Kubermatic

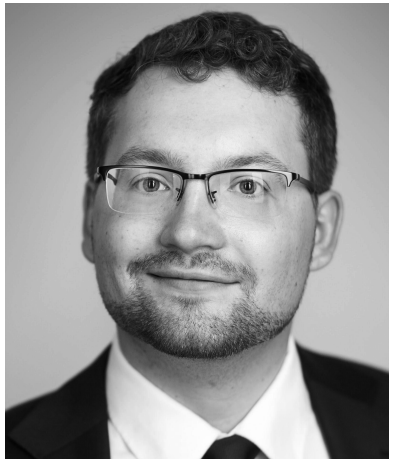
KubeCon | CloudNativeCon North America 2020 Virtual

A Future Journey:

How to Migrate 100 Clusters Between Clouds Without Downtime?

Tobias Schneck
Head of Professional Service

tobi@kubermatic.com
@toschneck
toschneck



Manuel Stößel
Systems Architect / Tech Lead

manuel@kubermatic.com
@Manuel_Stoessel
@ManuStoessel

What Else?

- Part of Professional Services @ Kubermatic
- Supporting customers on their cloud-native journey
- Geeking out over Kubernetes and adjacent technologies

Kubermatic

Why migrate clusters?

Reasons for cluster migration scenarios

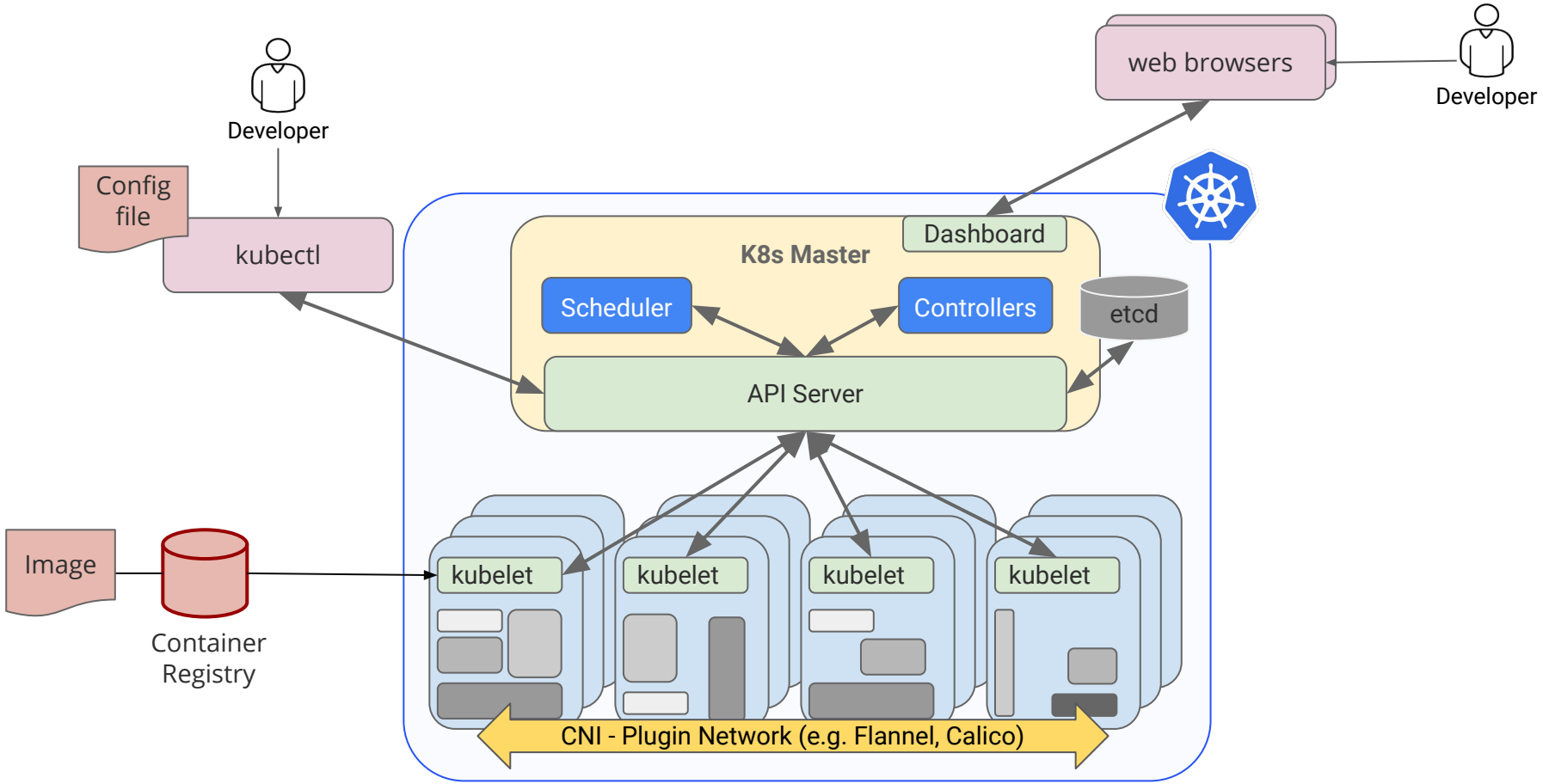
- Business Reasons
  - Better contract/conditions at another cloud provider ⇒ cost saving
  - Data center migration to/from (cloud) providers
  - Multi cloud strategy ⇒ decrease dependency to existing provider
- Technical Reasons
  - Location migration of data centers
  - Migrate to another network segments
  - Adaption of on-prem / cloud improvements at new data center provider
  - Data location of cloud offered service e.g. machine learning data

Kubermatic

What are the main challenges?

Kubernetes abstracts infrastructure, but:

- Consummation of infrastructure resources
  - (Virtual) Machines
  - Network:
    - Network IP Address Spaces
    - Routing, Firewall
    - Ingress / Egress Traffic
  - DNS
  - External Storage Systems
- Cloud dependent Kubernetes components
  - Cloud Controller Manager
    - Node controller - responsible for updating kubernetes nodes
    - Service controller - responsible for services of type `LoadBalancer`
    - Route controller - responsible for setting up network routes
  - Storage Classes
  - (sometimes) Overlay Networking


Developer
Config file
kubectl
web browsers
Developer
Dashboard
K8s Master
Scheduler
Controllers
etcd
API Server
Image
Container Registry
kubelet
kubelet
kubelet
kubelet
CNI - Plugin Network (e.g. Flannel, Calico)

Migration without downtime

⇒ Application workload has the highest priority!

- Ensure fundamental networking rules at any time
  - All containers within a pod can communicate (L4) with each other unimpeded.
  - All pods can communicate with all other pods without NAT.
  - All nodes can communicate with all pods (and vice-versa) without NAT.
  - The IP that a pod sees itself as is the same IP that others see it as.
- External dependencies need to be reachable
  - External routed IPs for Load Balancers / Node Port Service
  - DNS Names need to be reachable
- Storage
  - State needs to migrated without data loss

Scale Level of 100 clusters

- Larger organizations running a lot of clusters
  ⇒ different locations, org units, time zones

- Cluster users are only consumers
  ⇒ following the cluster as a service approach

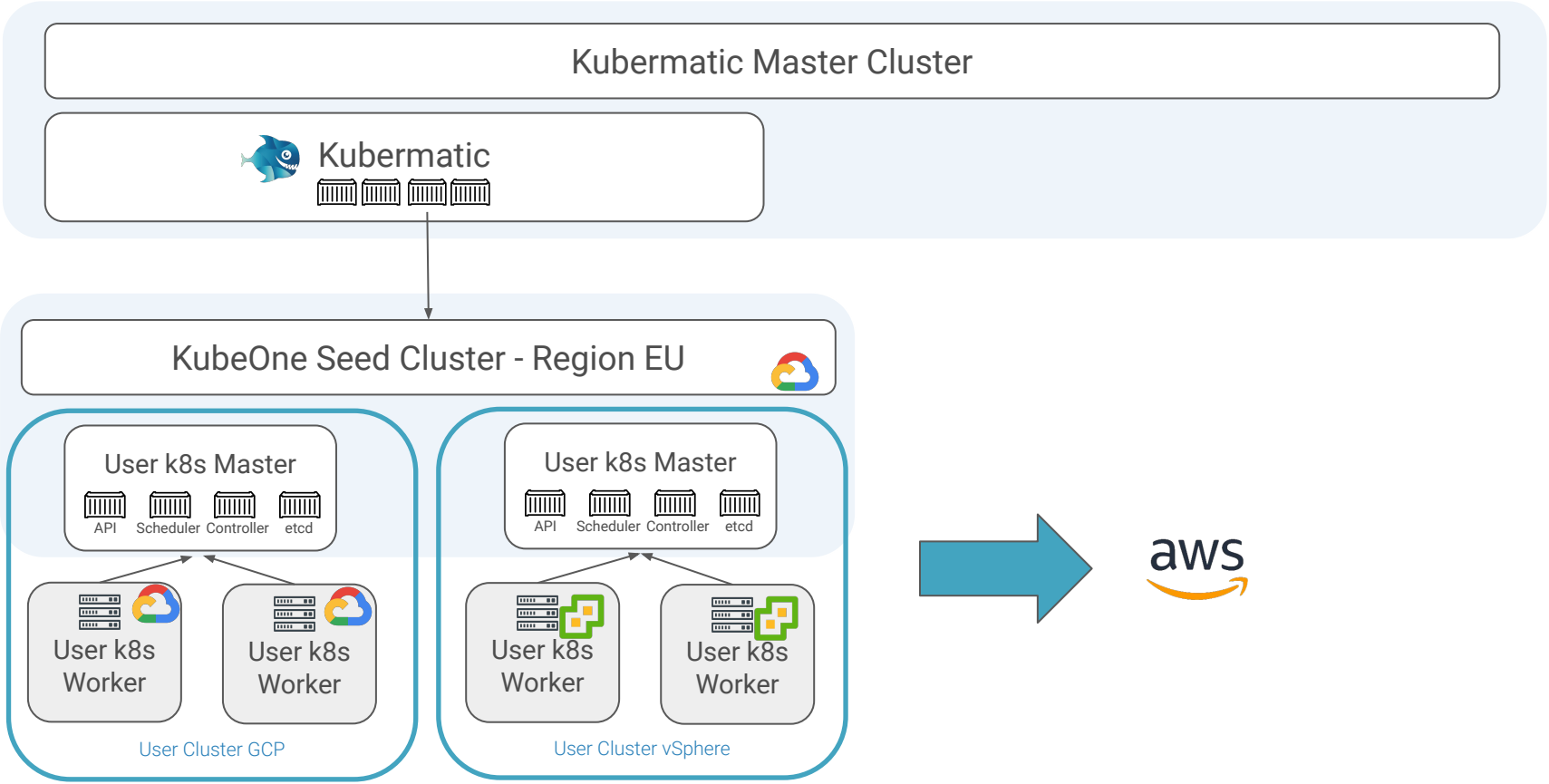
- Cluster connection and secrets needs to be stable
  => no change of interface

Kubermatic

Solution Approach

Status Quo

- Multi Cloud Setup with Kubermatic Kubernetes Platform (KKP)
  - Seed cluster hold containerized control plane of user clusters
  - Worker nodes provisioned by Cluster API conform Kubermatic machine-controller
  - Canal as default overlay network
- Target
  - Migrate user and seed cluster control planes and worker to different cloud
  - Keep external Cluster Endpoints stable
    - Control Plan: Kubernetes API Server endpoints
    - Application: DNS, Ingress
  - Out-of-Scope (for now): Storage replication
    - Assumption: Application Layer manages storage replication, e.g. etcd


Kubermatic Master Cluster
Kubermatic
KubeOne Seed Cluster - Region EU
User k8s Master
API
Scheduler
Controller
etcd
User k8s Worker
User k8s Worker
User Cluster GCP
User k8s Master
API
Scheduler
Controller
etcd
User k8s Worker
User k8s Worker
User Cluster vSphere
aws

Recommended Prerequisites

- Announce maintenance window and block cluster updates
- Ensure backups and recovery procedure for
  - Seed and user clusters
  - Application workload
- Create target cloud cluster as reference
- Ensure control of DNS entries

Kubermatic

Solution Approach

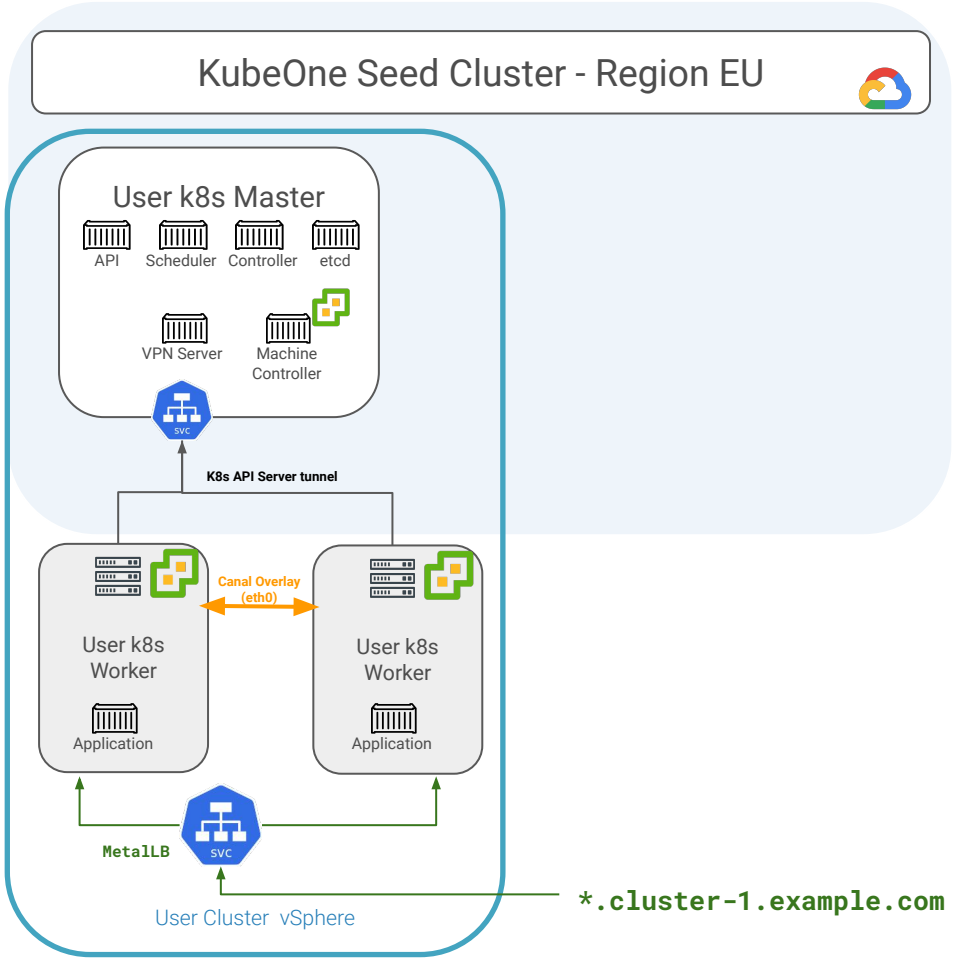
Migrate User Clusters

1) Migrate User Cluster Workers

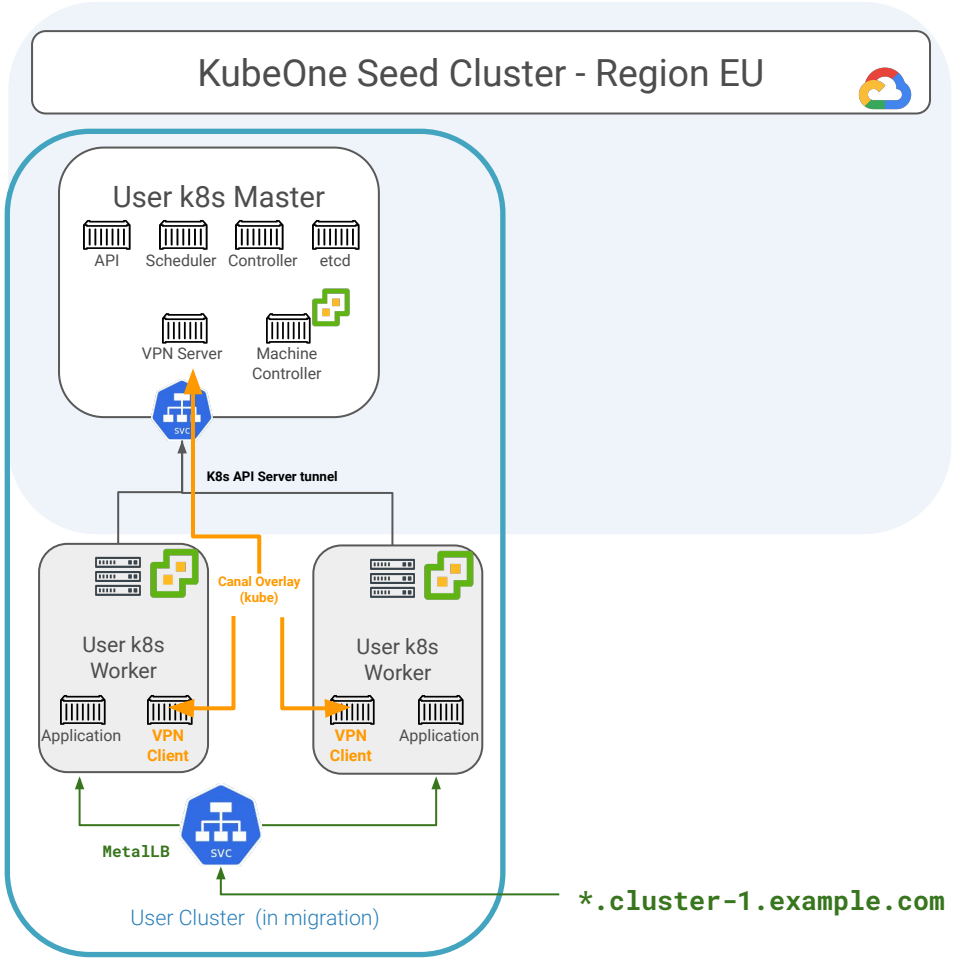
- Create new worker nodes in target cloud
  ⇒ Machine controller with **new Machine Deployment at target cloud**

- User worker nodes and Pods need to talk to each other at any time
  ⇒ Strap a **VPN overlay by DaemonSets** across current and target cloud
  ⇒ Route overlay **CNI traffic through VPN** network

- Ensure reachability
  => Keep old and create new cluster **Ingress endpoints**
  => **Transfer workload to new cloud**
  => **Delete after** workload / connectivity is ensured



Migrate User Cluster Worker Nodes:

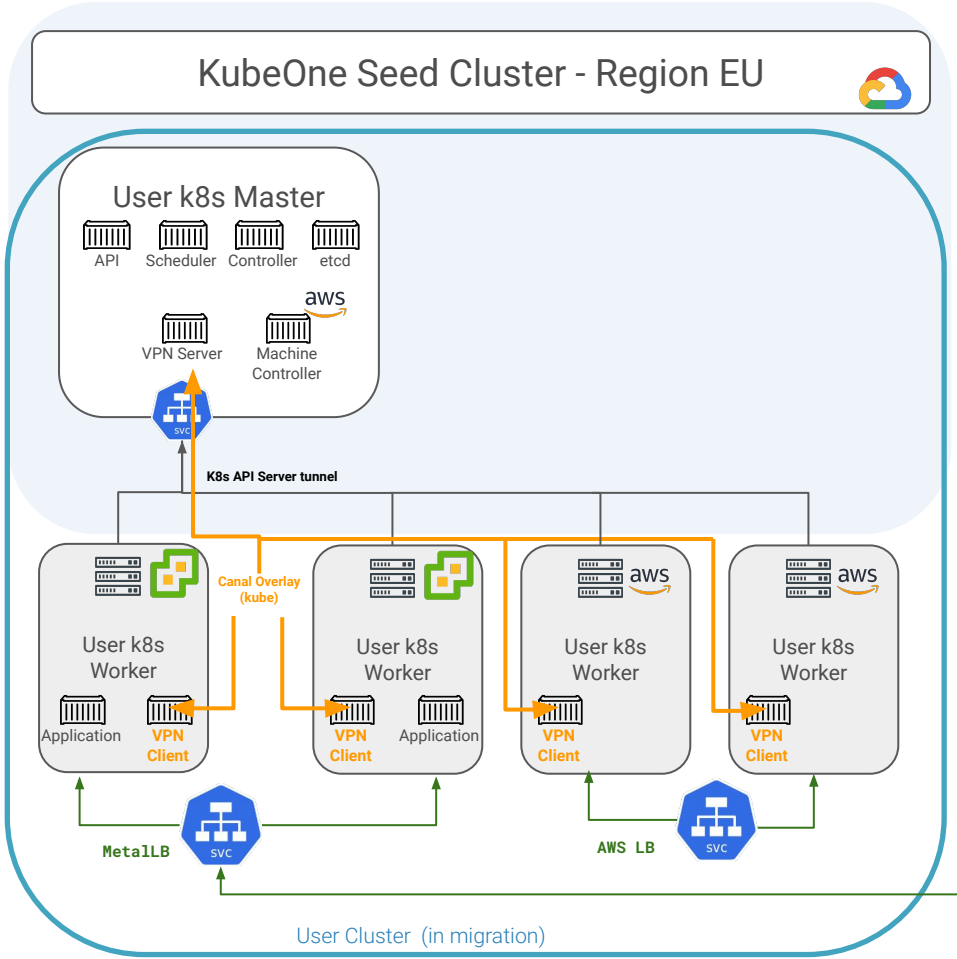


Migrate User Cluster Worker Nodes:

1. VPN Daemon Set with client-to-client communication

2. Route Overlay Traffic over VPN interface

3. Pause existing Cluster & Machine Deployment



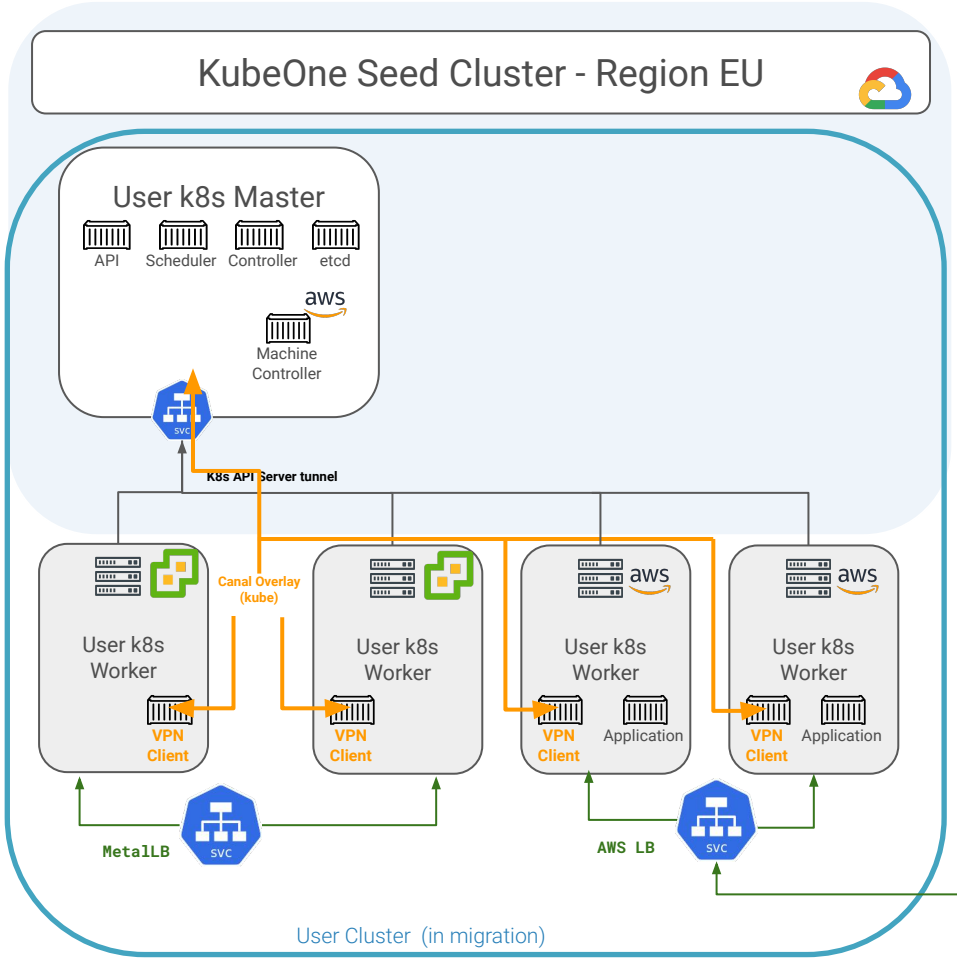
Migrate User Cluster Worker Nodes:

1. VPN Daemon Set with client-to-client communication
2. Route Overlay Traffic over VPN interface
3. Pause existing Cluster & Machine Deployment

4. Update Cluster Spec & Cloud Credentials

5. Unpause Cluster with new Cloud Provider

6. Apply new Machine Deployment

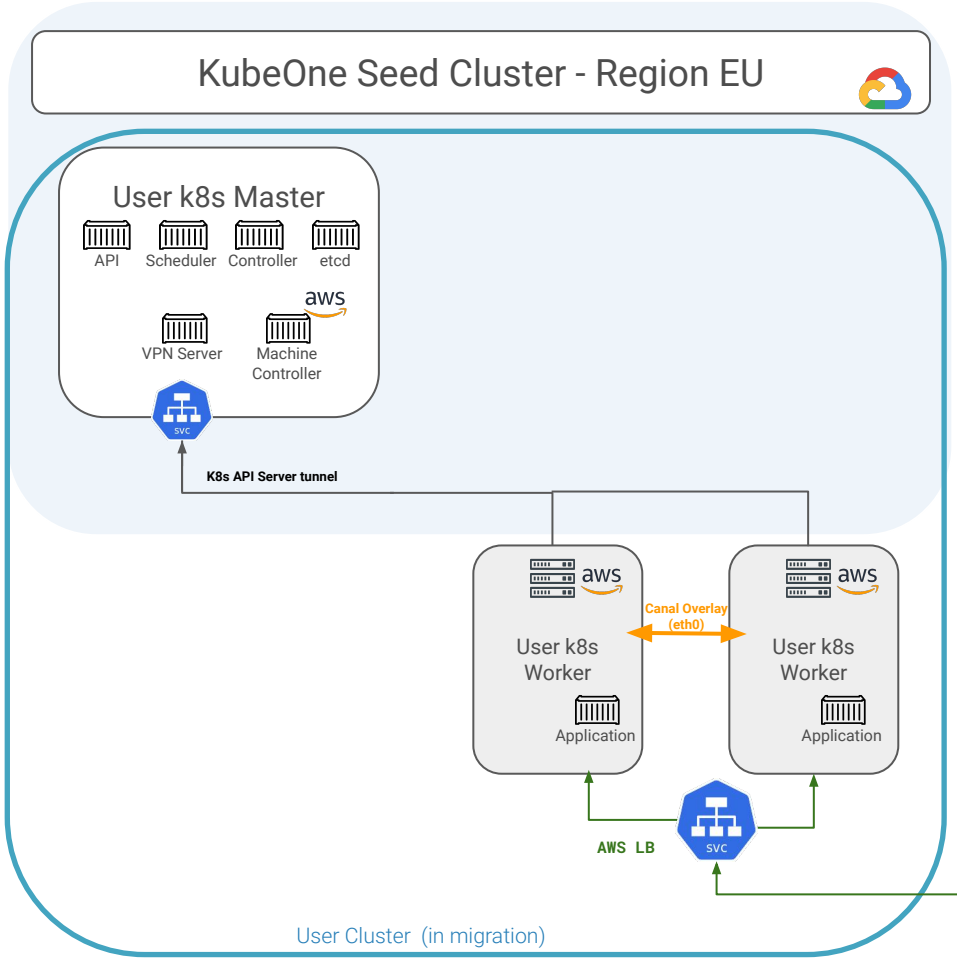


Migrate User Cluster Worker Nodes:

1. VPN Daemon Set with client-to-client communication
2. Route Overlay Traffic over VPN interface
3. Pause existing Cluster & Machine Deployment
4. Update Cluster Spec & Cloud Credentials
5. Unpause Cluster with new Cloud Provider
6. Apply new Machine Deployment

7. Test new cluster ingress entrypoint

8. Migrate Workload and update DNS



Migrate User Cluster Worker Nodes:

1. VPN Daemon Set with client-to-client communication
2. Route Overlay Traffic over VPN interface
3. Pause existing Cluster & Machine Deployment
4. Update Cluster Spec & Cloud Credentials
5. Unpause Cluster with new Cloud Provider
6. Apply new Machine Deployment
7. Test new cluster ingress entrypoint
8. Migrate Workload and update DNS

9. Cleanup old cloud resource

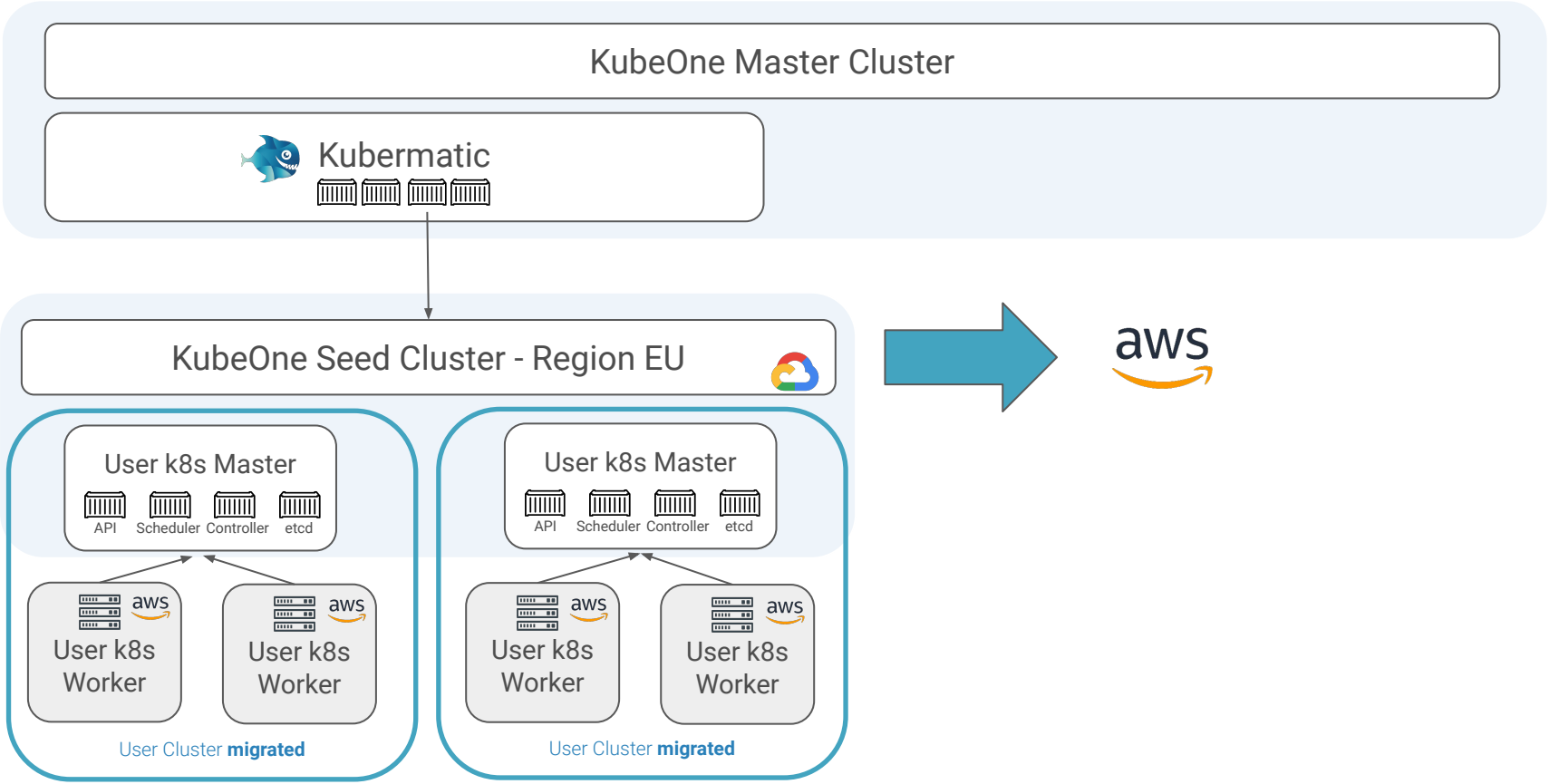
DEMO Time 

github.com/kubermatic-labs/cluster-migration

Kubermatic

Solution Approach

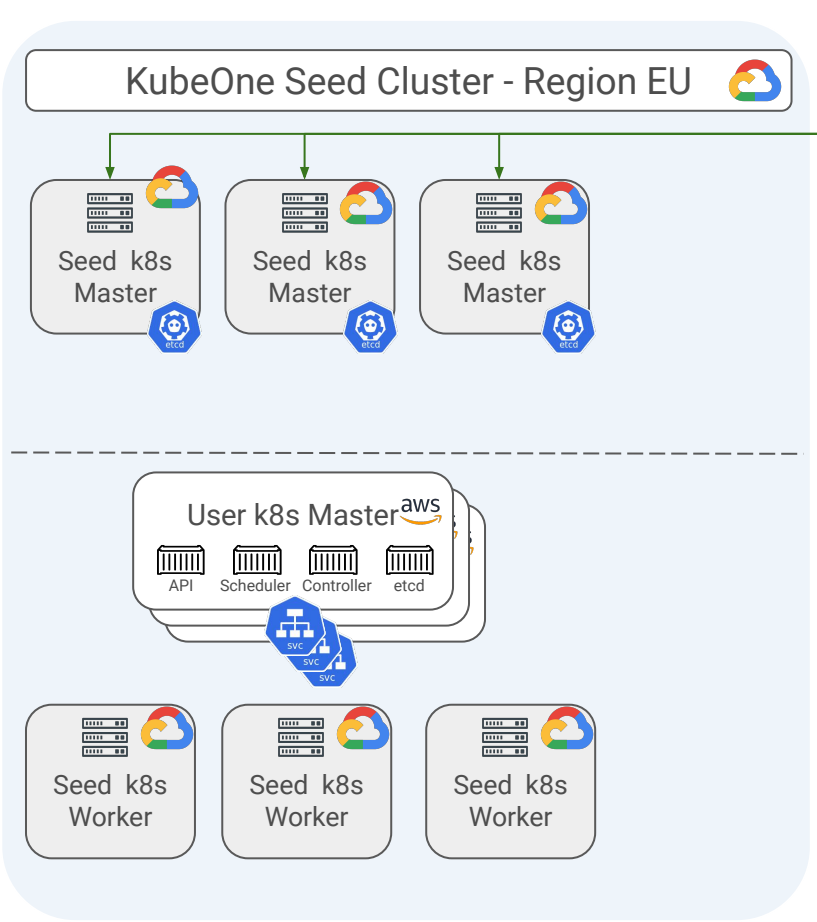
Migrate Seed Cluster


KubeOne Master Cluster
Kubermatic
KubeOne Seed Cluster - Region EU
aws
User k8s Master
API
Scheduler
Controller
etcd
User k8s Worker
User k8s Worker
User Cluster migrated
User k8s Master
API
Scheduler
Controller
etcd
User k8s Worker
User k8s Worker
User Cluster migrated

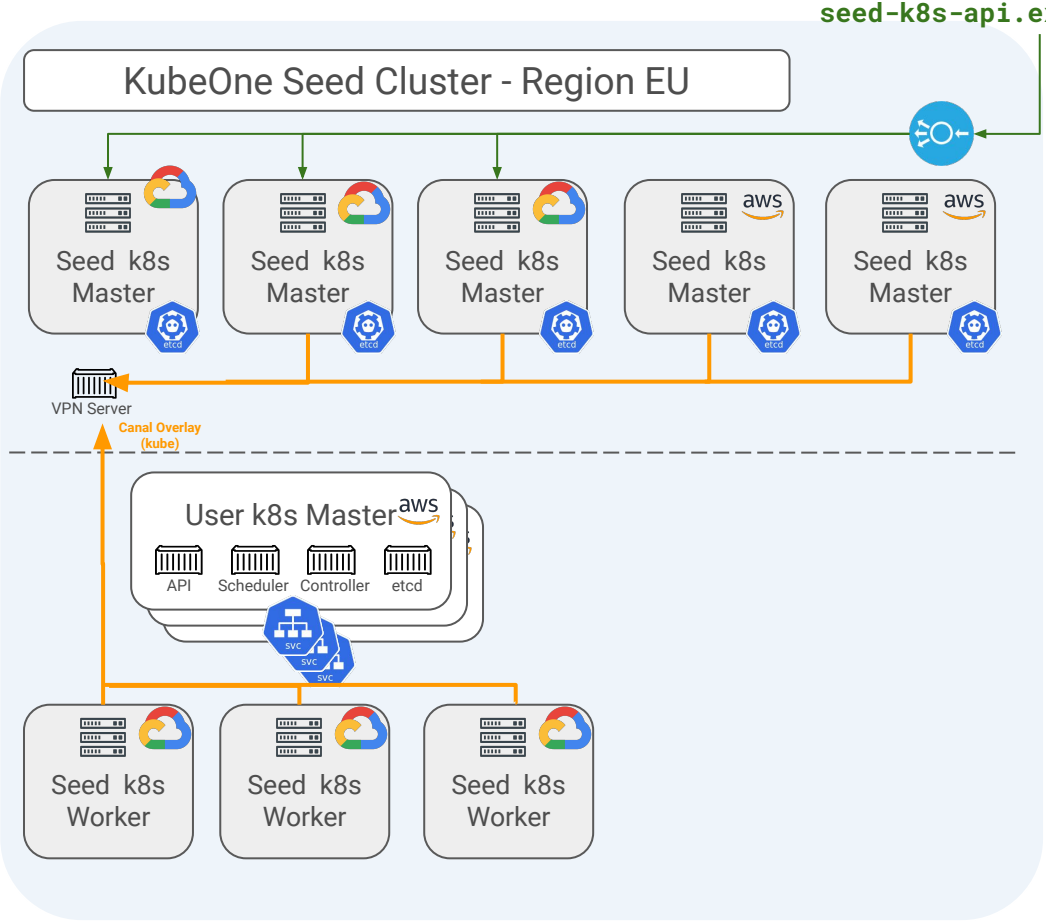
2) Migrate Seed Cluster

- Create new seed master nodes at new cloud
  => New Kubernetes API Load Balancer
  => API Endpoint needs to be updated by DNS
  => Block seed cluster upgrades to ensure worst case recovery

- Migrate user cluster control plane
  => Handle migration the same way (like user cluster workload)
  => Ensure etcd quorum and migration by data replication
  => Block user cluster upgrades to ensure worst case recovery



Migrate Seed Master Nodes:

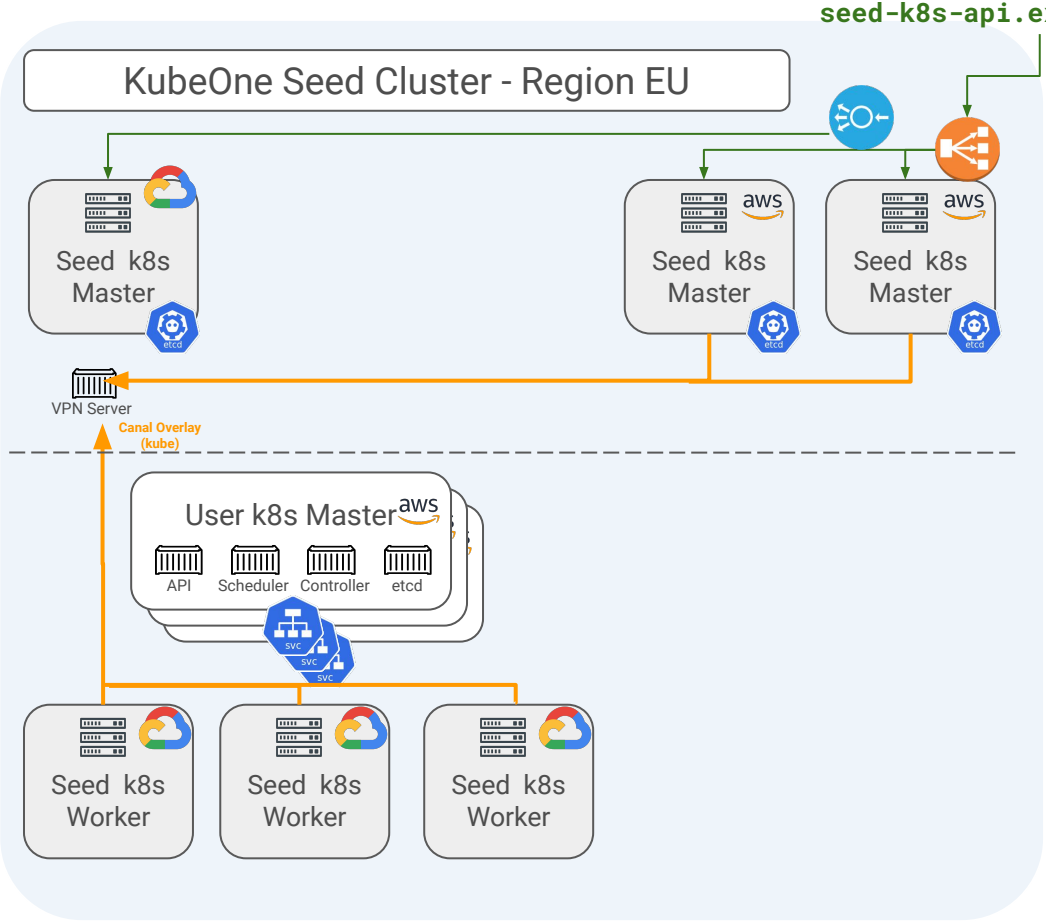


Migrate Seed Master Nodes:

1. Setup VPN Overlay

2. Pause existing Cluster & Machine Deployment

3. Create and join new 2 Master Nodes

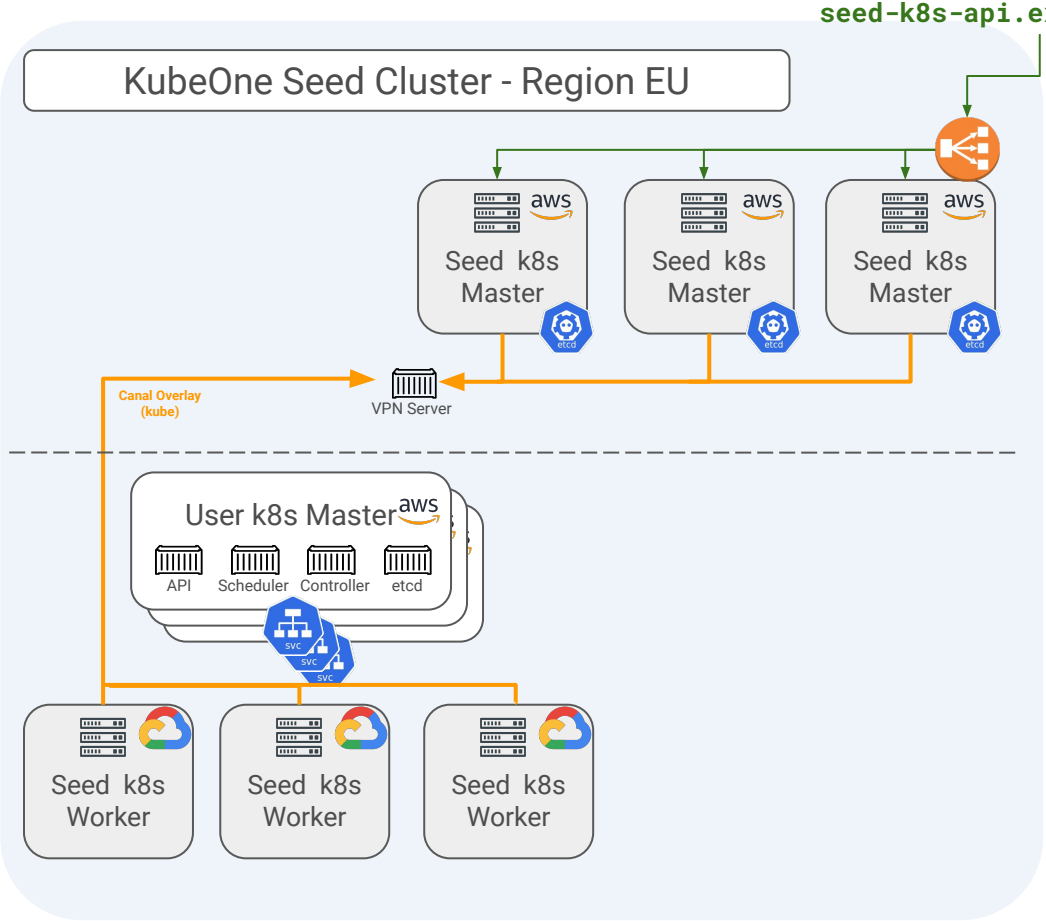


Migrate Seed Master Nodes:

1. Setup VPN Overlay
2. Pause existing Cluster & Machine Deployment
3. Create and join new 2 Master Nodes

4. Add new LB Service & Update DNS

5. Remove 2 old Master Nodes and move etcd quorum to new cloud



Migrate Seed Master Nodes:

1. Setup VPN Overlay
2. Pause existing Cluster & Machine Deployment
3. Create and join new 2 Master Nodes
4. Add new LB Service & Update DNS
5. Remove 2 old Master Nodes and move etcd quorum to new cloud

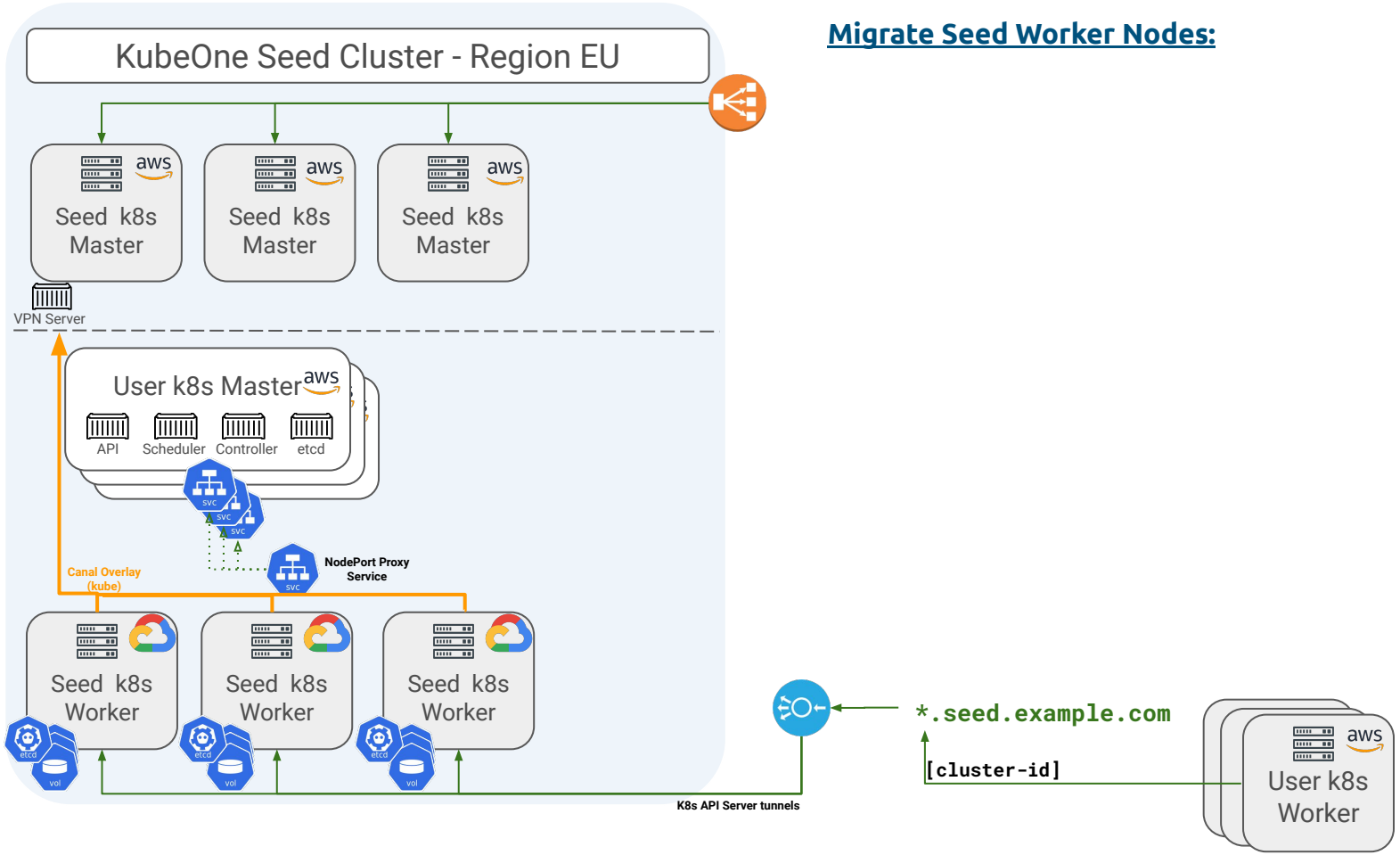
6. Create 3rd Master Node at new cloud and remove last old Master Node

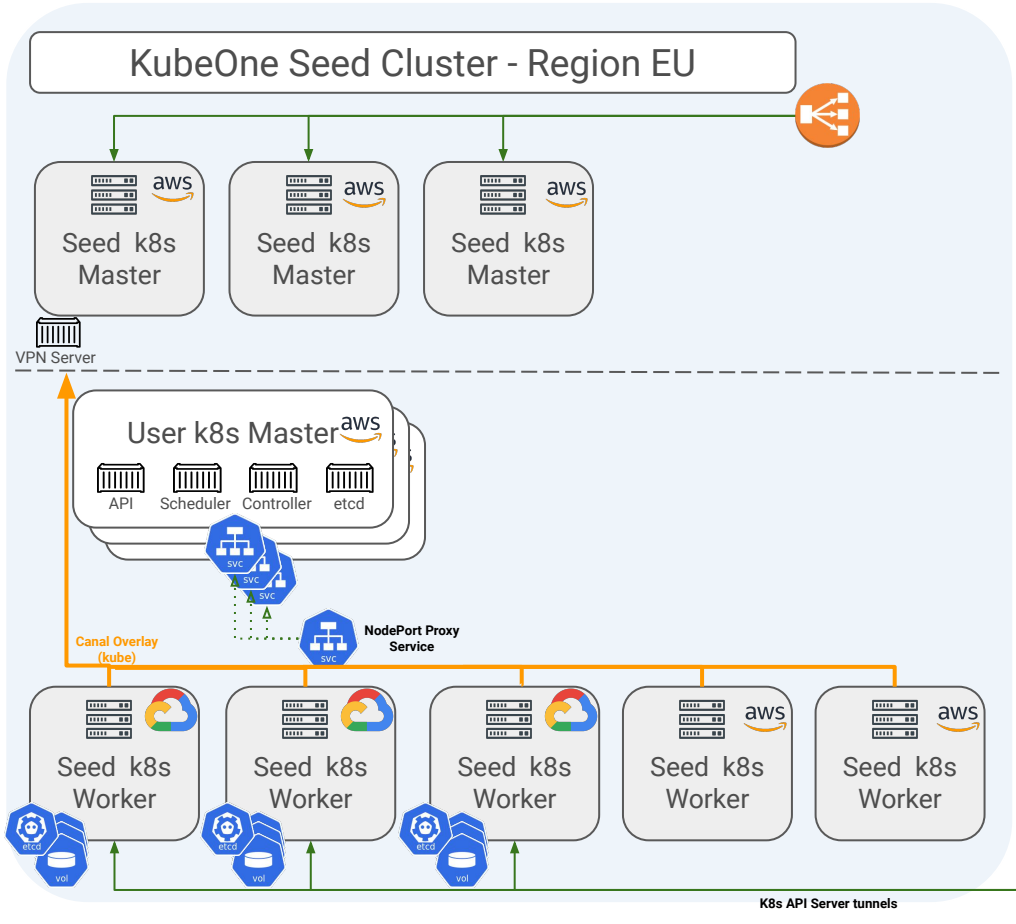
Kubermatic

Solution Approach

Migrate Seed Cluster Workers

Migrate Seed Worker Nodes:

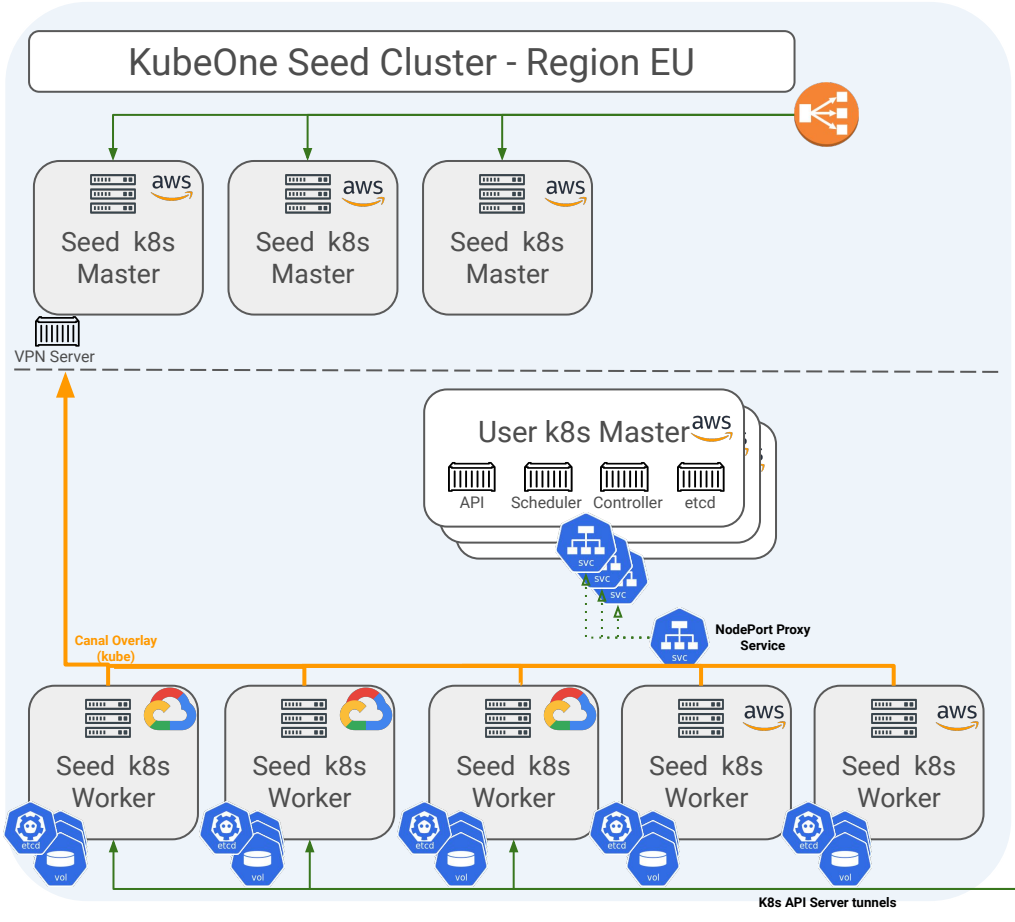




Migrate Seed Worker Nodes:

1. VPN Overlay, Pause existing Cluster, Machine Deployment

2. Create 2 new Workers (migration steps similar to user cluster)

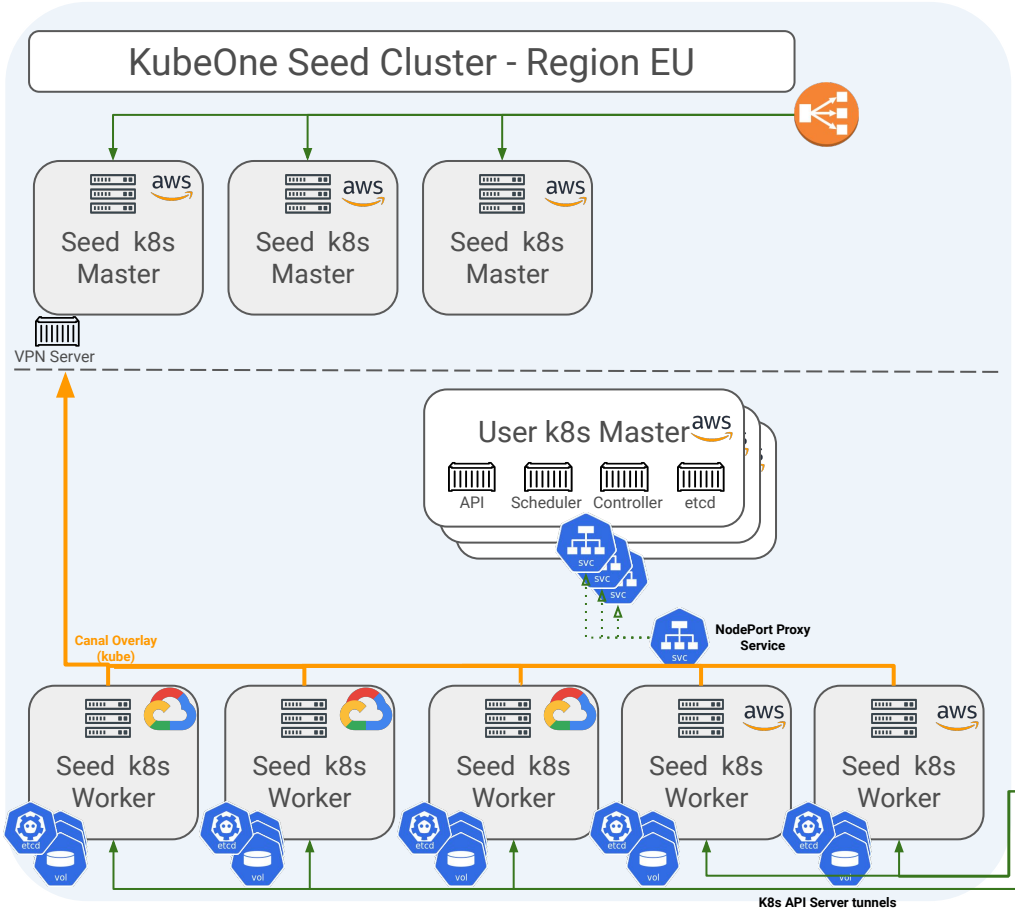


Migrate Seed Worker Nodes:

1. VPN Overlay, Pause existing Cluster, Machine Deployment
2. Create 2 new Workers (migration steps similar to user cluster)

3. Taint existing workers as non-schedule

4. Scale up etcd count of user cluster to 5
⇒ data replicated by etcd

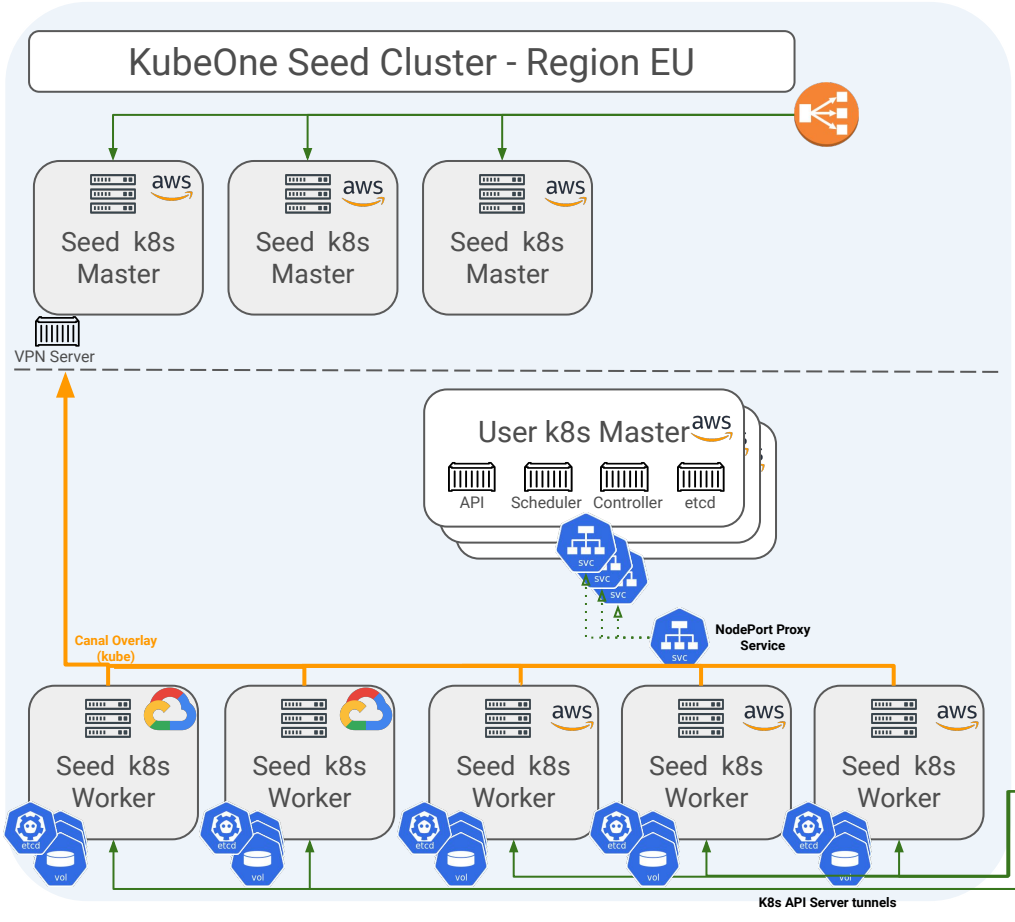


Migrate Seed Worker Nodes:

1. VPN Overlay, Pause existing Cluster, Machine Deployment
2. Create 2 new Workers (migration steps similar to user cluster)
3. Taint existing workers as non-schedule
4. Scale up etcd count of user cluster to 5

⇒ data replicated by etcd

5. Create new LB for NodePort Proxy and update DNS



Migrate Seed Worker Nodes:

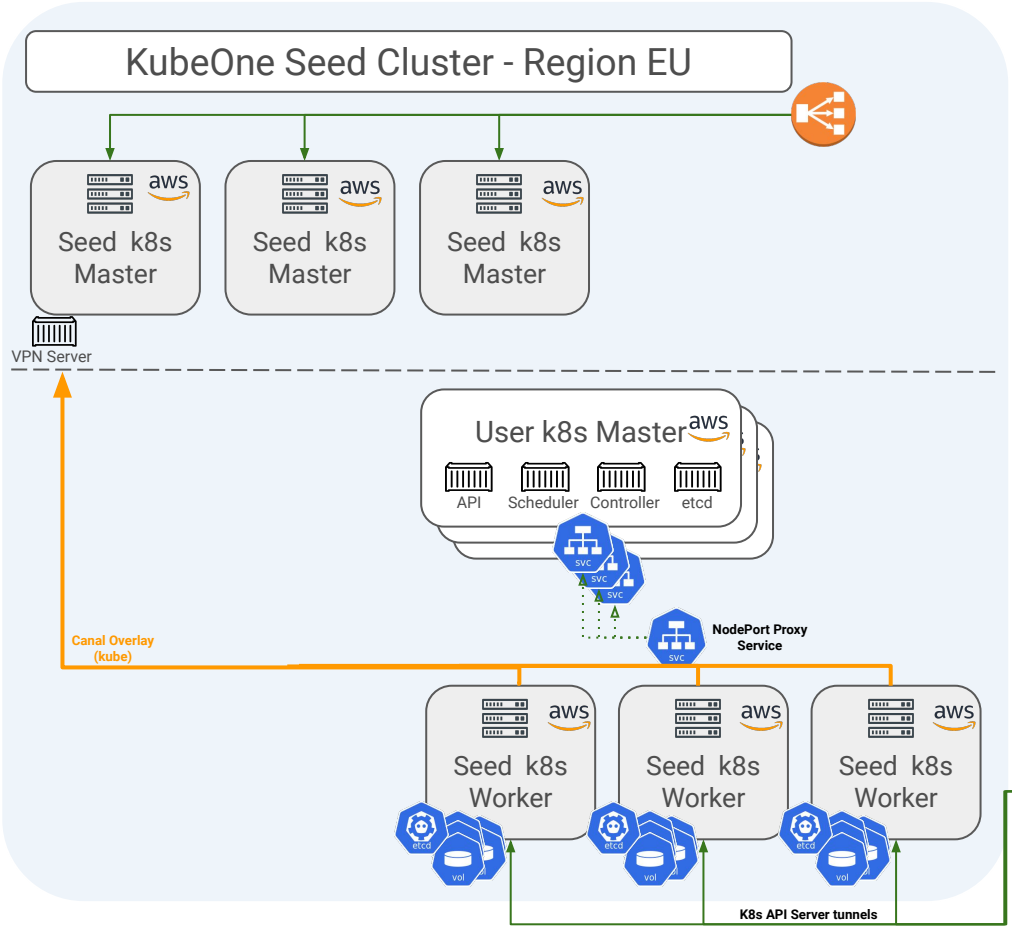
1. VPN Overlay, Pause existing Cluster, Machine Deployment
2. Create 2 new Workers (migration steps similar to user cluster)
3. Taint existing workers as non-schedule
4. Scale up etcd count of user cluster to 5

⇒ data replicated by etcd

5. Create new LB for NodePort Proxy and update DNS

6. Add 1 new worker and drain 1 old workers

⇒ etcd quorum migrated to new cloud



Migrate Seed Worker Nodes:

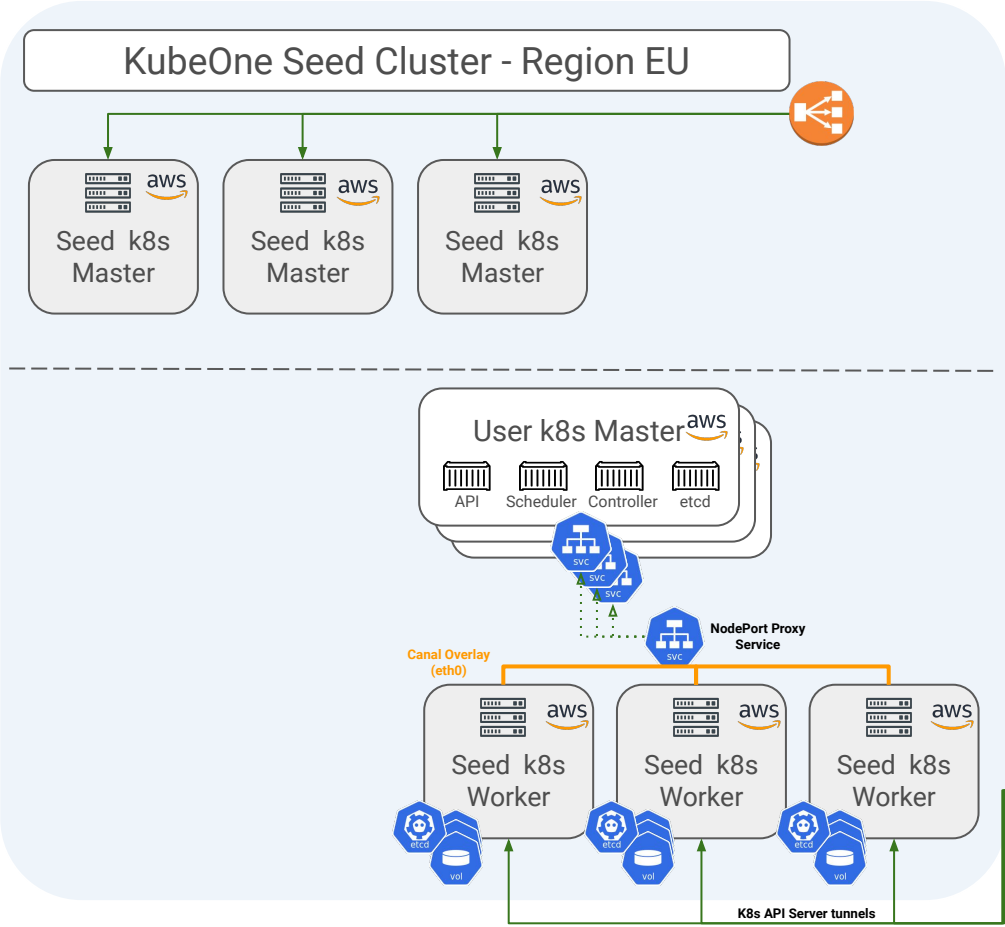
1. VPN Overlay, Pause existing Cluster, Machine Deployment
2. Create 2 new Workers (migration steps similar to user cluster)
3. Taint existing workers as non-schedule
4. Scale up etcd count of user cluster to 5

⇒ data replicated by etcd

5. Create new LB for NodePort Proxy and update DNS
6. Add 1 new worker and drain 1 old workers

⇒ etcd quorum migrated to new cloud

7. Drain missing worker nodes, cleanup old cloud



Migrate Seed Worker Nodes:

1. VPN Overlay, Pause existing Cluster, Machine Deployment
2. Create 2 new Workers (migration steps similar to user cluster)
3. Taint existing workers as non-schedule
4. Scale up etcd count of user cluster to 5

⇒ data replicated by etcd

5. Create new LB for NodePort Proxy and update DNS
6. Add 1 new worker and drain 1 old workers

⇒ etcd quorum migrated to new cloud

7. Drain missing worker nodes, cleanup old cloud

8. Scale down etcd count of user cluster to 3

9. Remove VPN Overlay

Outlook

- Automate clean up procedure
- Manage migration by Operator
  - Health checks
  - Wait conditions for migration steps
- Stabilize VPN connection
  - Multiple VPN servers
  - Soft switchover between VPN / Host network overlay
  - Evaluate Wireguard usage
- Support Seed cluster Kubernetes Version >1.17.X
  - Managed fields feature in >1.18.X stops the patching of cloud provider fields via kubectl

Kubermatic

Any Questions?

Kubermatic

Thank you for your attention!

marketing@kubermatic.com

@kubermatic

kubermatic/kubermatic