Stakater

Effective Disaster Recovery Strategies for Kubernetes

About me

- CEO @ Stakater
- Container platform evangelist & fullstack developer & architect and instructor (13+)
- Explore our open source contributions in Kubernetes ecosystem: https://github.com/stakater

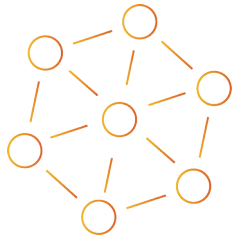
Stakater is a ***Kubernetes*** experts company enabling enterprises to realize the full potential of Kubernetes and its ecosystem, by assisting their journey from Strategy to Development and Operations.

TODAY'S CHALLENGES FOR HIGH AVAILABILITY (HA) AND DISASTER RECOVERY (DR)

TODAY CIO’s REQUIRE MORE

REQUIREMENTS

- No outages
- React quickly to demand
- Provide service 24 x 7 x 365



IMPLEMENTATION

- DevOps, CI/CD
- Rolling upgrades
- Redundancy
- HA architectures
- HA operational processes

CONTINUOUS SERVICE DELIVERY

Stakater

What is Business Continuity ?

Business Continuity is having a strategy to deal with major disruptions and disasters.

Backup

Copying your data to another environment or location

Disaster Recovery

Providing you with an environment that is capable of sustaining your business

Why is Business Continuity important ?

24 / 7 Availability

Competitive Advantage

Malware Protection & Security

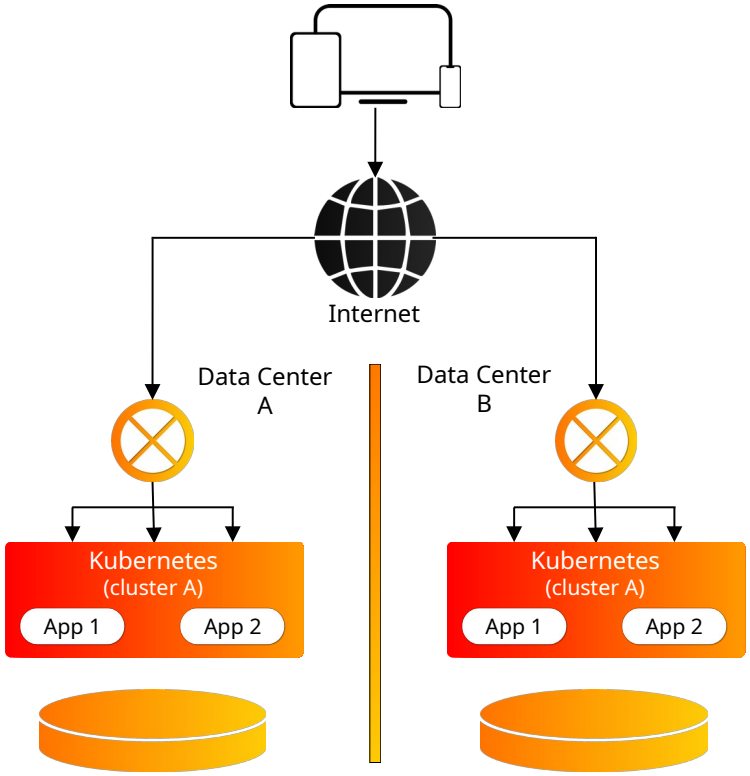
Reputation & Revenue

CHALLENGES

- Efficiency
- Complexity
- Cost

INTRODUCTION TO HIGH AVAILABILITY (HA)

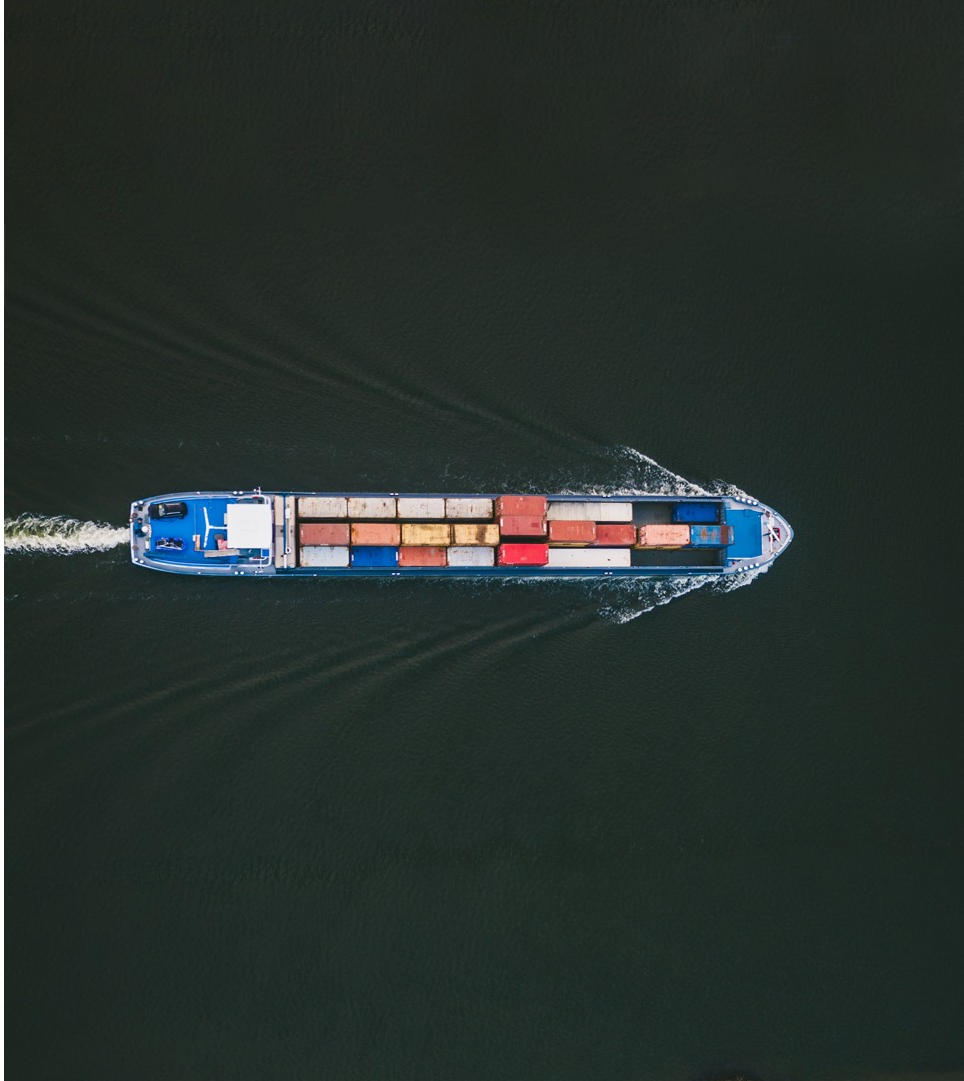
- Single site application (Traditional)
  - No disaster recovery
- DR application (Traditional)
  - Failover to a secondary site
- Multi site application (Cloud-native)
  - Deploy across multiple sites
  - Disaster recovery is built-in Simply re-scale
  - Recommended three sites vs two



WHAT EVERYONE WANTS?

Expectation from our IT infrastructure

Availability and uptime for our apps



But eventually, something may go wrong

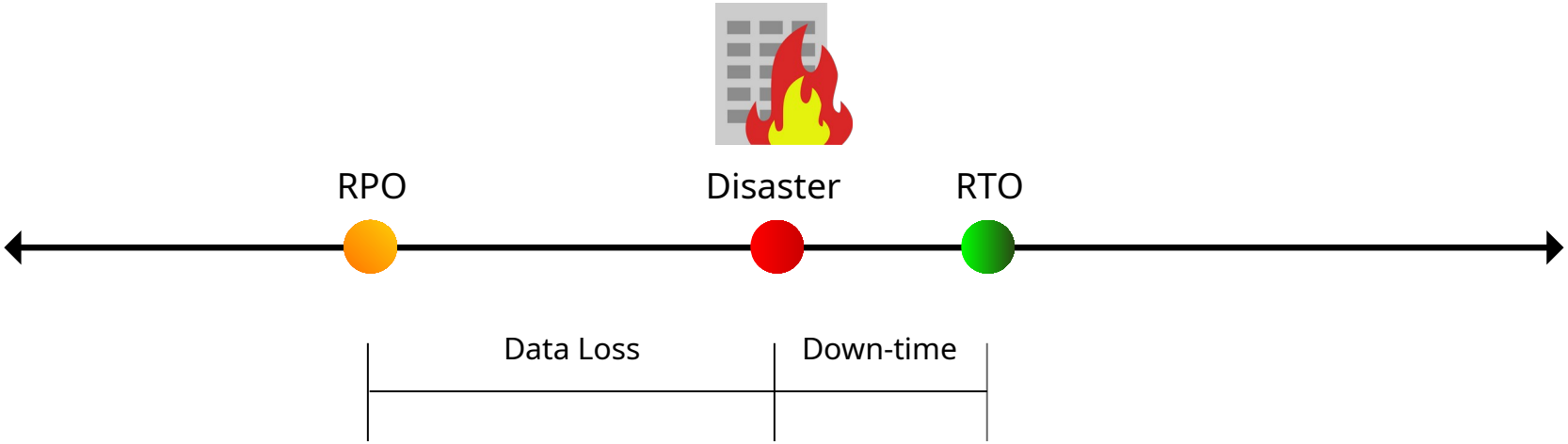
Time for...

Disaster Recovery

Disaster Recovery Metrics

Recovery Point Objective (RPO)

Recovery Time Objective (RTO)



PRE-CONTAINER ERA

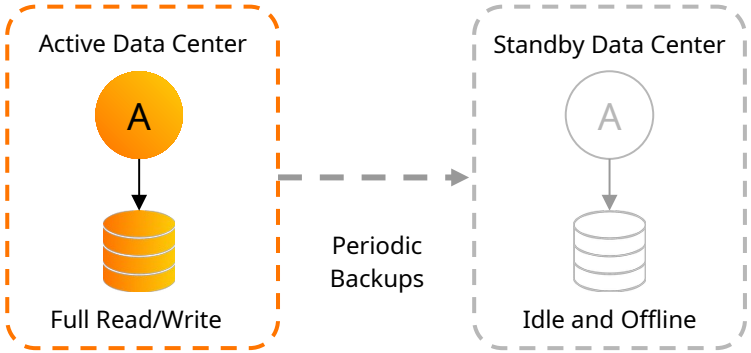
Stakater

Pre-Container ecosystem

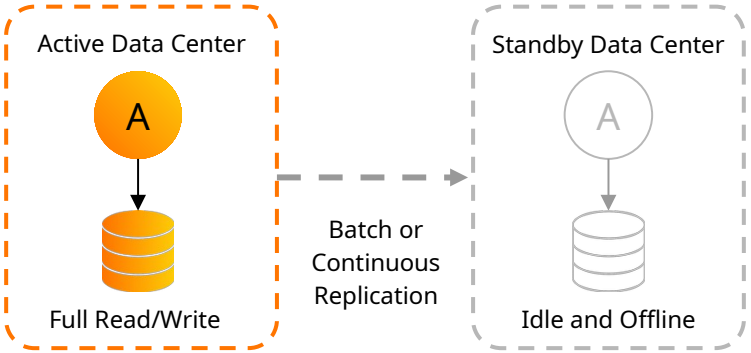
- Strong relation between **applications** deployed on specific **servers**

- Disaster Recovery
  - Regular backups server including software and configuration on the server
  - Restore backup on new working server

TRADITIONAL DR APPROACHES



COLD STANDBY

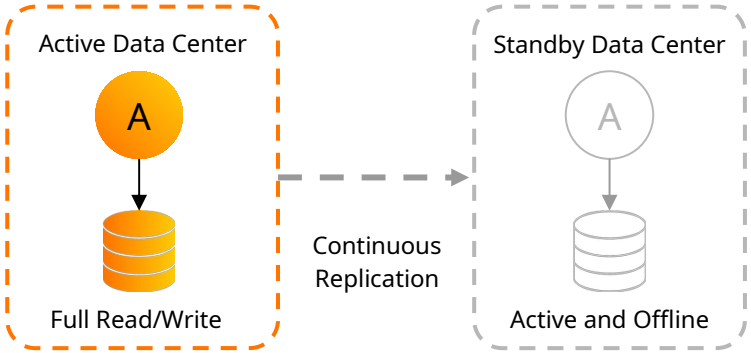


WARM STANDBY

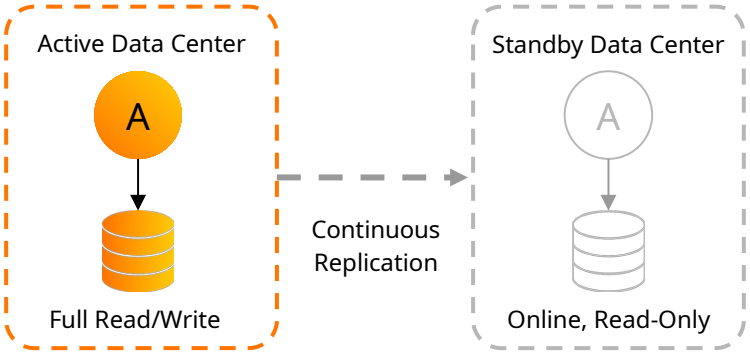
- Application and data unavailable until standby is brought online
- Data loss highly likely

Stakater

TRADITIONAL DR APPROACHES



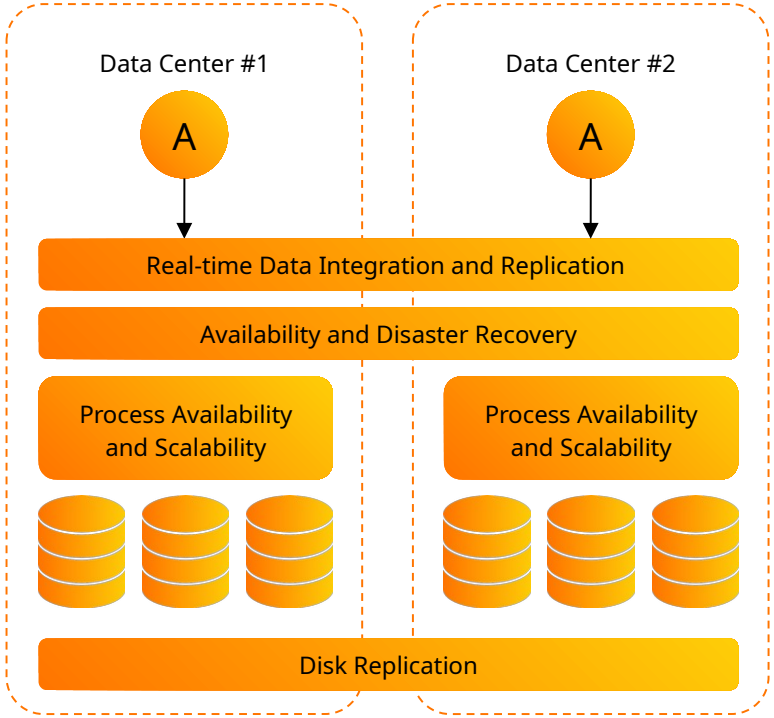
HOT STANDBY



HOT STANDBY, READ REPLICAS

- Standby available for immediate use in case of active data center failure
- Failure or read-only data processing utilization in standby data center

TRADITIONAL DR APPROACHES



ACTIVE - ACTIVE

- Additional Technology
- Configuration
- Coordination
- Complicated & difficult

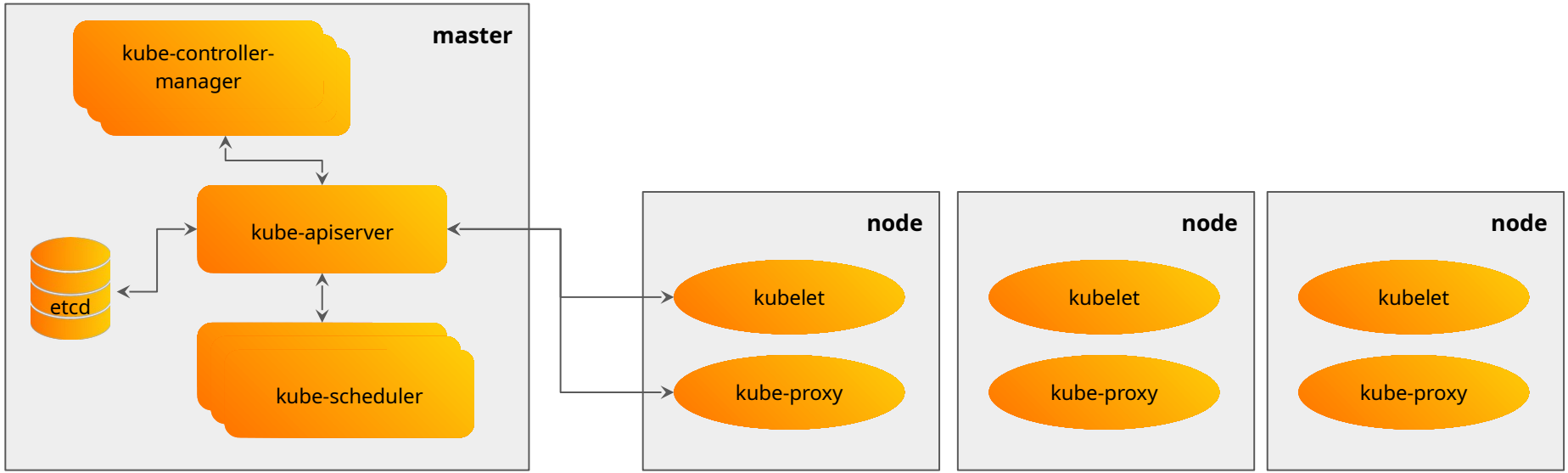
Stakater

CONTAINER ERA

Kubernetes ecosystem

Application not deployed on any fixed server

- Could be scheduled on any of the worker nodes



Kubernetes Backup & Recovery

WHY BACKUP

- Natural or man-made disasters
- Human error
- Hackers, Malware
- Legal standards or compliance regulations

WHAT TO BACKUP

- Namespaces
- Container Images
- Deployment Configurations
- Access Controls
- Certificates
- Persistent Volumes

Analysis

- Stateful Components
- Stateless Components

Stateful components

- ETCD
  - Kubernetes cluster state (kubernetes API objects)
  - Application configurations
- Persistent Volumes
  - Application persistent data (e.g. database, message queue, etc.)

Stateless components

- Rest of Kubernetes control plane
- Kubernetes worker node components

ETCD

- What is ETCD?
- How to backup ETCD?
- How to restore ETCD?

What is etcd?

- key value store used as Kubernetes' backing store
- consistent and highly available
- stores all cluster state and data
  - all kubernetes objects in the cluster

Stakater

How to backup etcd?

Option # 1: Take snapshot of the storage volume

Option # 2: Use built-in snapshot feature of etcd

Option # 3: Backup the kubernetes objects/resources

Restore etcd

- etcd can be restored from snapshots
- recover data of a failed cluster.

Stakater

Persistent Volumes

- CSI Volumes
- Non CSI Volumes

CSI VOLUMES

Stakater

Backup (CSI Volumes)

- Create Volume snapshot from Persistent Volumes

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshot
metadata:
  name: new-snapshot-test
spec:
  volumeSnapshotClassName: csi-hostpath-snapclass
  source:
    persistentVolumeClaimName: pvc-test
```

```
$ kubectl create -f your_snapshot_file.yaml
```

- Some CSI drivers may not have implemented the volume snapshot functionality.

Restore (CSI Volumes)

Provision a new volume using snapshot

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: restore-pvc
spec:
  storageClassName: csi-hostpath-sc
  dataSource:
    name: new-snapshot-test
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
```

Stakater

NON-CSI VOLUMES

Stakater

Backup and Restore (non-CSI Volumes)

Open source solutions such as:

- VMware Velero (formerly Heptio Ark)
- Rook

etc.

RESTORE K8S PLATFORM OPERATIONS

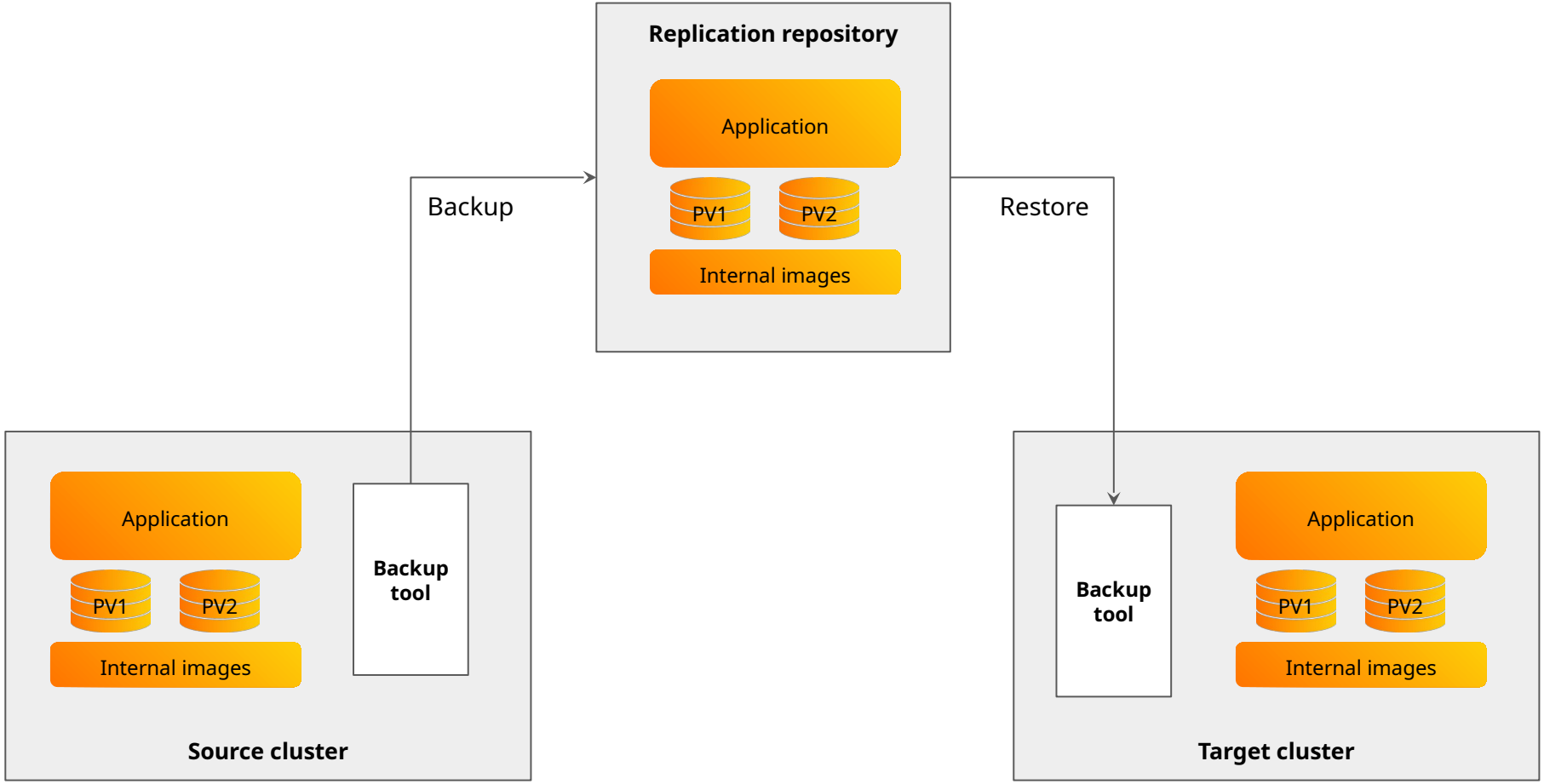
Restore Platform Operations

5 strategies:

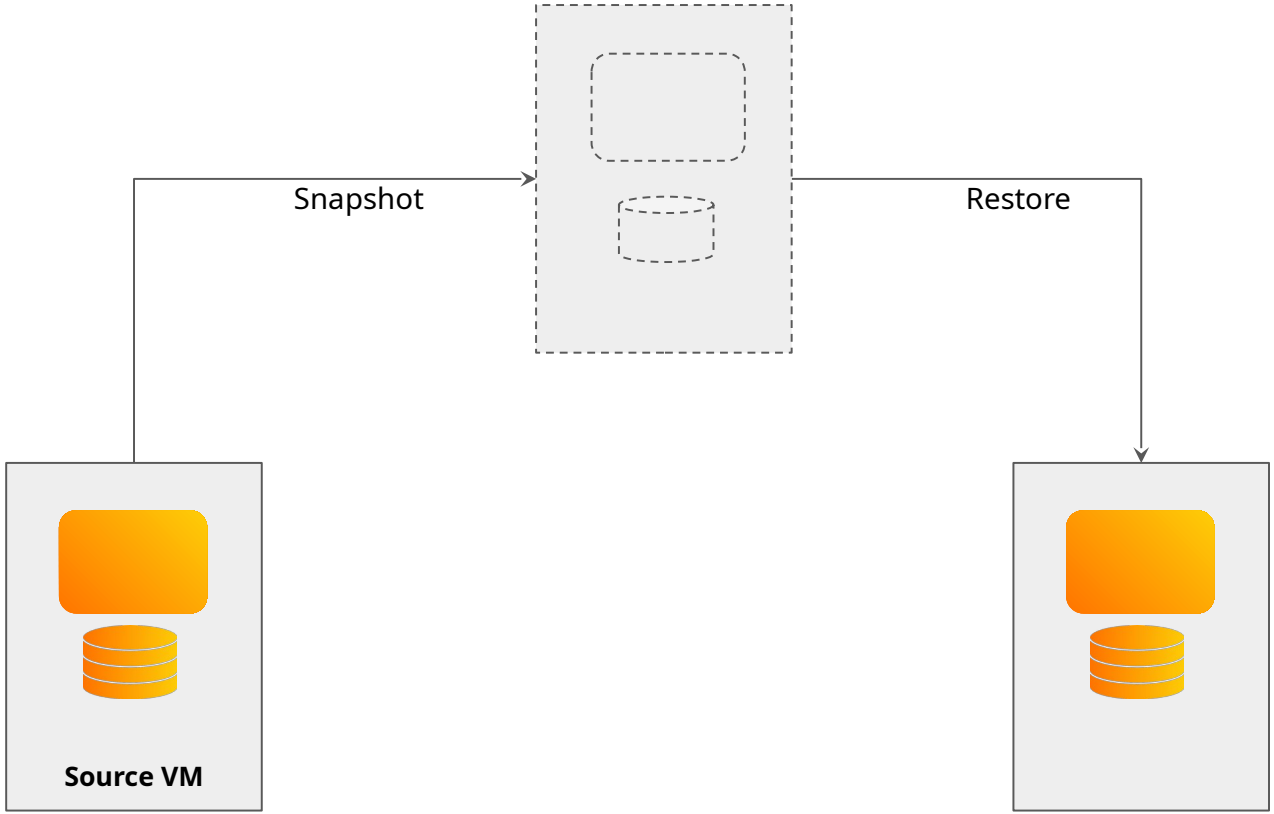
1. Platform backup/restore
2. Restore VMs from a snapshot
3. Failover to another cluster
4. Failover to other site (multi-site)
5. Re-Build from scratch (GitOps)

Option # 1 - Platform Backup/Restore

Stakater



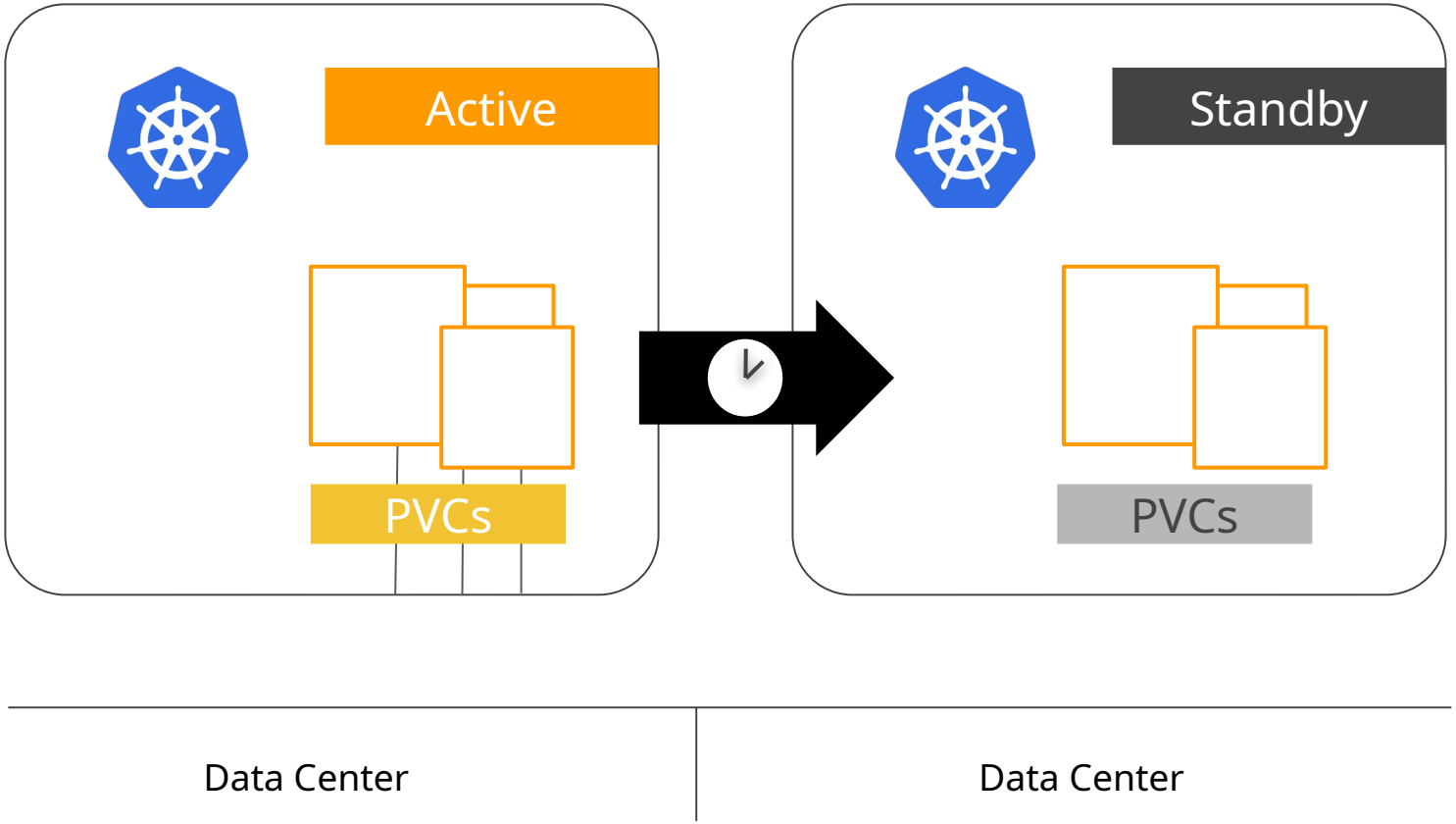
Option # 2 - Restore VMs from a snapshot



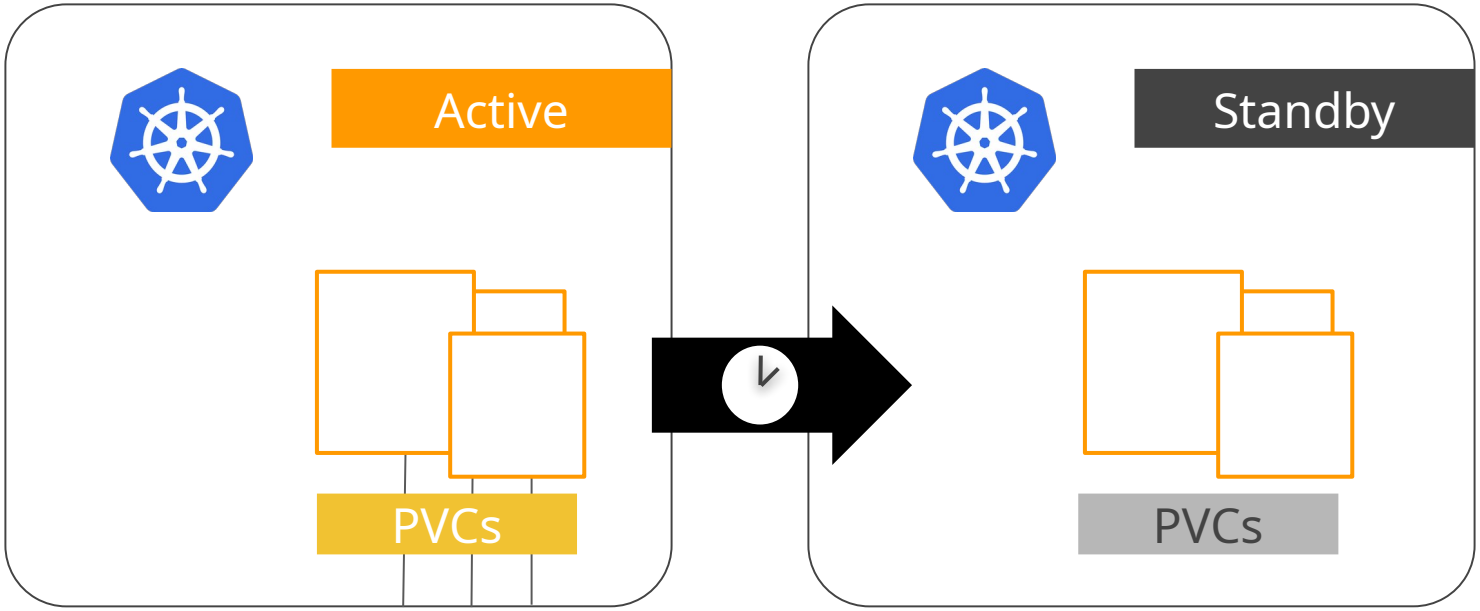
Option # 3 - Failover to another cluster

What is Failover

- In case of failure of one cluster, use a failover cluster
- Both clusters are near identical
  - Infrastructure identical
  - Stateless applications identical
  - Configuration and secrets possibly different
- Clusters can be kept in sync with parallel CI/CD



Option # 4 - Failover to other site (multi-site)



Option # 5 - Rebuild from scratch (GitOps)

Stakater

What is GitOps

Git as a single source of truth for declarative infrastructure

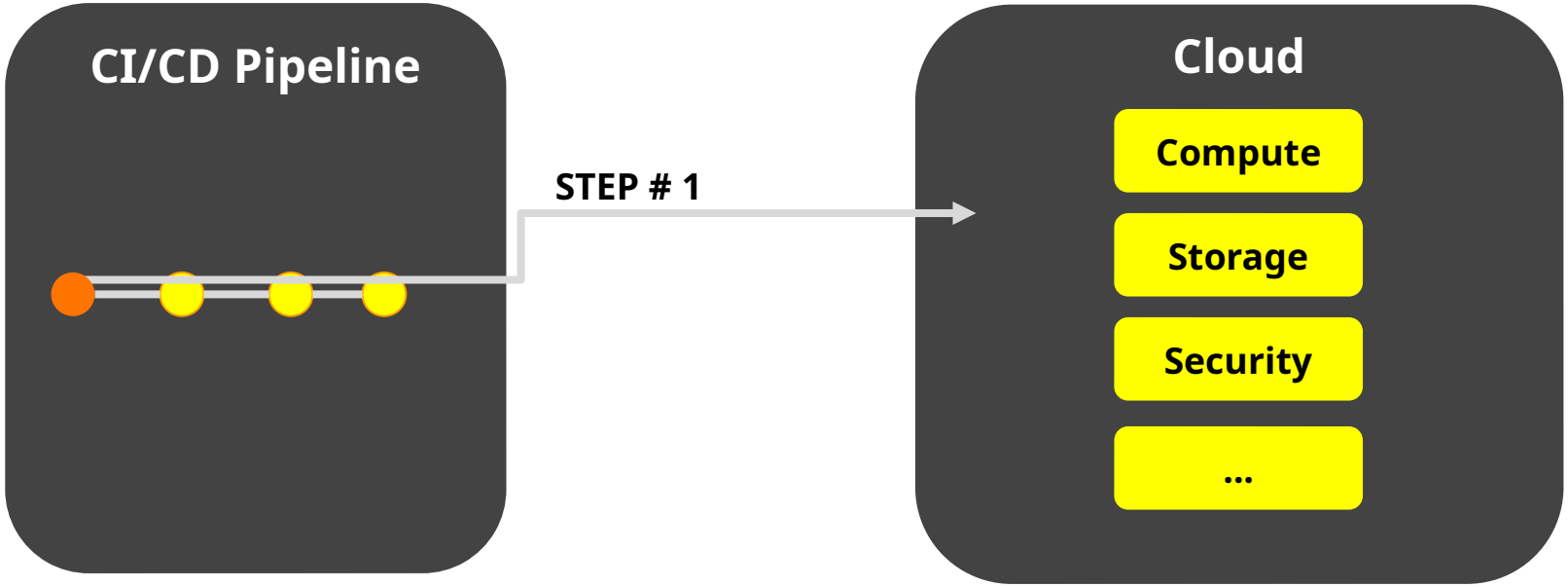
1. Entire system is described declaratively. State is maintained and versioned.
2. Self-healing agents can ensure any manual or erroneous changes are automatically corrected.
3. Observable changes via Git Diffs
4. Changes are approved and applied with Git operations
5. Easy rollbacks via Git revert/rollback/fork

Everything as Code

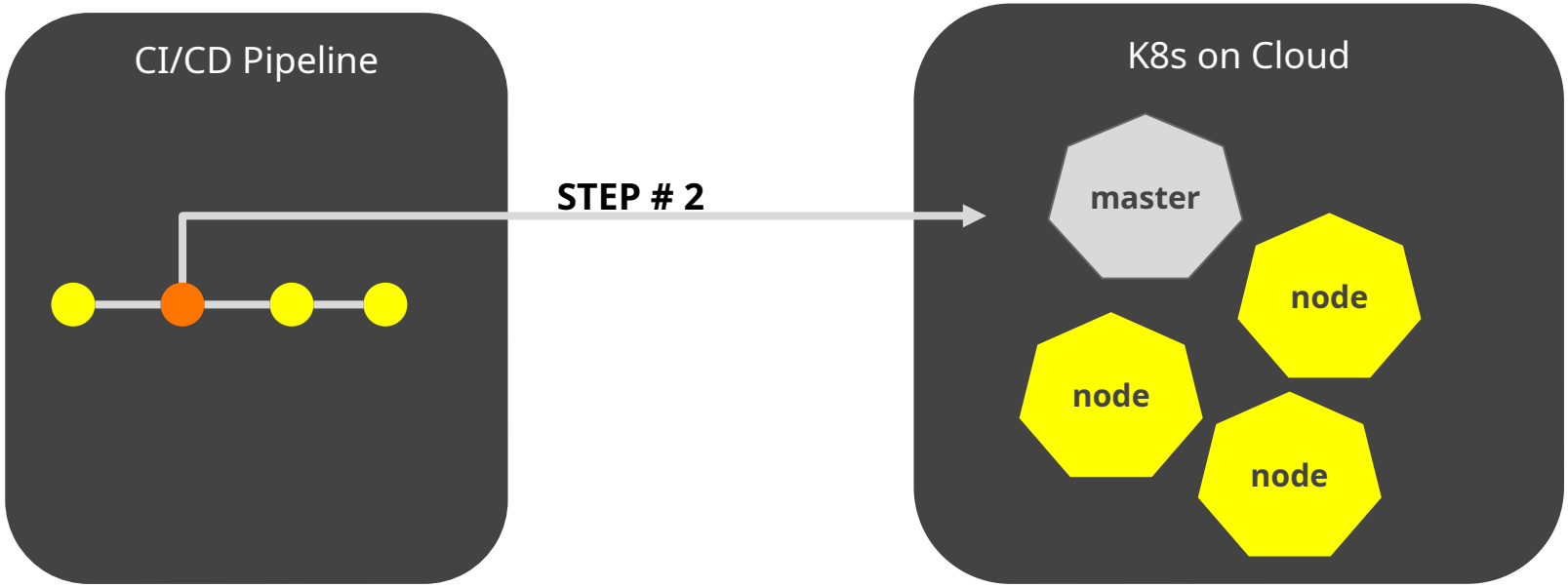
- Infrastructure
  - Cloud configurations
  - VM configurations
  - Kubernetes cluster set up
- Deployments
  - Tools
  - Applications
  - Configuration

Stakater

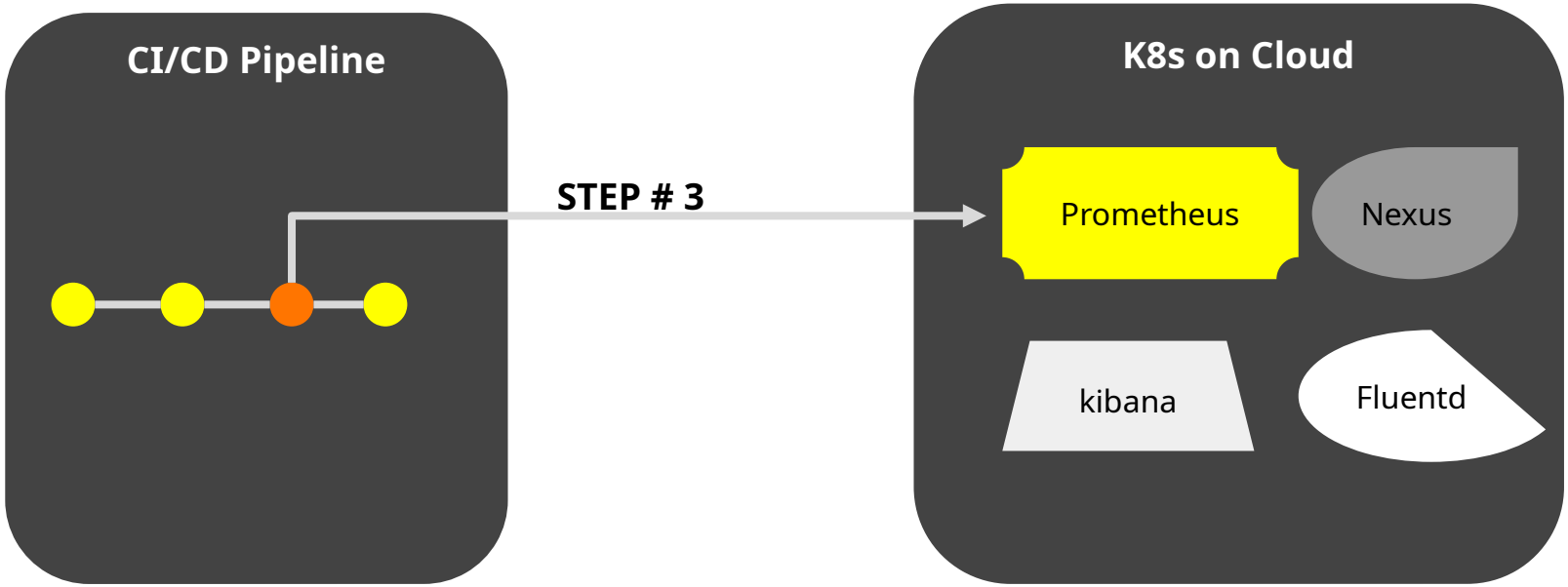
Cloud Infrastructure as Code



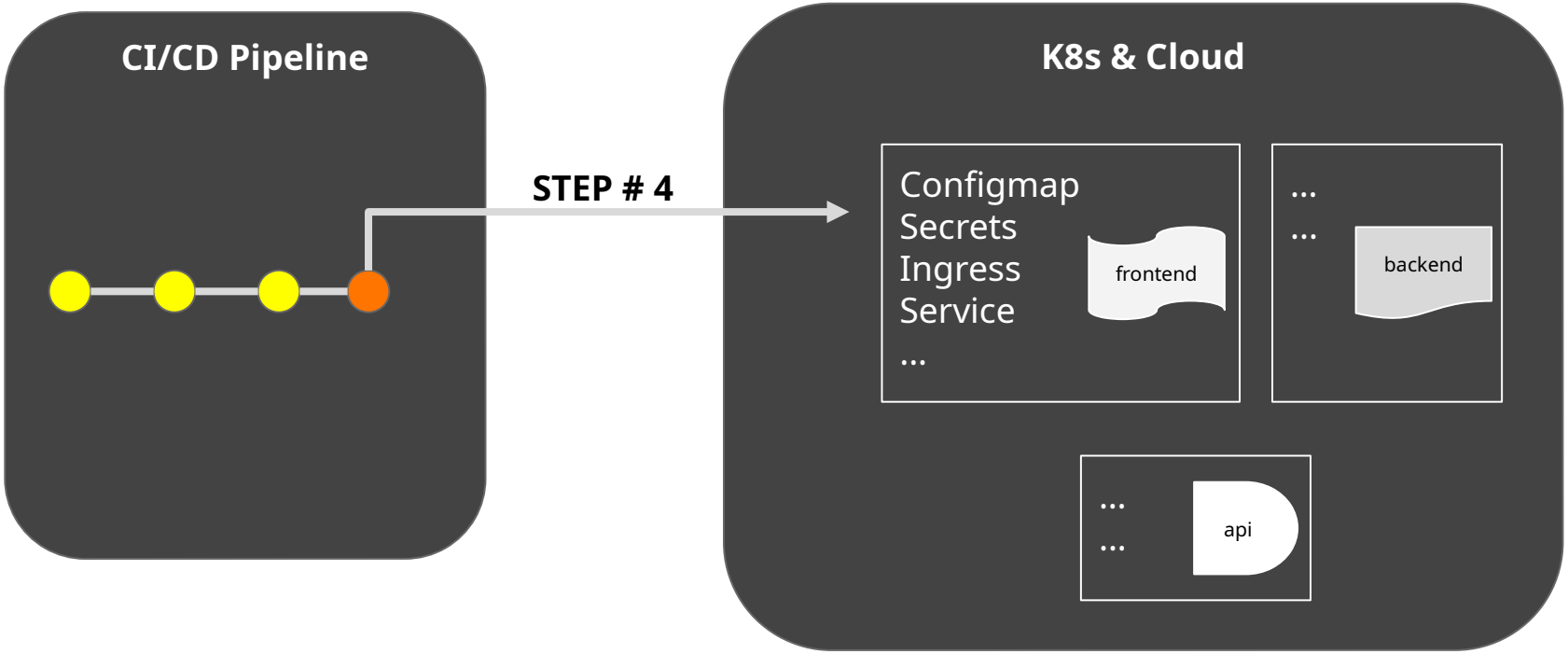
Kubernetes Cluster as Code



Tools as Code



Applications as Code



Summary

Stakater

CONSIDERATIONS

- Plan ahead
- Create backup and recovery procedures to best meet your specific requirements
- Test and rehearse failure + recovery scenarios in non-production environment
- Chaos Engineering tools
- Consider failures on every layer of infrastructure and kubernetes
  - Master (single and multiple)
  - Worker
  - Storage

Thank you!


hello@stakater.com


www.stakater.com


github.com/stakater


medium.com/stakater


youtube.com/channel/UCl9t7syUHPc2XXnuOiimZbg


linkedin.com/company/stakater


Stakater