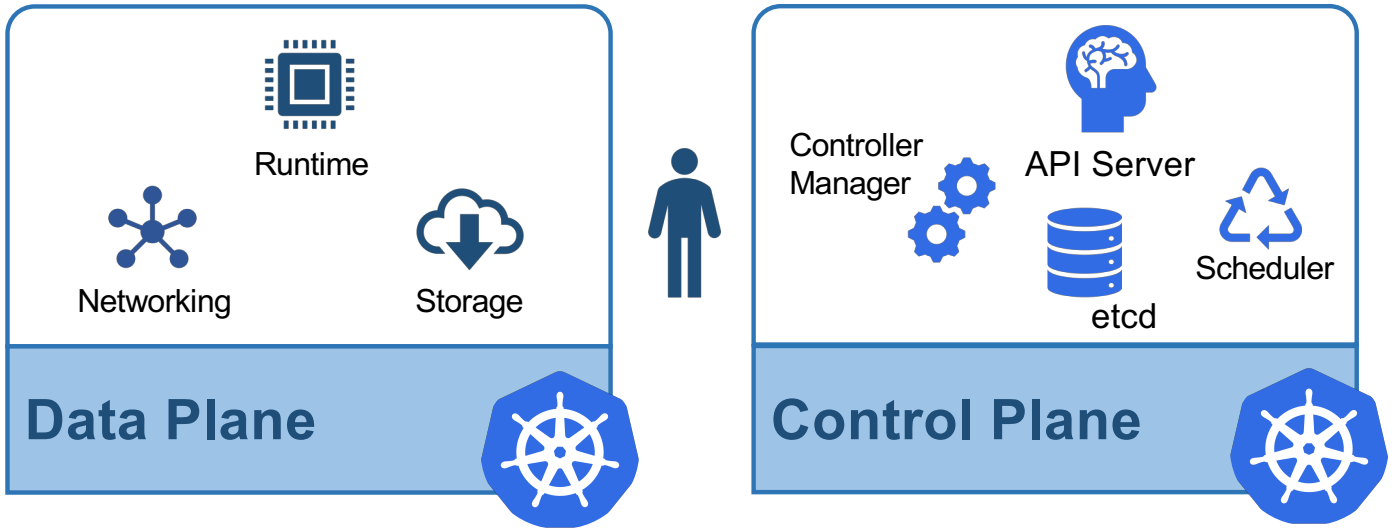
Virtual Cluster - A Practical Kubernetes Hard Multi-tenancy Solution

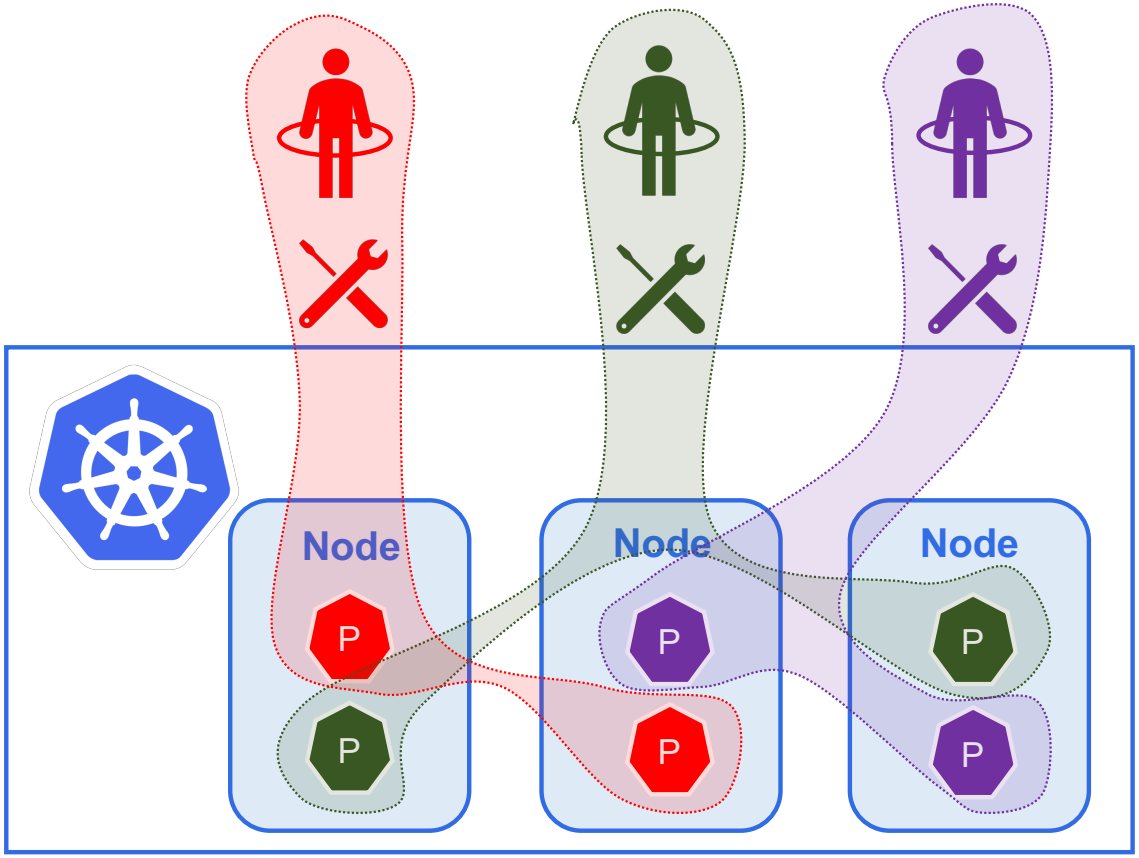
Fei Guo, Alibaba

Multitenancy: A battle in Kubernetes

Multiple users use the shared cluster resource in an **isolated** manner is a hard problem.



Is it possible?



Complete Control

Plane Isolation

+

Zero Tenant

Integration Effort

=

Virtual Cluster

The speaker

Fei Guo, Senior Staff Engineer, Alibaba Cloud

- Cloud native application platform team
- Serverless & Workload & Edge

- Design
- Challenges & Solutions
- Experiments
- Related work & Project Status
- Demo

Disclaimer

- This talk solely addresses the K8s controller plane isolation problems.
- Data plane isolation techniques will not be discussed. State of the art solutions may be referred if available.

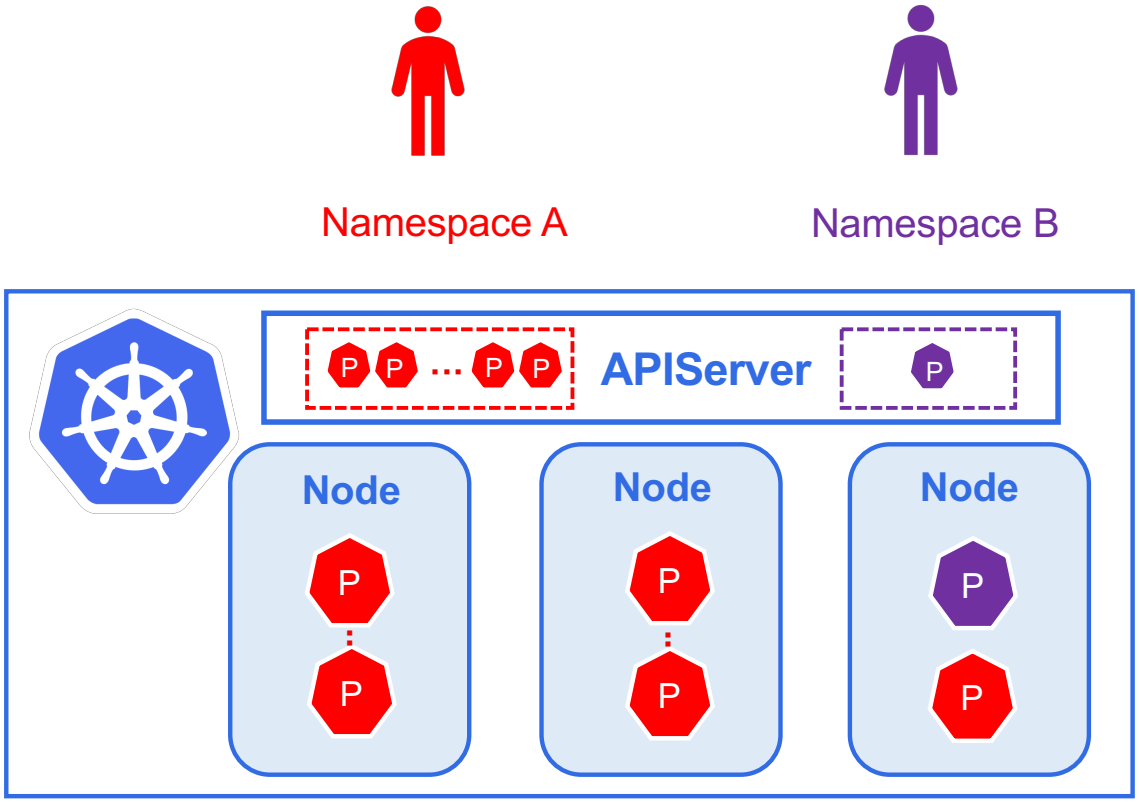
DESIGN

Threats

- Users are untrustworthy.
  - Exposing cluster scope resources is dangerous.
  - Generating harmful usage pattern intentionally or unintentionally.
  - They may serve other users.
- Containers are not safe.

Typical cloud scenarios that may apply internally as well

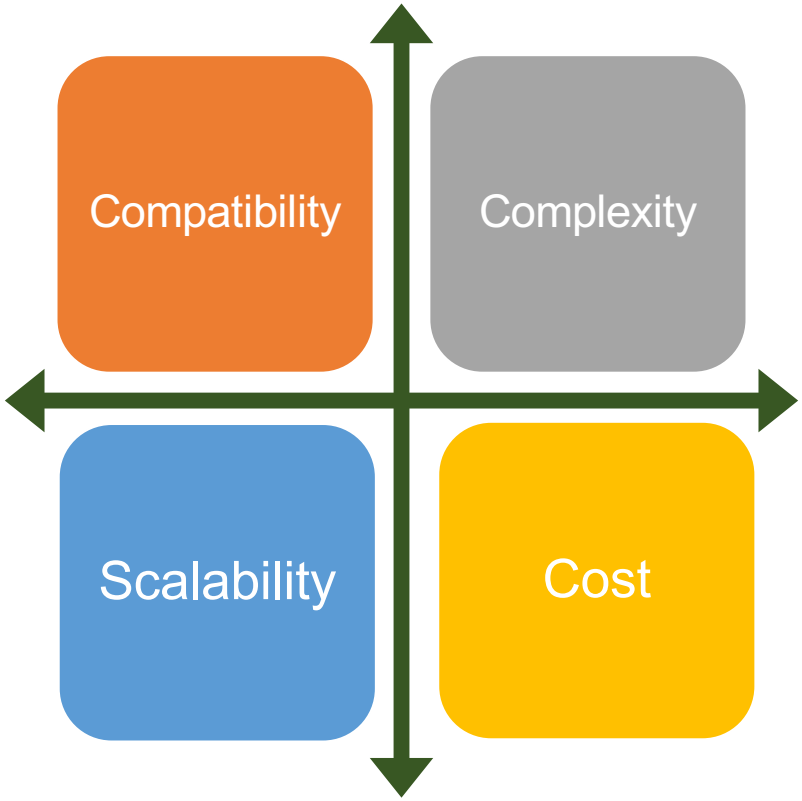
Namespace is insufficient



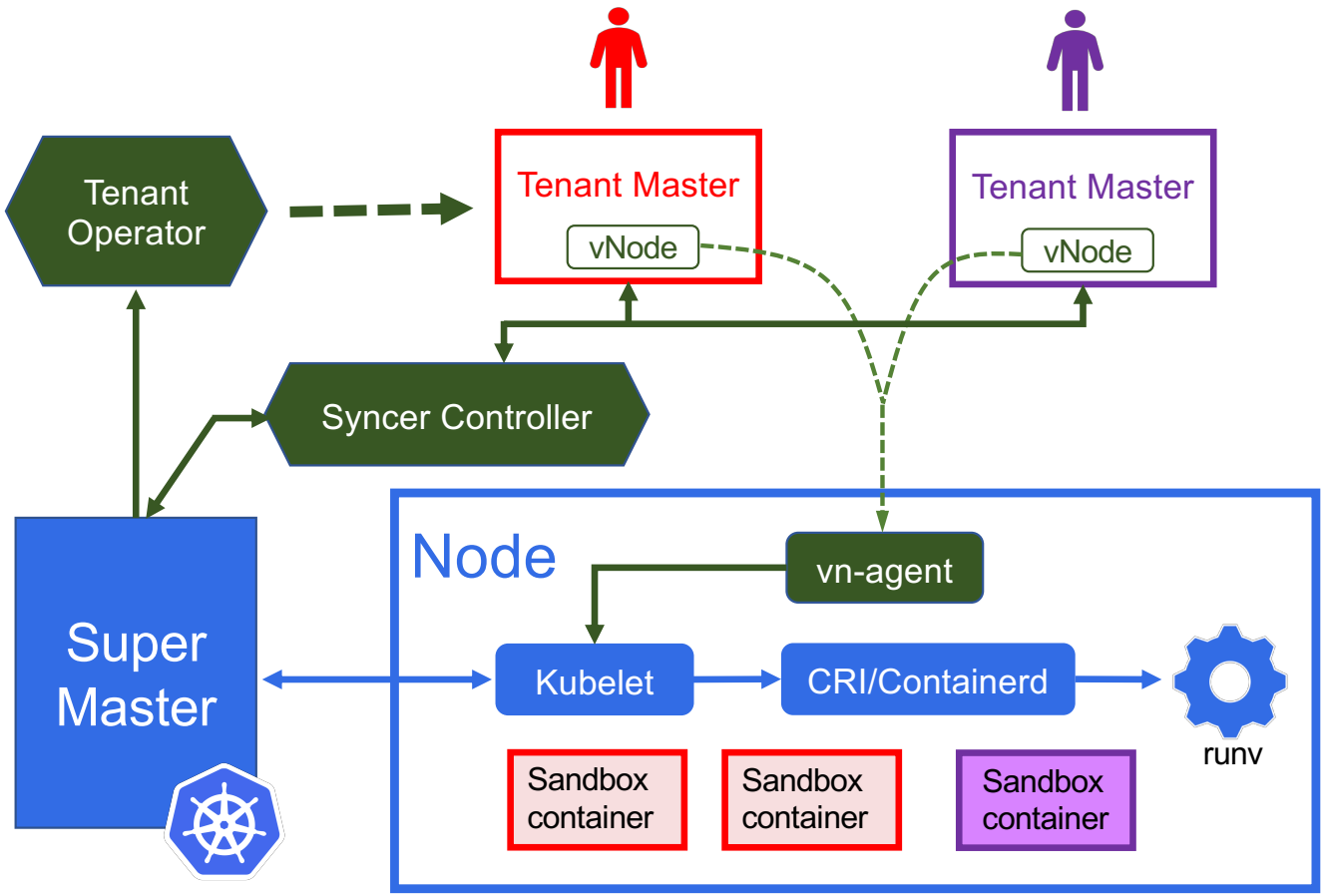
- Performance Interference
  - Starvation
  - Priority inversion
- Information leakage
- Installation disallowed
  - No CRD
  - No Webhooks
  - No Clusterroles

Principles

The solution space



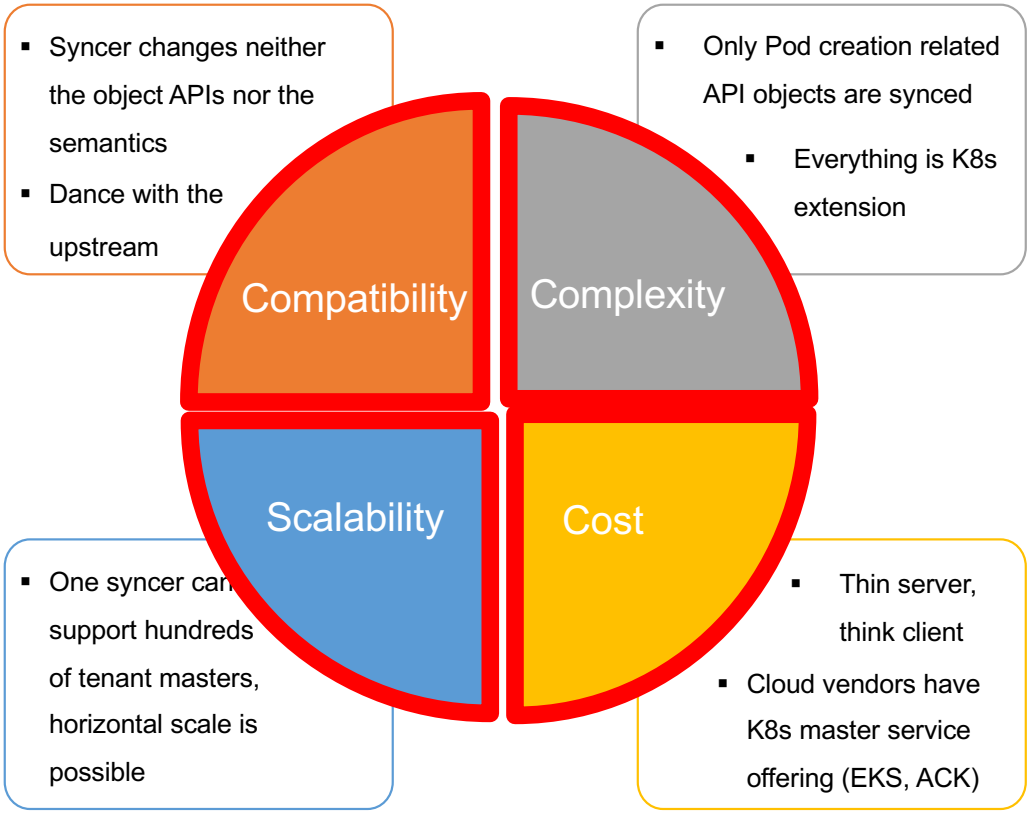
Architecture



Object States Maintainers

Resource Provider

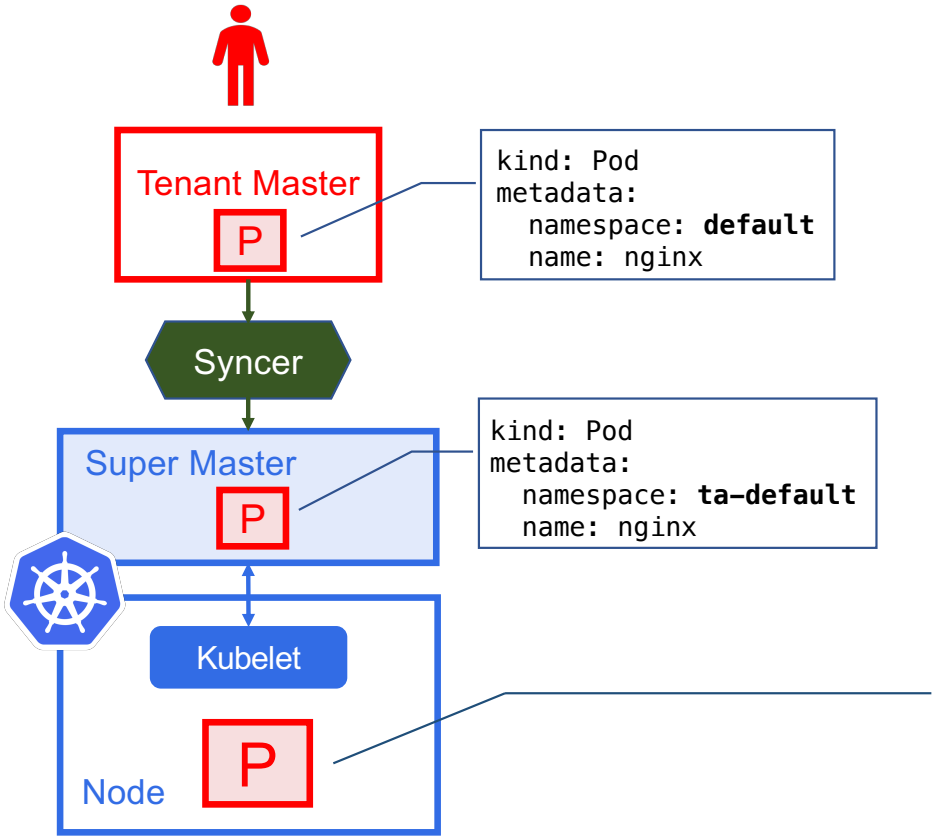
Justifications



- No noisy neighbors
- Complete view isolation
- Limited blast radius for security vulnerability
- Full manageability

CHALLENGES & SOLUTIONS

A “virtual” cluster view



User finds Pod running in a virtual tenant K8s

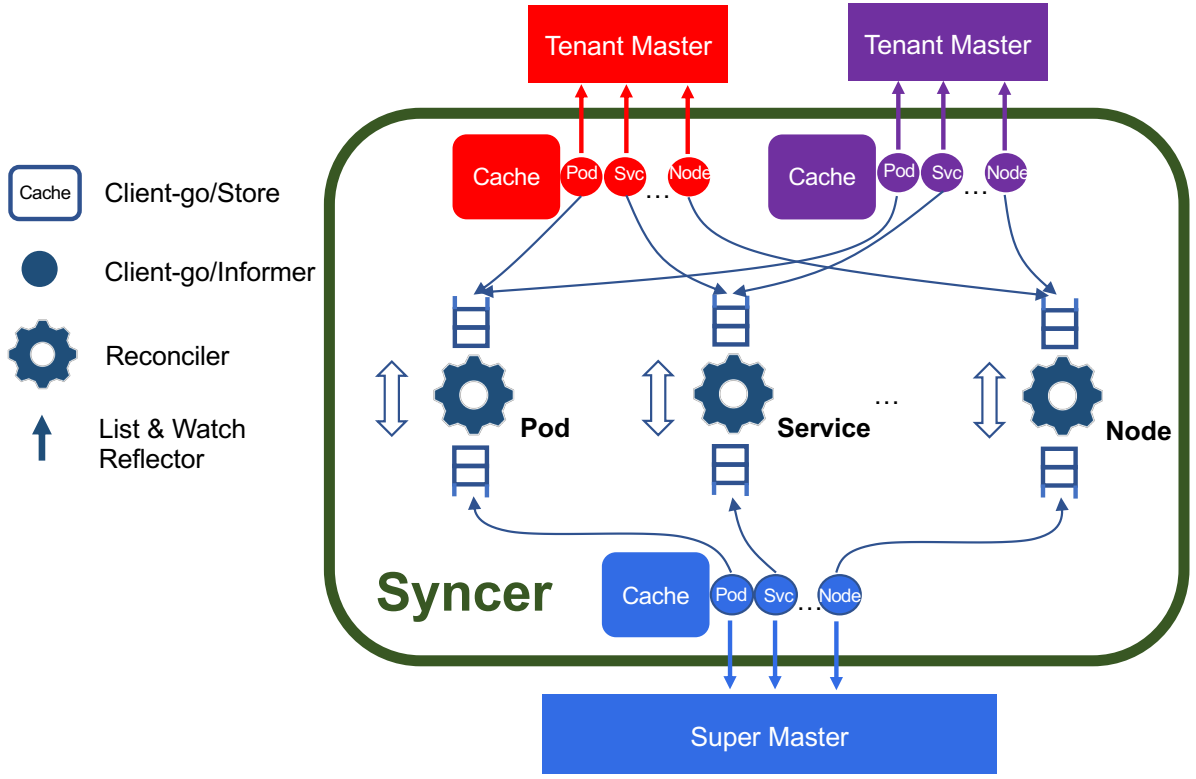
```
sh-4.4# env
HOSTNAME=curl-top
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_PORT=tcp://10.96.0.1:443
PWD=/
HOME=/root
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
TERM=xterm
SHLVL=1
KUBERNETES_SERVICE_PORT=443
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
KUBERNETES_SERVICE_HOST=10.96.0.1
_=/usr/bin/env
sh-4.4# cat /etc/resolv.conf
nameserver 10.96.0.10
search default.svc.cluster.local svc.cluster.local cluster.local
options ndots:5
sh-4.4# ls /run/secrets/kubernetes.io/serviceaccount/
ca.crt  namespace  token
sh-4.4# cat /run/secrets/kubernetes.io/serviceaccount/namespace
default
```

The magician - syncer

- Manipulate the Pod template (like a mutation webhook), no change in Kubelet is required
  - env variables
  - Service account secrets
  - Host alias & DNS config
- Ensure the data consistency
  - Tenant master is the source of truth for SPEC.
  - Super master is the source of truth for STATUS.
- User is not aware of the super master
  - Zero integration effort, **it just works**.

Syncer cannot be a hammer

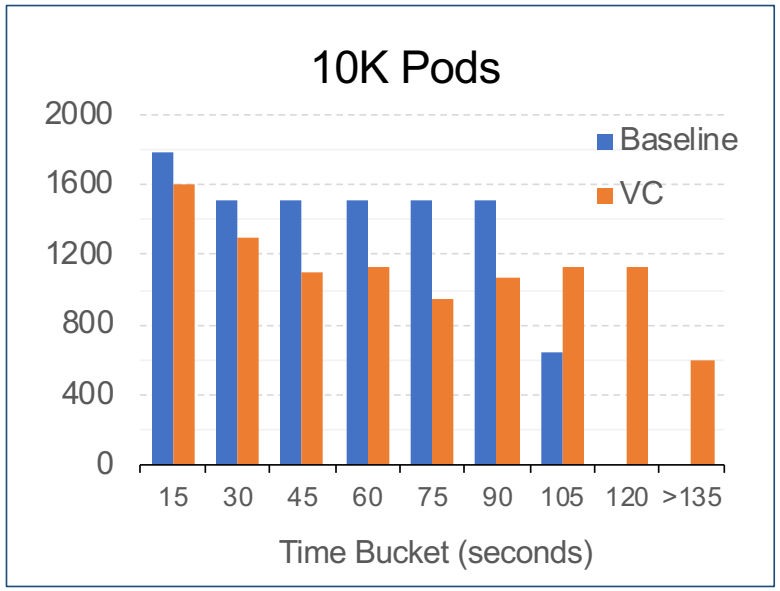
Synchronization based on the object states in the informer caches



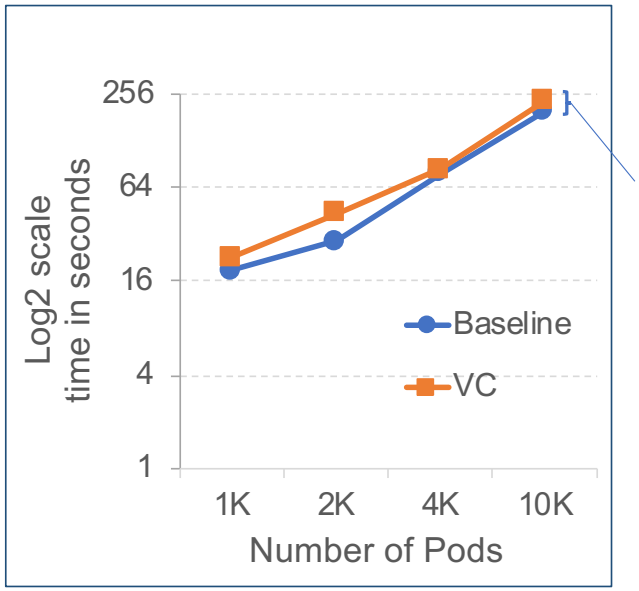
EXPERIMENTS

Stress tests

- 100 tenant masters, up to 10K Pods concurrent creations in total
- One syncer
- 100 virtual kubelets installed in the super master



The histogram of Pods creation time



The wall-clock time of creating all Pods

Syncer cost

- Syncer resource consumption does not scale.
- One syncer can support hundreds of tenant masters.
  - Syncer is stateless, state recovery can be done in < 1 minute upon restart.
  - It can be horizontally scaled.
- In normal cases, the extra latency added by the syncer is less than a few milliseconds.

RELATED WORK & PROJECT STATUS

Other solutions

- K3v (https://github.com/ibuildthecloud/k3v)
  - Dedicated control plane – modified K3s
  - Per tenant syncer
- Arktos (https://github.com/futurewei-cloud/arktos)
  - Modify APIServer to support new tenant APIs
  - Shared control plane
- Virtual Kubelet
  - Simplified provider interfaces, struggle for compatibility

Project status

- Multitenancy WG project (https://github.com/kubernetes-sigs/multi-tenancy/tree/master/incubator/virtualcluster)
- Kubernetes conformance tests pass rate : 99%
- Complete UT and e2e tests (>70% code coverage)
- Support cloud and on-prem K8s
- Already used in cloud serverless product
- Adopted by the community

DEMO

QUESTIONS ?