Containerization and its applications for CERN Accelerator Controls

Rémi Voirin

01/12/2021 – Academic Training Lecture – Controlling the CERN Accelerator Complex (3/4)

Overview

Key concepts

- UNIX process isolation: jails, namespaces, containers
- Container orchestration
- Microservices vs monoliths
- Containers at CERN and in Controls

Prerequisites

- UNIX systems basics (kernel, process, init)
- Network basics (DNS, IP address)
- Overall structure of CERN (in particular the A&T Sector, BE and IT Departments)

Outline

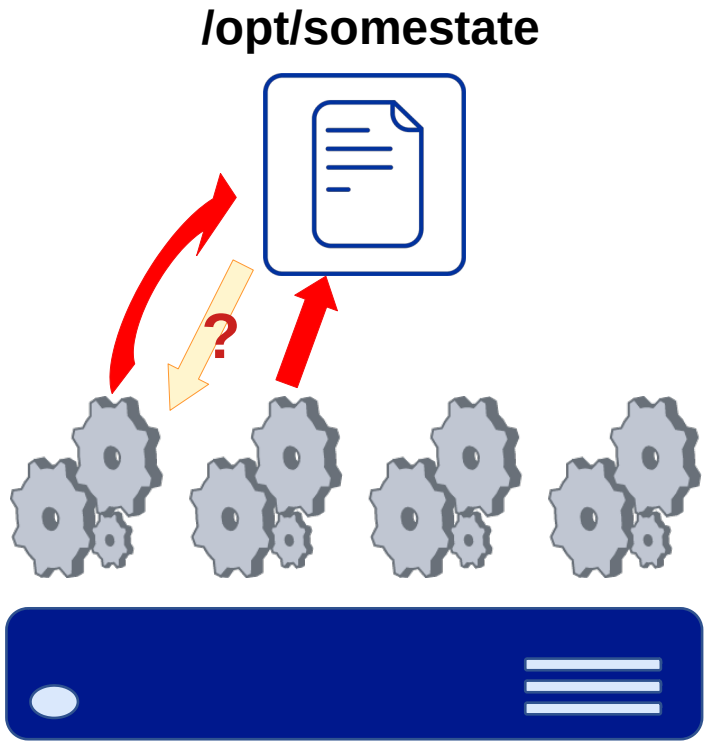
Isolation mechanisms in UNIX systems

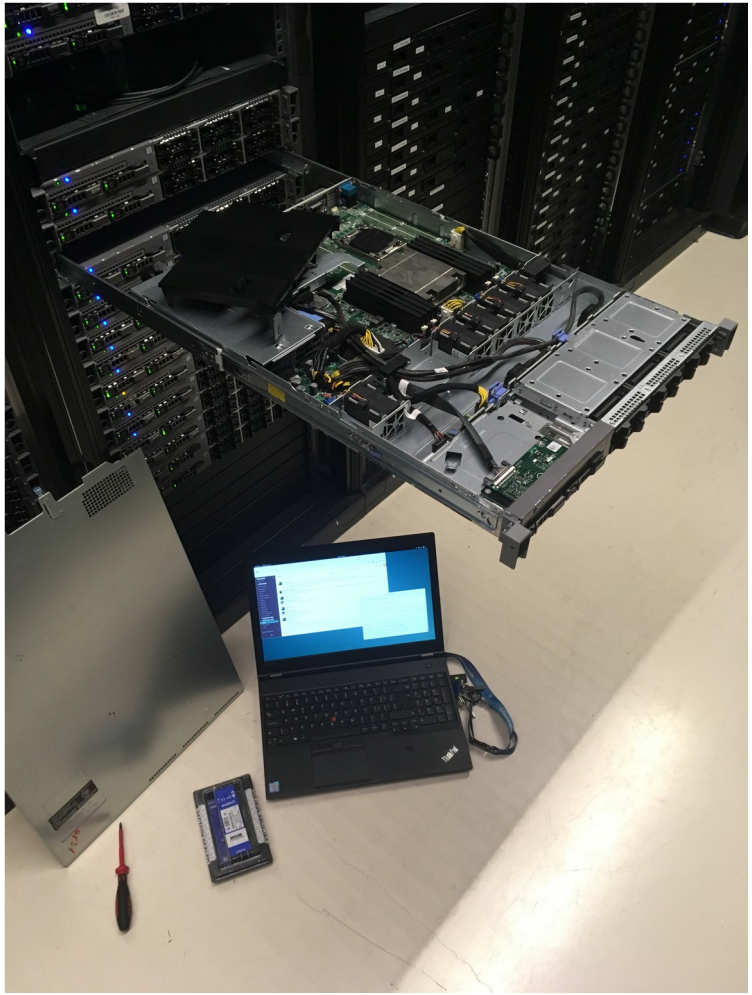
Containers in industry and at CERN

Status and use cases for Accelerator Controls

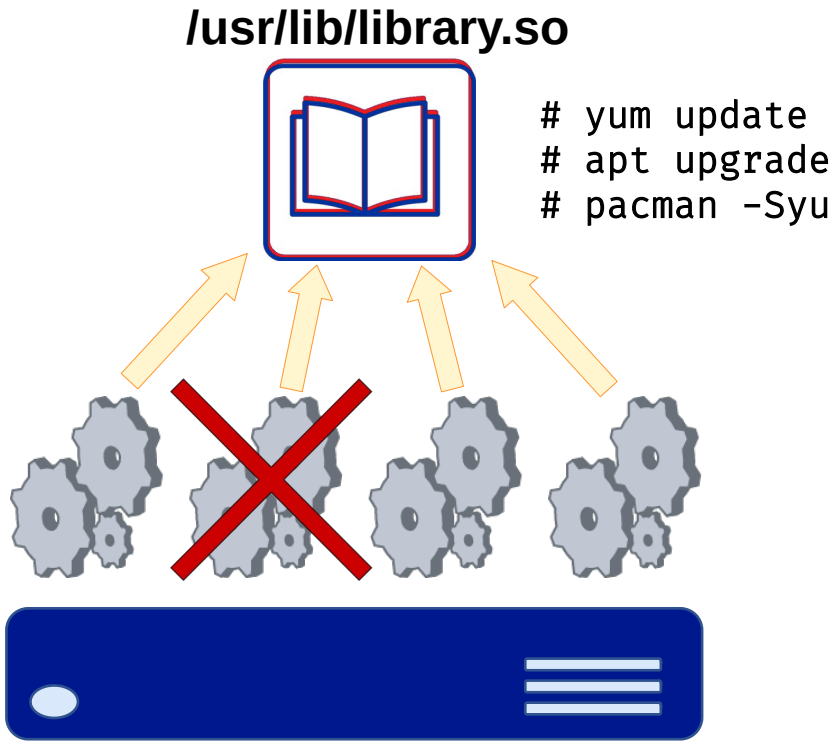
Table of Contents

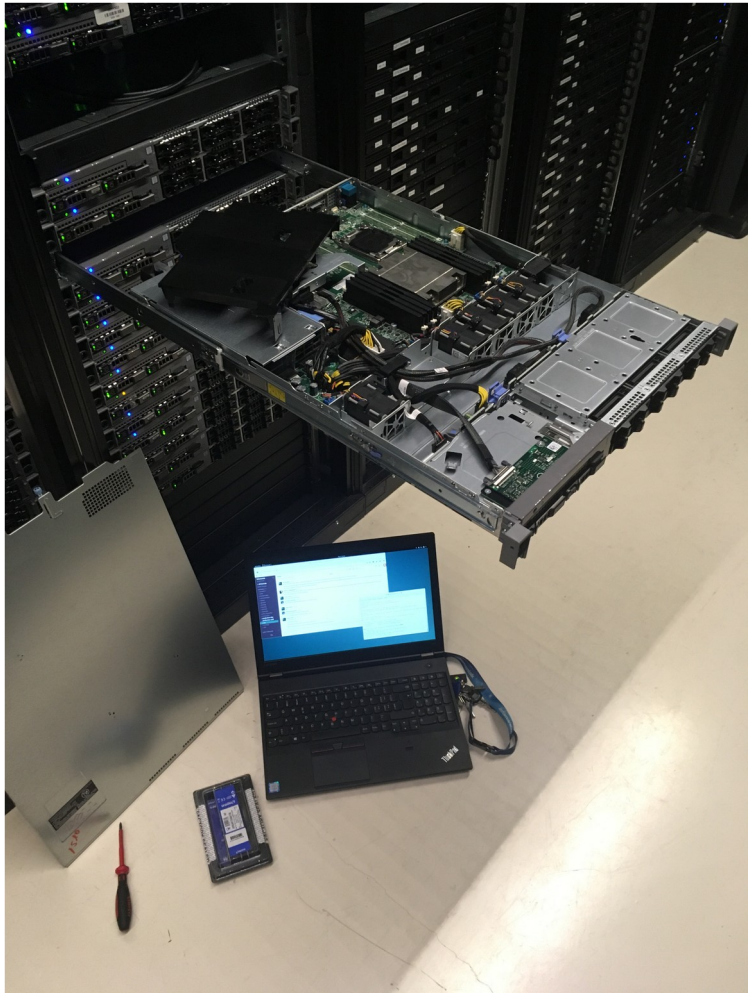
Purposes of isolation



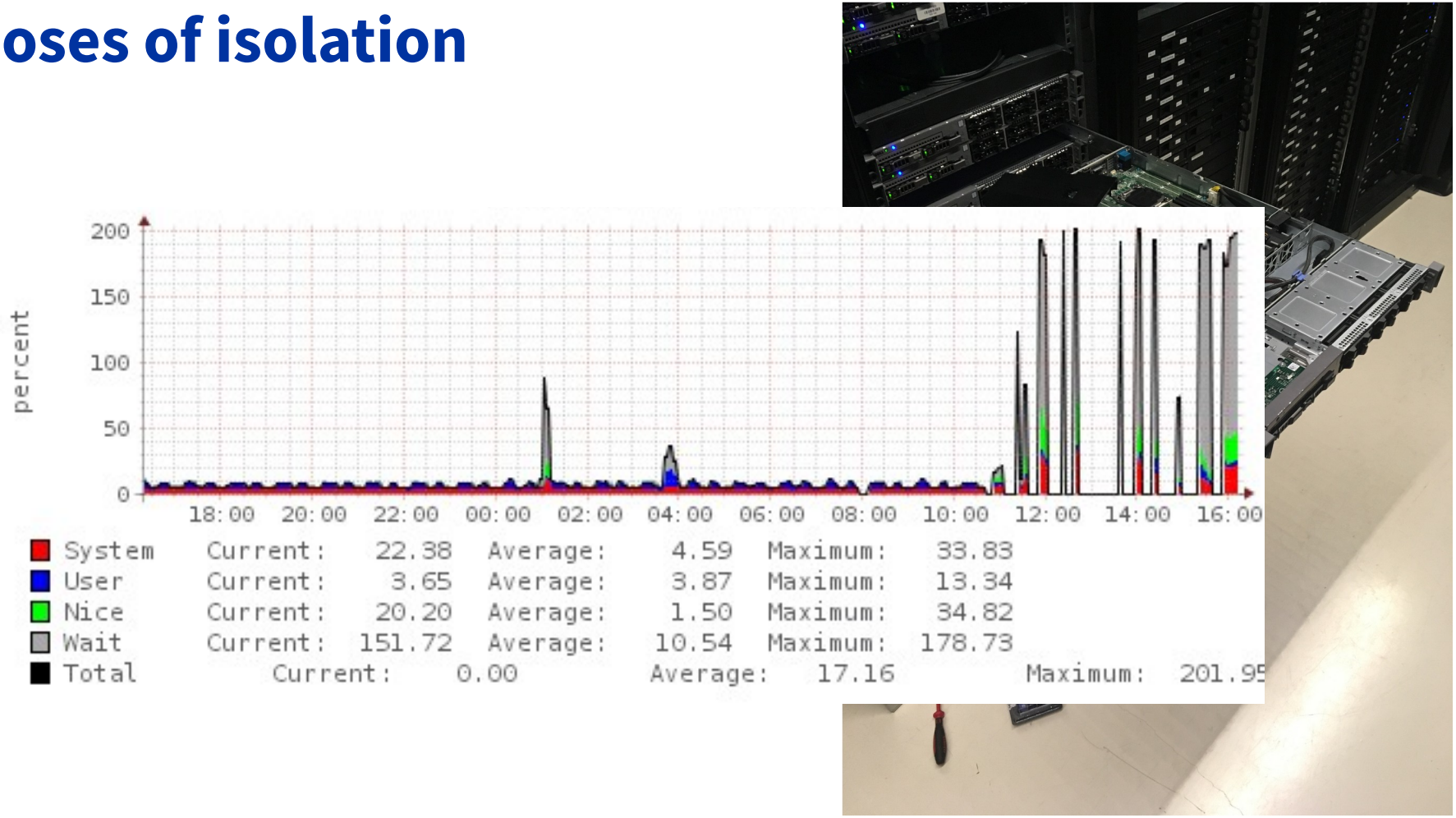


Purposes of isolation

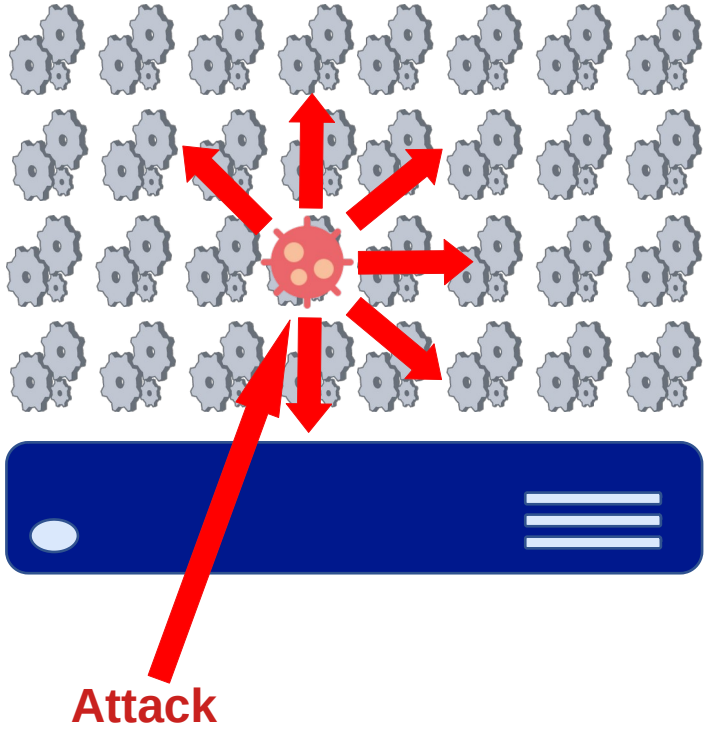


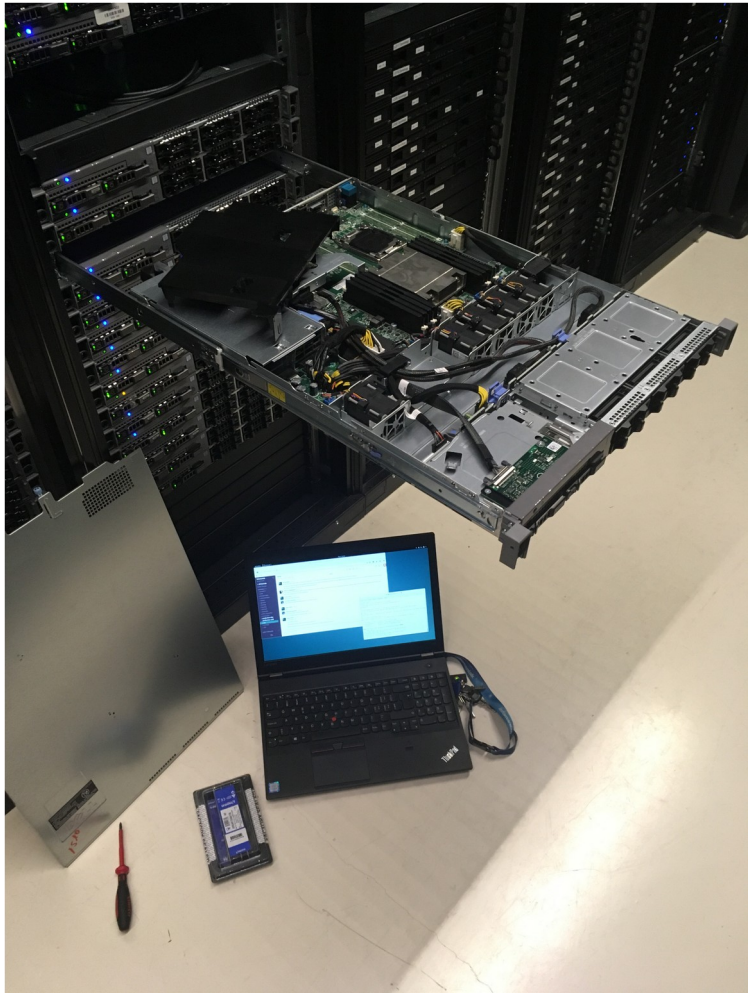


Purposes of isolation



Purposes of isolation





Technology example: FreeBSD jails

Released in 2000 in version 4.0

- Complete system sandboxing
- Main goal: **enhanced security**
  - Separate file hierarchy (visible filesystem)
  - Separate visible processes
  - Separate users
  - Different installed packages
  - Different IP addresses

Technology example: FreeBSD jails

DEMO TIME!

- More info on jails:
  - FreeBSD Jails Quickstart
    - https://zenlot.medium.com/freebsd-jails-quick-start-64235c6a4361
  - Official FreeBSD documentation
    - https://docs.freebsd.org/en/books/handbook/jails/

Technology example: Linux kernel namespaces

First namespace introduced in 2002 in version 2.4.19

- Eight system calls for eight isolation areas:
  - Mount (CLONE_NEWNS)
  - InterProcess Communication (CLONE_NEWIPC)
  - Time (CLONE_NEWTIME)
  - Time-sharing (CLONE_NEWUTS)
  - User (CLONE_NEWUSER)
  - PID (CLONE_NEWPID)
  - network (CLONE_NEWNET)
  - cgroups (CLONE_NEWCGROUP)

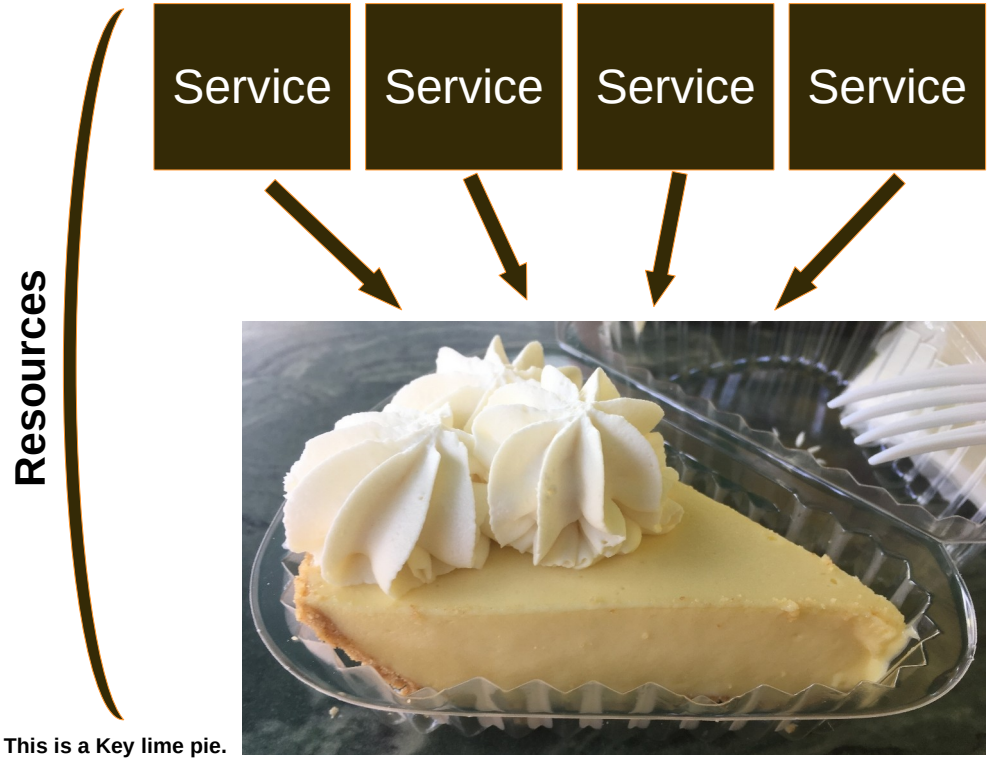
For more info:

```
$ man 7 namespaces
```

The cgroup namespace

Linux Control Groups

- Underlying concepts:
  - Slices
  - Resources
- Control Groups can be applied to resources



The cgroup namespace

DEMO TIME!

- Learn more at:
  - `$ man 7 cgroups`
  - Inside cgroups
    - https://0xax.gitbooks.io/linux-insides/content/Cgroups/linux-cgroups-1.html
  - Series of detailed articles
    - https://www.redhat.com/sysadmin/cgroups-part-one

Industrializing isolation mechanisms

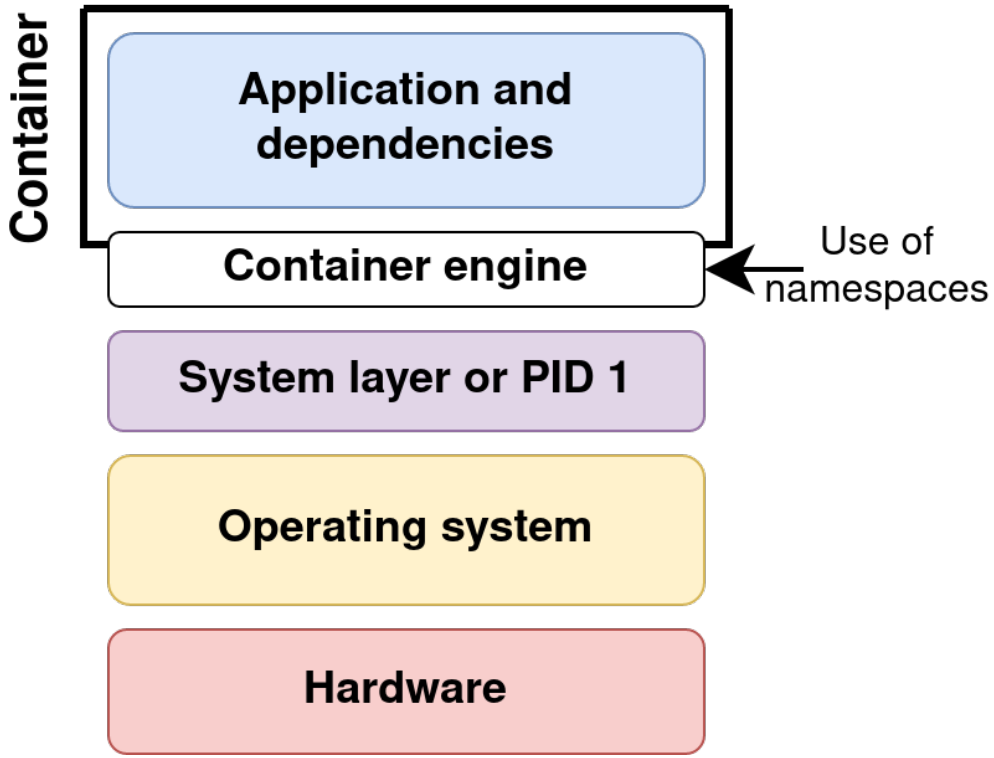
Docker (2013)

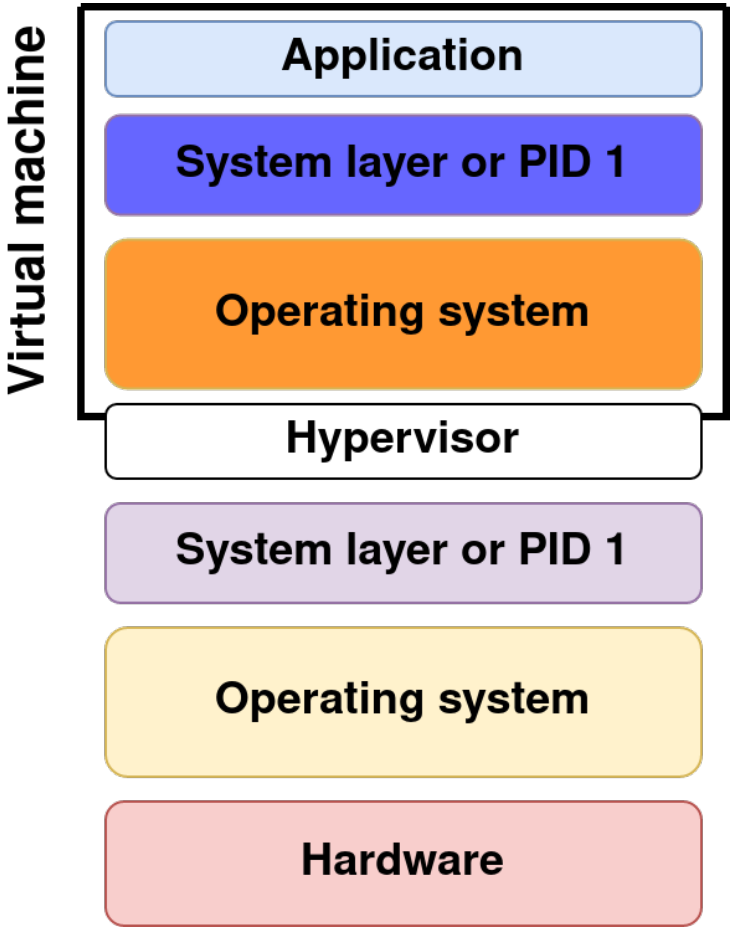
- Provided an ecosystem around Linux namespaces
  - Public registry to store “container images”: the Docker Hub
  - One command to deploy a containerized application

The Open Container Initiative (2015)

- Goal: “design standards for operating-system level virtualization” (i.e. containerization)
- Released specifications on storing container images and running containers
- “OCI-compliance” in Docker, Podman, Singularity,...

Containerization vs virtualization





Containerization: new paradigm

Vocabulary

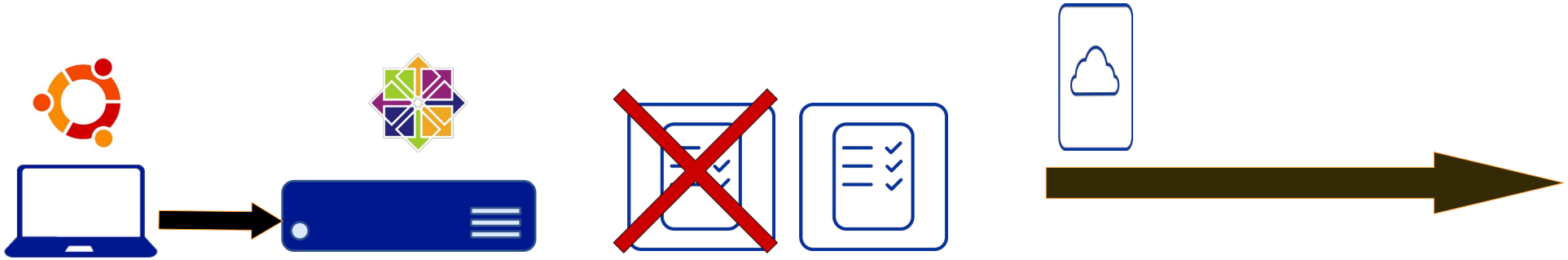
- Package / stored application (rpm, deb, binary,…) → Image
- Repository → Registry
- Deploy and start an application → Run (a container)
- Download → Pull
- Upload → Push

Table of Contents

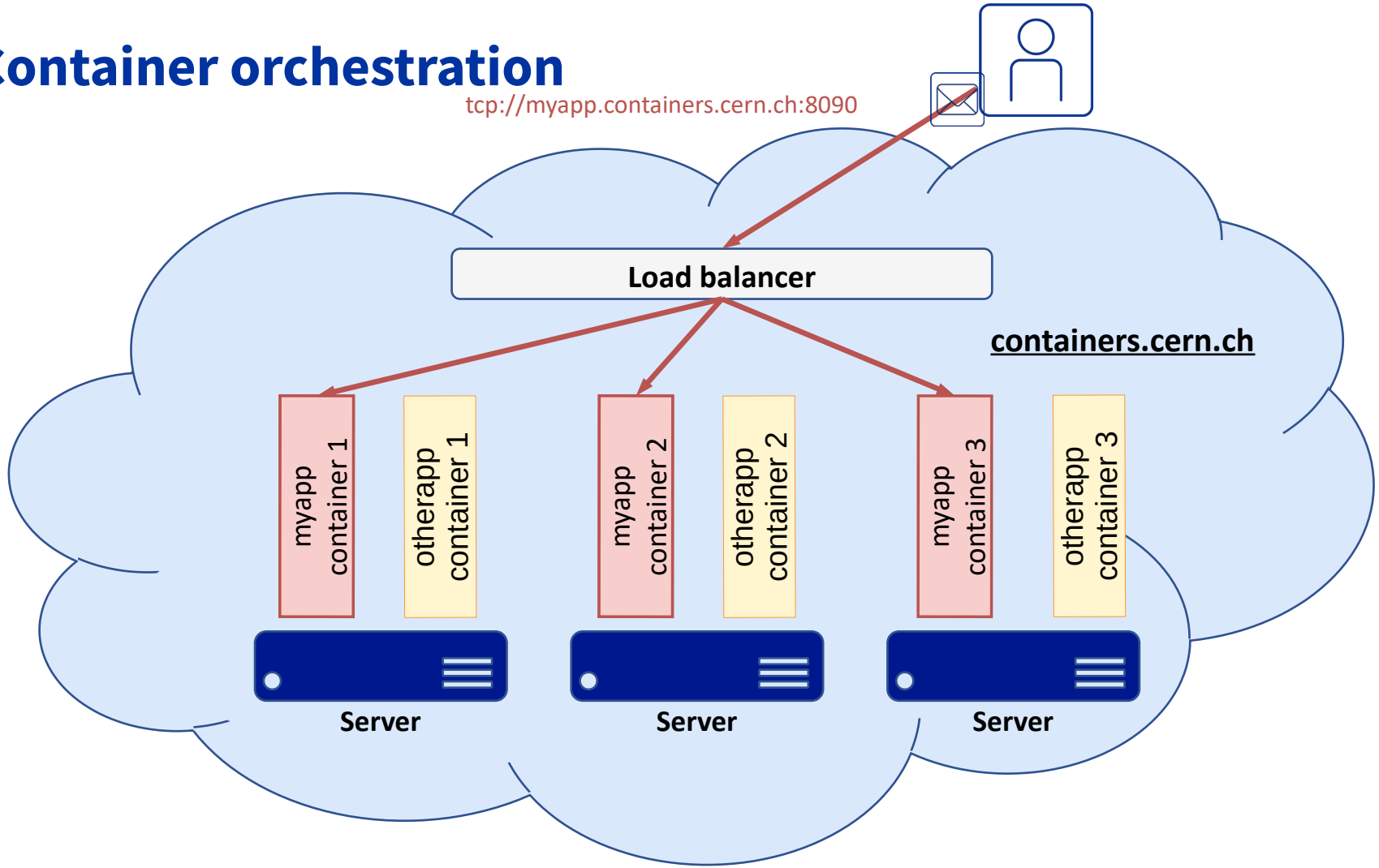
Plain containerization: general use cases

Containers are becoming the norm in the industry for:

- Idempotent execution between development and operational environments
- Streamlining the use of DevOps tools
- Managing application dependencies
- Encapsulating legacy solutions



Container orchestration



Container orchestration

What you get

- Scalability at hand, can be fully automated
- Load balancing “for free” and specific deployment strategies
- Zero downtime if you run multiple replicas

Requirements

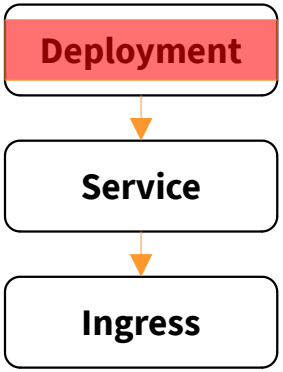
- Hardware and geographical redundancy
- To get all the goodness: microservice architecture
- Describe your application to deploy it

Example: Machine-Learning Platform

Curtesy of Jean-Baptiste de Martel

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: simple-ann-subpackage--secondmodel-simple-ann-default-deployment
  labels:
    app: simple-ann-subpackage--secondmodel-simple-ann-default
spec:
  replicas: 3
  selector:
    matchLabels:
      app: standalone-model
      modelType: simple_ann-subpackage--secondmodel
      modelParameters: simple_ann_default
  template:
    metadata:
      labels:
        app: standalone-model
        modelType: simple_ann-subpackage--secondmodel
        modelParameters: simple_ann_default
    spec:
      containers:
       - name: simple-ann-subpackage--secondmodel-simple-ann-default
         image:    registry.cern.ch/acc/simple_ann-subpackage--secondmodel
```

Kubernetes PoC
for accelerators

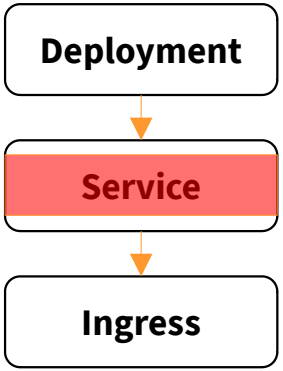


Example: Machine-Learning Platform

Curtesy of Jean-Baptiste de Martel

```yaml
apiVersion: v1
kind: Service
metadata:
  name: simple-ann-subpackage--secondmodel-simple-ann-default
spec:
  selector:
    app: standalone-model
    modelType: simple_ann-subpackage--secondmodel
    modelParameters: simple_ann_default
  ports:
    - protocol: TCP
      port: 5000
      targetPort: 5000
```

Kubernetes PoC
for accelerators



Example: Machine-Learning Platform

Curtesy of Jean-Baptiste de Martel

```yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: simple-ann-subpackage--secondmodel-simple-ann-default
  namespace: default
  annotations:
    kubernetes.io/ingress.class: traefik
    traefik.ingress.kubernetes.io/rule-type: PathPrefixStrip
#    traefik.ingress.kubernetes.io.rewrite-target: /predict
    traefik.frontend.entryPoints: "http"
spec:
  rules:
  - host:            .cern.ch
    http:
      paths:
      - path: /simple_ann-subpackage--secondmodel--simple_ann_default
        backend:
          serviceName: simple-ann-subpackage--secondmodel-simple-ann-default
          servicePort: 5000
```

Load balancer

Kubernetes PoC
for accelerators

Deployment

↓

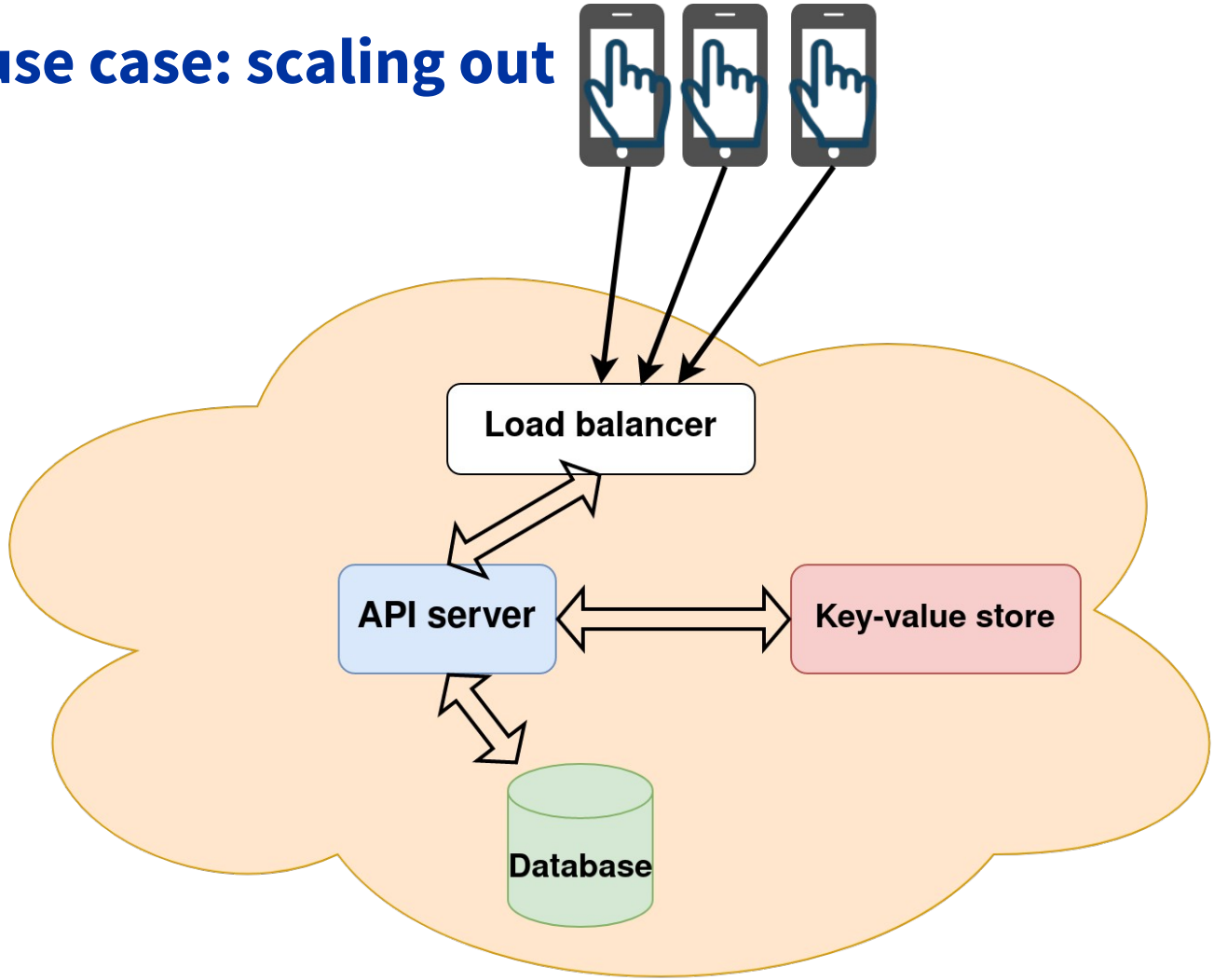
Service

↓

Ingress

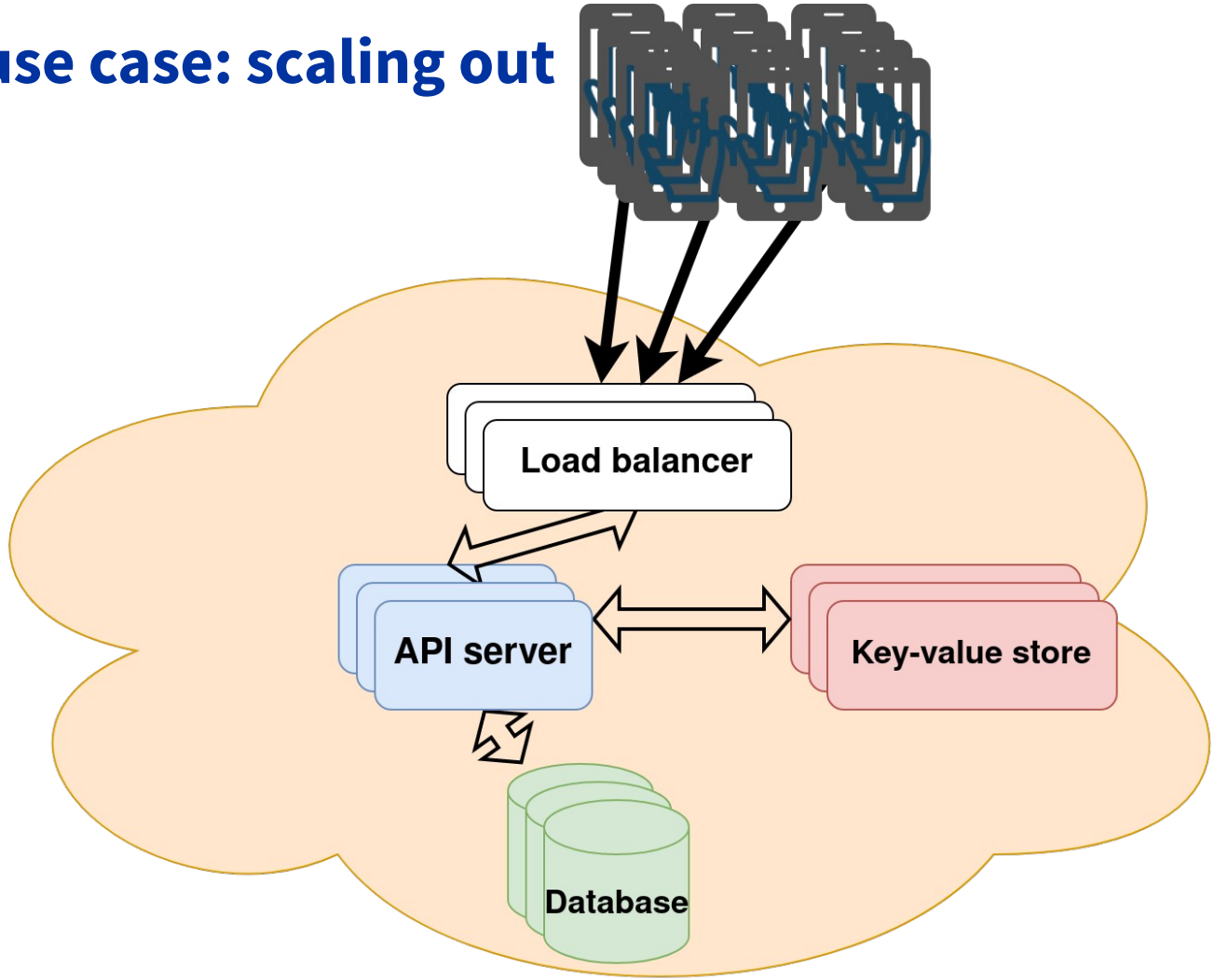
Orchestration use case: scaling out

Infrastructure costs: $



Orchestration use case: scaling out

Infrastructure costs: $$$



On-premise or cloud provider?

… and many more!

Containers at CERN (outside of Controls)

- Plain containers: use your existing system and GitLab, IT images
- Container orchestration: two main choices

Info on training in extra slides

Doc: https://clouddocs.web.cern.ch/containers/quickstart.html

Kubernetes from the Cloud Container Service

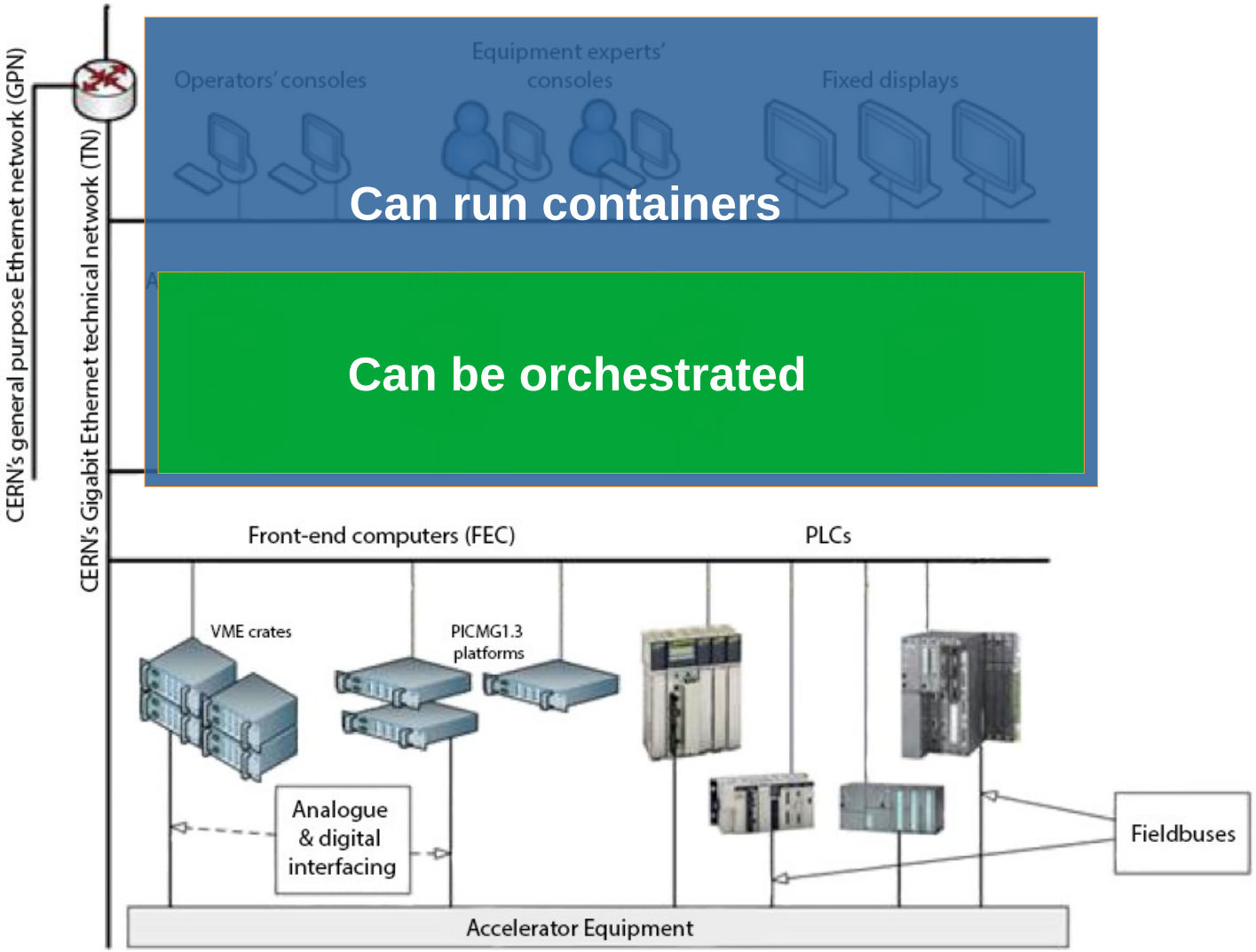
- Available for all orchestrated deployments
- More powerful but steeper learning curve
- Fine tunable
- GPN (supported) and for accelerators (PoC)

Doc: https://cern.service-now.com/service-portal/?id=kb_article&n=KB0004358

OPENSHIFT

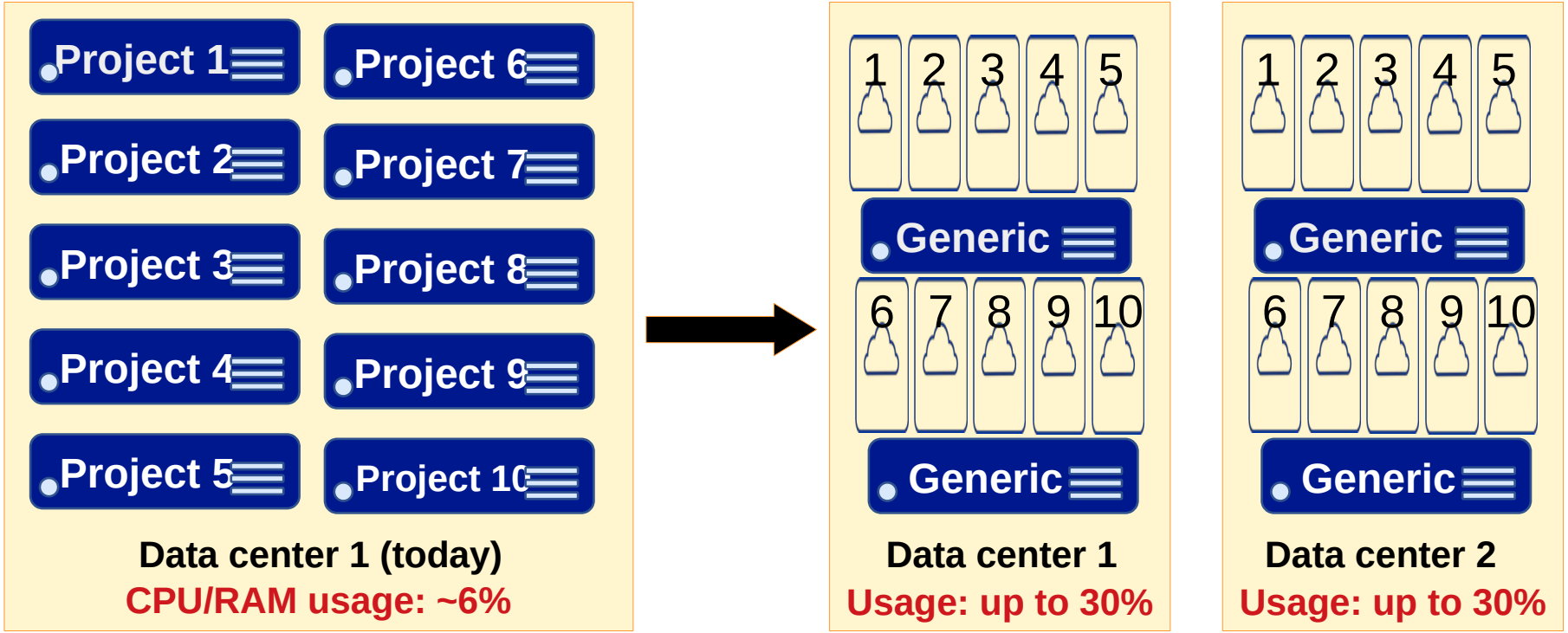
- Made for hosting web applications at CERN
- User-friendly web interface
- Many built-in features e.g. TLS certificates
- Operates from the GPN only

Table of Contents


CERN's general purpose Ethernet network (GPN)
CERN's Gigabit Ethernet technical network (TN)
Operators' consoles
Equipment experts' consoles
Fixed displays
Can run containers
Can be orchestrated
Front-end computers (FEC)
PLCs
VME crates
PICMG1.3 platforms
Analogue & digital interfacing
Fieldbuses
Accelerator Equipment

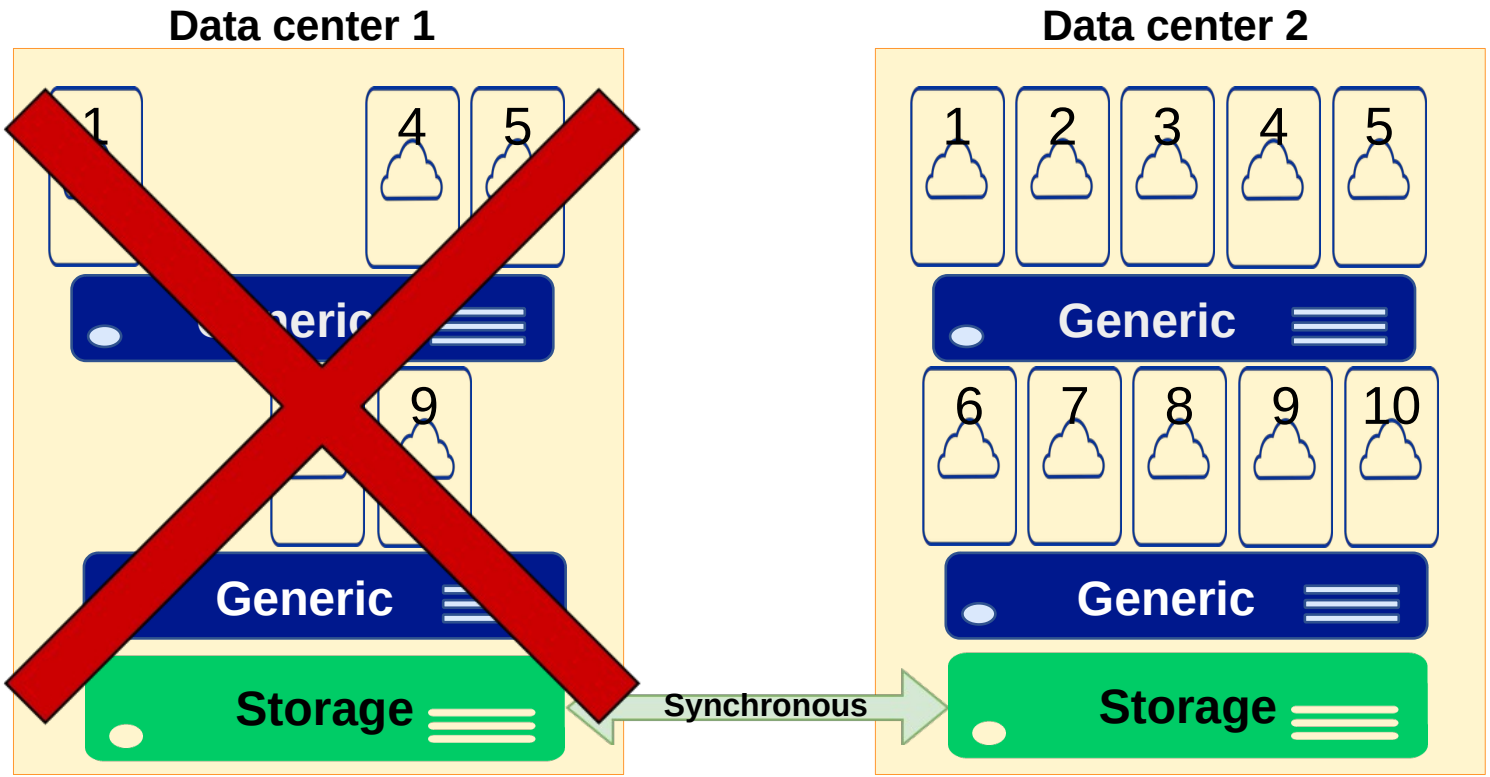
Leveraging container orchestration in Controls

Improved usage of the middle-tier bare metal infrastructure + data center redundancy



Leveraging container orchestration in Controls

Quick failover of all applications



Leveraging container orchestration in Controls

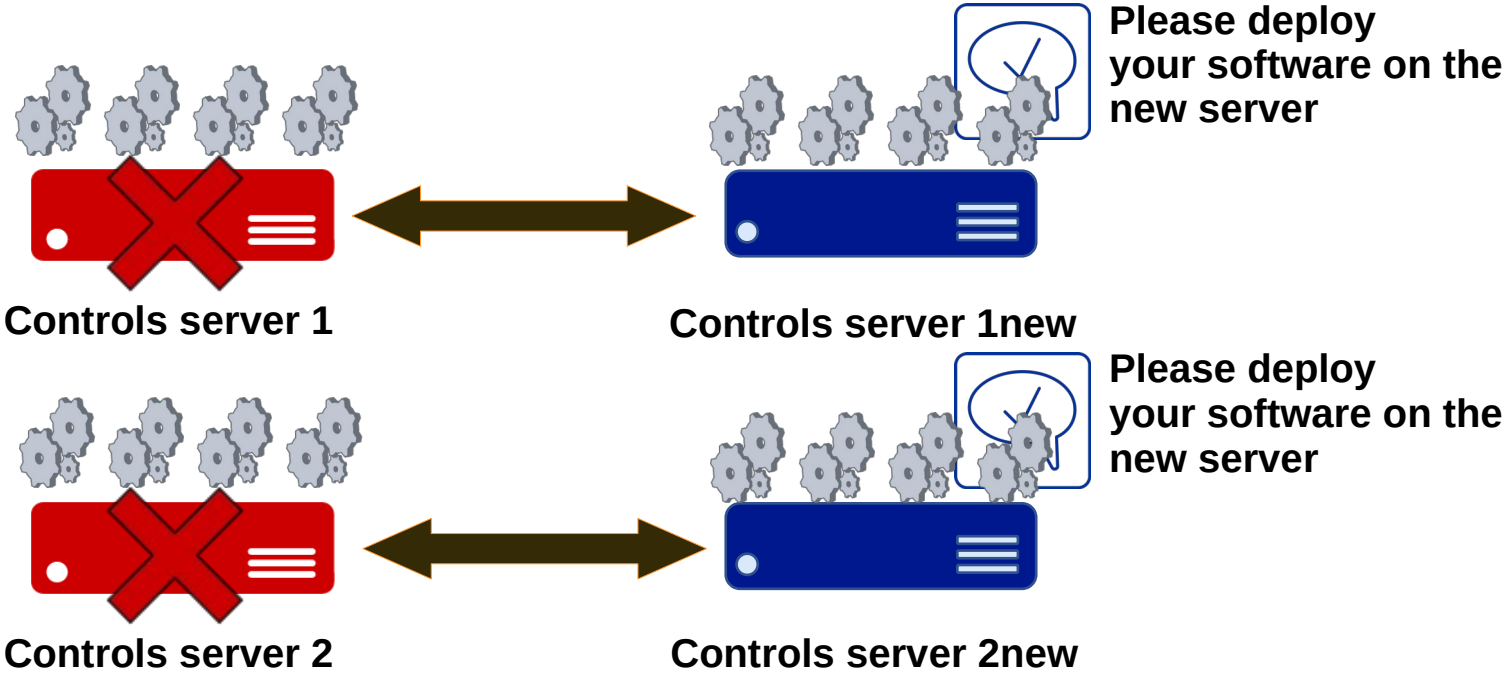
Curtesy of Marine Gourber-Pace

Shorter Controls sysadmin days (1/3)

							CERN Xmas holidays					
	46	47	48	49	50	51	52		Week 1		2	3
	15.Nov	22.Nov	29.Nov	06.Dec	13.Dec	20.Dec	22.Dec	27.Dec	03.Jan	05.Jan	10.Jan	17.Jan
LINAC4		Stop on 26									LN4 source only	
PSB												
CPS												
SPS												
LINAC3	Stop on 19											
ISOLDE stand alone												tbc
LHC Powering Campaign						tbc			CONTROLS SYS ADMIN DAYS [4 -7 Jan]			
CLEAR												
SM18						tbc						
INDIVIDUAL SYSTEM TESTS with remote controls LN4 PSB CPS SPS		LIN4 only										
SPS beam Dump tests (SBDS) (SY-ABT)		Tests on 22Nov only										
LHC CRYOGENICS						tbc					tbc	tbc
LEGEND:	operate	shutdown										

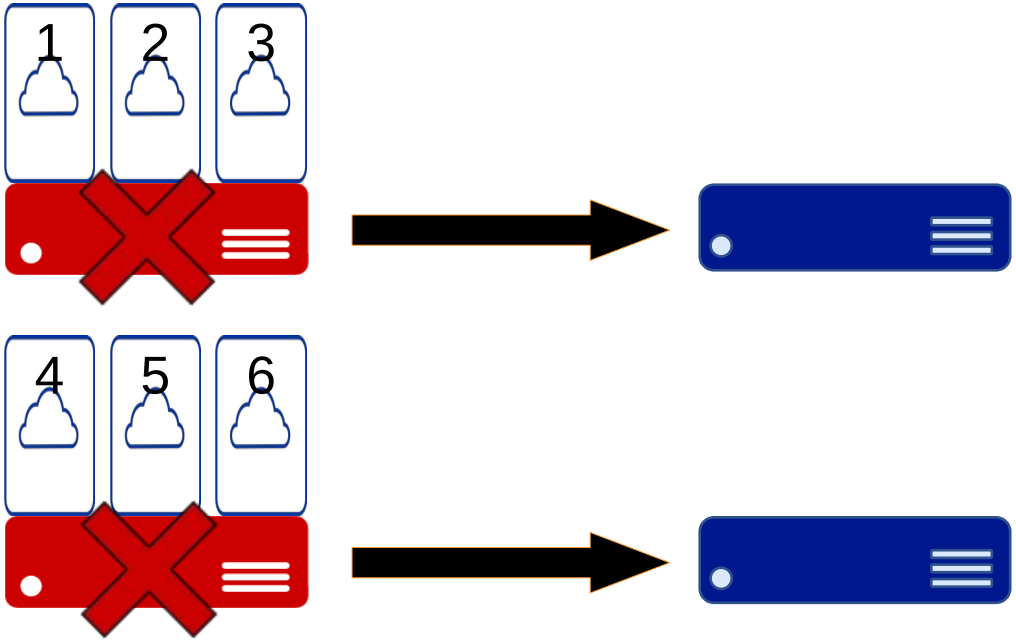
Leveraging container orchestration in Controls

Shorter Controls sysadmin days (2/3)



Leveraging container orchestration in Controls

Shorter Controls sysadmin days (3/3)

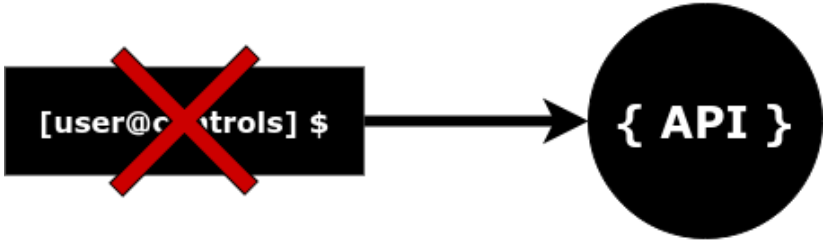


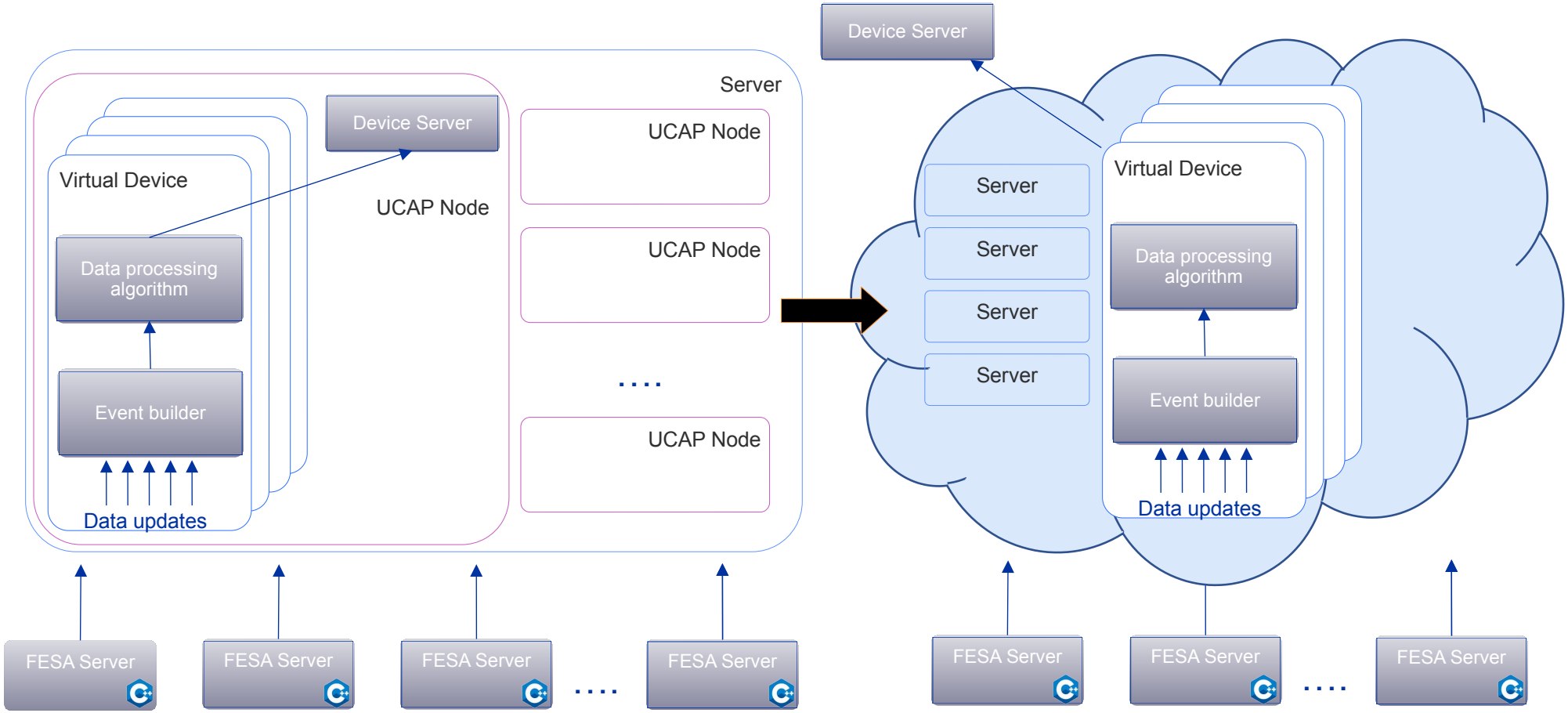
Hardware to decommission **Pre-provisioned servers**

Moving away from the “Linux shell” interface

Container orchestration changes the way the middle-tier is exposed to developers

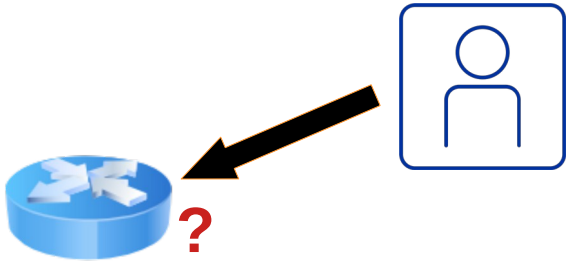
- Give access to APIs instead
- More security by design
- Easier for developers
- Some use cases would be easy to translate
  - REST APIs (cf Bartek Urbaniec’s presentation tomorrow)
  - UCAP
  - Other scalable components (e.g. Grafana)

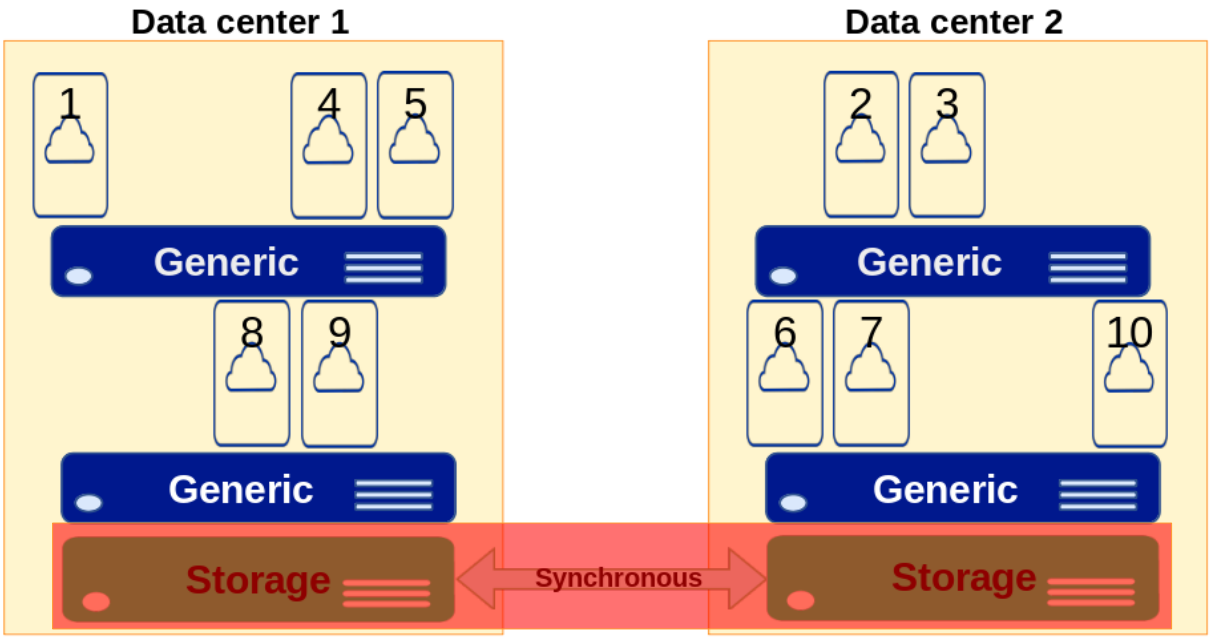



Server
Device Server
UCAP Node
Virtual Device
UCAP Node
Data processing algorithm
UCAP Node
Event builder
....
UCAP Node
Data updates
FESA Server
FESA Server
FESA Server
....
FESA Server
Device Server
Server
Server
Server
Server
Virtual Device
Data processing algorithm
Event builder
Data updates
FESA Server
FESA Server
....
FESA Server

Container orchestration requirements

Rethink network and storage





In the meantime, we address the need for plain containerization.

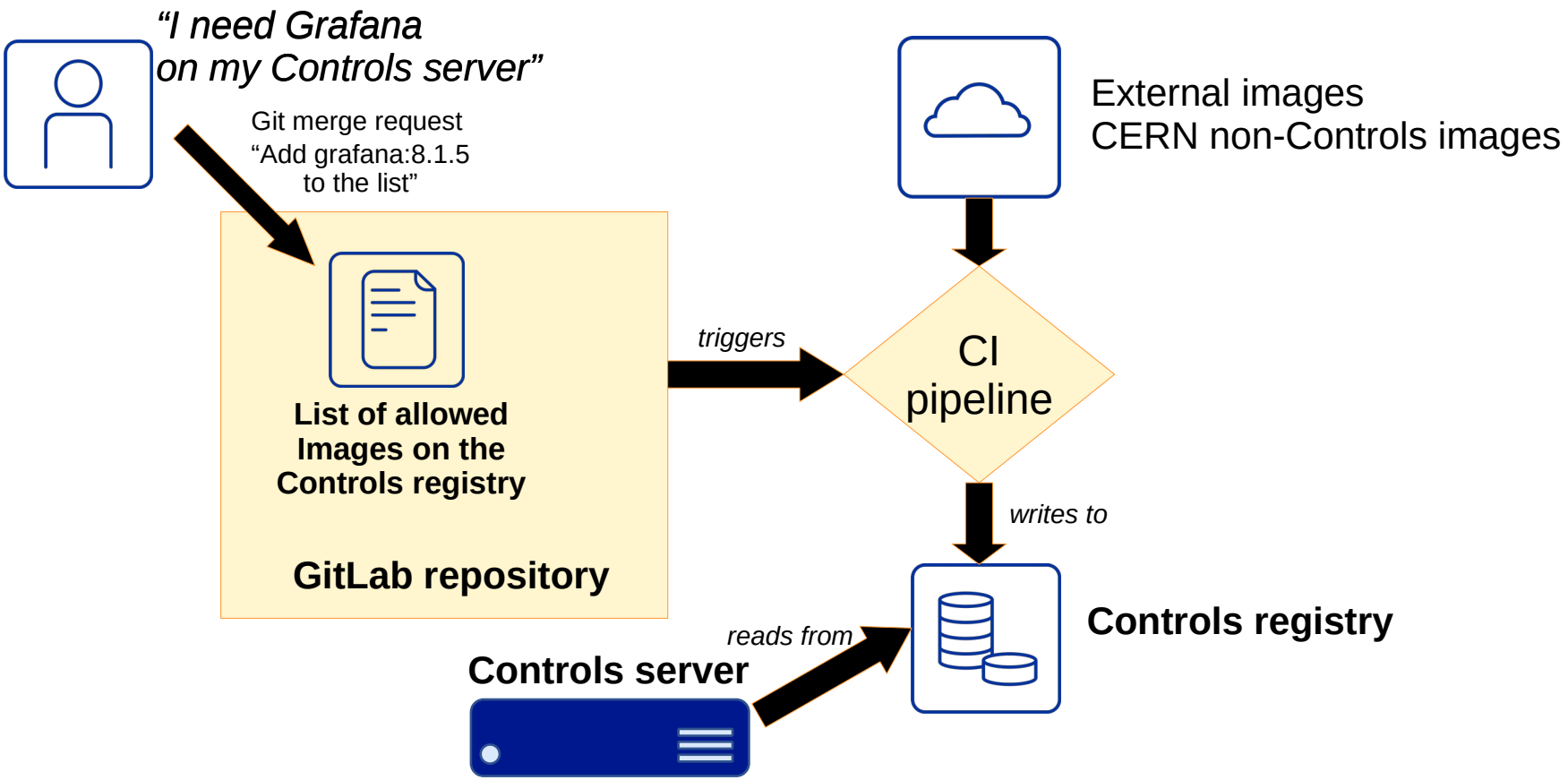
Plain containerization: overview

Purpose: offer the ability to run containers on servers and technical consoles

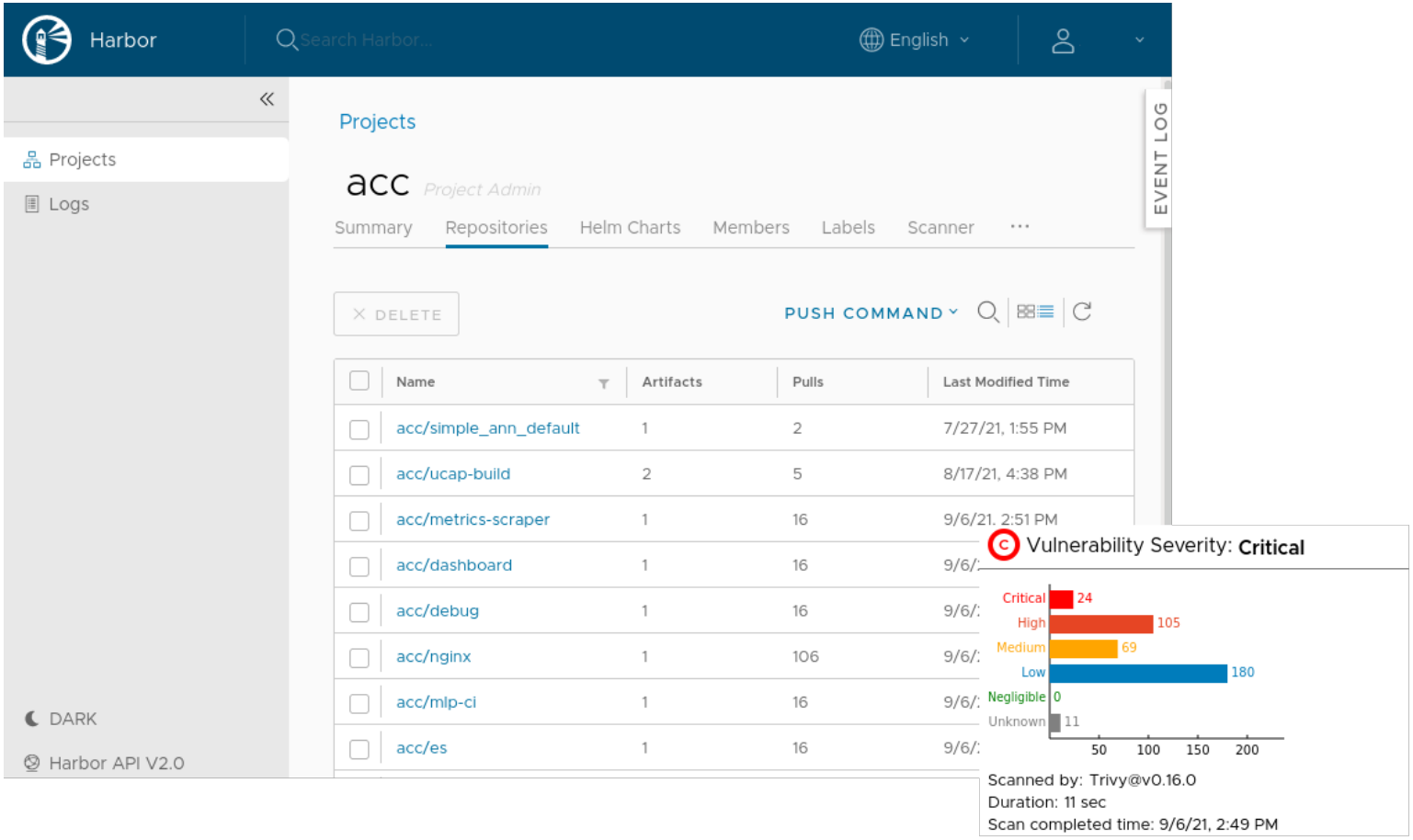
What we need:

- **Base images** that our developers can rely on *(details in extra slides)*
- **An image registry** to store container images
- **A container engine** to run containers on hosts

Plain containerization: registry approval mechanism



Plain containerization: registry

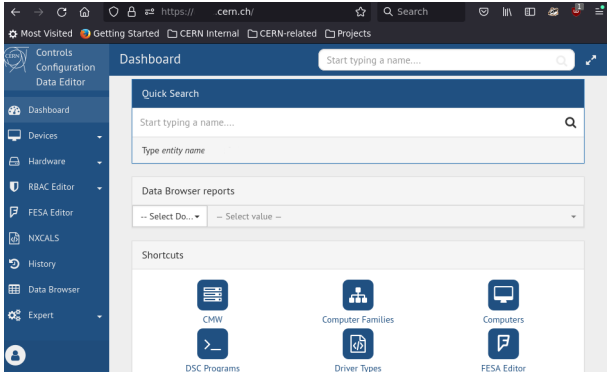


Plain containerization: engine

podman

Podman is our container engine of choice:

- Containers run as ordinary Controls services



1. Declare the containerized service

Controls server

2. Deploy and run in one command

Plain containerization: engine

podman

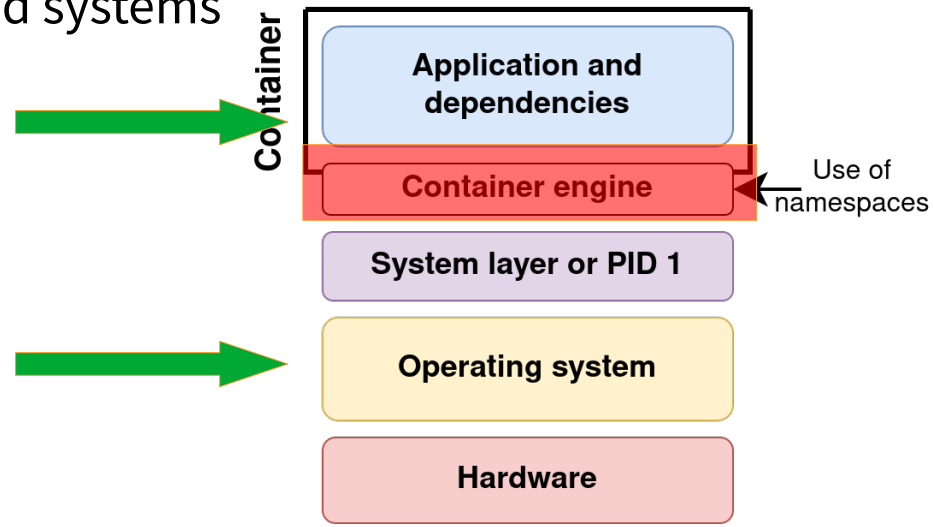
Podman is our container engine of choice:

- Containers run as ordinary Controls services
- Rootless for security and practicality
- Daemonless architecture
- Community-driven project

Plain containerization: pros and cons

Drawbacks

- **More complex debugging** and diagnostics on the sysadmin side
- Risk of discouraging investment in getting rid of **legacy software**
- Need to manage **obsolescence** of containerized systems



Plain containerization: pros and cons

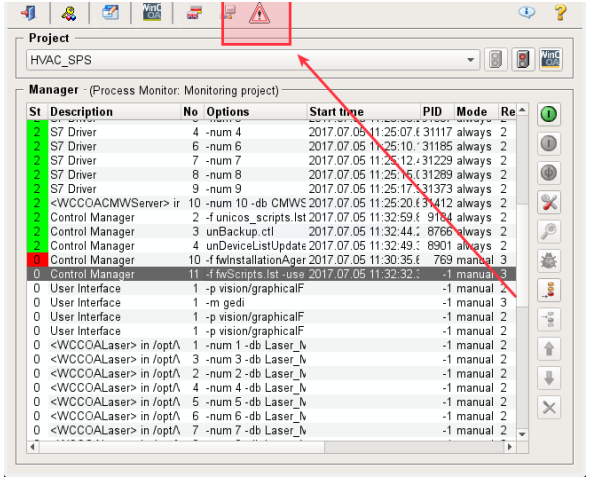
Advantages (1/2)

- **Unify building/packaging process with CI/CD templates**
  - Auto versioning and easy rollback
- **Relocatability, portability**
  - “Certified to run anywhere” when transferring to other labs

Plain containerization: pros and cons

Advantages (2/2)

- **Decoupling between the operating system and software**
  - Avoid constraints with new hardware
  - WinCC OA upgrades vs CentOS/RHEL upgrades
- **Encourage good development practices**
  - e.g. send logs to the tracing infrastructure
  - Implement Software Bill of Materials when available



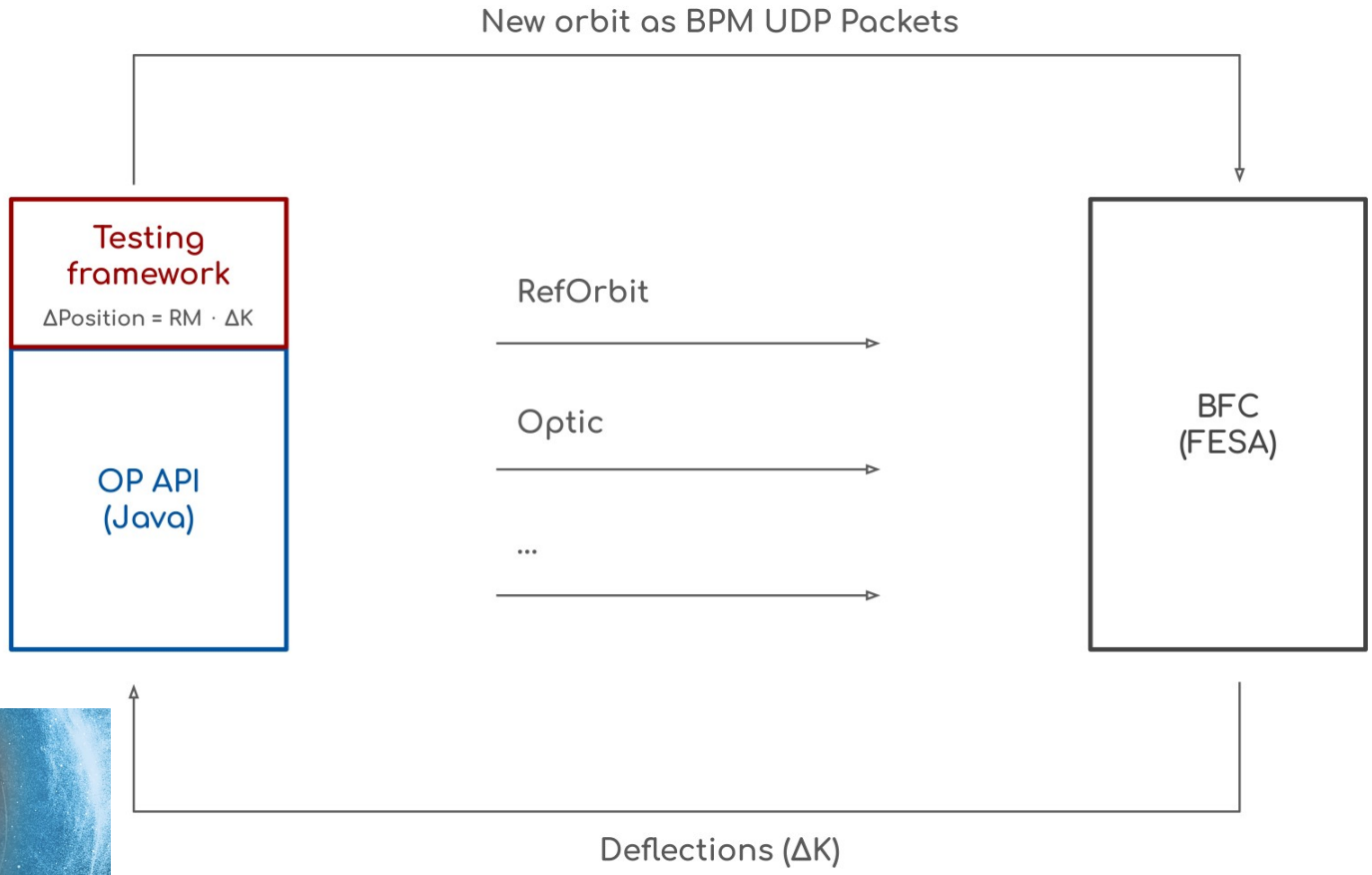
Summary: containerized and orchestrated applications

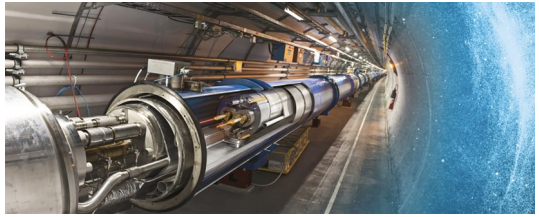
	Non-containerized	Containerized	Orchestrated
Unified packaging			
Portability			
Decoupling OS/software			
Ensure good dev practices			
Deploy with API			
Replication & scale-out			
Zero-downtime strategy			
Load-balancing	Manual	Manual	Integrated feature

Unless tied to specific hardware (e.g. timing cards), all applications can benefit from containerization.

A list of active use cases is in extra slides, now let's focus on one: the LHC Orbit Feedback.

Example use case: LHC Orbit Feedback testing





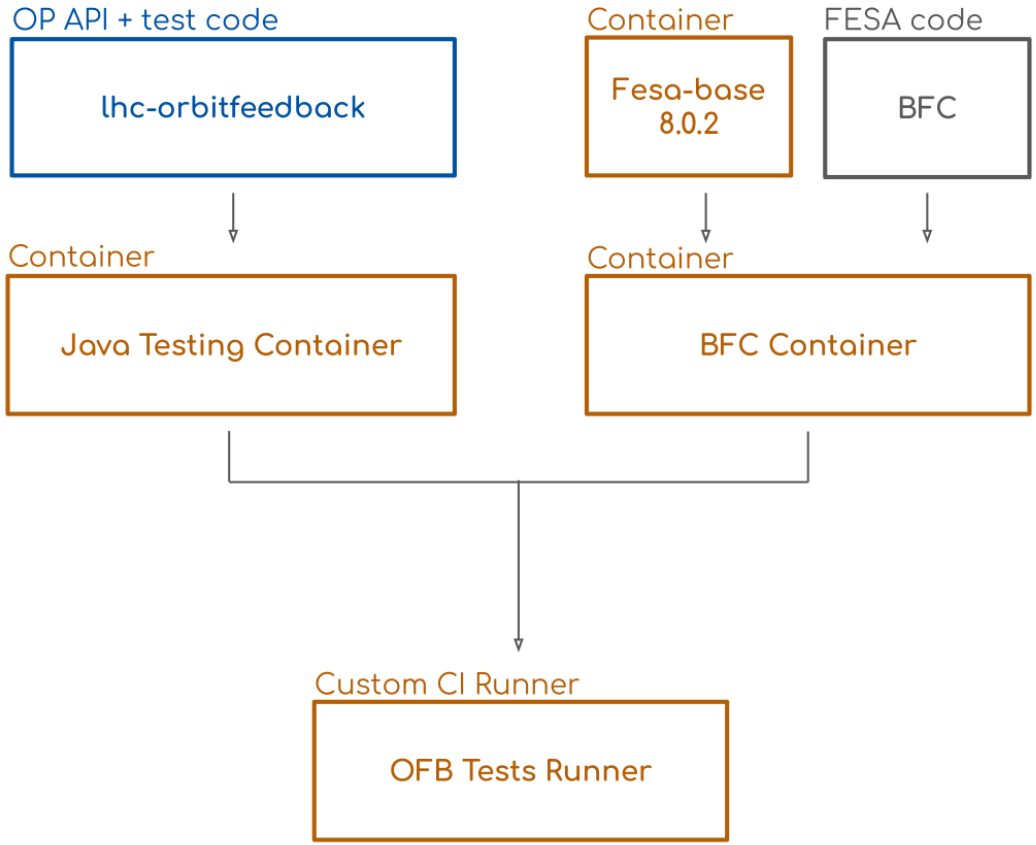
Curtesy of Andrea Calia

Example use case: LHC Orbit Feedback testing

- Goal: not having to rely on physical servers with timing cards
- Run on commit and daily
  - Based on Gitlab CI Pipelines
  - Currently ~100 unit tests
- Detailed history of issues
  - “This has happened before”
- Better development cycle
  - “It worked on my machine!”

More information and full presentation on the Evian Workshop 2021 page: https://indico.cern.ch/event/1077835/timetable/

Note: some of these components are not officially supported (e.g. Fesa-base)



Curtesy of Andrea Calia

Summary

Current containerization and orchestration come from twenty years of development

- From Linux kernel namespaces to Kubernetes

Containerization technologies are broadly used in industry

- Use cases: CI/CD pipelines, portability, scalability, redundancy,...

We're brought plain containerization while working on necessary Controls network & storage improvements

- First step for developers: take a look at the Containerisation FAQ at https://wikis.cern.ch/display/CONT/Containerisation+Guide+and+FAQ
- Help is available on the Mattermost channel be-dep #Containers
- The support line is acc-adm-support@cern.ch for now

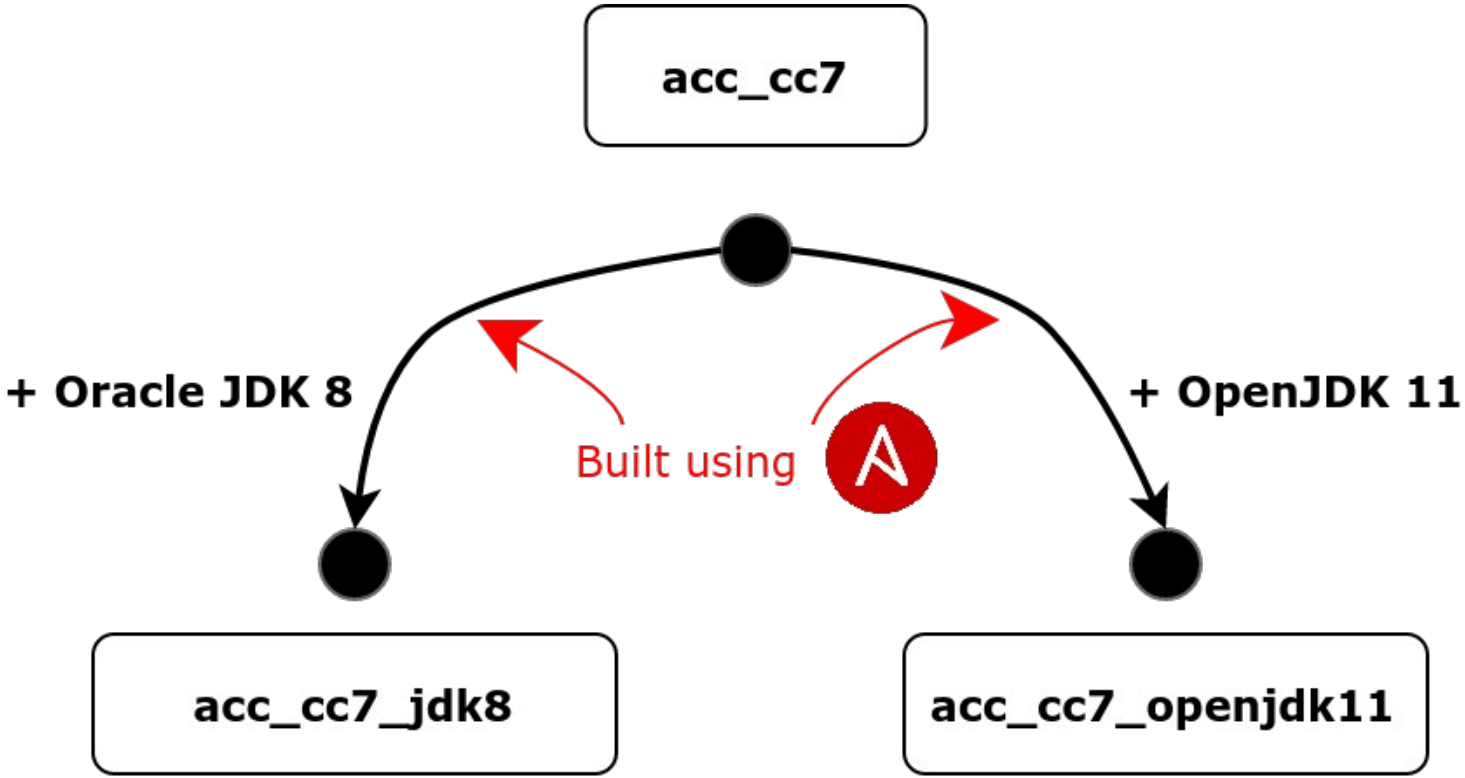
Thanks for your attention!

Extra slides

Kubernetes training

- Currently offered at CERN
  - Provided by the Linux Foundation
  - Multiple choices: from Application Development to Security
  - For developers: CKAD is recommended
    https://www.cncf.io/certification/ckad/
  - LMS https://lms.cern.ch/ekp/servlet/ekp?PX=N&TEACHREVIEW=N&CID=EKP000043693&TX=FORMAT1&LANGUAGE_TAG=en&DECORATEPAGE=N
    or look for “Kubernetes” in Catalog > Search

Plain containerization: base images



Container project: base images

```
dnf -y --installroot=${WORKDIR}/rootfs
--nodocs install ${PACKAGES}
```

Install base CentOS packages in subdirectory

Apply custom configuration files

Create an archive of the work directory

```
tar -C ${WORKDIR}/rootfs
-cf ${WORKDIR}/rootfs.tar .
```

Convert to OCI image

```
podman import -
acc_cc7:latest < rootfs.tar
```

Plain containerization: base images

GitLab CI

acc-co › docker › acc-base-images › **Pipelines**

All 57 | Finished | Branches | Tags

Run Pipeline | Clear Runner Caches | CI Lint

Filter pipelines

Status	Pipeline	Triggerer	Commit	Stages	
passed	#2098034 Scheduled latest		master 284d619f .gitlab-ci.yml: Add prune		00:21:24 2 hours ago
failed	#2095193 Scheduled latest		master 284d619f .gitlab-ci.yml: Add prune		00:17:49 1 day ago

Plain containerization: active use cases

- Currently:
  - Standard way to deploy software (e.g. SourceGraph, Nexus)
  - Unified deployment and operational environments (e.g. LHC Injector Chain Timing Sequence Manager)
  - Replicate production in local environments (e.g. LHC Orbit Feedback)
- Foreseen:
  - Decoupling software upgrades from operating system upgrades (e.g. WinCC OA 3.16 on EL8 consoles)
  - Unless tied to specific hardware (e.g. timing cards), all applications can benefit from containerization

home.cern