KubeCon | CloudNativeCon

Europe 2020

Virtual

Sharing Clusters: Learnings From Building a Namespace On-Demand Platform

Lukas Gentele, DevSpace Technologies Inc.

Hi! I am Lukas.

- CEO @ DevSpace Technologies Inc.
- Currently working on loft.sh
- Open-Source Maintainer: devspace.sh, kiosk.sh etc.
- Live in San Francisco (prev. Mannheim, Germany)

@LukasGentele

loft.sh

Who is this talk for?

- For IT Teams / Admins / SREs who:
  - are already managing Kubernetes clusters
  - need to make Kubernetes available for engineering teams
  - plan to host multiple engineers (or teams) in shared dev clusters

- Challenge: **Kubernetes Multi-Tenancy**

What is Multi-Tenancy?

Single-Tenant k8s

1 Team/App per Cluster

Multi-Tenant k8s

Sharing Large Clusters

Single-Tenant k8s

1 Team/App per Cluster

This is way too expensive!

Multi-Tenant k8s

Sharing Large Clusters

Very difficult to get right!

Hard vs Soft Multi-Tenancy

- **Hard Tenancy**
  - No trust between tenants
  - Multiple users from multiple different places are sharing the cluster
- **Soft Tenancy**
  - Tenants usually part of the same organization
  - *Small* chance of malicious actors from outside the organization
  - Prevent accidents & ensure stability rather than harden against attacks

Learning #1
Centralize user management & authentication

- **You probably already have users in a 3rd party system**
  - Active Directory
  - Google, Microsoft, ...
  - GitHub, GitLab, Bitbucket, ...

- **SSO for Kubernetes via dex**
  - CNCF sandbox project
  - OpenID Connect & OAuth2 Provider
  - Supports various identity providers (including LDAP, SAML)
  - Option: Provide kube-config with service account after successful authentication

Learning #2
Restrict users <u>but</u> use smart defaults (UX matters)

- **Pod Security Policy (!)**
- **Resource Quotas**
  - Problem: Users MUST specify cpu/memory limits if quota is enabled for cpu/memory
  - Solution: Set defaults via LimitRange => Mutating Admission Controller
- **Network Policies**
  - Default: deny all
  - Allow: traffic within the same namespace
  - Allow: internet traffic for containers
  - Allow: requests to the cluster-internal DNS (or DNS in general)

Learning #3
Automate as much as possible

- **Templates for**
  - RBAC
  - Resource Quotas & Limit Ranges
  - Network Policies
- **OPA for dynamic admission control**
  - Hostname validation for ingress resource (+ ingress annotations)
  - Hostname validation for certificate resource
  - Block certain storage and network related configurations (e.g. LoadBalancer services)
- **Cluster-wide services (e.g. ingress controller, cert-manager)**

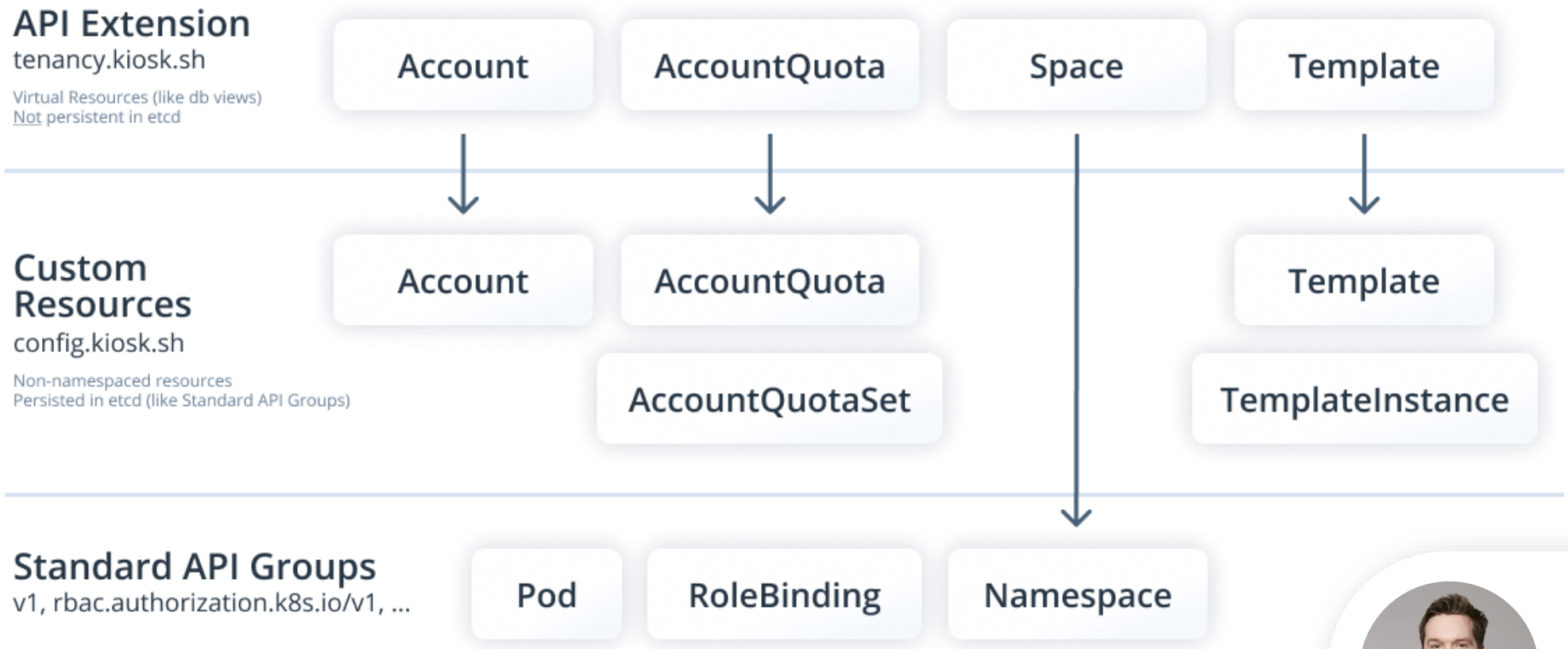
Learning #4
Store <u>everything</u> in Kubernetes (+ git)

- **Use annotations, labels, secrets and config maps to store information about owners/tenants etc.**

- **GitOps**
  - History / audit log via commits + easy rollback via git revert
  - Approval process via pull/merge requests & code owners
  - Great for platform state != user workloads

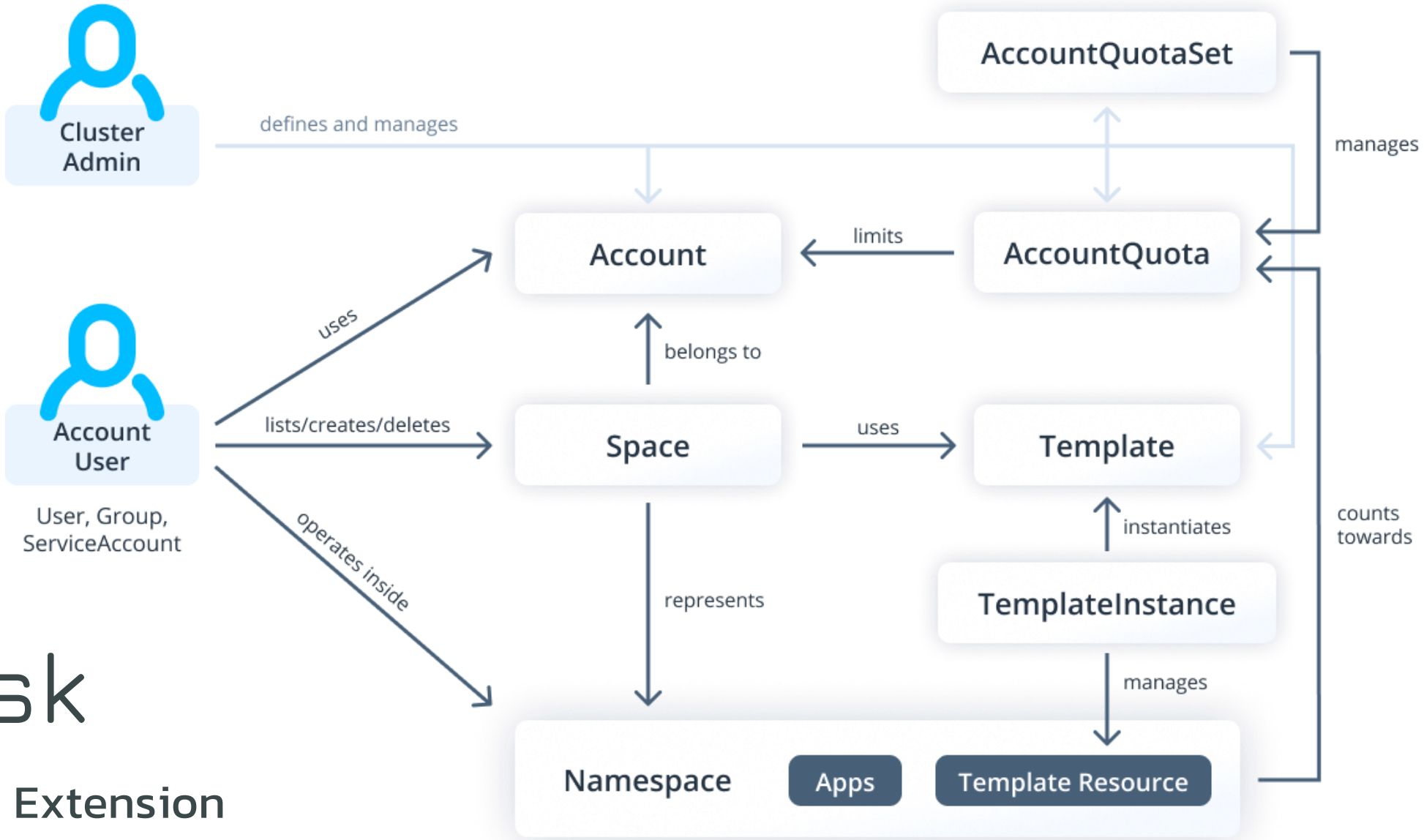
- **CRDs for even more control & automation**
- **Extra fancy: API server extension for “list problem” of RBAC**

kiosk Multi-Tenancy Extension for k8s



Fabian Kramm
kiosk Maintainer
fk@devspace.cloud

@LukasGentele



kiosk

Multi-Tenancy Extension For Kubernetes

@LukasGentele

loft.sh

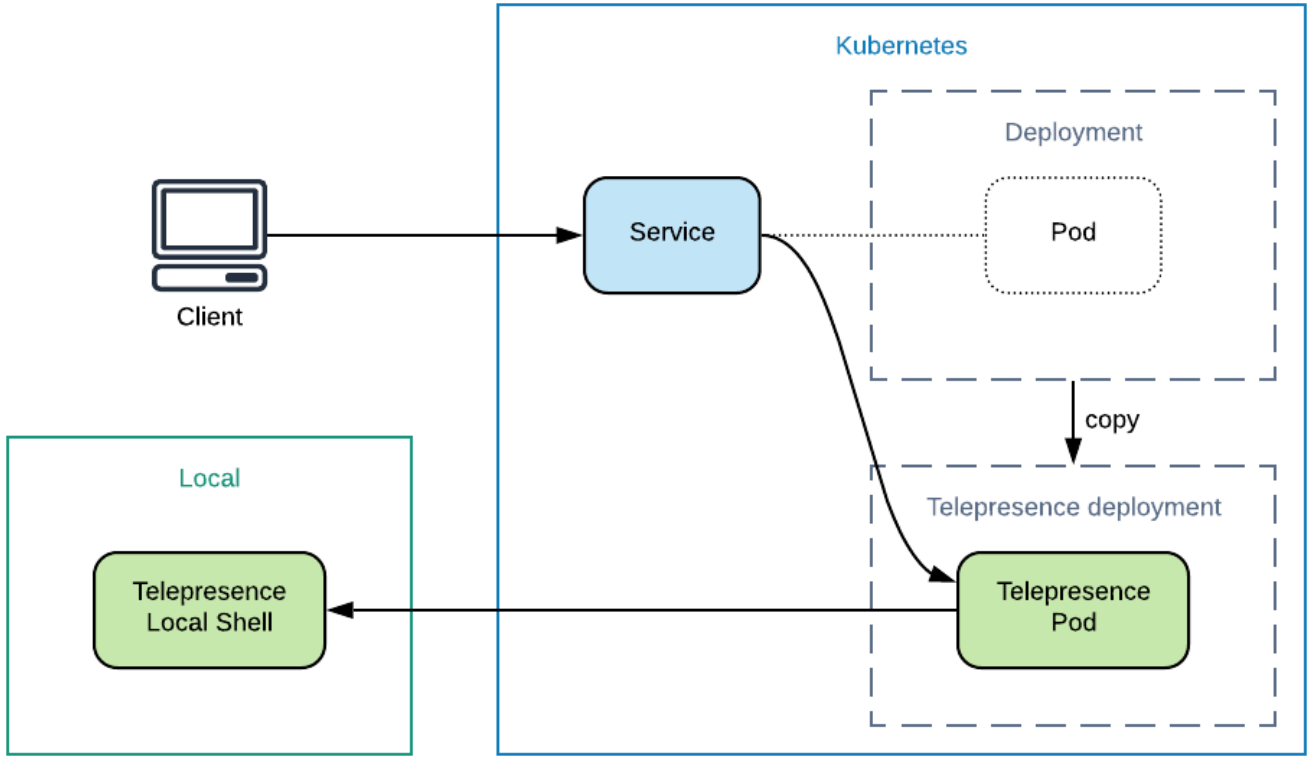
Learning #5
Do not hide Kubernetes but make it easier to use

- **Engineers need direct access to Kubernetes**
  - To verify new features fast without going through the full CI cycle (= pre-commit)
  - To debug container startup and inspect the state of their applications (logs etc.)
  - To attach debuggers, trace requests between microservices
- **kubectl is an API client, it's not a dev tool**
  - Leave dev tooling and deployment workflows up to the engineering teams
  - Raise awareness for any of the 50+ open-source dev tools for Kubernetes

SKAFFOLD DevSpace TILT TELEPRESENCE

Proxy-based Dev Experience

Telepresence

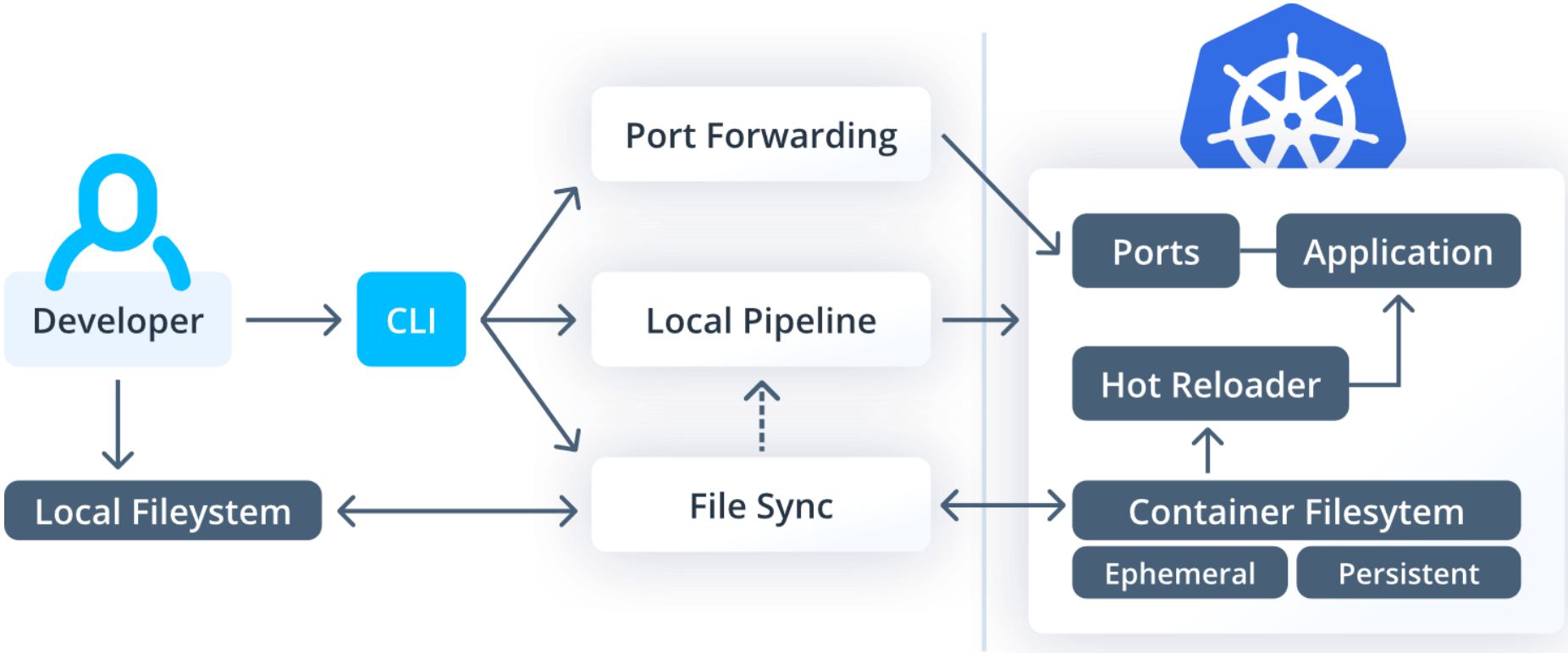


Source: https://medium.com/swlh/local-development-with-telepresence-256911cb21e9

- **Idea**
  - Deploy a two-way proxy instead of actual pod
  - Mount volumes and network via local telepresence CLI and in-cluster component
- **Drawbacks**
  - Bad Windows support
  - Slow Network (Databases?)
  - Issues with Volume Mounts

File-Sync-based Dev Experience

Skaffold, DevSpace, Tilt



@LukasGentele

loft.sh

Learning #6
Monitor cost and identify idle namespaces

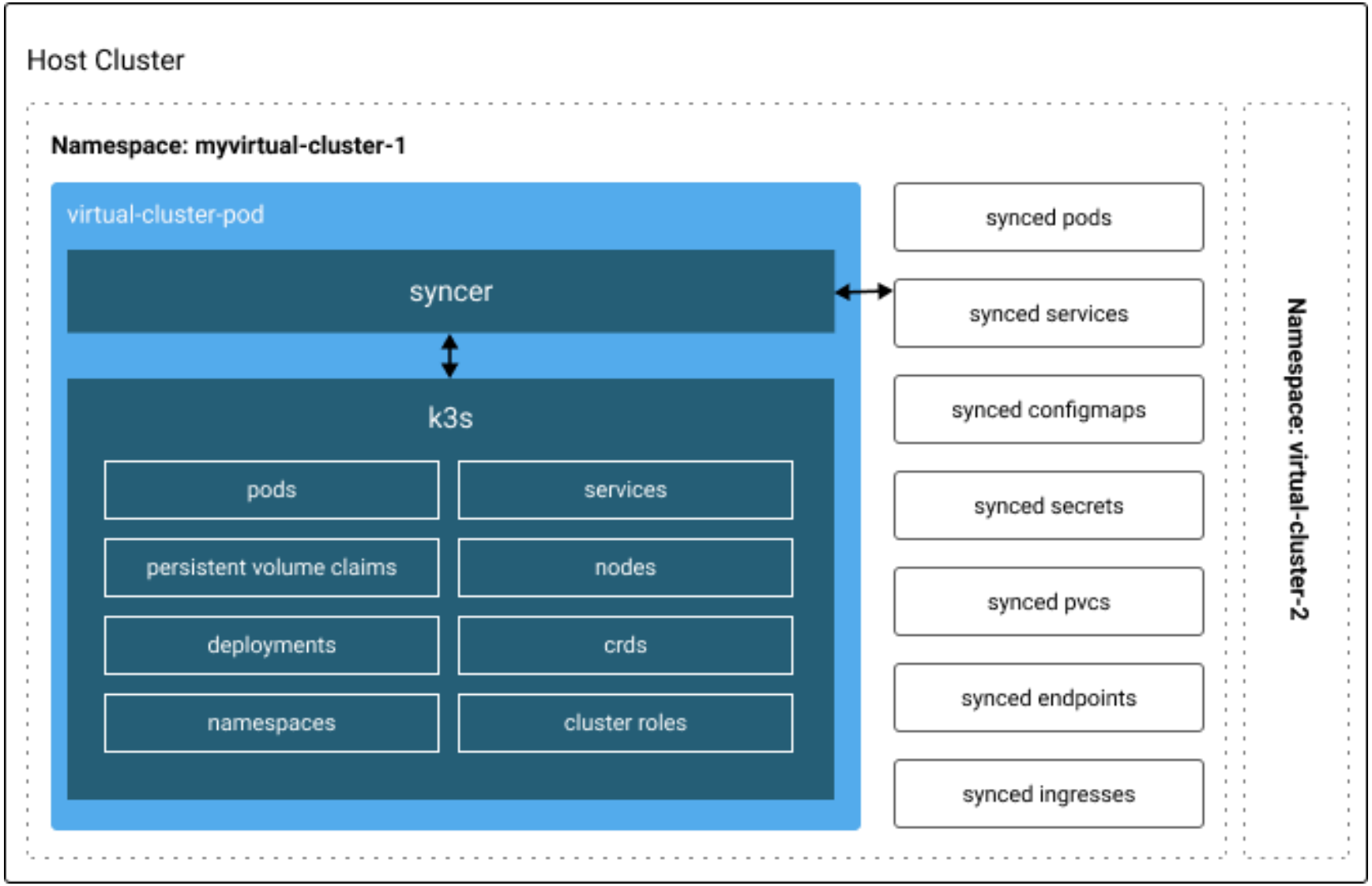
- **Use cluster auto-scaling (if possible)**
- **If you've done a great job with the platform, engineers**
  - will be quick to spin up and deploy a lot of things
  - will also be terrible at turning things off
- **Automate the shutdown of idle namespaces**
  - based on fixed schedule: cluster-turndown by kubecost (GKE/EKS)
  - based on ingress network traffic: idling in OpenShift
  - based on last k8s API server requests: sleep mode in loft (any k8s cluster)

Learning #7
Sometimes users need more than just namespaces

- **Namespace-based multi-tenancy has limitations**
  - What if users need CRDs?
  - What if users want to install Helm charts that use RBAC?
  - What if users need different k8s version or a beta feature?

- **<u>Virtual Clusters</u> can solve this problem**
  - Let users provision virtual Kubernetes clusters instead of namespaces
  - Unlocks access to cluster-wide settings and resources for users
  - Better isolation than with namespaces (separate k8s control planes)
  - Sharing of host cluster services and resources is still possible

vClusters



loft

github.com/loft-sh/virtual-cluster

@LukasGentele

loft.sh

My Learnings From Building A Namespace-as-a-Service Platform

#1 Centralize user management & authentication

#2 Restrict users but use smart defaults (UX matters)

#3 Automate as much as possible

#4 Store <u>everything</u> in Kubernetes (+ git)

#5 Do <u>not</u> hide Kubernetes but make it easier to use

#6 Monitor cost and identify idle namespaces

#7 Sometimes users need more than just namespaces

Learn more

- **Talk:** Making an Internal Kubernetes Offering Generally Available (James Wen, Spotify, KubeCon 2019 NA)

- **Articles:**
  - How to Save More Than 2/3 of Engineers' Kubernetes Cost
  - Securing the Kubernetes API with OPA
  - An Introduction to Kubernetes Network Policies for Security People
  - Introduction to Virtual Kubernetes Clusters

- **SIG Multi-Tenancy**

Get in touch.

- Connect on Twitter: @LukasGentele
- Email me: lg@devspace.cloud
- Check out what I'm working on:
  - loft.sh
  - devspace.sh
  - kiosk.sh