Kubernetes and WLCG

Ricardo Rocha
CERN IT-CM-RPS

Kubernetes

Google Cloud Platform

OPENSHIFT

Rancher
Kubernetes Engine

MAGNUM
an OpenStack Community Project

kubeadm

Common declarative API for basic infrastructure : compute, storage, networking

Healthy ecosystem of tools offering extended functionality

Cloud Native Landscape

v20180425

See the interactive landscape at l.cncf.io

Greyed logos are not open source

App Definition and Development

Database and Data Warehouse

Vitess (CNCF Incubating)

Streaming

NATS (CNCF Incubating)

Source Code Management

Application Definition and Image Build

Continuous Integration / Continuous Delivery (CI/CD)

Orchestration & Management

Scheduling & Orchestration

kubernetes (CNCF Graduated)

Coordination & Service Discovery

CoreDNS (CNCF Incubating)

Service Management

gRPC (CNCF Incubating) · envoy (CNCF Incubating) · LINKERD (CNCF Incubating) · Open Policy Agent (CNCF Sandbox)

Runtime

Cloud-Native Storage

ROOK (CNCF Sandbox)

Container Runtime

containerd (CNCF Incubating) · rkt (CNCF Incubating)

Cloud-Native Network

CNI (CNCF Incubating)

Provisioning

Host Management / Tooling

Infrastructure Automation

Container Registries

Secure Images

Notary (CNCF Incubating) · TUF (CNCF Incubating)

Key Management

spiffe (CNCF Sandbox) · SPIRE (CNCF Sandbox)

Cloud

Public

Private

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

l.cncf.io

CLOUD NATIVE Landscape

CLOUD NATIVE COMPUTING FOUNDATION

Redpoint Amplify

Platforms

Certified Kubernetes - Distribution

Certified Kubernetes - Hosted

Certified Kubernetes - Installer

Non-Certified Kubernetes

PaaS/Container Service

Kubernetes Certified Service Provider

Observability & Analysis

Monitoring

Prometheus (CNCF Incubating)

Logging

fluentd (CNCF Incubating)

Tracing

Jaeger (CNCF Incubating) · OpenTracing (CNCF Incubating)

Kubernetes Training Partner

Serverless

Special

Graduated

Kubernetes
Orchestration

Prometheus
Monitoring

Envoy
Network Proxy

CoreDNS
Service Discovery

containerd
Container Runtime

Fluentd
Logging

Jaeger
Distributed Tracing

Vitess
Storage

TUF
Software Update Spec

Helm
Package Management

Benefits

Simplified Infrastructure

Monitoring, Lifecycle, Alarms

Simplified Deployment

Uniform API, Replication, Load Balancing, Configuration

Periodic Load Spikes

International Conferences, Reprocessing Campaigns

Architecture

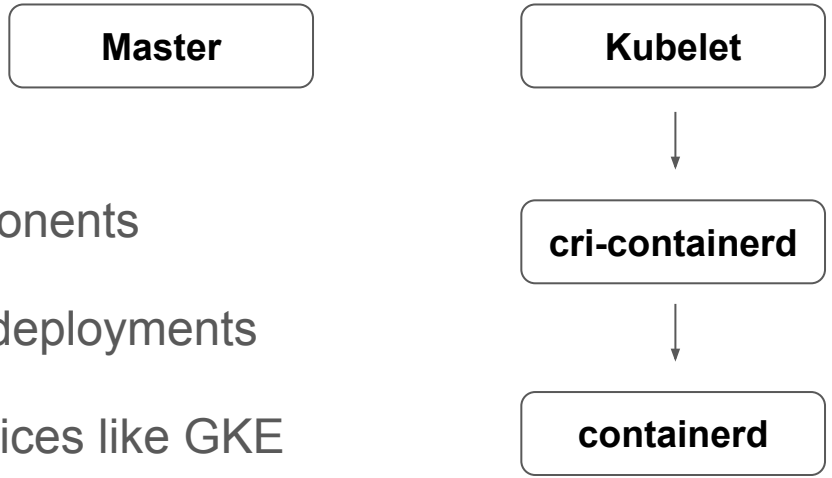
Our current stack at CERN

Integration effort focusing on these components

Pretty much standard with most popular deployments

- Including public cloud managed services like GKE
- And on-premise deployment tools like Rancher / k3s

There is value in reusing a common stack



Architecture

Our current stack at CERN

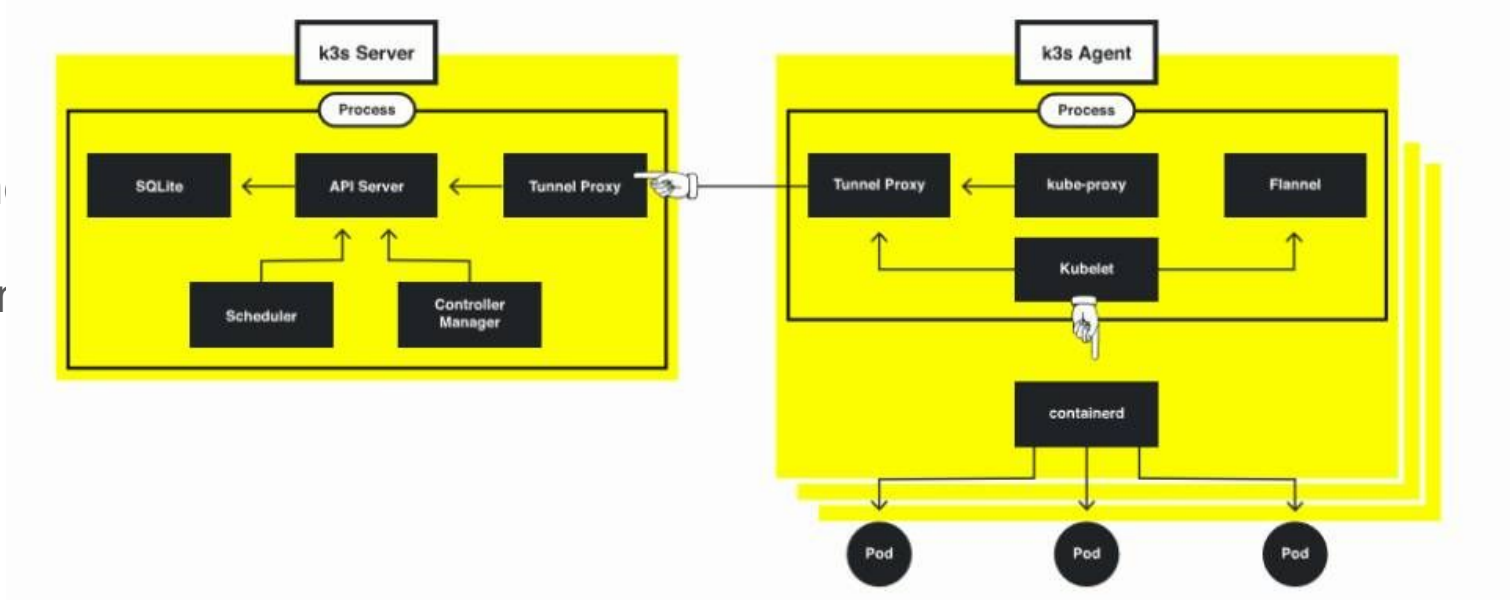
Integration effort focusing on these components

Pretty

In

Ar

There



Master

Kubelet

cri-containerd

Architecture

Master

Kubelet

Our current stack at CERN

Integratio

ainerd

Pretty mu

Inclu

nerd

And

There is

containerd node images

containerd is an important building block and the core runtime component of Docker. There are two OS images using containerd as the main container runtime directly integrated with Kubernetes: `cos_containerd` and `ubuntu_containerd`.

For debugging or troubleshooting on the node, you can interact with containerd using the portable command-line tool built for Kubernetes container runtimes: `crictl`. `crictl` supports common functionalities to view containers and images, read logs, and execute commands in the containers. Refer to the crictl user guide for the complete set of supported features and usage information.

Warning: Docker cannot view or access containers or images managed by Kubernetes. Your applications should not interact with Docker directly. For general troubleshooting or debugging, use **`crictl`** instead.

containerd on Container-Optimized OS (`cos_containerd`)

`cos_containerd` is a variant of the Container-Optimized OS image with containerd as the container runtime directly integrated with Kubernetes.

`cos_containerd` requires **Kubernetes version 1.14.3** or higher.

Architecture

Master

Kubelet

Our cur

Integrat

Pretty m

Inc

And

There is

erd

rd

CRI Product Landscape

OPEN SOURCE LEADERSHIP SUMMIT

- **GKE**: containerd-based K8s clusters in **beta**/selectable; default is **Docker**
- **IBM Cloud**: containerd-based clusters in **production** (all versions)
- **Azure**: OSS acs-engine includes containerd; AKS uses **Docker**; (but CRI-O for OpenShift deployment)
- **Amazon**: EKS uses **Docker by default**; Firecracker using **containerd**
- **CloudFoundry**: Eirini project (CF on K8s) using **containerd**; pre-Eirini (non-K8s-based) used **runc**, now **containerd**
- **OpenShift**: prior versions used RHEL-Docker (1.12/13); **cri-o** GA in OpenShift during 2018
- **ICP**: IBM private cloud offering **defaults to Docker**; **containerd** in tech preview

@estesp

THE LINUX FOUNDATION

What we need

1. Recipes to deploy Kubernetes at sites
2. Easy way to deploy WLCG specific components
3. Provide efficient access to software

1. Deploy Kubernetes at sites

OpenStack sites?

- Magnum works well for CERN

Rancher seems to be a popular option

- Particularly k3s gained some traction (simplified deployment, no etcd)

Highly dependent if sites already have a layer to provision instances

As mentioned earlier, some stack choices can have an impact

- containerd if we want to rely on the remote snapshotter

2. Deploy common components

Helm is the package manager for Kubernetes

A **Chart** manages the deployment and config

of an application

Reusable, shareable units

Includes all required manifests, plus any required libraries for lifecycle

Separate **values** definitions for instance configuration

Repositories keep track of available versions: eg. https://registry.cern.ch/chartrepo

2. Deploy common components

Which components are shared?

Helm charts already available for

- csi-cvmfs: https://github.com/cernops/cvmfs-csi/pull/14
- condor, xcache, …: https://portal.slateci.io/applications

Others required?

Some discussion already in the past on lifecycle management

- Flux, ArgoCD, ...

3. Efficient access to software

Similar goals to what was there with the docker cvmfs graphdriver

- Efficient lazy pull for faster container startup

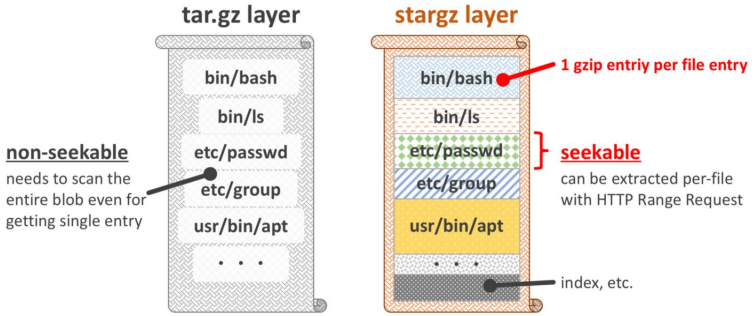
Most of the plumbing for containerd integration already done for (e)stargz

- https://github.com/containerd/stargz-snapshotter
- We have experimental clusters with this configuration at CERN
- Standards compliant, no registry changes, fits well our local use case
- `ctr-remote images optimize` required today before pushing to registry

3. Efficient access to software

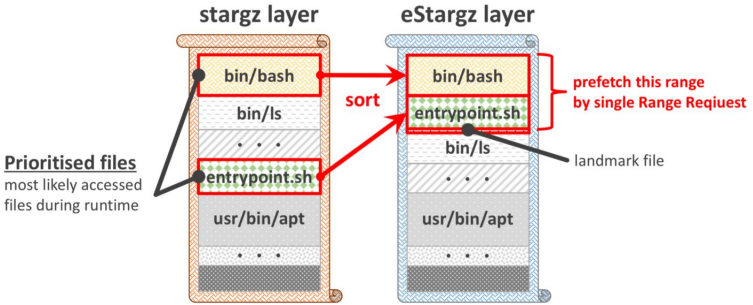
stargz (seekable stargz), a tar of tarred files

- Images compatible with non stargz runtimes
- HTTP range support in OCI registries



estargz (enhanced stargz), prefetch expected requests

- Profile required files from a sample workload
- Add a landmark file to the image with those



https://medium.com/nttlabs/startup-containers-in-lightning-speed-with-lazy-image-distribution-on-containerd-243d94522361

3. Efficient access to software

CERN Kubernetes clusters can already rely on containerd remote snapshotters

AFAIU the CVMFS solution is using the same concepts

- Should be straightforward to replicate the deployment we have

Questions?