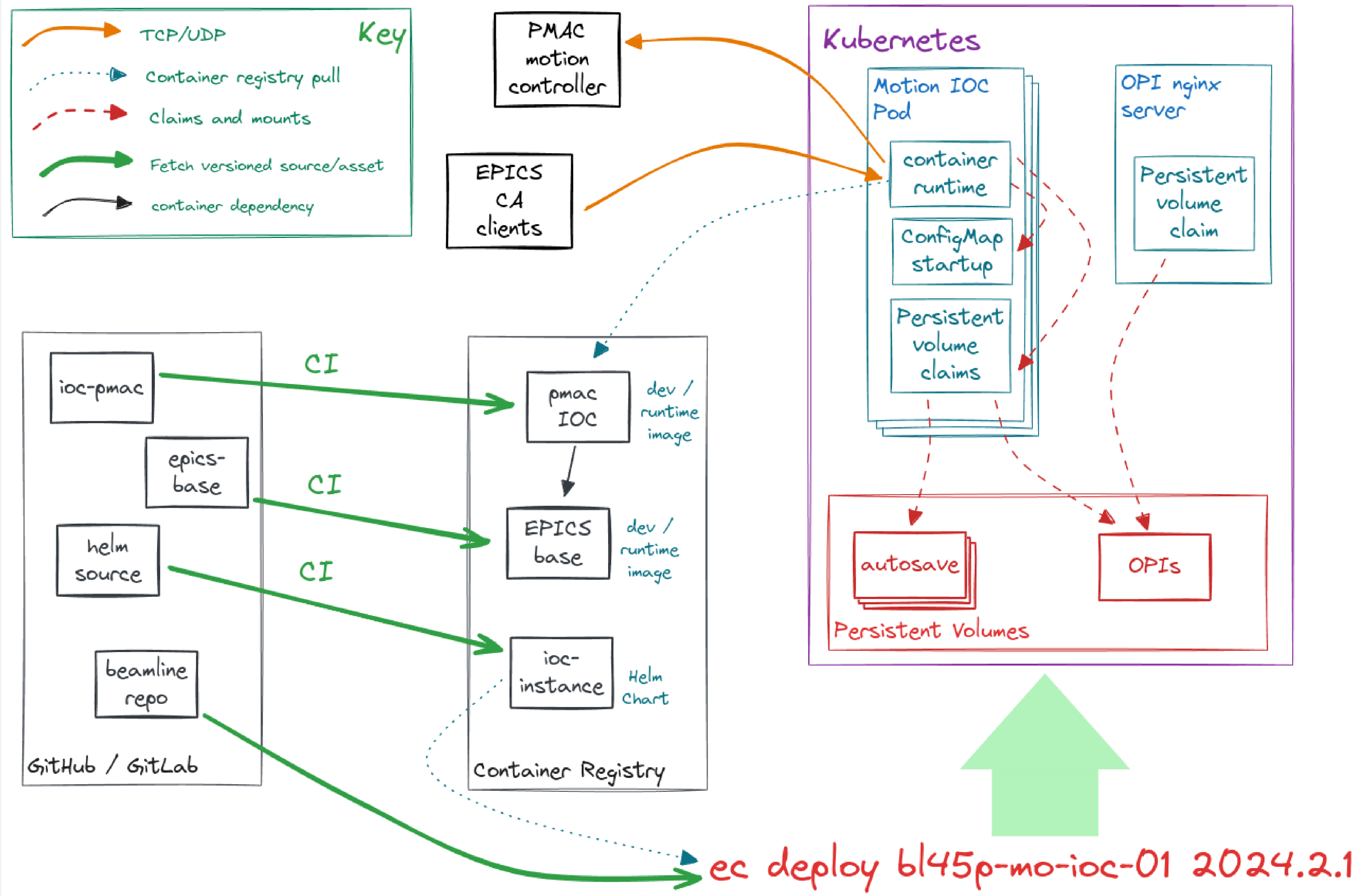
p47-beamline/bl47p-mo-ioc-01
Project: p47-beamline
Labels: argocd.argoproj.io/instance=p47-beamlin...
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... 2024.8.4
Path: services/bl47p-mo-ioc-01
Destinati... pollux
Namespa... p47-beamline
Created At: 08/21/2024 12:15:28 (a day ago)
Last Sync: 08/22/2024 12:18:12 (2 hours ago)

p47-beamline/epics-opis
Project: p47-beamline
Labels: argocd.argoproj.io/instance=p47-beamlin...
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... 2024.8.4
Path: services/epics-opis
Destinati... pollux
Namespa... p47-beamline
Created At: 08/21/2024 12:15:28 (a day ago)
Last Sync: 08/21/2024 13:08:13 (a day ago)

p47-beamline/epics-pvcs
Project: p47-beamline
Labels: argocd.argoproj.io/instance=p47-beamlin...
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... 2024.8.4
Path: services/epics-pvcs
Destinati... pollux
Namespa... p47-beamline
Created At: 08/21/2024 12:15:28 (a day ago)
Last Sync: 08/21/2024 12:15:45 (a day ago)

p47-beamline/p47
Project: p47-beamline
Labels:
Status: Healthy Synced
Repositor... https://github.com/epics-containers/p47-...
Target Re... main
Path: apps
Destinati... argus
Namespa... p47-beamline
Created At: 08/21/2024 12:15:22 (a day ago)
Last Sync: 08/21/2024 12:20:14 (a day ago)


Container Registry
ioc-dlsvacuum Generic IOC
Kubernetes Cluster
HA Control Plane
Control Plane Server
Control Plane Server
Control Plane Server
continuous integration
GitHub / GitLab
ioc-dlsvacuum
SR03-iocs instance config
specify generic IOC container image
send details to cluster
Deploy
fetch ioc instance configuration
schedule IOCs to available worker
schedule IOCs to available worker
Worker Server
Worker Server
Instrumentation Network
Pump Controller
Gauge Controller
etc ...
SR03 CIA


Key
TCP/UDP
Container registry pull
Claims and mounts
Fetch versioned source/asset
container dependency
PMAC motion controller
EPICS CA clients
Kubernetes
Motion IOC Pod
container runtime
ConfigMap startup
Persistent volume claims
OPI nginx server
Persistent volume claim
autosave
OPIs
Persistent Volumes
ioc-pmac
epics-base
helm source
beamline repo
GitHub / GitLab
CI
CI
CI
pmac IOC
dev / runtime image
EPICS base
dev / runtime image
ioc-instance
Helm Chart
Container Registry
ec deploy bl45p-mo-ioc-01 2024.2.1