Building Cloud Native Distributions

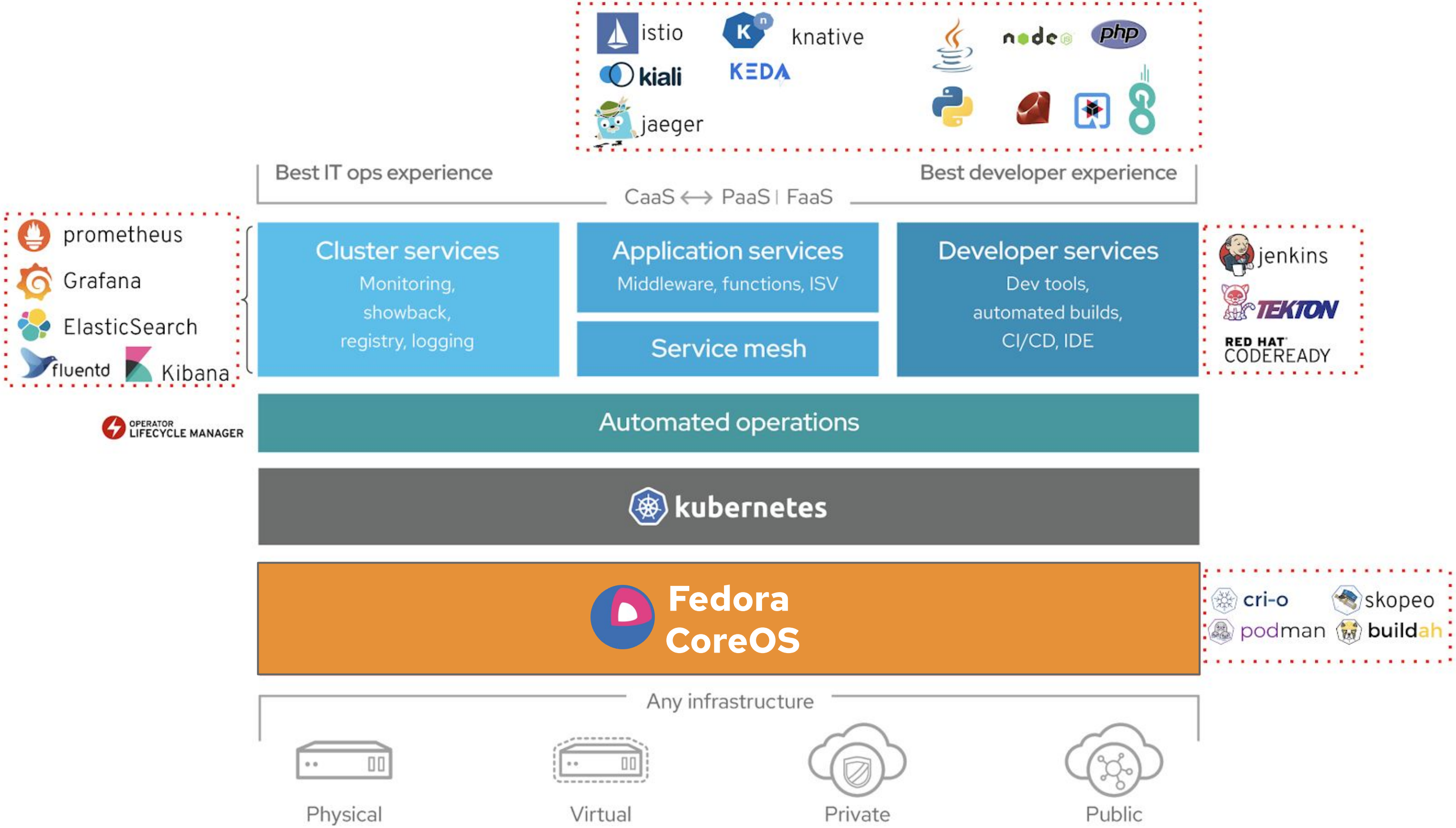
the Cloud Native Way

… or how we build OKD using Tekton

Alessandro Di Stefano
Christian Glombek
Sherine Khoury
Timothee Ravier

@aleskandro[@hachyderm.io]
@lorbus42 - @lorbus@mastodon.social
@srinerine[@hachyderm.io]
@siosm[@floss.social]

OKD - A(nother) K8s Distribution?



istio
knative
kiali
KEDA
jaeger

Best IT ops experience
CaaS ⟷ PaaS | FaaS
Best developer experience

prometheus
Grafana
ElasticSearch
fluentd
Kibana

Cluster services
Monitoring,
showback,
registry, logging

Application services
Middleware, functions, ISV

Service mesh

Developer services
Dev tools,
automated builds,
CI/CD, IDE

jenkins
TEKTON
RED HAT CODEREADY

OPERATOR LIFECYCLE MANAGER

Automated operations

kubernetes

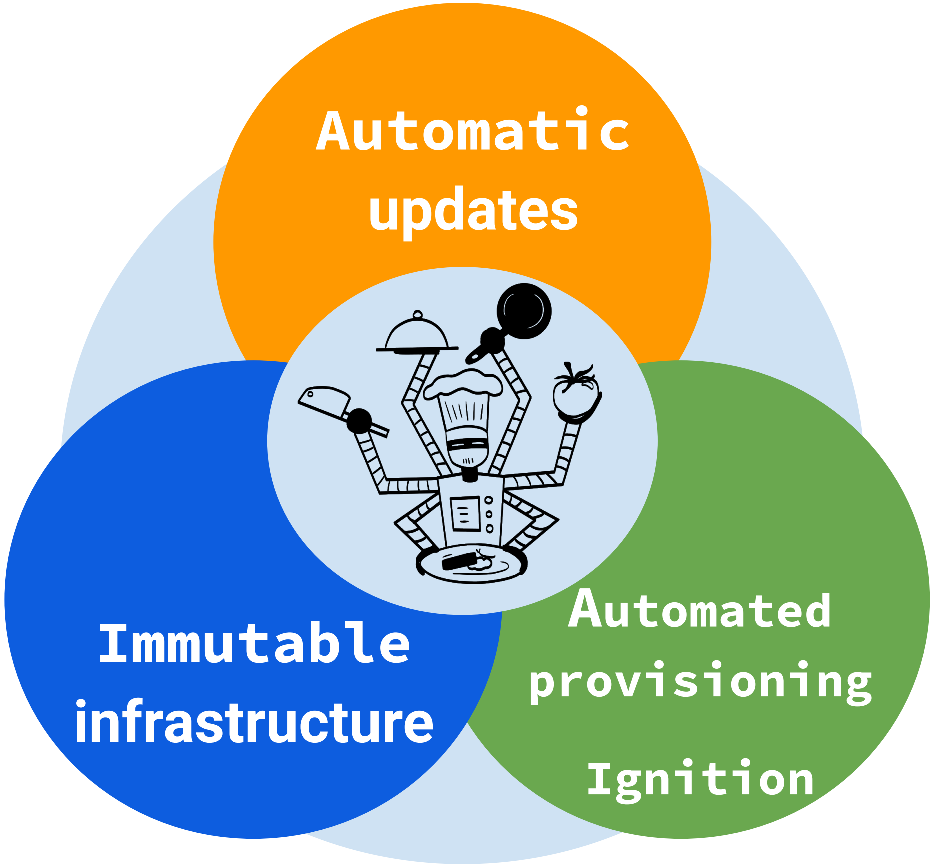
Fedora CoreOS

cri-o
skopeo
podman
buildah

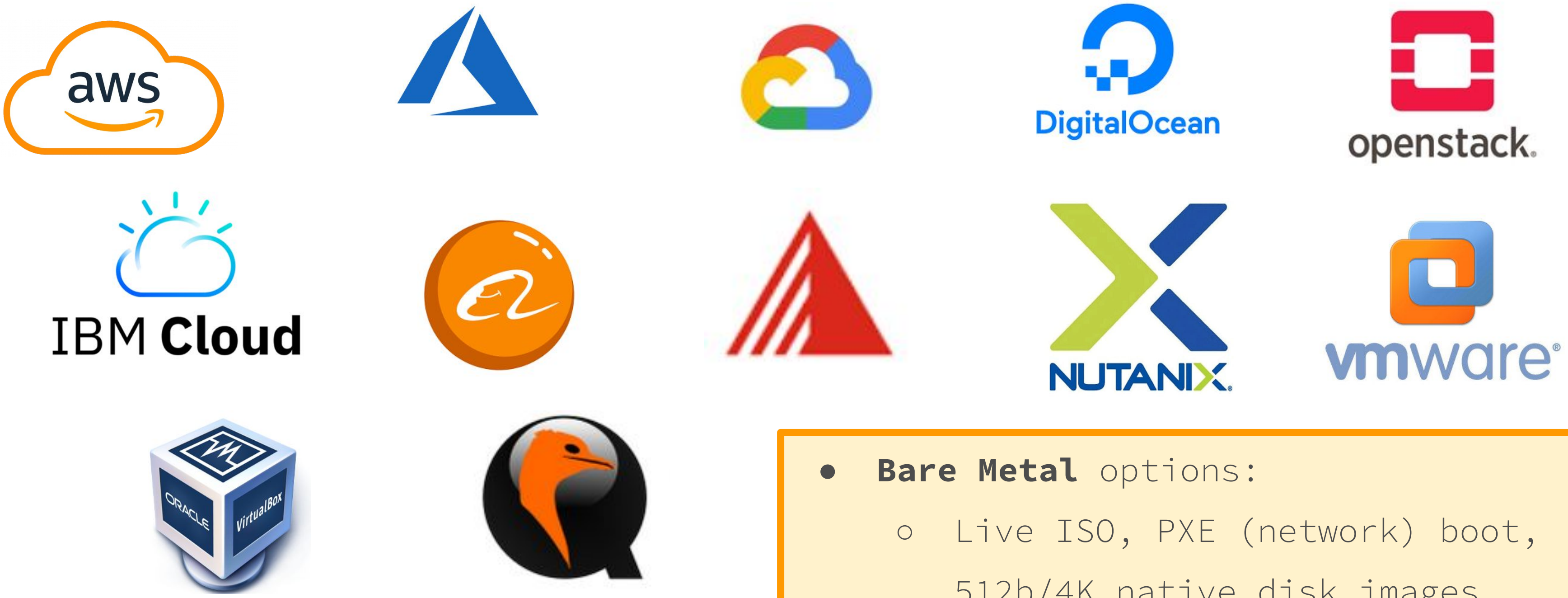
Any infrastructure

Physical
Virtual
Private
Public

Fedora CoreOS - An official Fedora edition

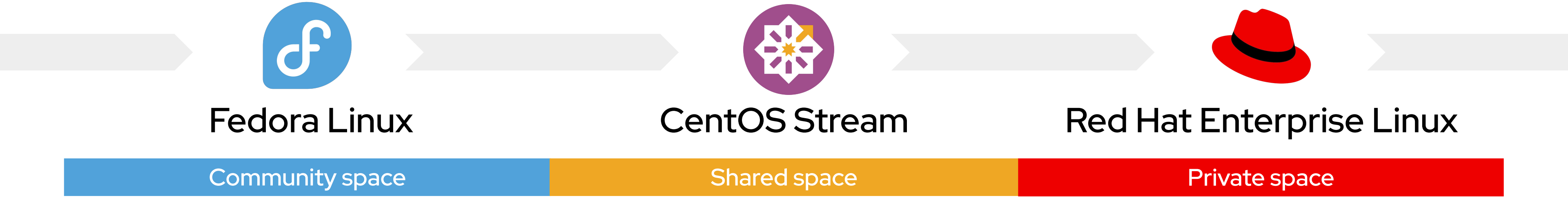


Platforms and architectures



- **Bare Metal** options:
  - Live ISO, PXE (network) boot, 512b/4K native disk images
- Support for **x86_64**, **aarch64**, **s390x** and **ppc64le(soon)**

Enterprise Linux ecosystem



Fedora Linux

CentOS Stream

Red Hat Enterprise Linux

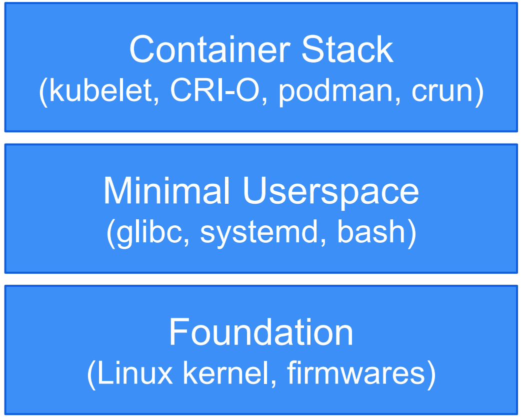
Community space

Shared space

Private space

CoreOS you said...

- **Flavors:**
  - Fedora CoreOS (FCOS)
  - RHEL CoreOS (RHCOS)
  - CentOS Stream CoreOS (SCOS)

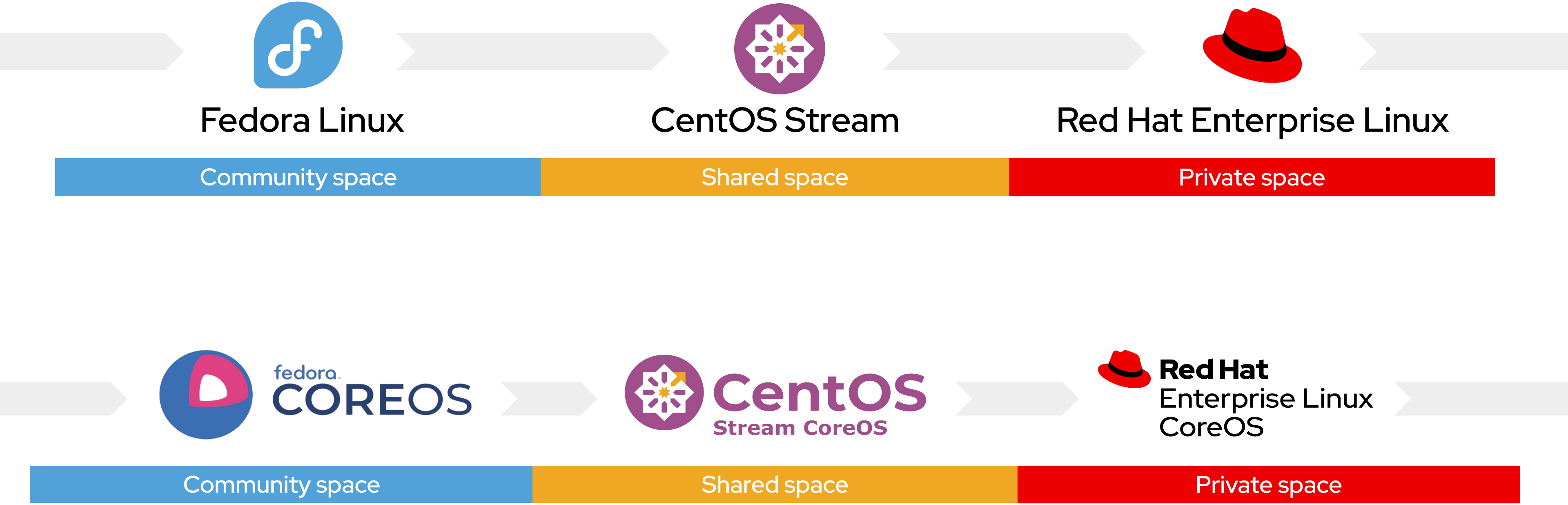


Container Stack
(kubelet, CRI-O, podman, crun)

Minimal Userspace
(glibc, systemd, bash)

Foundation
(Linux kernel, firmwares)

But what is CentOS Stream CoreOS (SCOS)?



Fedora Linux → CentOS Stream → Red Hat Enterprise Linux

Community space | Shared space | Private space

fedora COREOS → CentOS Stream CoreOS → Red Hat Enterprise Linux CoreOS

Community space | Shared space | Private space

So why build OKD on SCOS? (CentOS Stream CoreOS)

Fedora is 2-3y ahead of RHEL

+

FCOS is 2-3y ahead of RHEL CoreOS

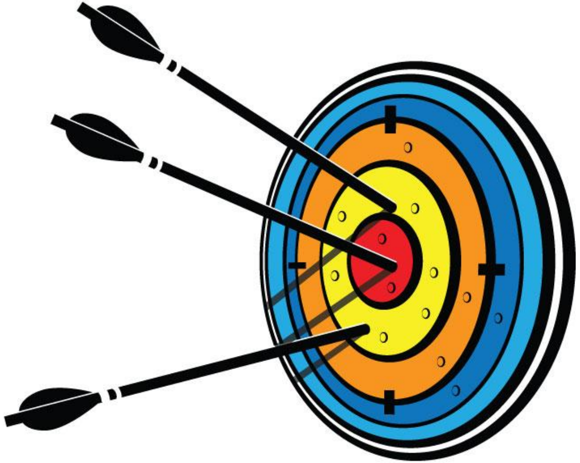
→

OKD users experienced issues that were less priority for Red Hat Engineers

Focused on OpenShift and RHEL CoreOS

So why build OKD on SCOS? (CentOS Stream CoreOS)

- More stable for OKD users
- Closer to OpenShift (on RHEL CoreOS)
- Same OKD community as for OKD on FCOS



Enters OKD Streams

OKD Stream /əʊ keɪ diː striːm/ **(noun):**

refers to a build, test, and release pipeline for ANY configuration of OKD.

Example: Build OKD on SCOS from the grounds up to improve RHEL 9 readiness signal for Red Hat OpenShift.

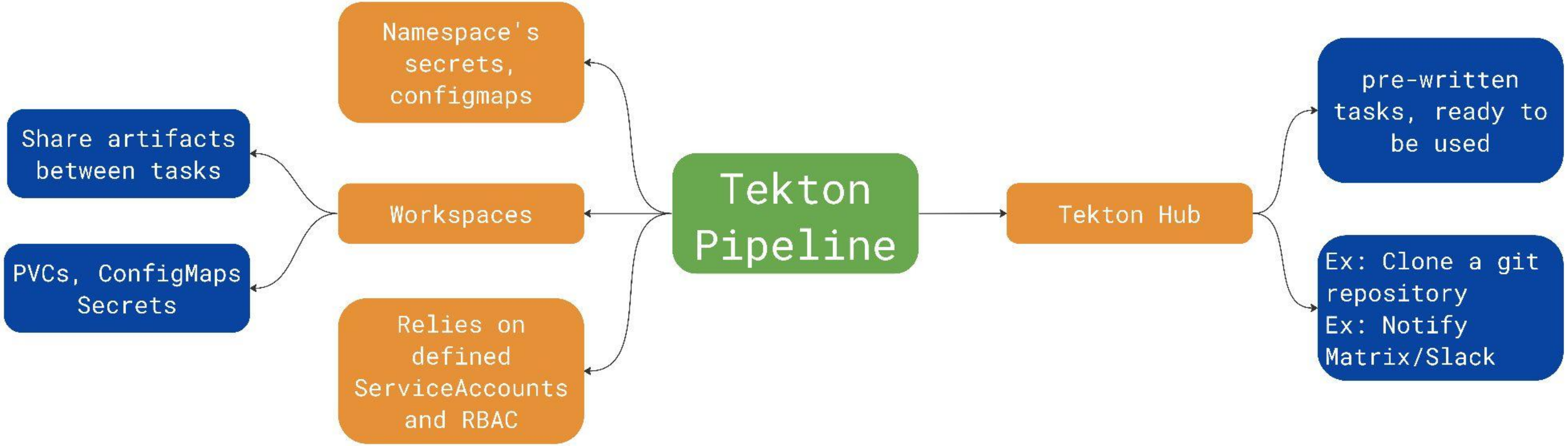
The power of Tekton

Prow Build

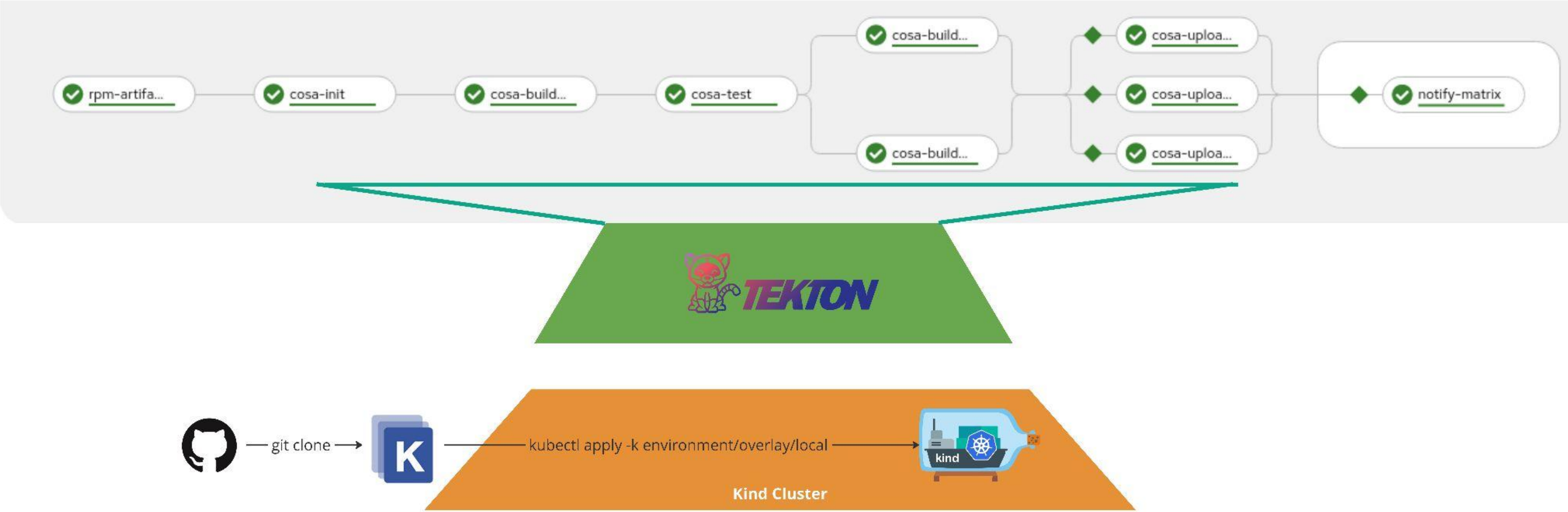
- Used for OKD / FCOS
- internal to Red Hat
- Inaccessible to the community
- Suited for development (PRs, history…)

Tekton Build

- Used for OKD / SCOS
- Within anyone’s reach
- Even with KinD

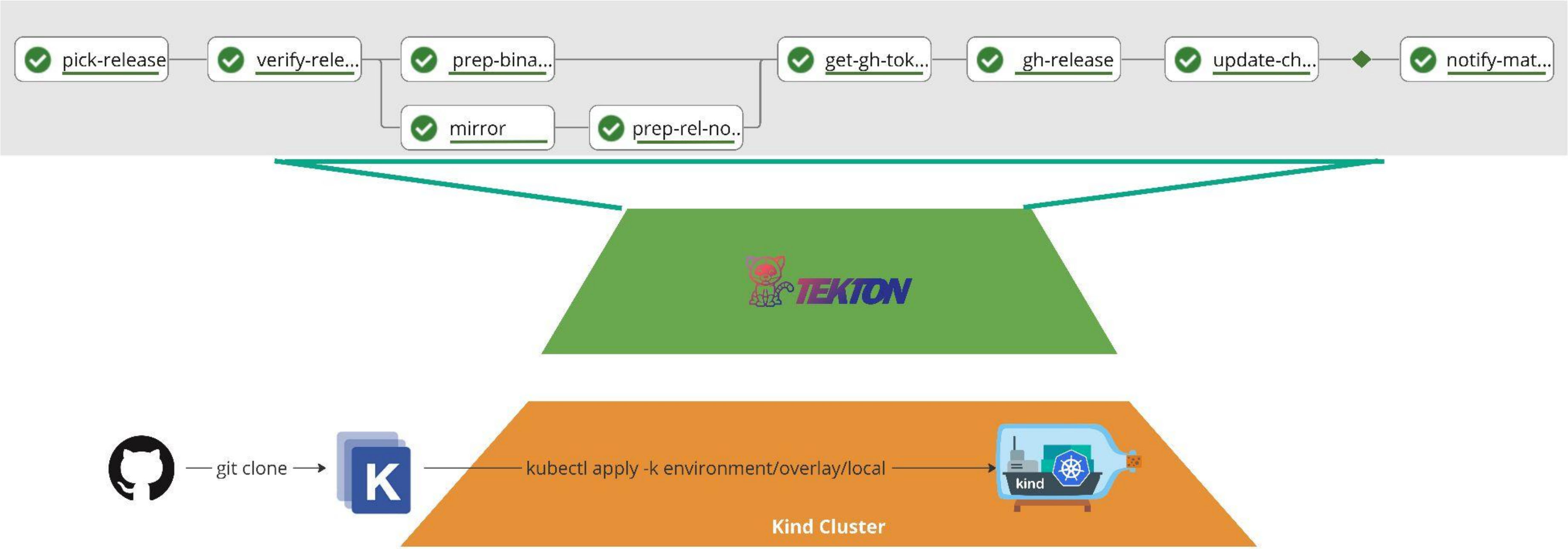

Namespace's secrets, configmaps
Share artifacts between tasks
Workspaces
PVCs, ConfigMaps Secrets
Relies on defined ServiceAccounts and RBAC
Tekton Pipeline
Tekton Hub
pre-written tasks, ready to be used
Ex: Clone a git repository
Ex: Notify Matrix/Slack

What’s cookin’ - Building and publishing SCOS



https://github.com/okd-project/okd-coreos-pipeline/

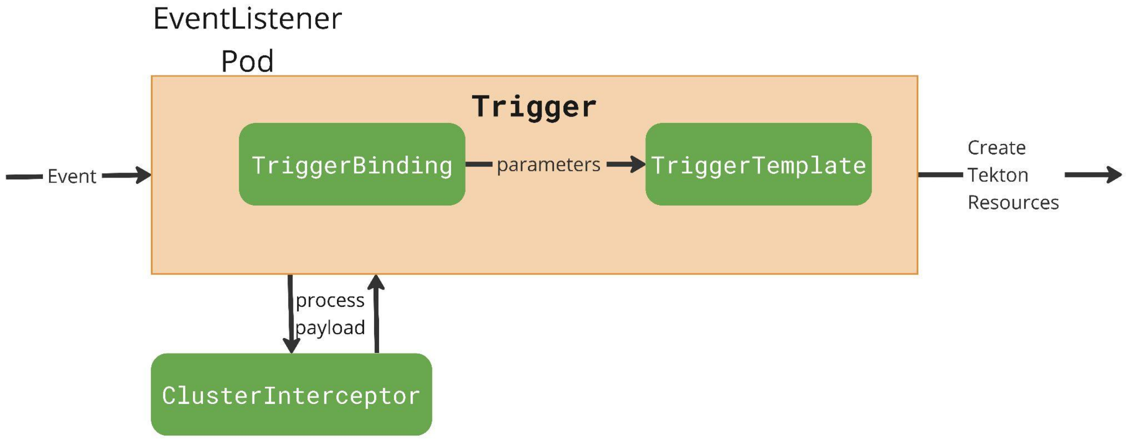
What’s cookin’ - Creating an OKD Release



https://github.com/okd-project/okd-release-pipeline/

What’s starting the pipelines?

- triggers.tekton.dev
- EventListener
- Trigger
  - TriggerBinding
  - TriggerTemplate
  - Interceptor
- Git, periodics, …

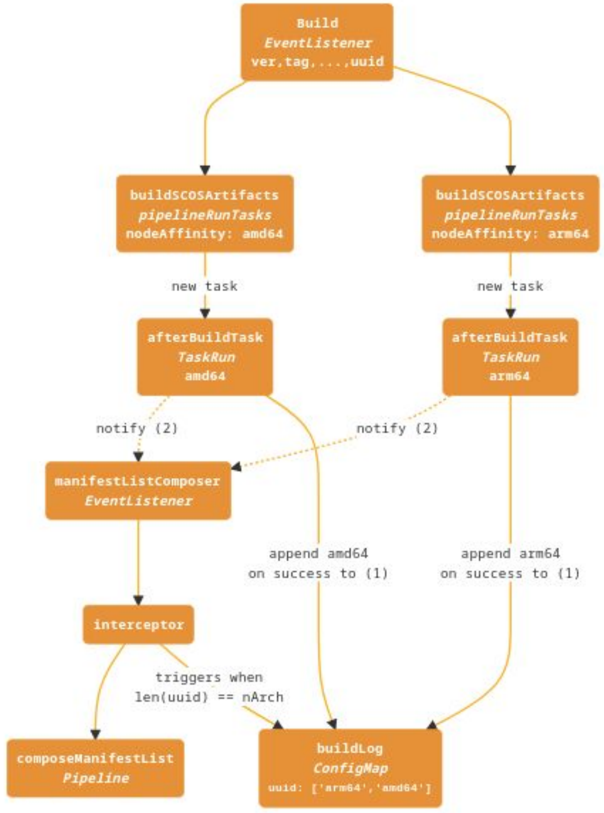


Next steps: multi-arch SCOS (via triggers)

- No arch-specific bindings in the Tekton resources
- The SCOS manifests sources can deliver arch-specific content
- Separate cloud boot images
- Unique manifest-list container-native image

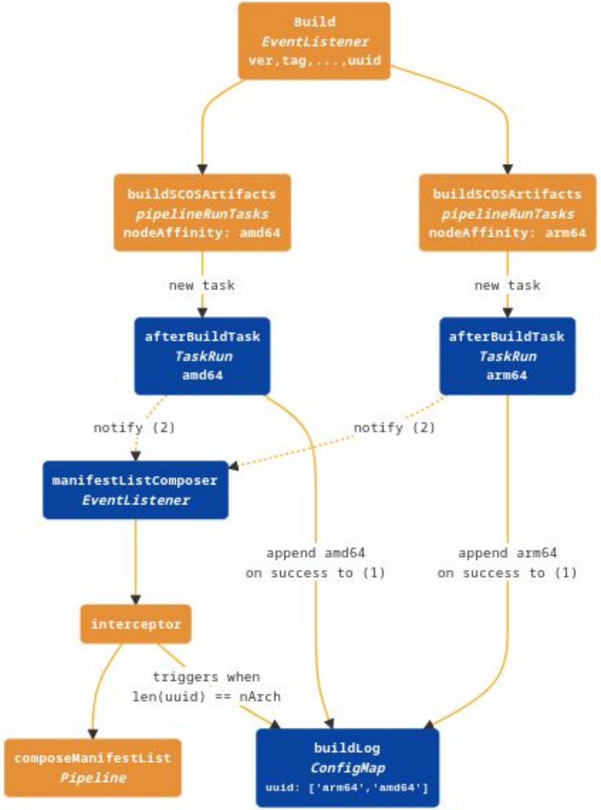
Next steps: multi-arch SCOS (via triggers)

- Multiple single-arch pipelineRuns
- Separate cloud boot images
- single-arch container images
- Manifest-list?



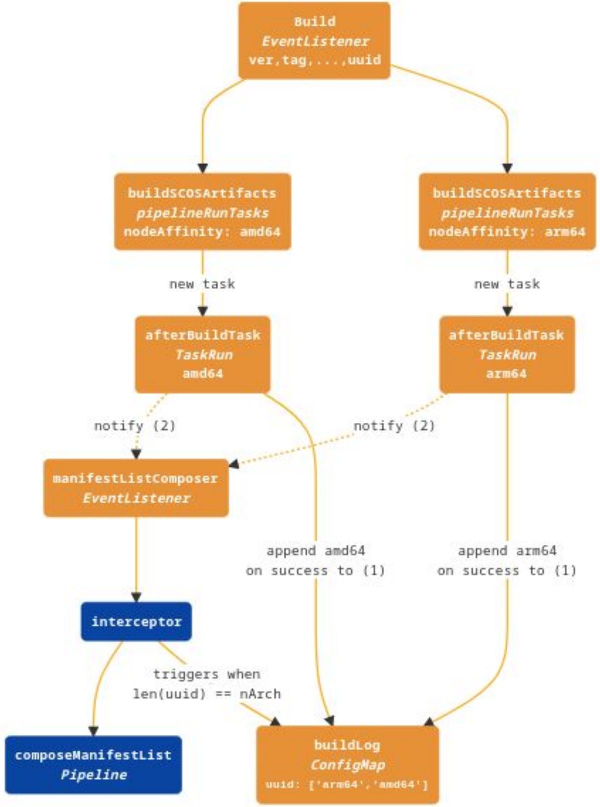
Next steps: multi-arch SCOS (via triggers)

- Notify the successful build event to the multi-arch EventListener
- non complete single-arch builds are gated by the interceptor
- The composeManifestList pipeline is triggered when all the single-arch artifacts are available

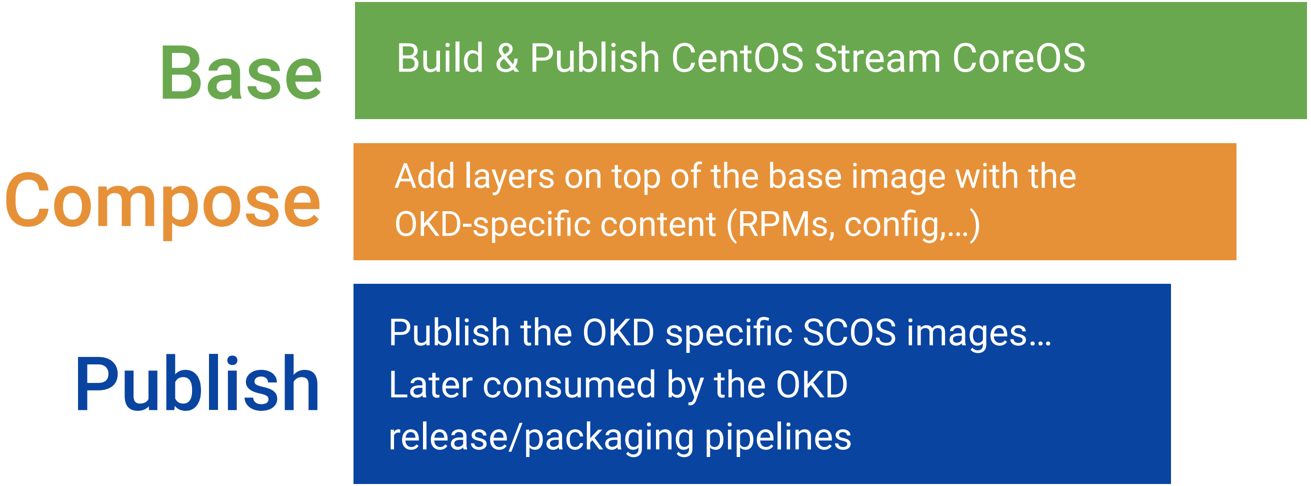


Next steps: multi-arch SCOS (via triggers)

- Notify the successful build event to the multi-arch EventListener
- non complete single-arch builds are gated by the interceptor
- The composeManifestList pipeline is triggered when all the single-arch artifacts are available



Next steps: Fedora CoreOS layering



Base

Build & Publish CentOS Stream CoreOS

Compose

Add layers on top of the base image with the OKD-specific content (RPMs, config,…)

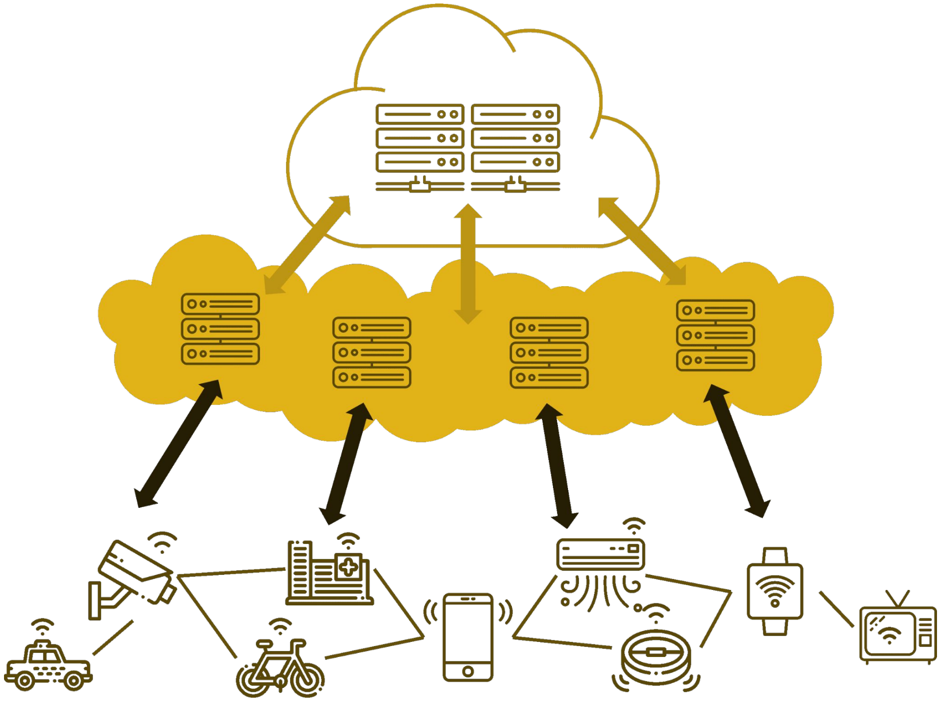
Publish

Publish the OKD specific SCOS images…
Later consumed by the OKD release/packaging pipelines

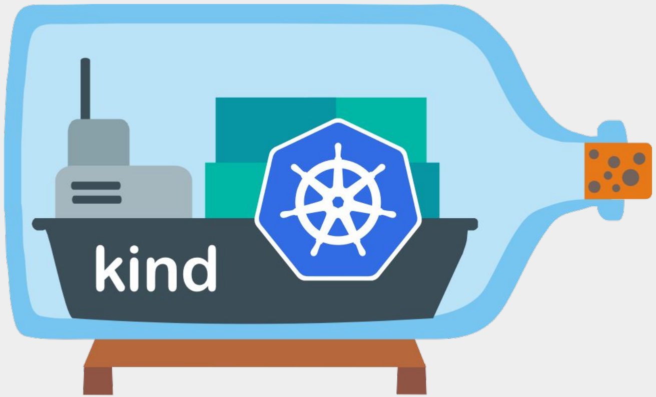
```
FROM quay.io/okd/centos-stream-coreos-9:4.13

RUN rpm-ostree install cri-o cri-tools conmon-rs && rpm-ostree cleanup -m && \

    ostree container commit #!!TRUNCATED!!
```

Next steps: layering

- Thin-clients and SMEs
- multimedia, automotive
- Edge computing
- IoT



Where can I use it?
- on Any Kubernetes cluster



For local experimentations

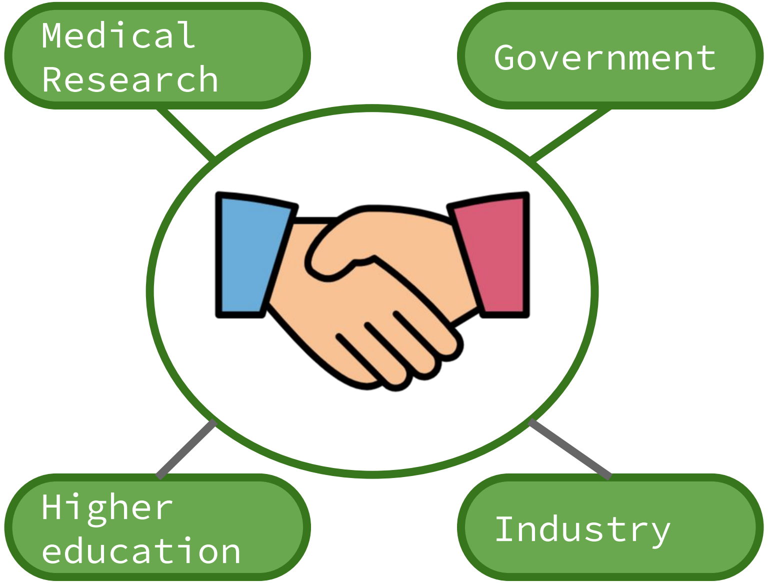
For official releases of OKD / SCOS

MOC Alliance

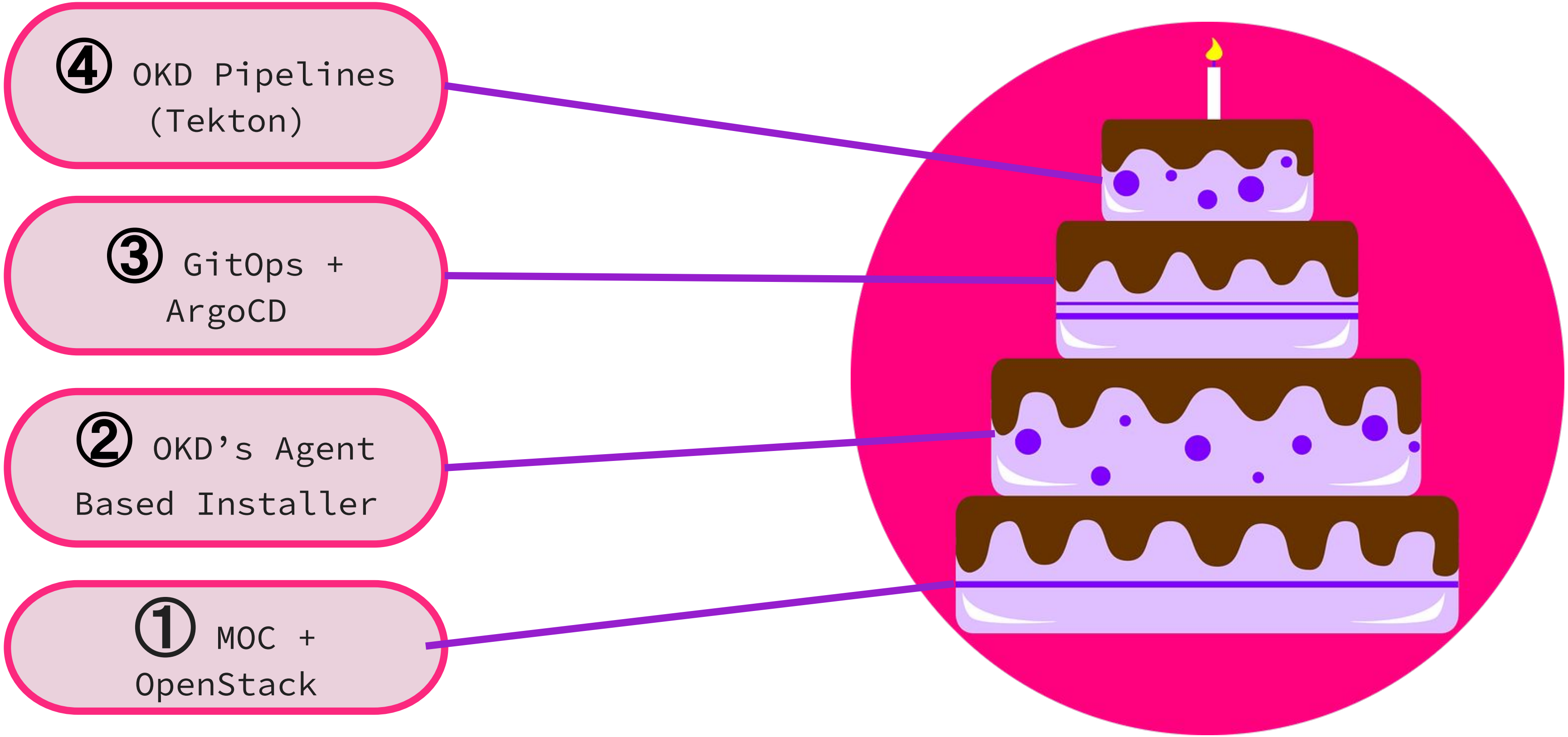
- A research-focused cloud

- Provides:
  - Bare metal machines - Donations
  - Controllable via OpenStack
  - Plus ESI - Elastic-Secure-Infrastructure

- Runs our OKD clusters
  - Long-term -> CD
  - Ephemeral -> Tests



Setting up MOC cluster with Argo and Tekton



④ OKD Pipelines (Tekton)

③ GitOps + ArgoCD

② OKD's Agent Based Installer

① MOC + OpenStack

What’s in it for you?

- **For development:** A more stable version of OKD
  - Closer to OpenShift release
- **For staging:** A preview version of the next OKD release
  - Every 3 weeks
- **For labs:**
  - Experiment with switching some operators of the payload
  - Fine tune the OS used by your cluster

Alessandro Di Stefano
Christian Glombek
Sherine Khoury
Timothee Ravier

@aleskandro[@hachyderm.io]
@lorbus42 - @lorbus@mastodon.social
@srinerine[@hachyderm.io]
@siosm[@floss.social]