KubeCon | CloudNativeCon

Europe 2020

Virtual

Zero Downtime Deployments:
Controlling Application Rollouts and Rollbacks

Christopher Hanson
RX-M, llc.

whoami

Christopher Hanson (Chris)
Senior Cloud Native Engineer at RX-M

Speaker

- KubeCon / CloudNativeCon Shanghai:
  - Helm
  - CNCF Certification Programs
- San Francisco Meetup: Managing Resources in K8s apps
  (can be found on the RX-M YouTube channel: https://youtu.be/ijWZinxdHEA)

Instructor – K8s, Helm, Prometheus, Spinnaker, Docker, OpenShift, Ansible, OpenStack (to name a few)

Consultant/Advisor – built a K8s Cassandra DBaaS, monitoring/logging pipeline using the EFK stack

Certified Kubernetes Application Developer
(CKAD-1900-0994-0100)
Certified Kubernetes Administrator
(CKA-1700-0131-0100)

@CloudNativChris

rx-m cloud native training & consulting

Kubernetes Controllers

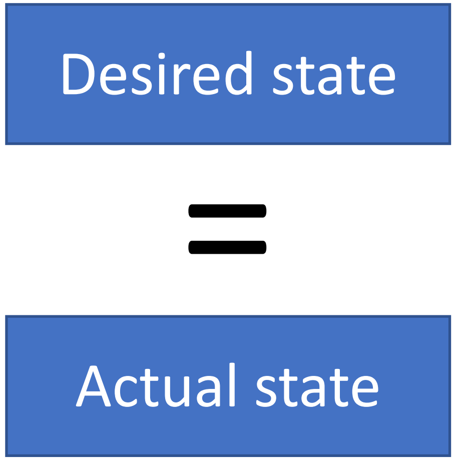
Controllers are control loops that watch the cluster's state

If necessary, the controller changes a resource's current state to match the desired state:

- An application replica fails, a new replica is created
- A resource is scaled up or down, Pods will be created or terminated to match the desired state

Kubernetes has several built-in controllers that satisfy common use cases:

- Deployments
- DaemonSets
- Jobs
- StatefulSets


Desired state

=

Actual state

Deployments

Describes the desired state for an application

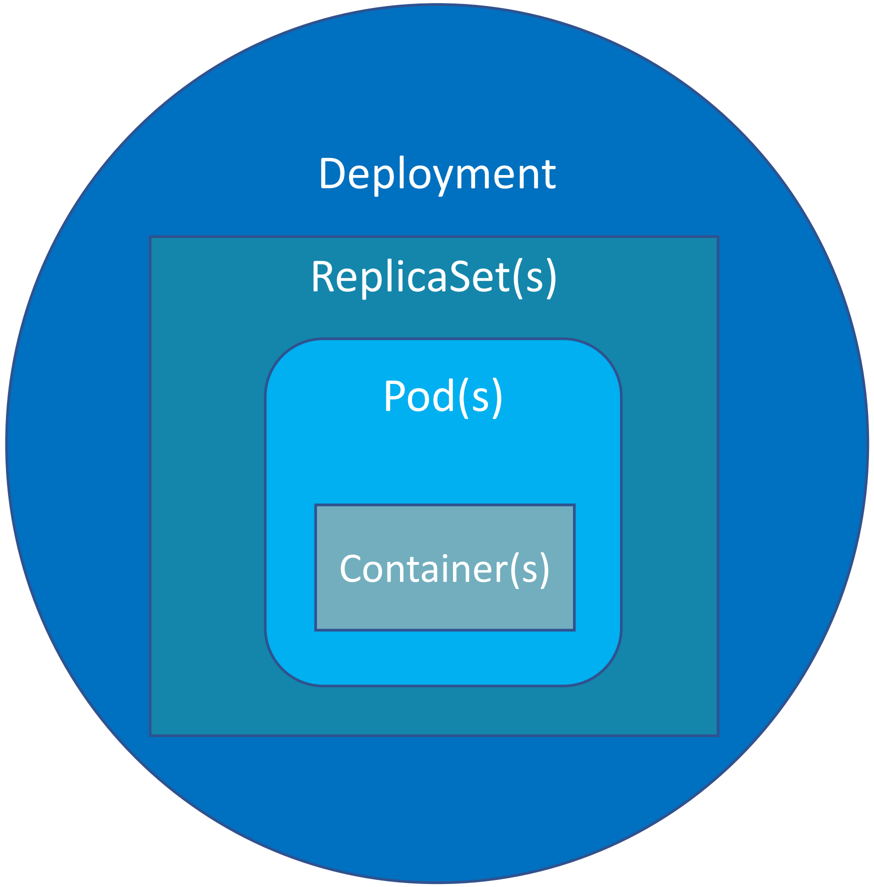
Wrapper for ReplicaSet (RS) controller, Pod, and application container

Creating/deleting a Deployment creates/deletes RS and Pods in a cascade

Rolling update feature changes the actual state to the desired state with zero downtime!

Supports a wide array of application types but feature set works best with *stateless* apps

- Pod hostnames not predictable/stable
- Can only use PVs/PVCs with ReadWriteMany access mode



ReplicaSets

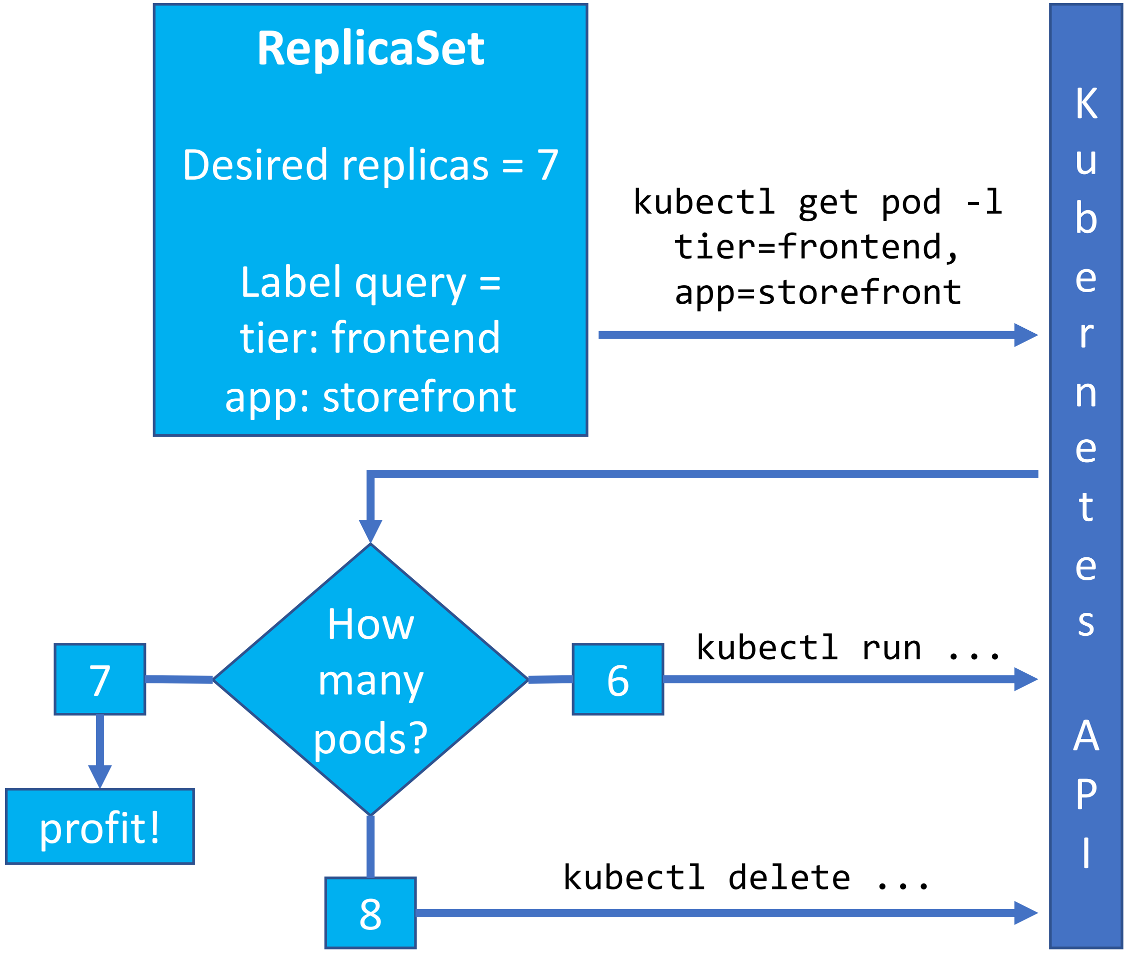
Reconcile desired vs actual replication factor

Guarantee availability of application replicas

Use selectors to query for, and add/remove target pods

Can be used independently of Deployments

When used *with* Deployment controller, part of rolling update feature



```
kubectl get pod -l
tier=frontend,
app=storefront
```

```
kubectl run ...
```

```
kubectl delete ...
```

Deployments & RSes

One-to-many relationships:

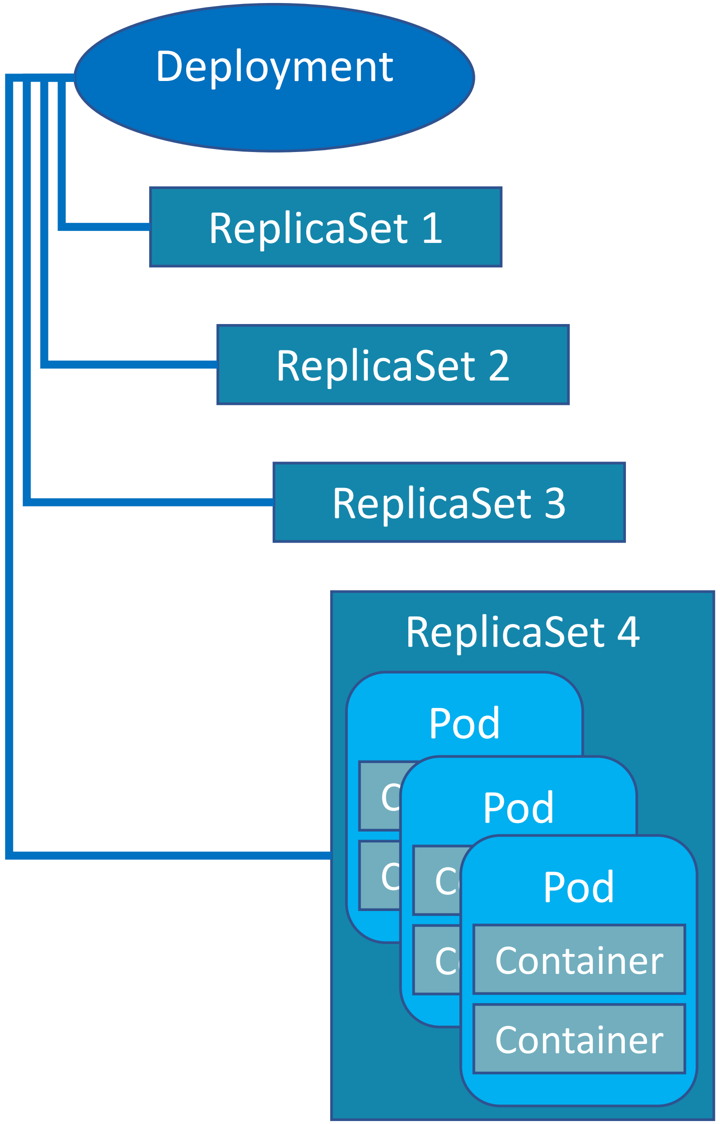
- Between Deployments and RSes
- Between RSes and Pods

RS is a "snapshot" of an application revision:

- Image change
- Configuration change
- etc.

Revision history – number of ReplicaSets to retain

- Enables/limits to *n* number of rollbacks
- Lightweight tracking of revisions to an app



Deploymets: Rolling Updates

Changes to Pods are rolled out at a controlled rate

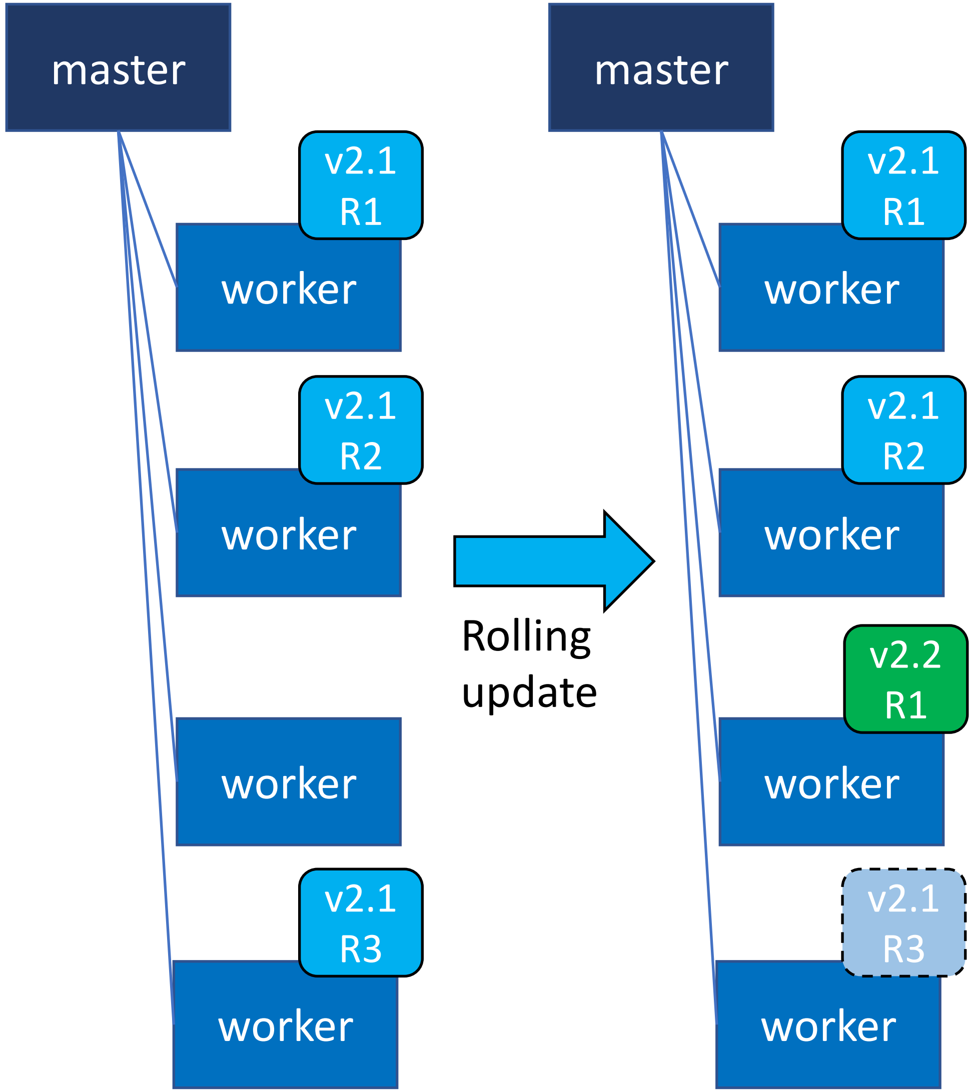
New Pods are rolled out:

- Before old Pods are terminated
- On Nodes with available resources

Default setting ensures that at least 75% of the Pod replicas are available throughout an update

Client traffic is load balanced across all available Pods, despite application version

In-progress rollouts can be watched for updates



Demo: Rolling Update

Rolling update demo:

Default behavior

Undo, status, and history

Deployment Rolling Update

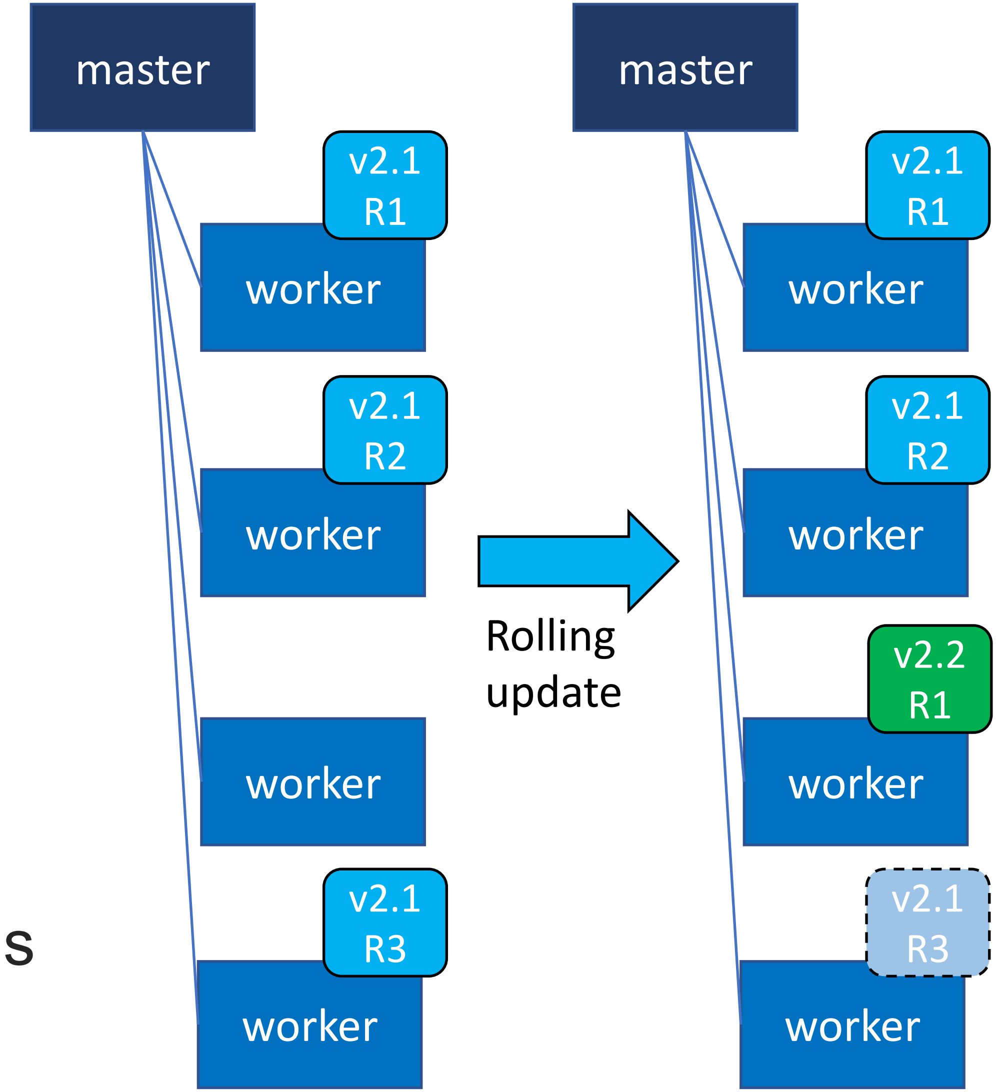
Pausing – a Deployment's rollout trigger can be paused at any time

- Issue several commands that make changes without trigging an update for each change
- During the rolling update to confirm settings before completing

Resuming – rolling updates can start or proceed

- All changes made *prior* to resuming will be rolled out during a single rolling update
- Finishes a rollout that was stopped in progress

Any subsequent change(s) to a Deployment that is not paused will trigger immediate rollouts



Demo: Rolling Update

Rolling update demo:

Pause & resume

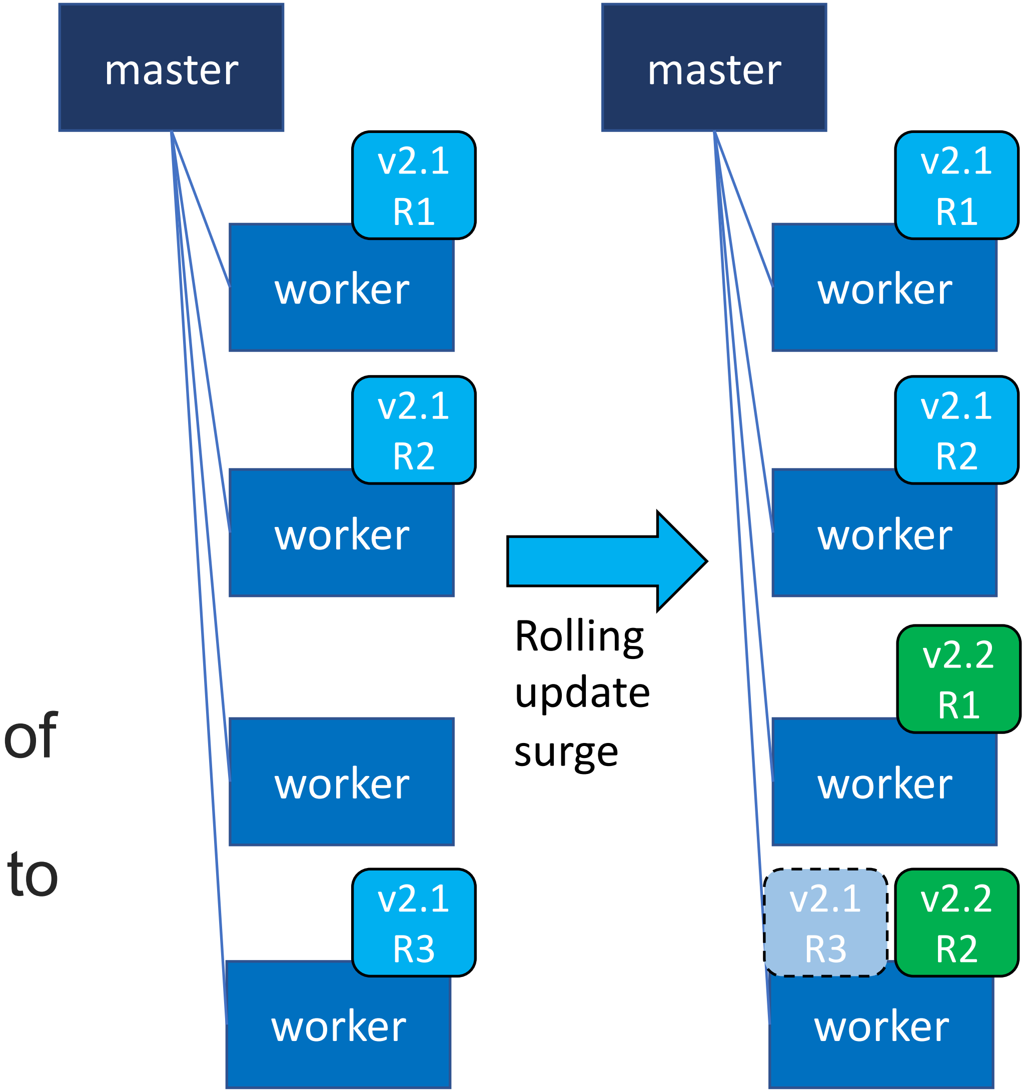
Controlling Rolling Updates

Max surge – number of pods *in addition to* the desired number that can be *scheduled* during a rolling update

- Controls how many new replicas roll out at a time
- Breaks updates into iterations/waves
- High surge % means more resources (cpu, memory) needed during rollout

Max unavailable – ensures a minimum number of pods are always available

- Guarantees that client traffic can be delivered to pods throughout the rolling update
- Cannot be 0
- Set at 100% will mean downtime!



Demo: Rolling Update

Controlling Rolling update demo(s):

Max surge

Max unavailable

StatefulSets

Support stateful applications that require:

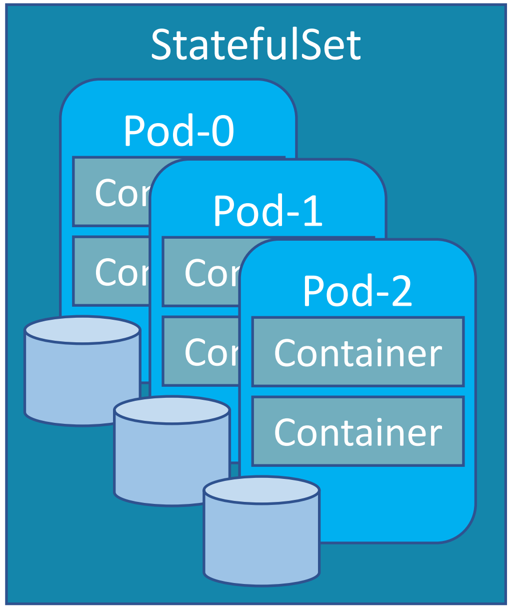
- Stable network identities – Pods have unique ordinals
- Persistent storage – embedded PVC template creates a PV for each pod

Create Pods and PVCs in a cascade

- Deleting a StatefulSet deletes Pods but not PVCs–safety first!

Pod identities tied to volumes

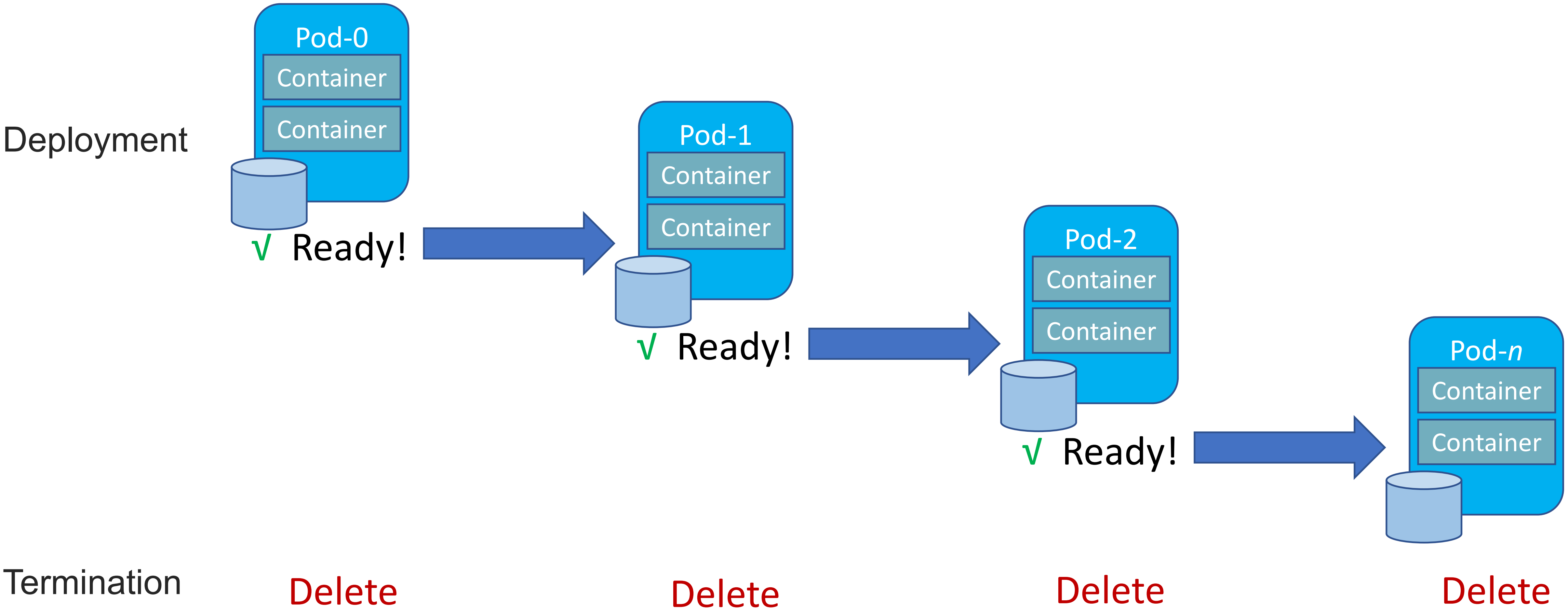
- Failed Pods replaced by Pods with identical identifiers match existing volumes and bind to them



StatefulSets

Pod ordinals provide guarantees about ordering:

- Sequential Pod deployments and scaling
- No guarantee during termination when a StatefulSet is deleted (workaround: scale to 0 first!)



StatefulSet Rolling Updates

Pods are deleted and recreated/replaced *on the same node*

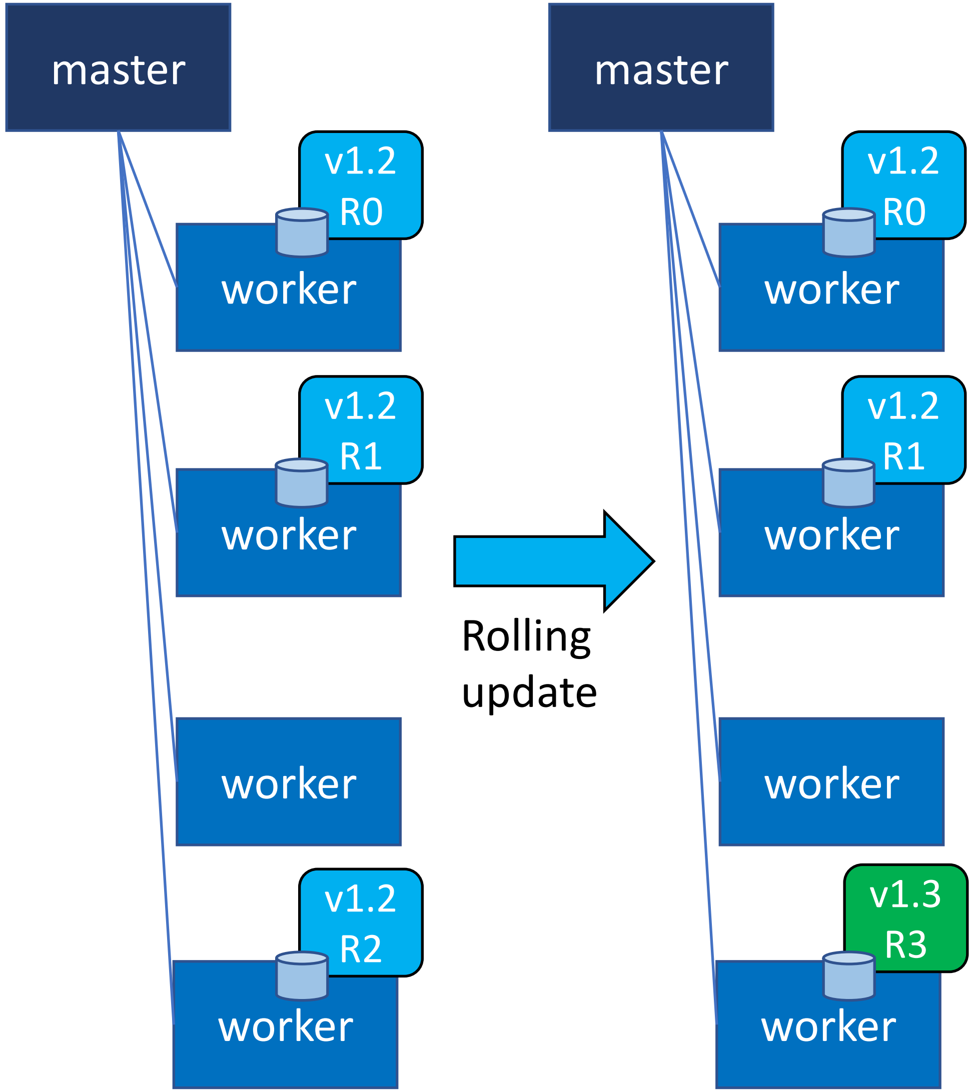
- Eliminates the need to detach/attach network volumes from/to Nodes

StatefulSet rolling updates only support undo and status commands

- History not really functional
- Pause/resume not supported

Forced rollbacks sometimes necessary

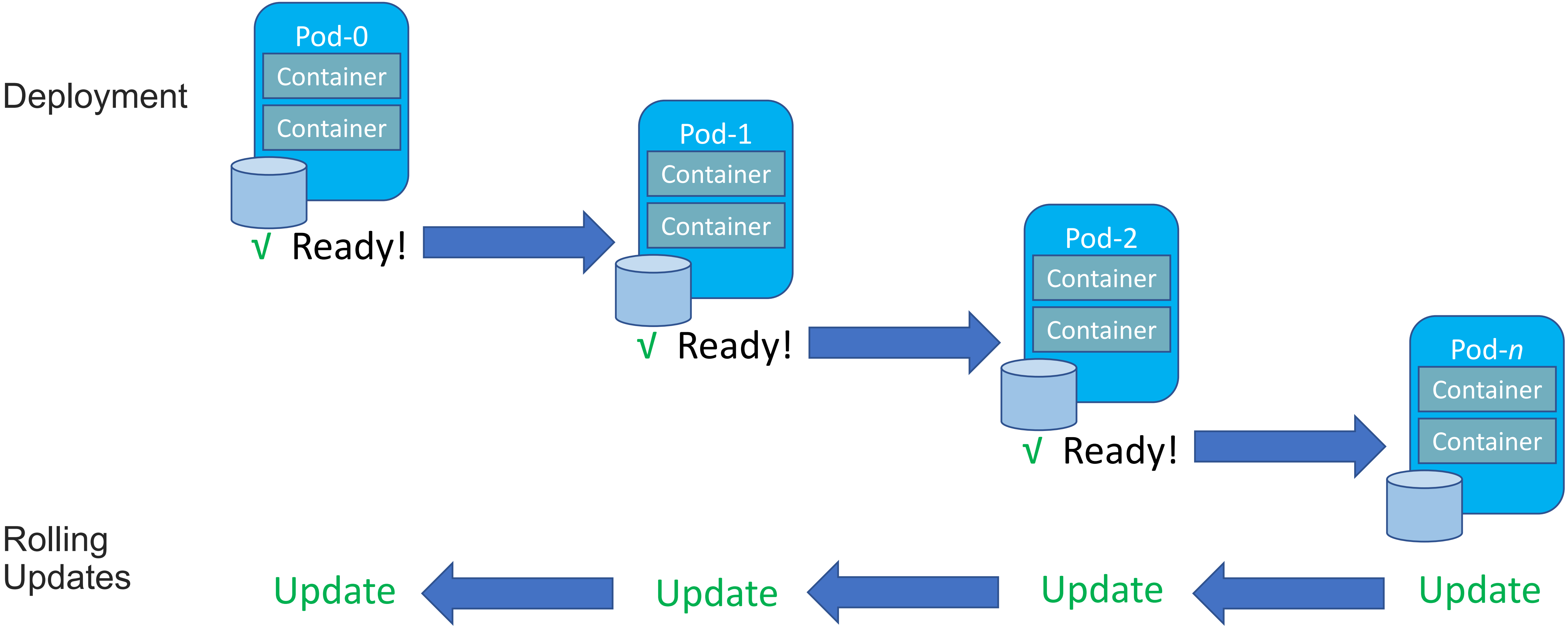
- If a Pod never reaches Ready state, reverting the change will not rectify the Pod, it must be manually deleted to force the rollback



StatefulSet Rolling Updates

Sequential/ordered Pod rollouts

- Pods are replaced/rolled out in reverse order



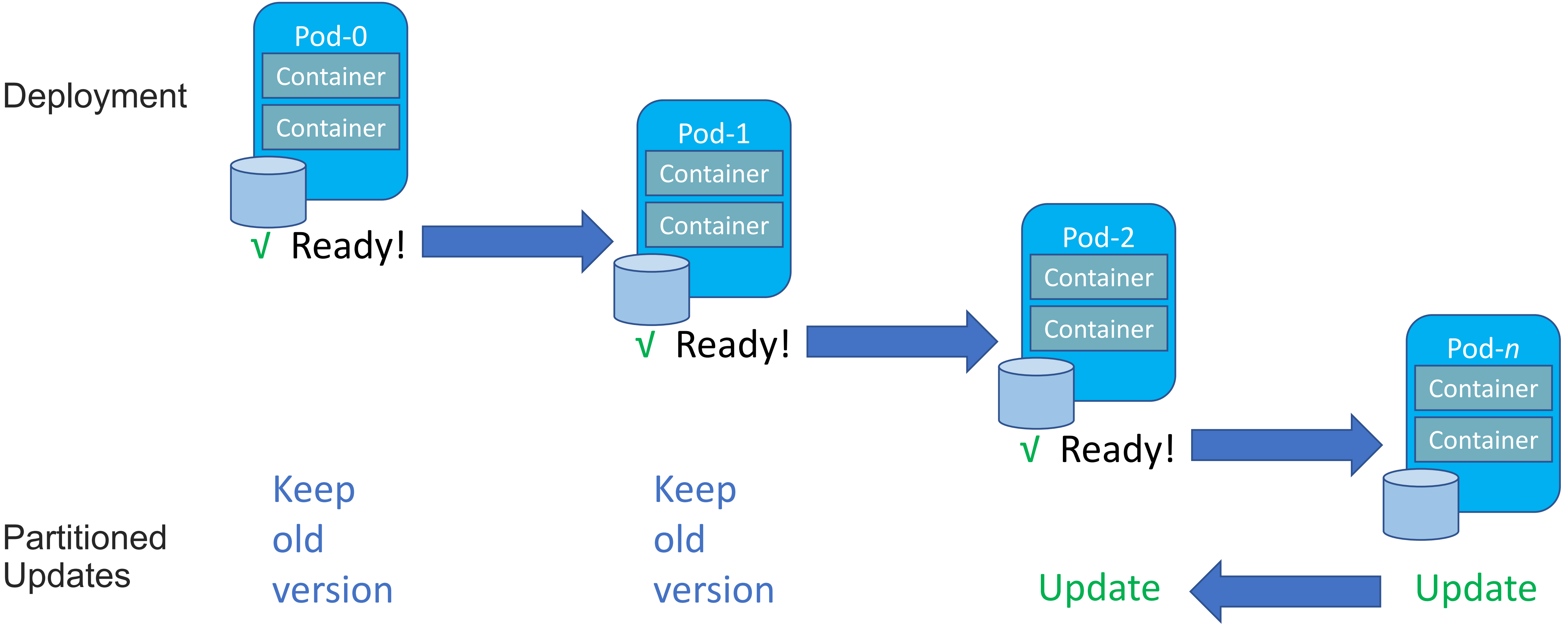
Demo: STS Rolling Update

Rolling update with network attached storage

Undo, status, history

StatefulSet Partitioned Updates

Pods >= partition number rollout the update
Pods < partition number remain at the previous version–even if deleted!



Demo: STS Rolling Updates

Partitioned update with local storage volumes

Summary

Kubernetes controllers provide features that provide zero-downtime rolling updates

Behaviors and features differ by controller:

- Deployments overlap new Pod replicas with old Pod replicas
- StatefulSets delete and replace Pods in place and require more from the application

KubeCon CloudNativeCon

Europe 2020

Virtual

Thank you!

KubeCon | CloudNativeCon

Europe 2020

Virtual

HELM

KEEP CLOUD NATIVE CONNECTED