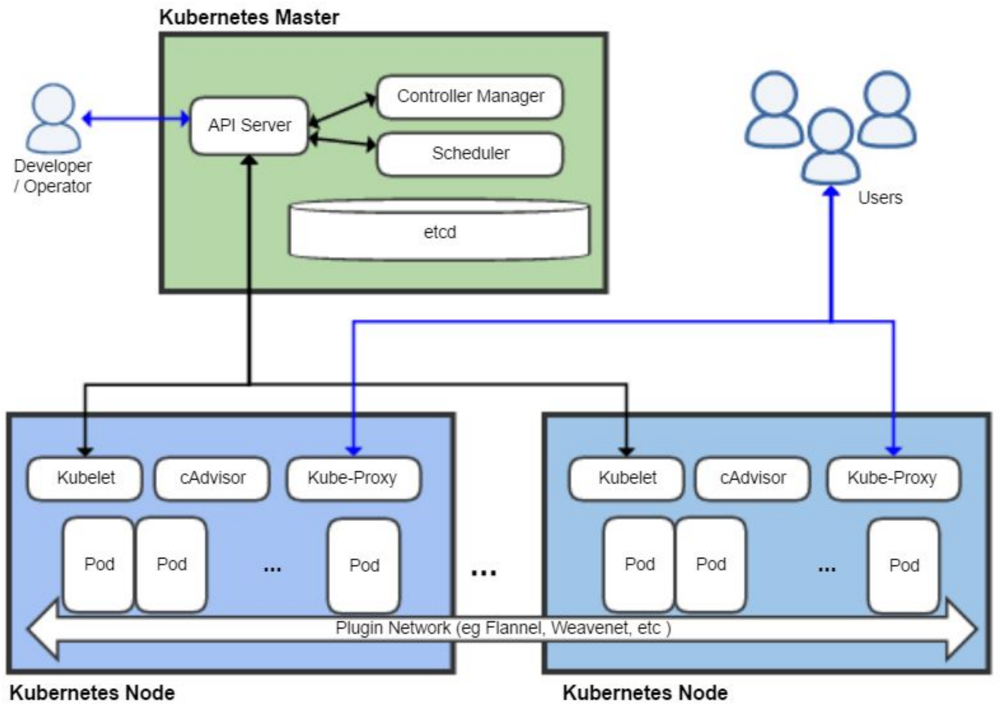
MicroK8s HA under the hood

Kubernetes with Dqlite

Konstantinos Tsakalozos - Senior Engineer

CANONICAL ubuntu

Kubernetes: ground rules



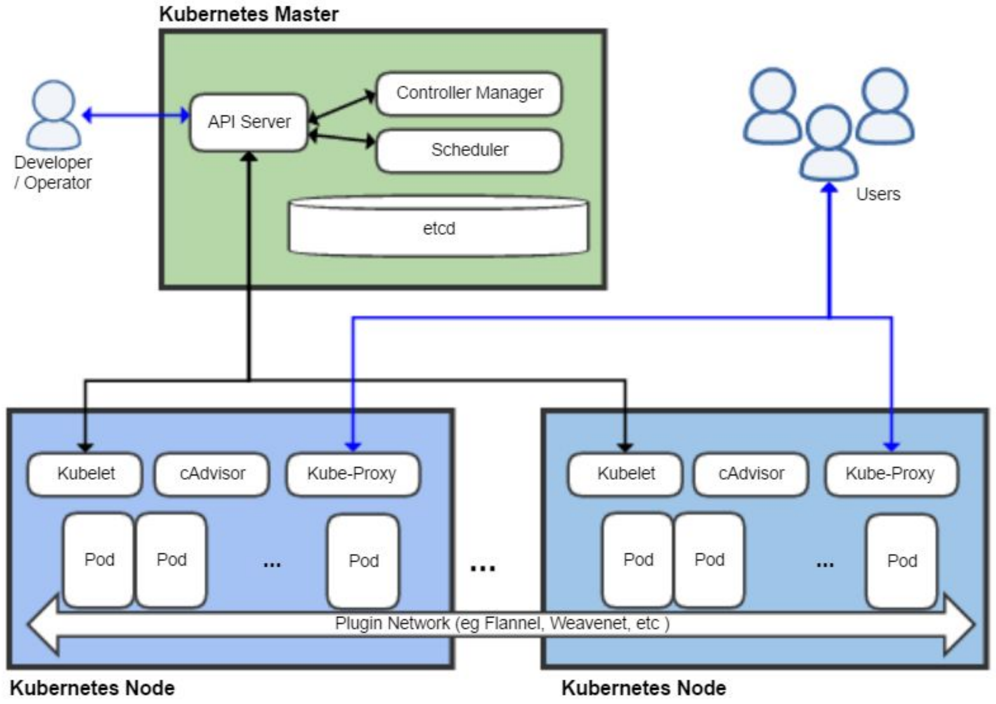
What is High Availability?

1. Elimination of single points of failure
2. Reliable crossover. In redundant systems
3. Detection of failures as they occur

High Availability Perception

For users

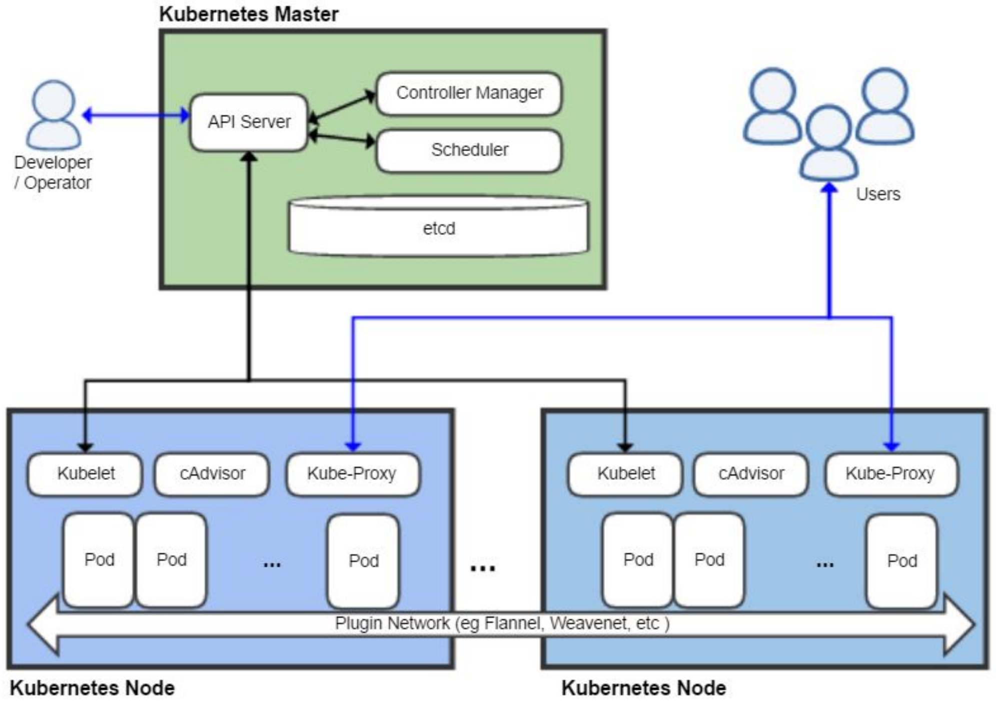
- Services are always available



High Availability Perception

For admins

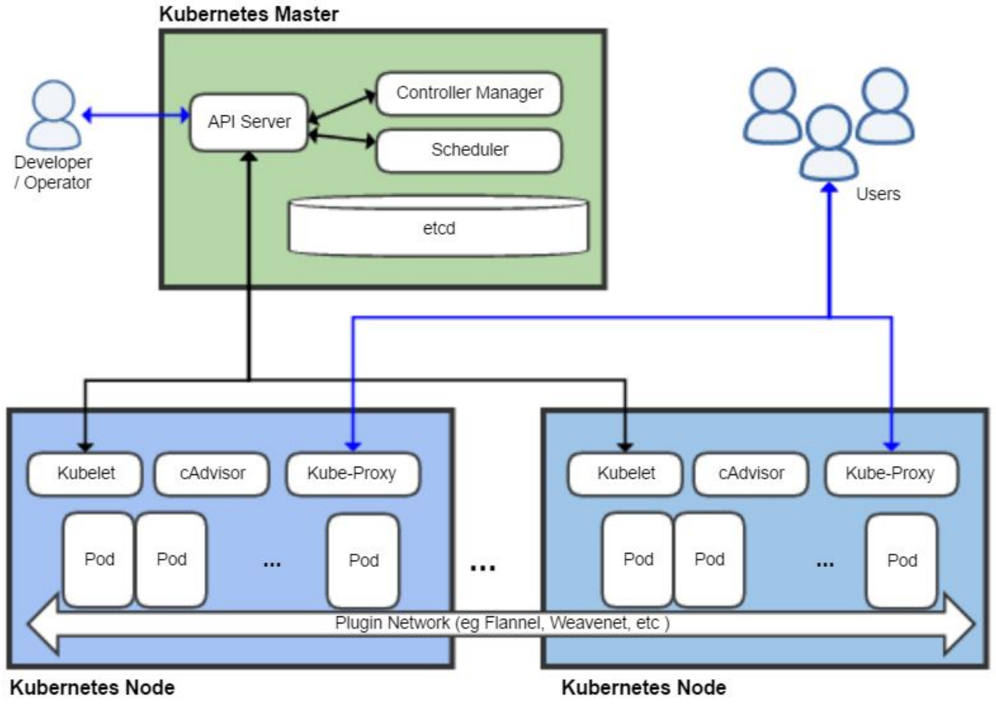
- Control plane is always available
- More than one nodes
- Workloads spread across nodes
- Reliable persistent storage



High Availability Perception

For Kubernetes itself

- Datastore is always available
- Clustering
- Persistent storage configured
- Load balancer floating IPs



MicroK8s is a k8s distribution

We focus on the datastore...

… and achieve much more

Lightweight Kubernetes

- CNCF conformant
- Minimal ops
- Efficient package
- Standalone or clustered
- X86 & ARM
- Edge & IoT
- Opinionated K8s

Focus on security

- Containerised Kubernetes
- Immutable container
- No moving parts, better security, simpler ops
- Automated, controllable updates
- Security patching

$ microk8s enable <features>

Juju

flannel

AMBASSADOR

MULTUS

cilium

K n

Kubeflow

HELM

CoreDNS

Prometheus

MetalLB

fluentd

elastic

Grafana

NVIDIA GPU

LINKERD

Istio

JAEGER

High-availability Zero-ops Self-healing

Zero-ops HA

Stop worrying about the control plane

- Datastore embedded into the API server
- Dqlite: the most popular embedded database made distributed
- At least three nodes needed
- Replication: API server ⇔ datastore

Zero-ops HA

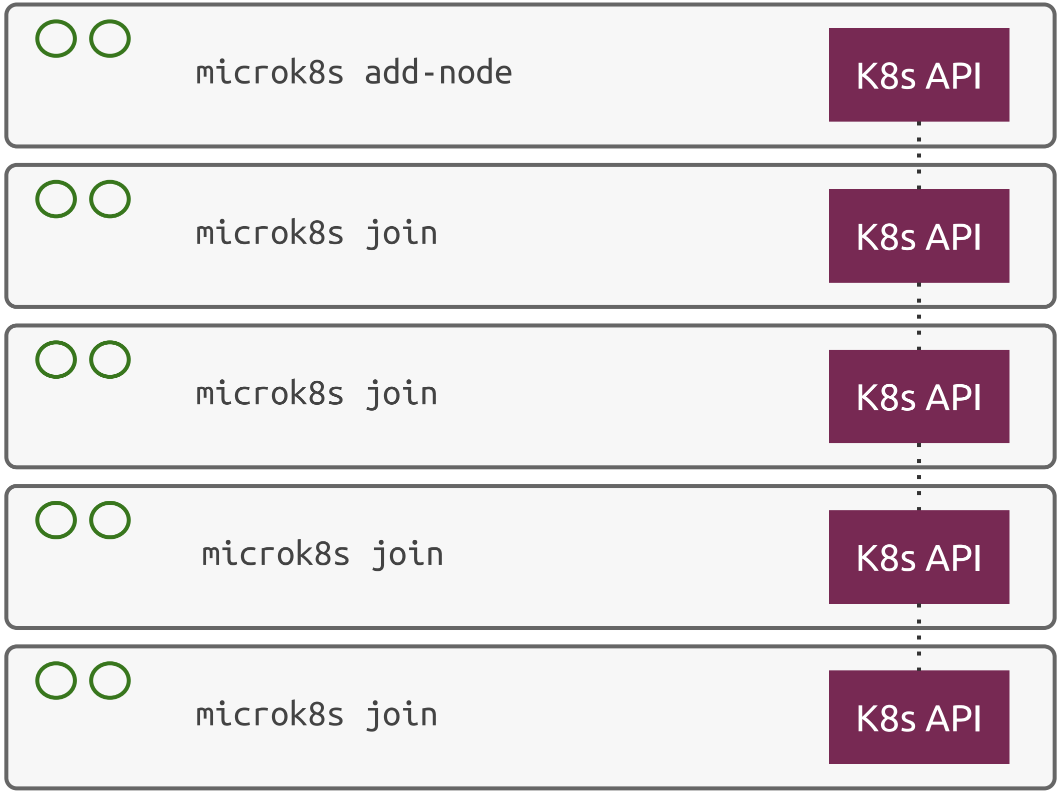
Stop worrying about the workers

- Every node is also a worker
- API server replication ⇔
- Datastore replication
- AND worker replication

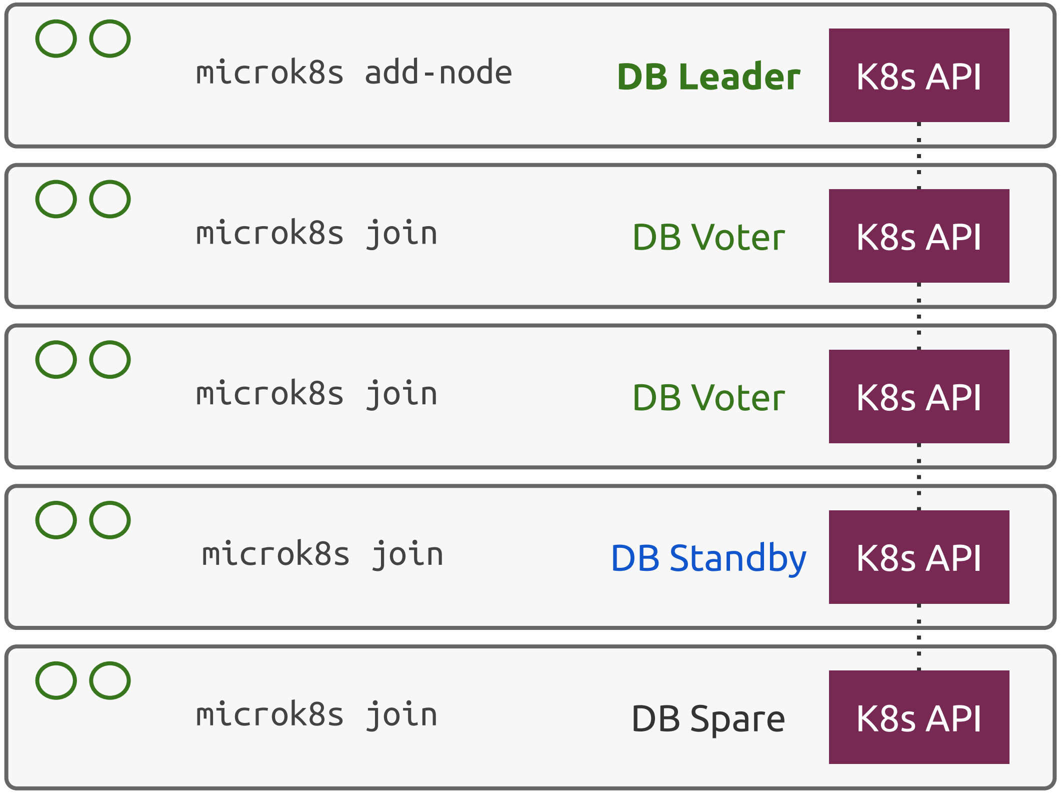
A single command to cluster

```
microk8s join
```

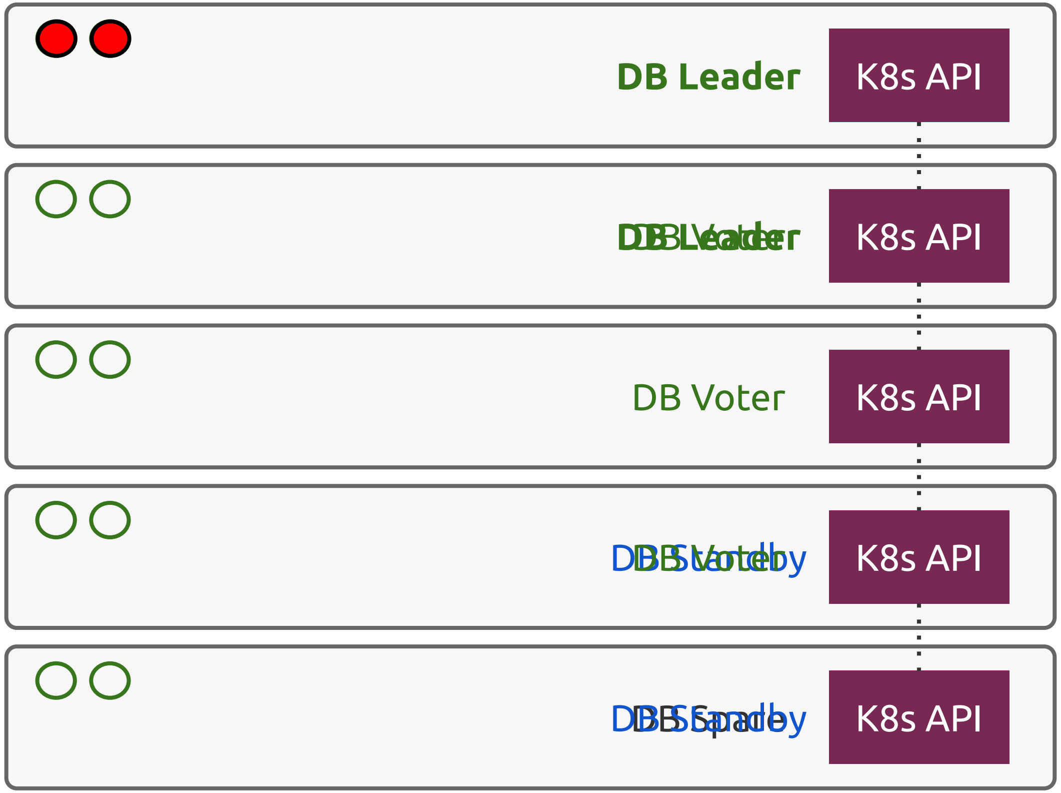
Simple clustering



Zero-ops HA clustering



Self-healing HA cluster



Demo!

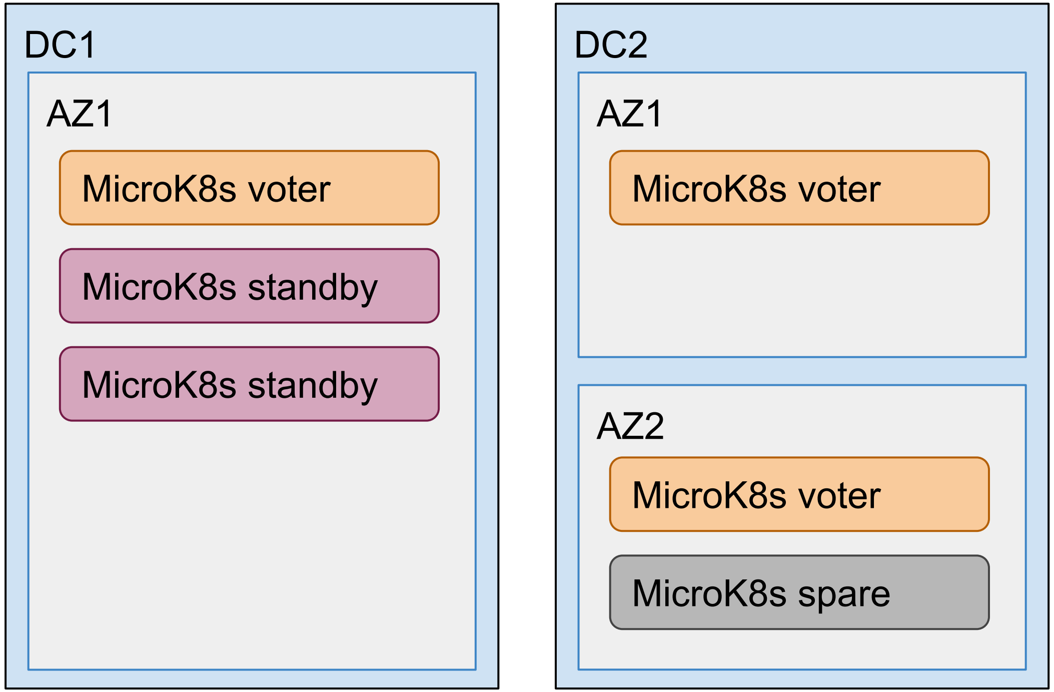
Why dqlite and not etcd?

- Reliability
  - SQLite is the most widely used DB
  - A very well understood distributed SQLite
  - Perfect for embedded devices
- Frictionless
  - Transparent operations
  - No DBadm needed
- Ownership
  - Long term performance gains

Autonomous High Availability

- At least three nodes
  - Two nodes stand-by
  - Spare
- Extra nodes
  - One leader
  - Two voters
- Node role transitions happen within seconds from node failure

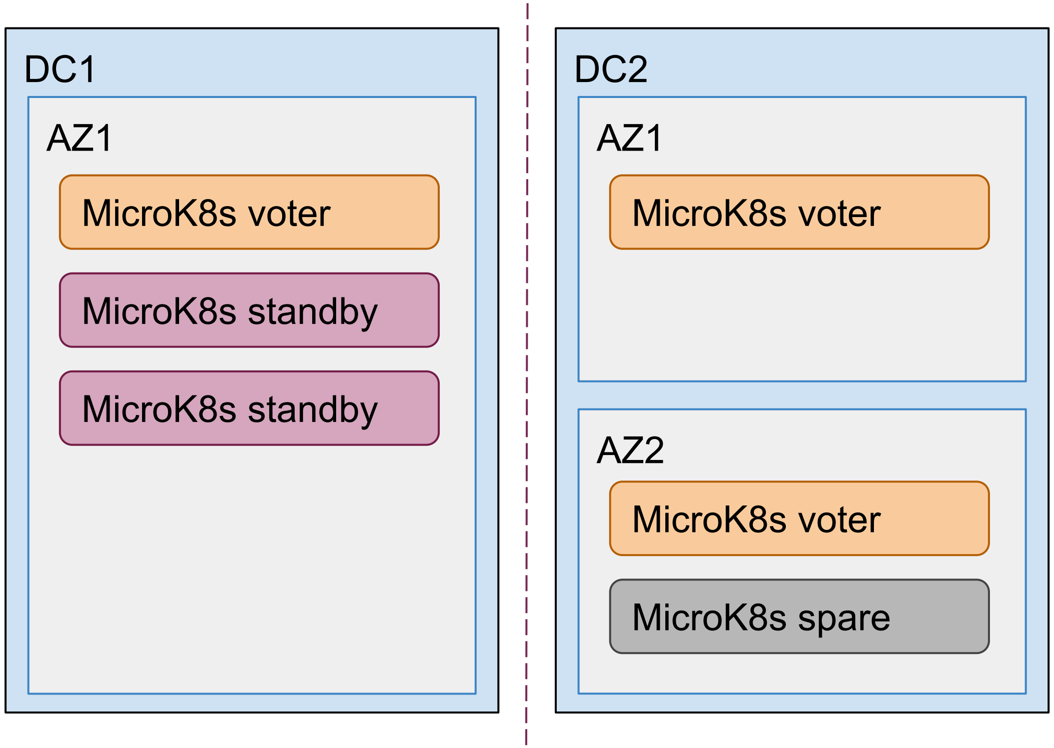
What happens if...



DC1 and DC2 get disconnected

If leader is on DC1

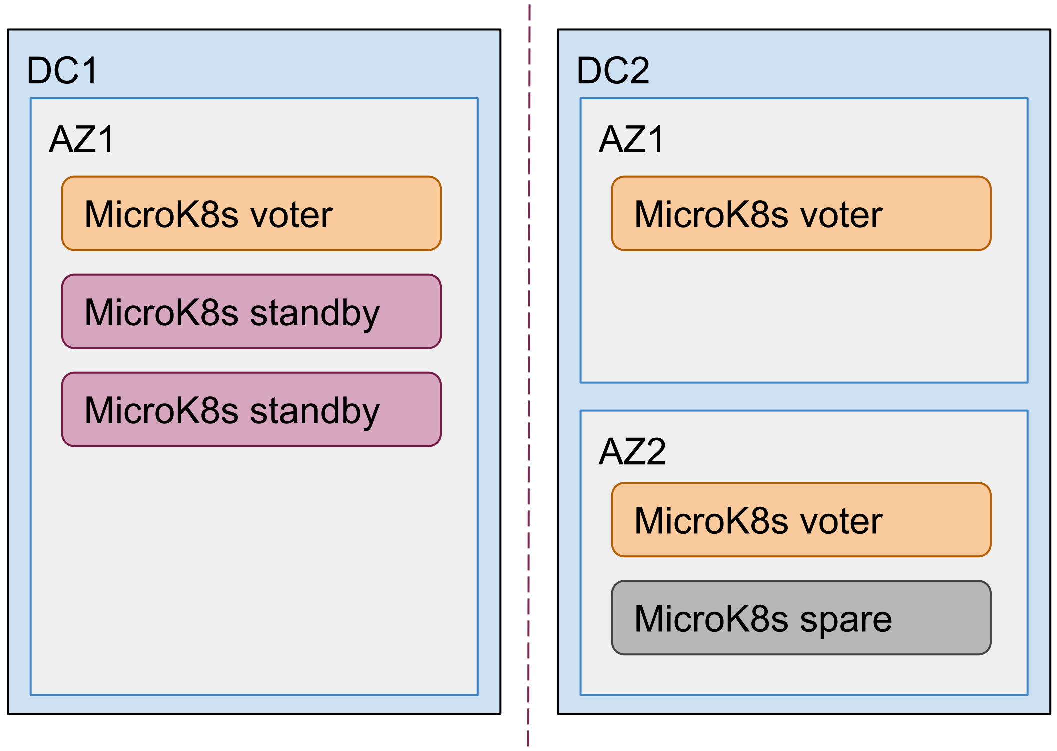
- The leader will step down because he lost majority
  - in ~ 1 second
- Voters on DC2 vote for a new leader
- DC1 freezes
- Spare node on DC2 becomes voter



DC1 and DC2 get disconnected

If leader is on DC2

- No election needed
- DC1 freezes
- Spare node on DC2 becomes voter



What’s next?

- Failure domains
  - FD-aware deployments
  - Spread voters across FDs
- Weighted voter placement
  - Hints for dqlite-hosting candidate nodes
- Performance improvements
  - CPU and memory footprint

Resources

MicroK8s GitHub: github.com/ubuntu/microk8s

MicroK8s web: microk8s. io

#microk8s channel on slack.kubernetes.io

Snaps web: snapcraft.io

Charmed Kubernetes: ubuntu.com/kubernetes/docs

Cool K8s and Ubuntu demos on YouTube: youtube.com/celebrateubuntu/

Try MicroK8s today!

www.microk8s.io

CANONICAL ubuntu