Storage Landscape for Containerized Stateful Applications

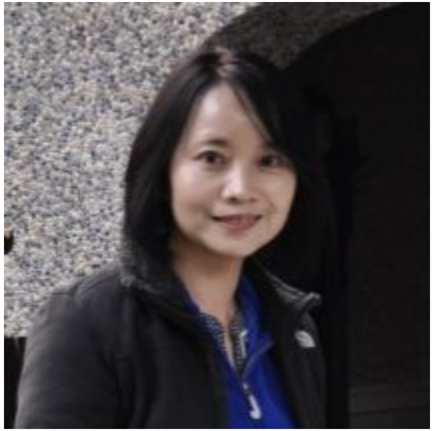
CLOUD NATIVE
COMPUTING FOUNDATION

Speakers

CLOUD NATIVE
COMPUTING FOUNDATION

Speakers

Alex Chircop
Founder and CTO,
StorageOS



Xing Yang
Lead Architect,
OpenSDS

Luis Pabón
MTS Engineer,
Portworx

Storage Landscape White Paper

CLOUD NATIVE
COMPUTING FOUNDATION

Why is storage important?

- There's no such thing as a stateless architecture, **applications store state somewhere.**

- Cloud native is about supporting patterns such as **portability**. Containers on their own do not enable portability.

- **Interoperating** with storage increases cloud native's relevance and leads to better applications.

Storage Landscape White Paper Outline

http://bit.ly/cncf-storage-whitepaper

- Definition of the attributes of a storage system
- Definition of the layers in a storage solution with a focus on terminology and how they impact the attributes
- Definition of the data access interfaces in terms of volumes and application APIs
- Definition of the management interfaces

Inside a storage solution …

Storage solutions have …

- a variety of **interfaces** suitable for different use cases
- **multiple layers** of functionality

- The different components of an overall storage solution impact the **attributes** of a storage system:

- **Availability**
- **Scalability**
- **Performance**
- **Consistency**
- **Durability**

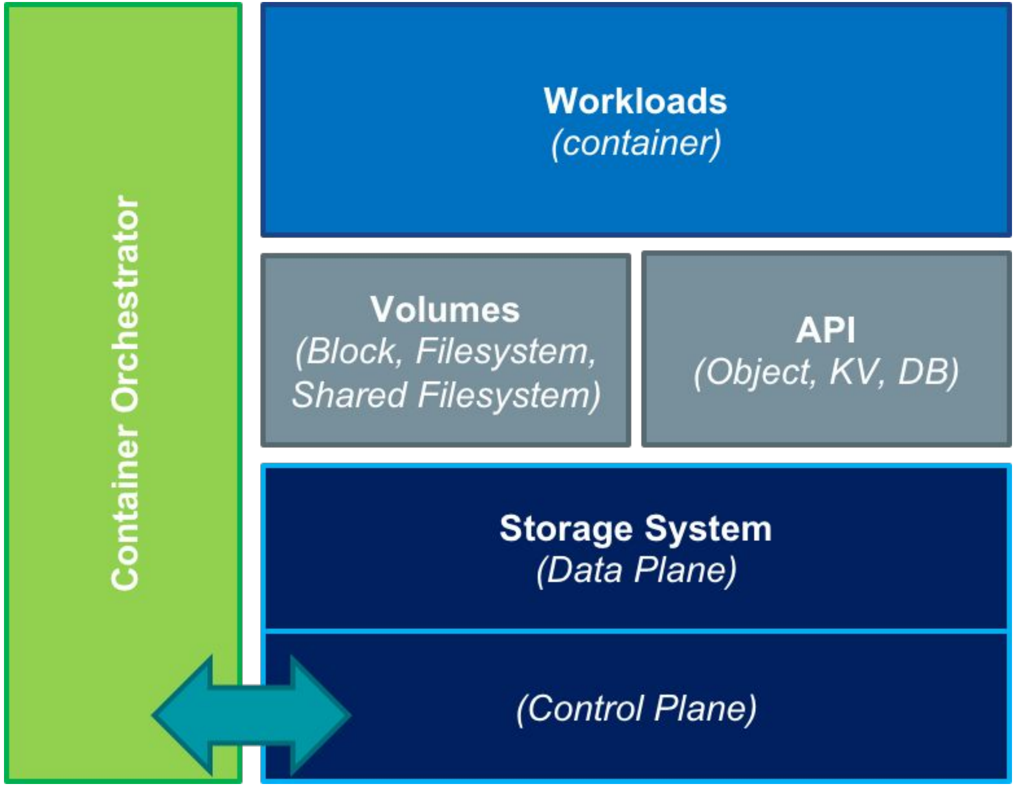
Storage Attributes

Availability	Scalability	Performance	Consistency	Durability
• Failover • Moving access between nodes • Redundancy • Data Protection	• Clients • Operations • Throughput • Components	• Latency • Operations • Throughput	• Delay to access correct data after a commit • Delay between commit and data being committed to non-volatile store	• Data protection • Redundancy • Bit-Rot

Instantiation & Deployment

Instantiation	Description
Hardware	**Deployed as hardware solution in a datacenter.** This limits the portability of the application and generally means that such systems cannot be deployed in a public cloud environment
Software	**Deployed as software components** on commodity hardware, appliances or cloud instances. Software solutions tend to be more **platform agnostic** and can be installed both on-premises as well as cloud environments. Some software defined storage systems can also be **deployed as a container** and **deployment can be automated by an orchestrator**.
Cloud Services	**Consumed from public cloud providers**. Cloud services provide storage services in cloud environments.

Data Access Interfaces



Storage can be accessed via **Data Access Interfaces**:

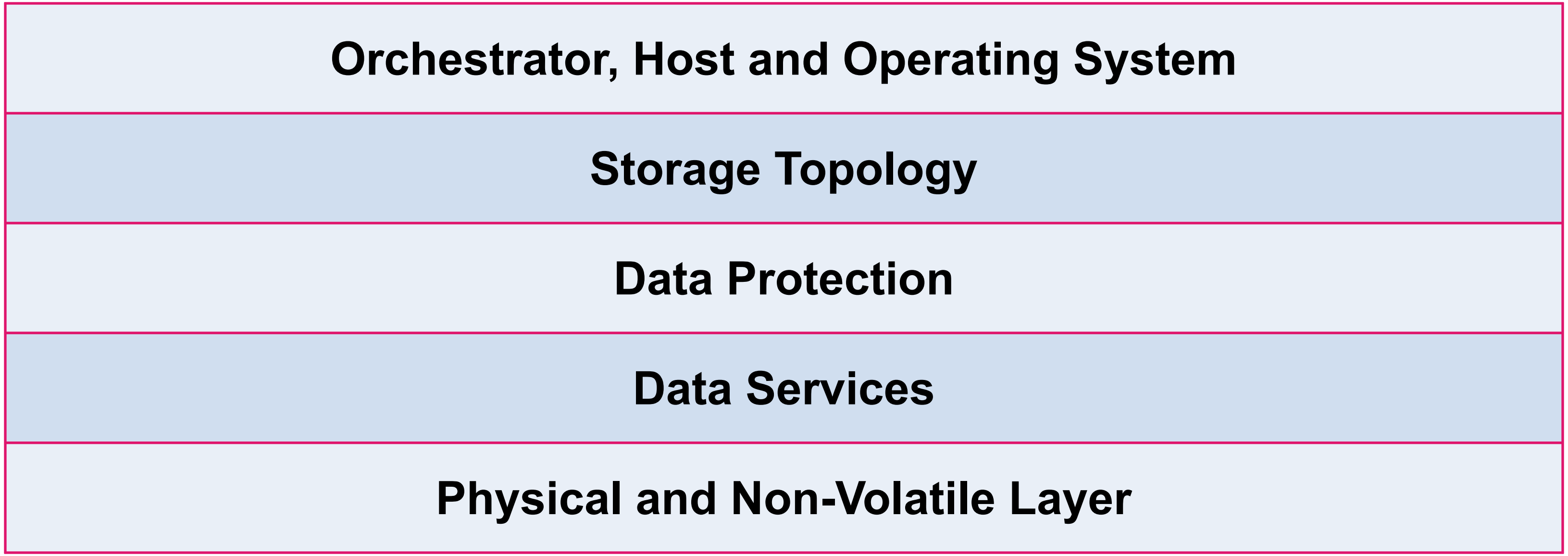
- **Volumes** – accessed through a more traditional file interface in a **block** or **filesystem** interface
- **API** – other ways to persist data such as **object stores**, **KV stores** or **databases**

Comparison: Data Access Interfaces

Data Access Interface	Most suited	Least suited
Block	● Availability ● Low latency performance ● Good throughput performance for individual workloads	● Capacity scaling ● Sharing data with multiple workloads simultaneously
Filesystem	● Sharing data with multiple workloads simultaneously ● Optimised throughput for aggregated workloads	● Strong file locking integrity when filesystems are shared
Object Store	● Availability ● Large capacities (PB scale) ● Durability ● Sharing data with multiple workloads simultaneously ● Optimised throughput for parallelised workloads	● Low Latency performance

***The information in this table are generally accepted attributes and measurements for object stores, file systems and block stores.*

Storage Layers



Orchestrator, Host and Operating System

Storage Topology

Data Protection

Data Services

Physical and Non-Volatile Layer

Orchestrator, Host and Operating System

Orchestrator, Host and Operating System

Layers that are overlaid on a Data Access Interface as part of orchestration

Influences

- Availability
- Scalability
- Performance

Volumes: volume managers, bind mounts, overlay filesystems

API: discovery, load balancers, meshes, ingress

Storage Topology

Storage Topology

Architectures and Topologies of Storage Systems

Influences

- Availability
- Scalability
- Performance
- Consistency

Centralised

Distributed

Sharded

Hyperconverged

Comparison between Local, Remote, and Distributed Systems

	Local	**Remote**	**Distributed**
Availability	**Limited by failure of components locally and ability to failover.** If a node fails, the local storage is isolated to the local node.	**May be limited by single points of failure.** Workloads can move to another node and reconnect to the remote storage.	**Clients may access numerous nodes, and any storage node failures can be mitigated.** The additional complexity of distributed systems may add operational complexity which may in turn affect availability or the ability to recover errors.
Scalability	**Limited by local architecture** (1 node; typically TB)	**Limited by monolithic architecture** (2-16 nodes; typically 10s-100s of TB)	**Scale by adding additional systems.** (3-1000s nodes; often supports PB)
Consistency	**Yes** (storage system implementation is easy)	**Yes** (storage system implementation is harder with more nodes)	**Yes** (storage system implementation is hardest)
Durability	**Limited by local components (less)**	**Limited by monolithic architecture (more)**	**Scaling out to additional systems increases durability (most)**
Performance	**Limited by local components, can benefit low-latency applications (100us-5ms, GB/sec)**	**Similar to local, but additional overhead in network transport (500us-5ms, GB/sec)**	**Scaling out to additional systems increases performance (500us-5ms, TB/sec)**

*** The information in this table are generally accepted attributes and measurements among local, remote, and distributed storage systems.*

Data Protection

Data Protection

How data is protected through redundancy

Influences	
- Availability - Scalability - Performance - Consistency	**RAID & Mirrors** **Erasure Coding** **Replicas**

Data Services

Data Services

Data services which complement the core storage function

Influences

- Availability
- Durability
- Performance

Replication

Snapshots and Clones

Encryption

Physical and Non-Volatile Layer

Physical and Non-Volatile Layer

Terminology that is often used in both storage products and services

Influences

- Durability
- Performance

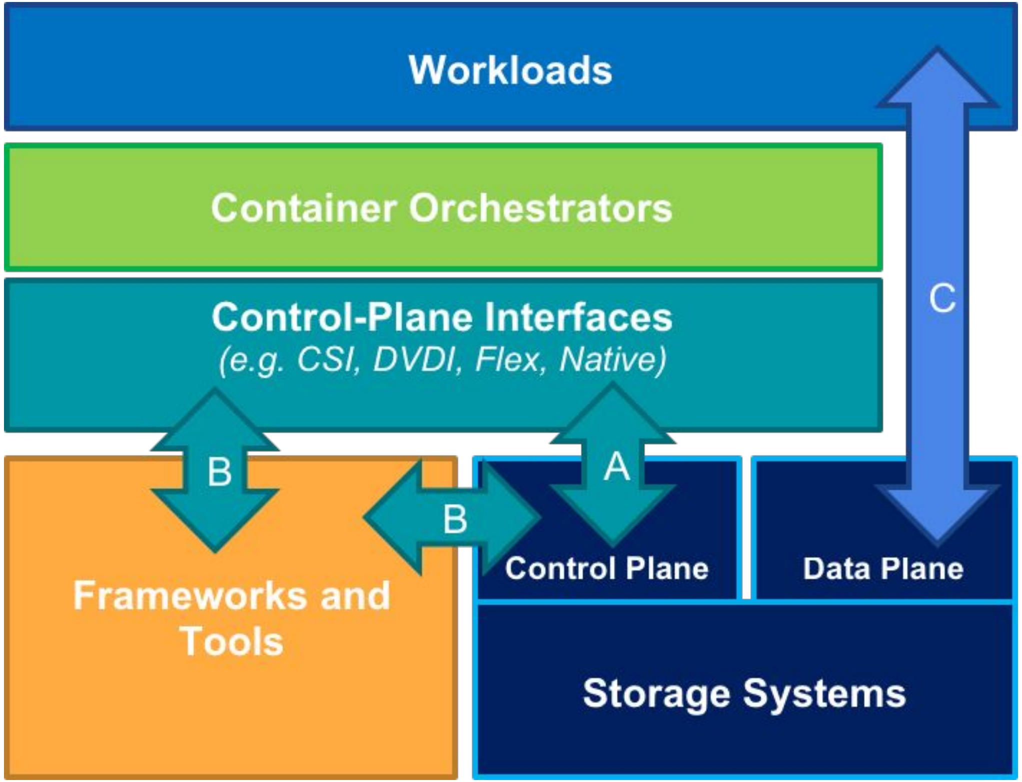
Spinning Disk (e.g. SATA, SAS & SCSI)

Solid State Disk

Non Volatile Memory (e.g. NVMe)

Cache (memory & otherwise)

Orchestration and Management Interfaces



Container Orchestration system (**CO**) uses an interface to interact with a storage system

The storage system can:

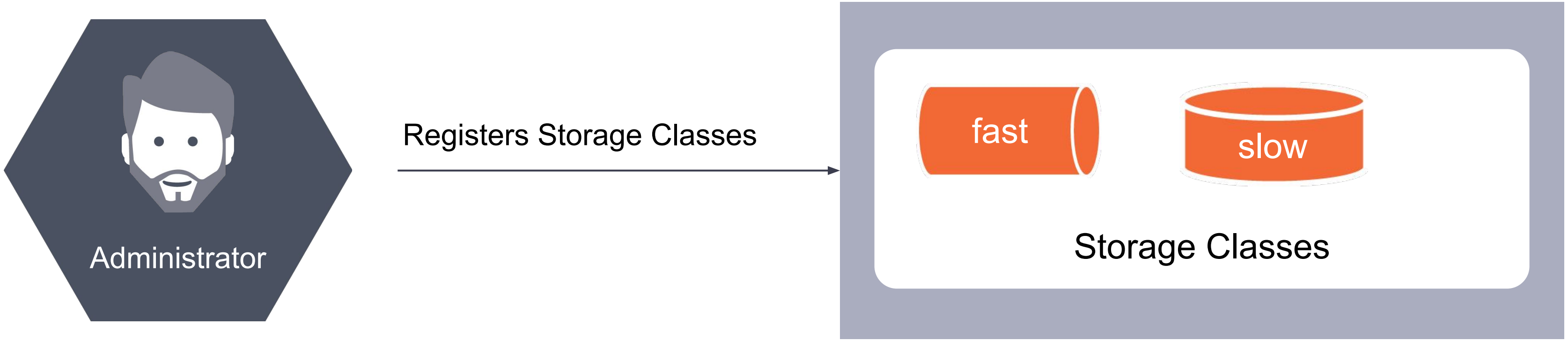
- **(A)** support control-plane API directly
- **(B)** interact via an API Framework layer or other Tools

Workloads consume **(C)** storage via a data access interface

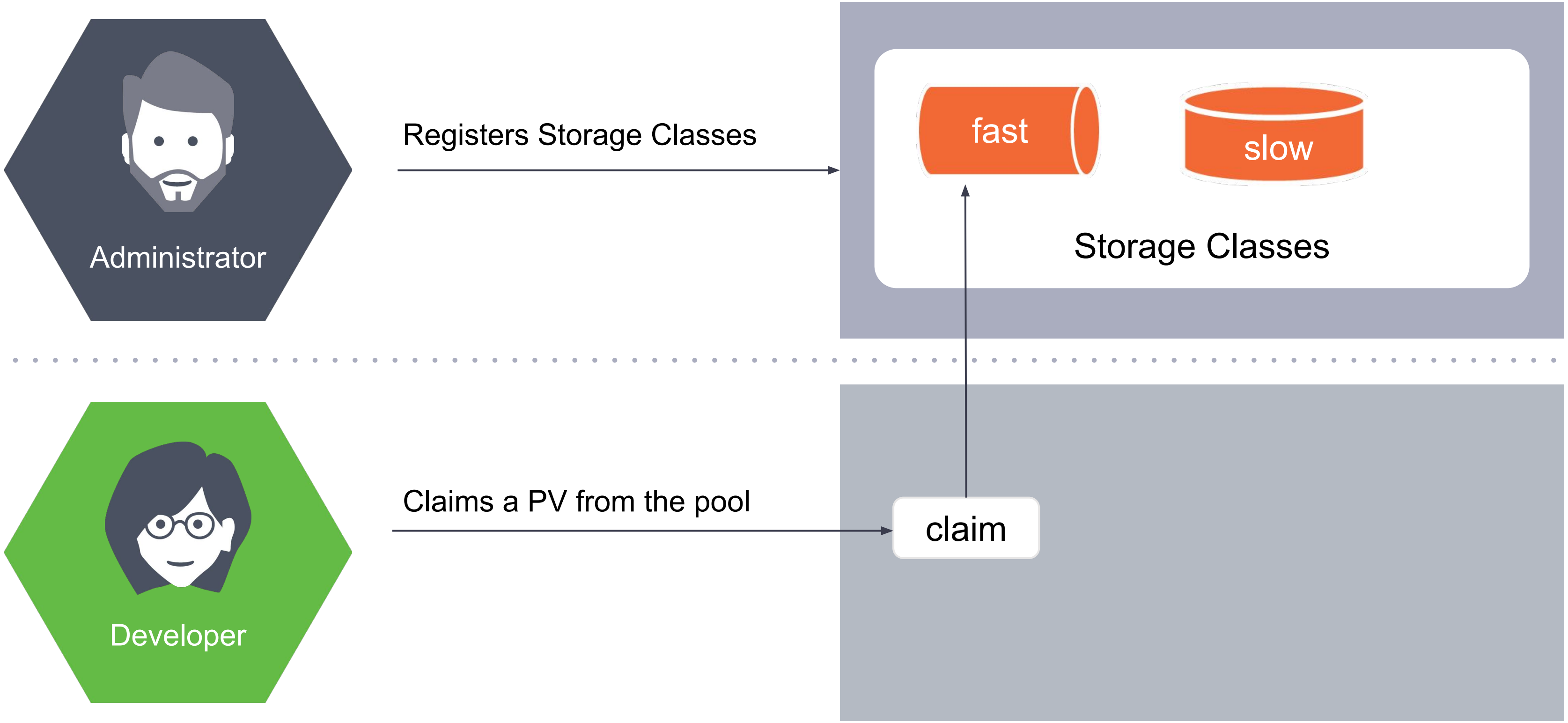
Storage for Stateful Applications

CLOUD NATIVE
COMPUTING FOUNDATION

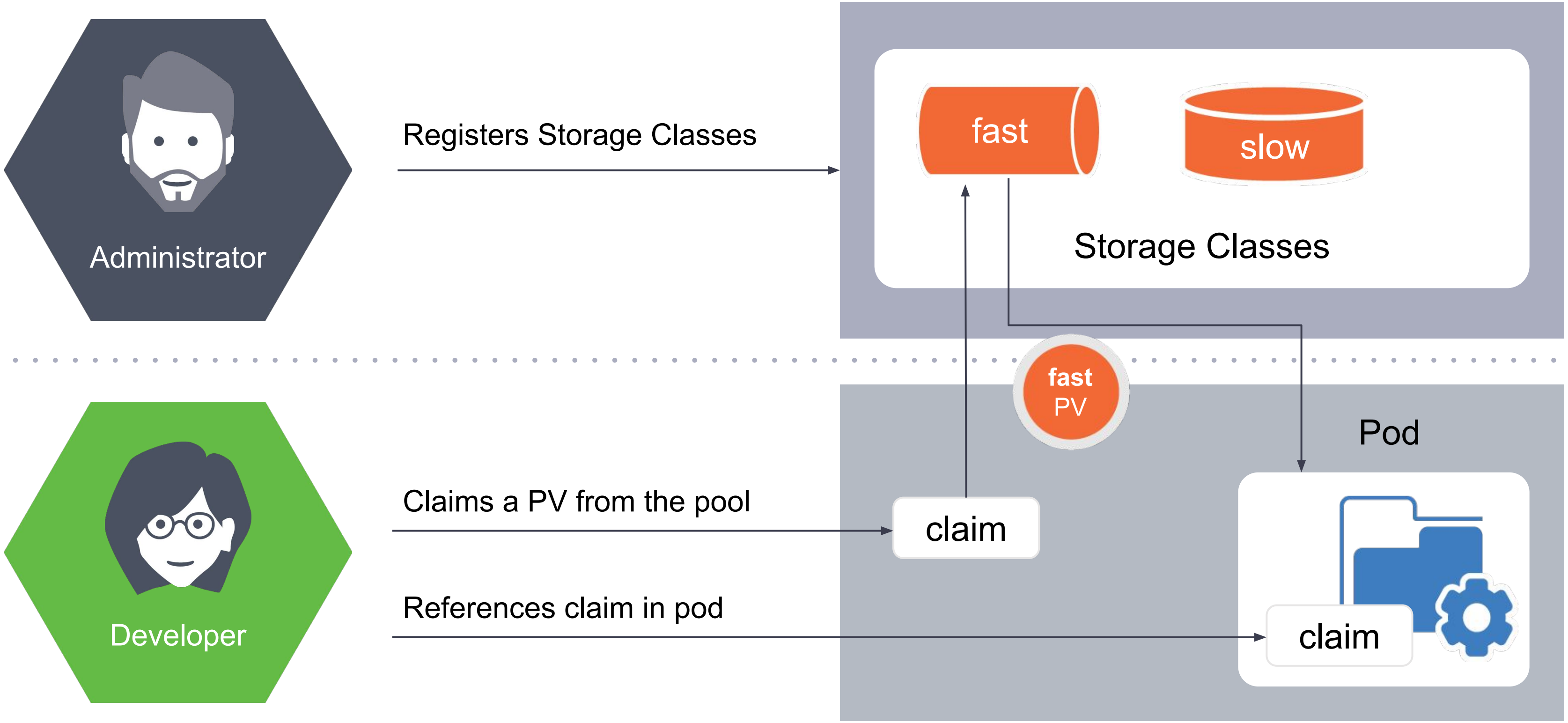
Dynamic Provisioning in Kubernetes



Dynamic Provisioning in Kubernetes

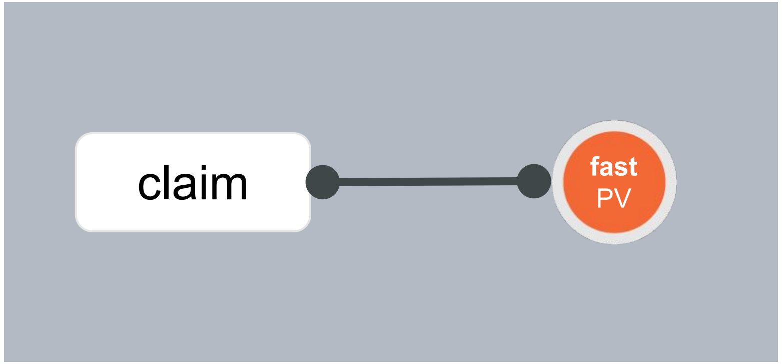


Dynamic Provisioning in Kubernetes



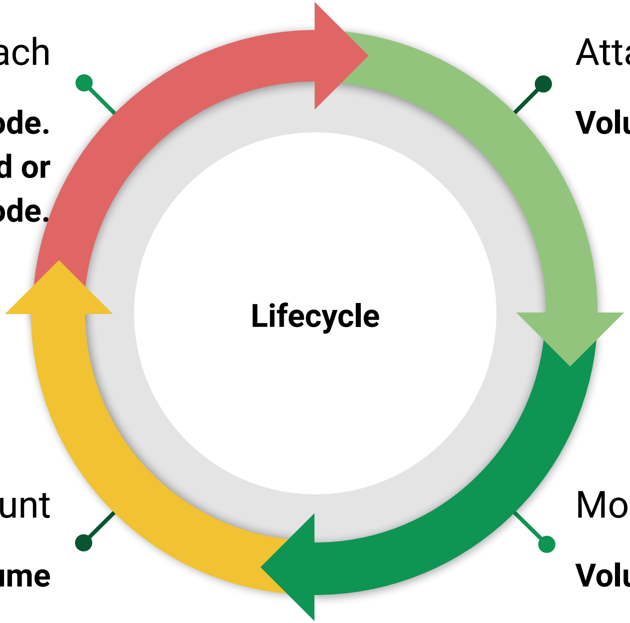
Dynamic Provisioning in Kubernetes

Resulting in a *binding*:



The *claim* can now be used with an application

Kubernetes Volume Lifecycle



Lifecycle

Attach

Volume is attached to a node

Mount

Volume is now available to be used

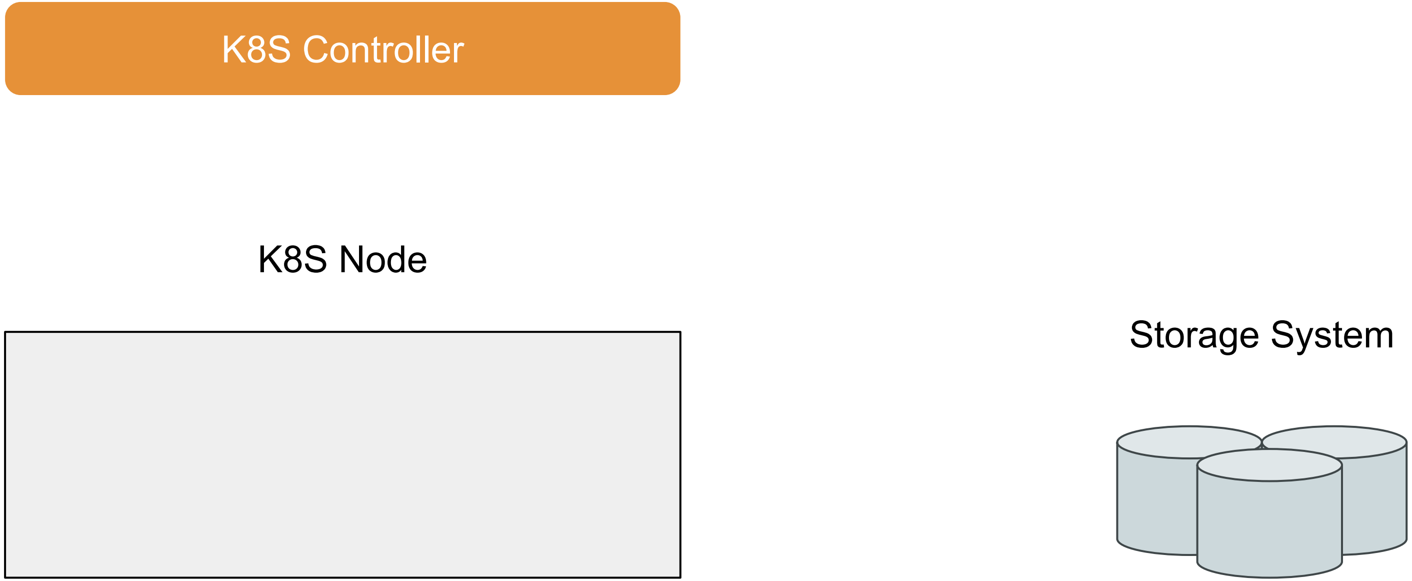
Unmount

Preparation to remove volume

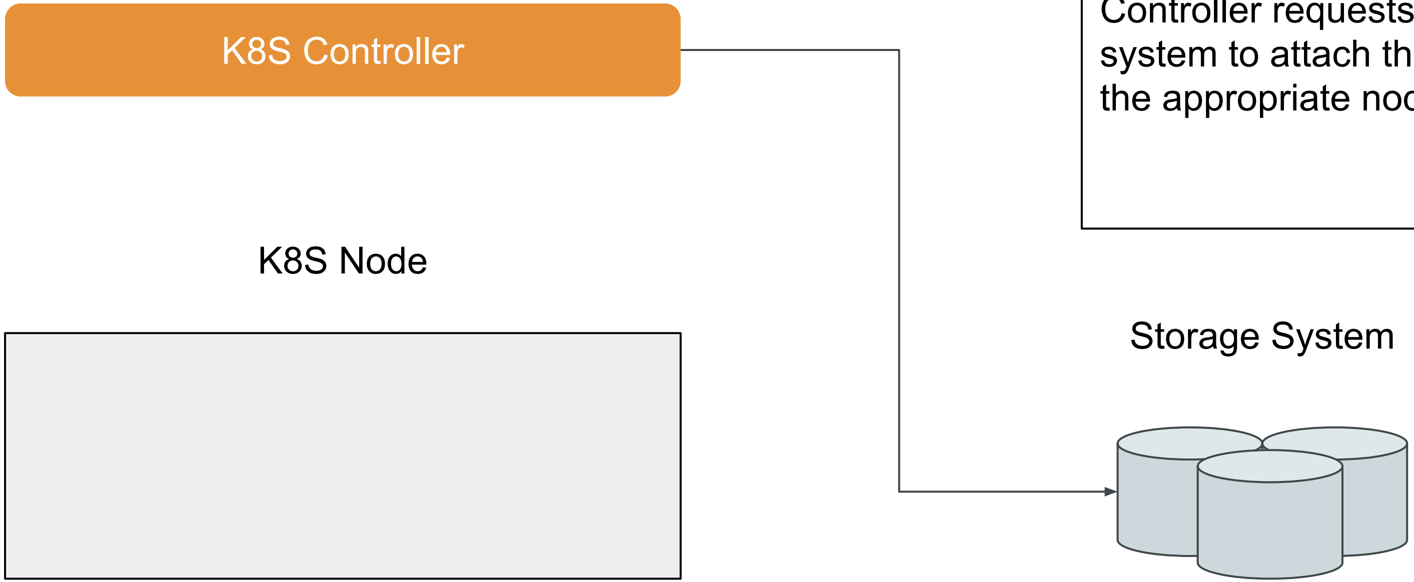
Detach

Volume is removed from the node. Volume may now be deleted or reattached in another node.

Providing Storage in Kubernetes

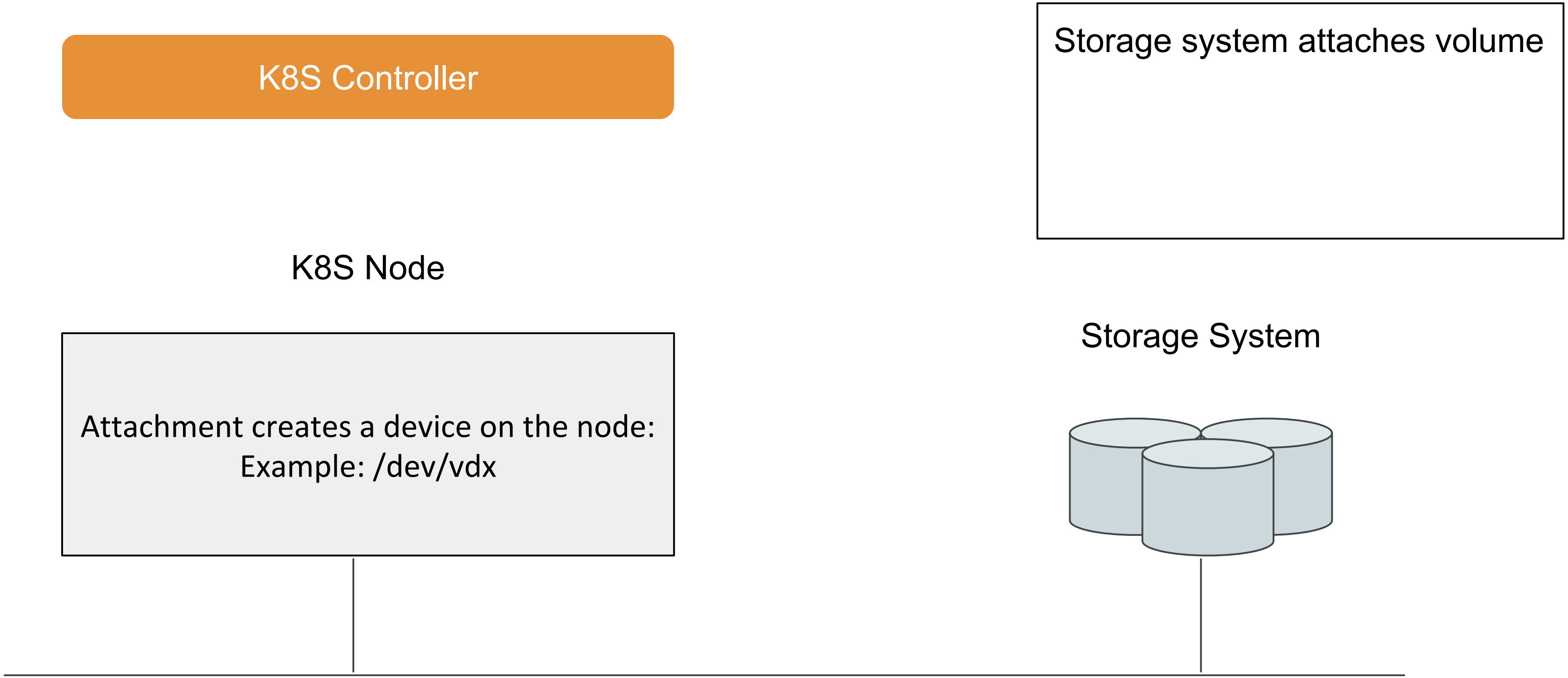


Providing Storage in Kubernetes



Controller requests storage system to attach the volume on the appropriate node

Providing Storage in Kubernetes



K8S Controller

Storage system attaches volume

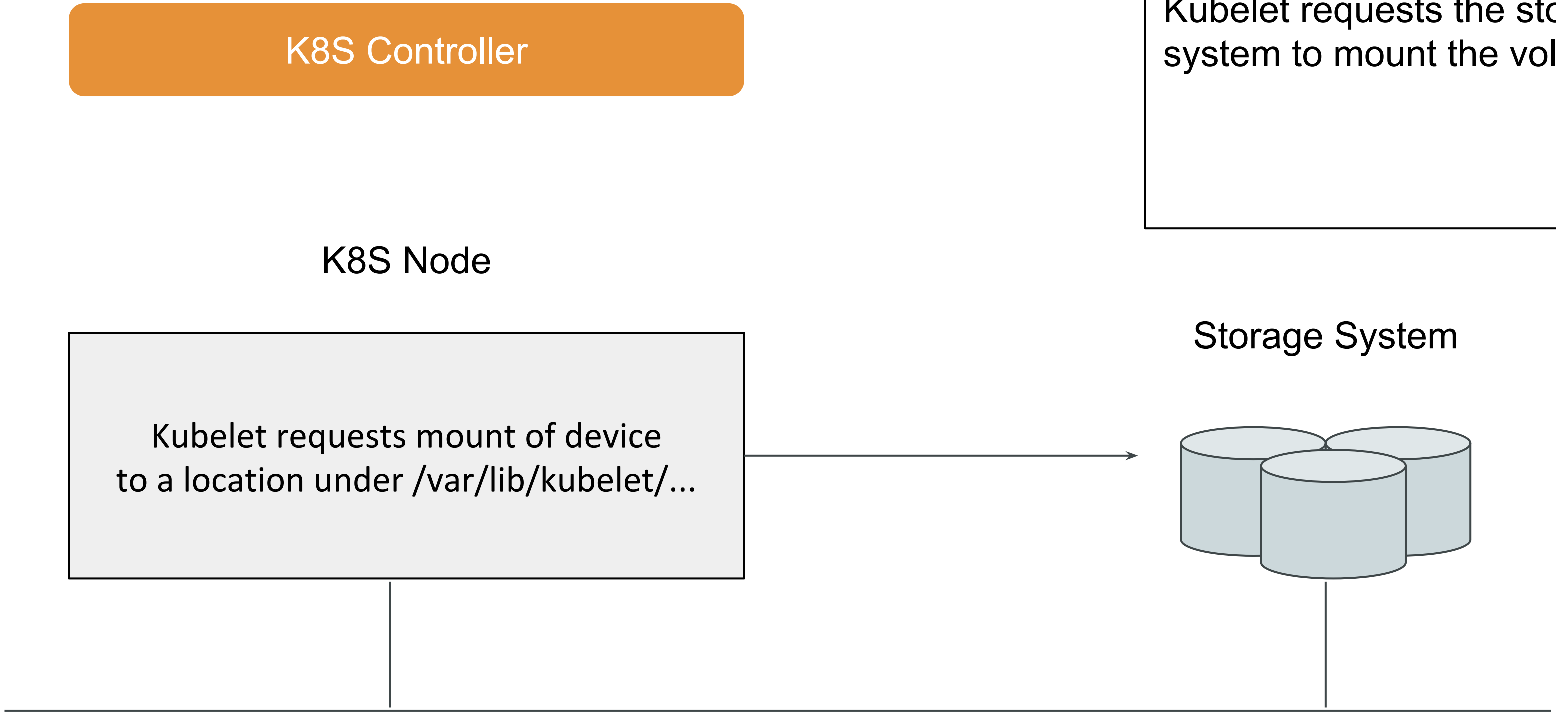
K8S Node

Attachment creates a device on the node:
Example: /dev/vdx

Storage System

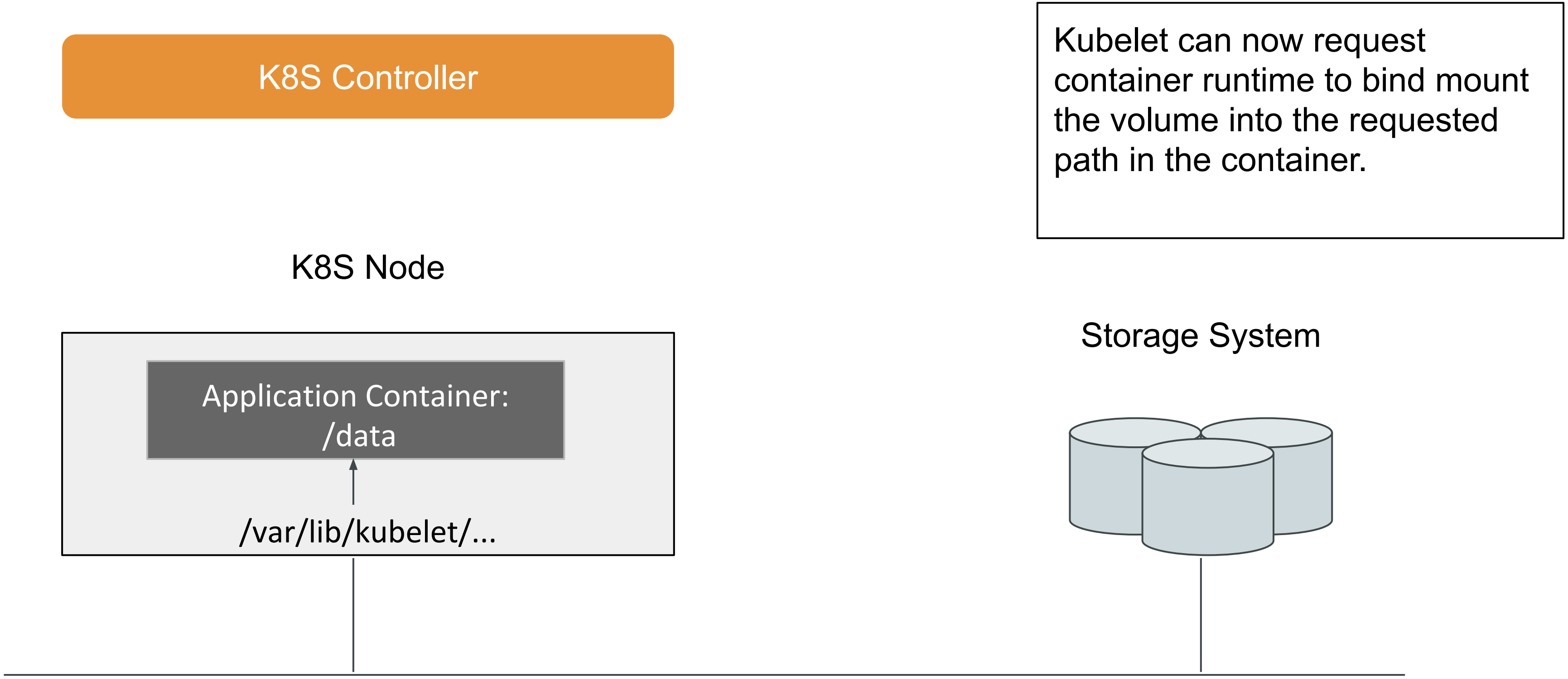
Providing Storage in Kubernetes

Kubelet requests the storage system to mount the volume



K8S Controller
K8S Node
Kubelet requests mount of device to a location under /var/lib/kubelet/...
Storage System

Providing Storage in Kubernetes



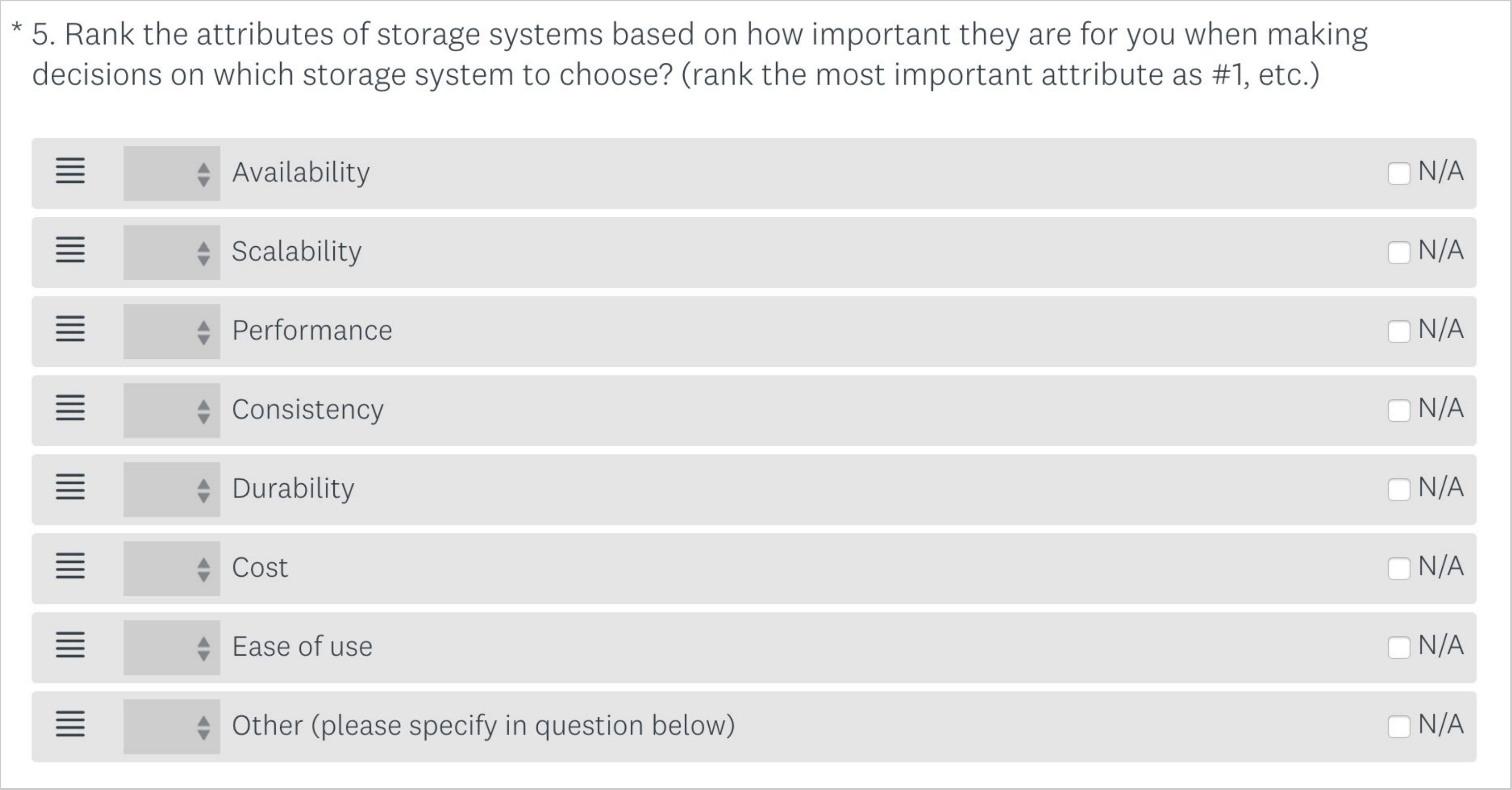
Kubelet can now request container runtime to bind mount the volume into the requested path in the container.

Persistence for Kubernetes Stateful Applications

Type	Access Mode	Description
Block	ReadWriteOnce	● Mount required ● Normally, accessed exclusively by only one node.
Filesystem	ReadWriteMany	● Mount required ● Accessible by multiple nodes
Object Store, Databases		● No mount required, therefore storage access is not managed by Kubernetes ● Accessible using APIs

Storage Survey

http://bit.ly/cncf-storage-survey



* 5. Rank the attributes of storage systems based on how important they are for you when making decisions on which storage system to choose? (rank the most important attribute as #1, etc.)

≡	Availability	□ N/A
≡	Scalability	□ N/A
≡	Performance	□ N/A
≡	Consistency	□ N/A
≡	Durability	□ N/A
≡	Cost	□ N/A
≡	Ease of use	□ N/A
≡	Other (please specify in question below)	□ N/A

KubeCon | CloudNativeCon

North America 2019

Nov. 18 - 21, 2019
San Diego, CA
kubecon.io

Q&A

CLOUD NATIVE
COMPUTING FOUNDATION