KubeCon | CloudNativeCon

North America 2018

Kubernetes Scalability:
A multi-dimensional analysis

Shyam Jeedigunta (@shyamjvs)
Maciej Rozacki (@mrozacki)

Background

FAQs by several devs/teams:

- What scale does k8s support?
- What do we mean when we say “it scales”?
- Why are clusters << 5000 nodes running into scale problems?
- Why aren’t we testing various possible configurations?

Goal

Address those concerns by:

- Explaining what scalability really means
- Eliminating few common misconceptions
- Describing some currently known scalability limits in K8s
- Knowing how we can explore our scalability bounds together

KubeCon | CloudNativeCon

North America 2018

Understanding Scalability

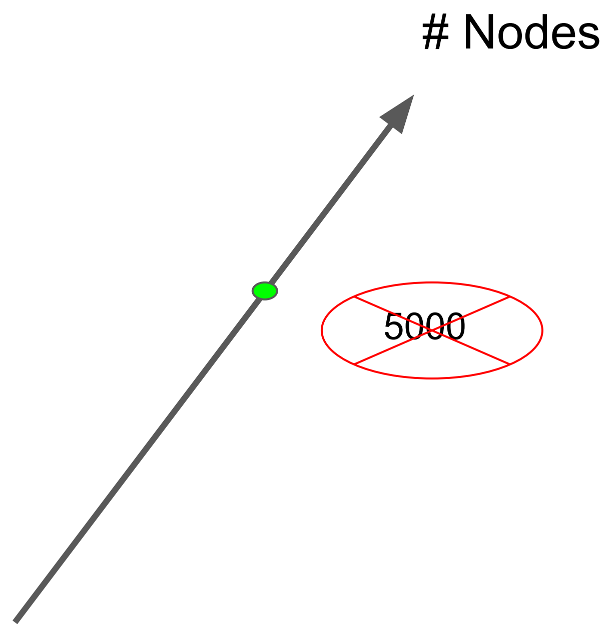
Scalability Limits

Scalability is **not a single number** (like 5000)

Yes, we *"`support`"* upto 5000 nodes in k8s

But that's not even close to the whole story!

Let's see what is...



Scalability Envelope

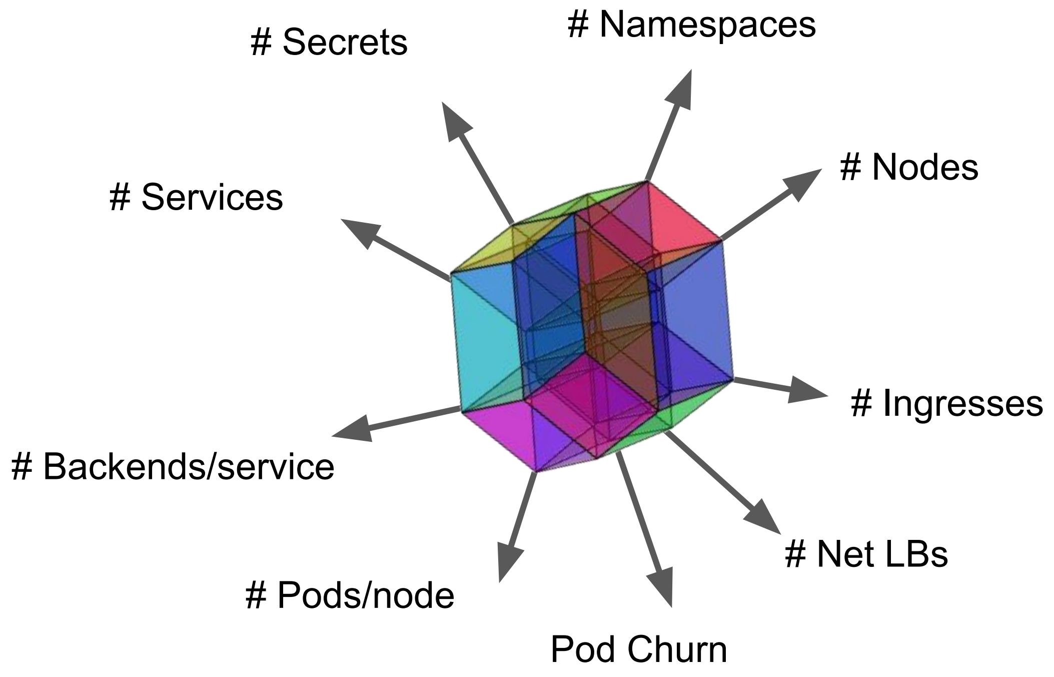
Scalability is a **subspace of configurations**

Think of it as a ~ higher-dimensional cube (not really a cube… see next slide)

If you're within the envelope, you're *safe*

By *safe*, we mean:

- Performance SLOs are satisfied
- Your k8s cluster is not badly broken



Source of hypercube image: http://www.gregegan.net/APPLETS/29/29.html

Properties of the Envelope

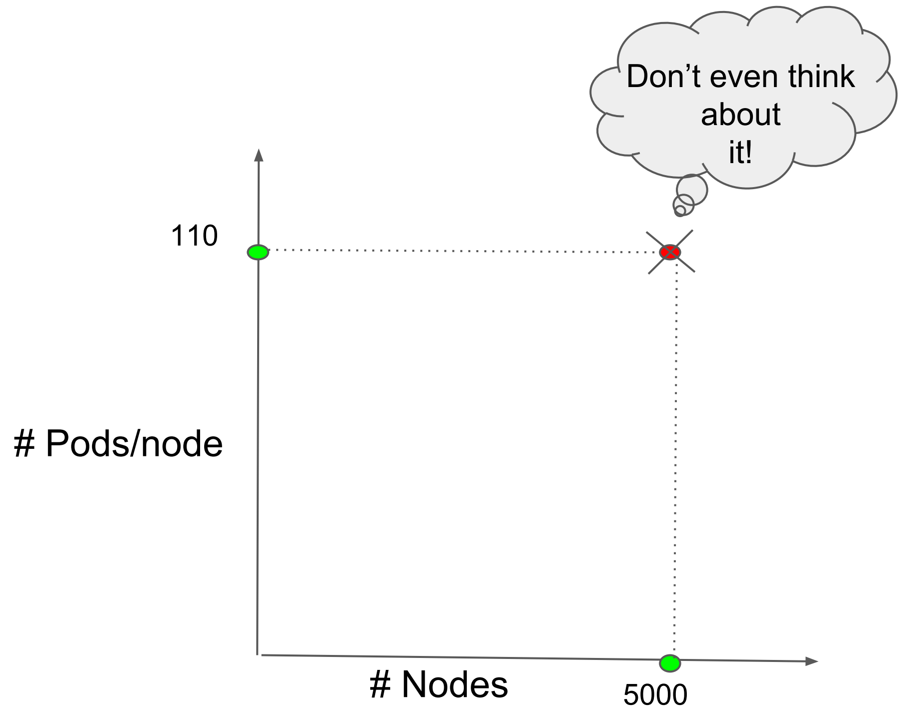
1. **NOT a cube**

Because...
the dimensions are sometimes NOT independent.

So if we support $X_1 = A$ and $X_2 = B$

$\not\Rightarrow$

we support $(X_1 = A, X_2 = B)$



E.g

Properties of the Envelope

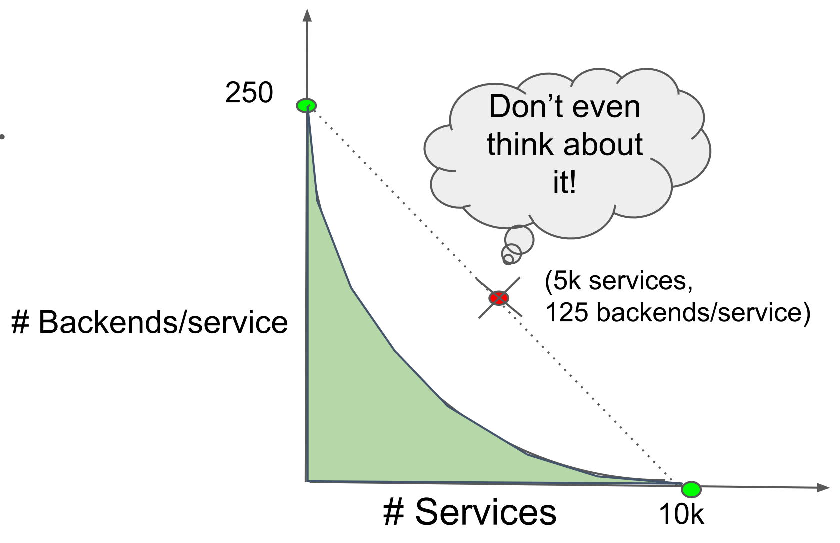
2. NOT convex

Because...
the dimensions are sometimes NOT linearly dependent.

So if we support configuration A and configuration B

$\not\Rightarrow$

we support configuration (A+B)/2



E.g

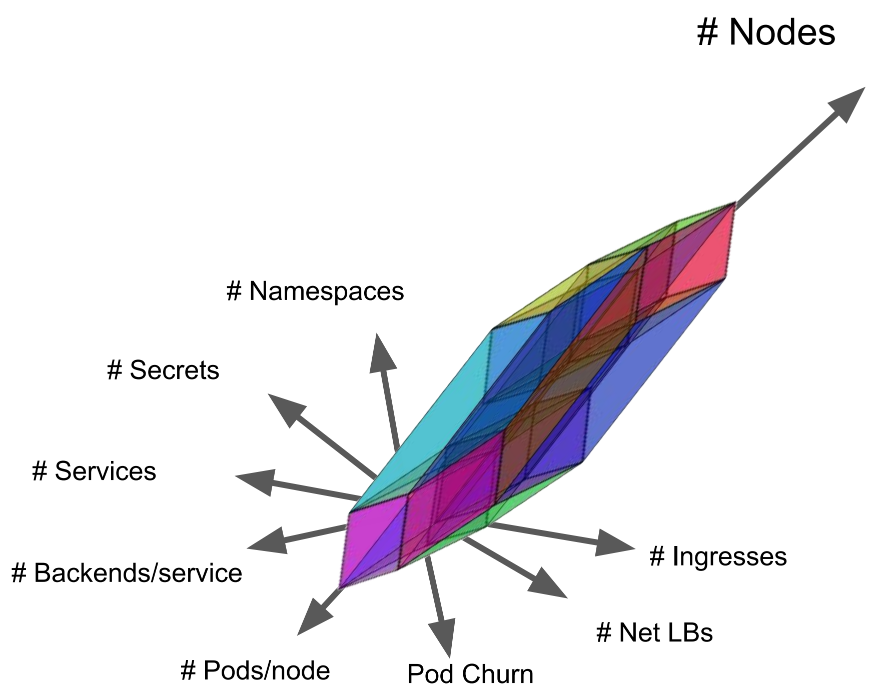
Properties of the Envelope

3. Tapers along each axis

As you move farther along one dimension, your cross-section wrt other dimensions gets smaller.

So don't push too many dimensions at once!

Note that it means even a 5-node cluster can break if you push too much along some dimension(s).



E.g

Properties of the Envelope

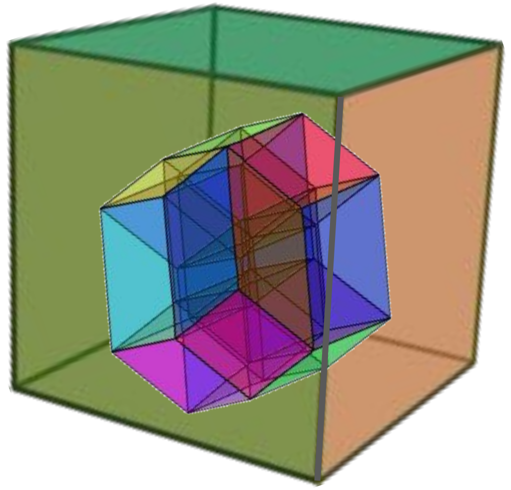
4. Bounded

No axis can be arbitrarily pushed (even if all others are kept at minimum).

We have hard limits - mainly due to etcd size. So...

Total #Objects (built-in API objects + CRDs) ≤ X (~300,000)*

is a bounding box.



*It's a crude limit and assumes etcd size is 4GB (it may change in future)

Source of cube image: https://en.wikipedia.org/wiki/Hypercube Source of cropped hyperbola image: http://inspirehep.net/record/1454384

Properties of the Envelope

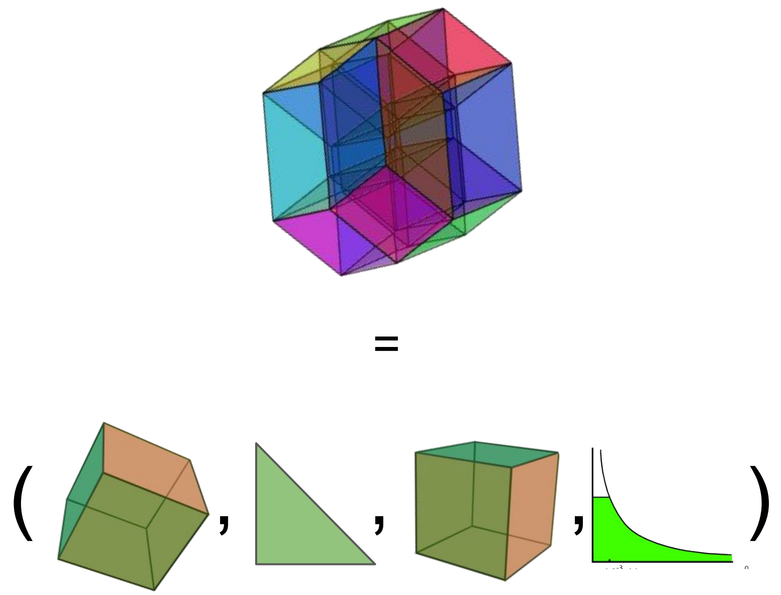
5. Decomposable into smaller envelopes

Precisely computing the envelope boundaries is too ***hard*** a problem (O(2^#dimensions)).

Luckily, we can ~break it into simpler envelopes, due to some independence among the dimensions.

Each envelope == some constraint

Let’s look at those…



Source of cube image: https://en.wikipedia.org/wiki/Hypercube Source of cropped hyperbola image: http://inspirehep.net/record/1454384

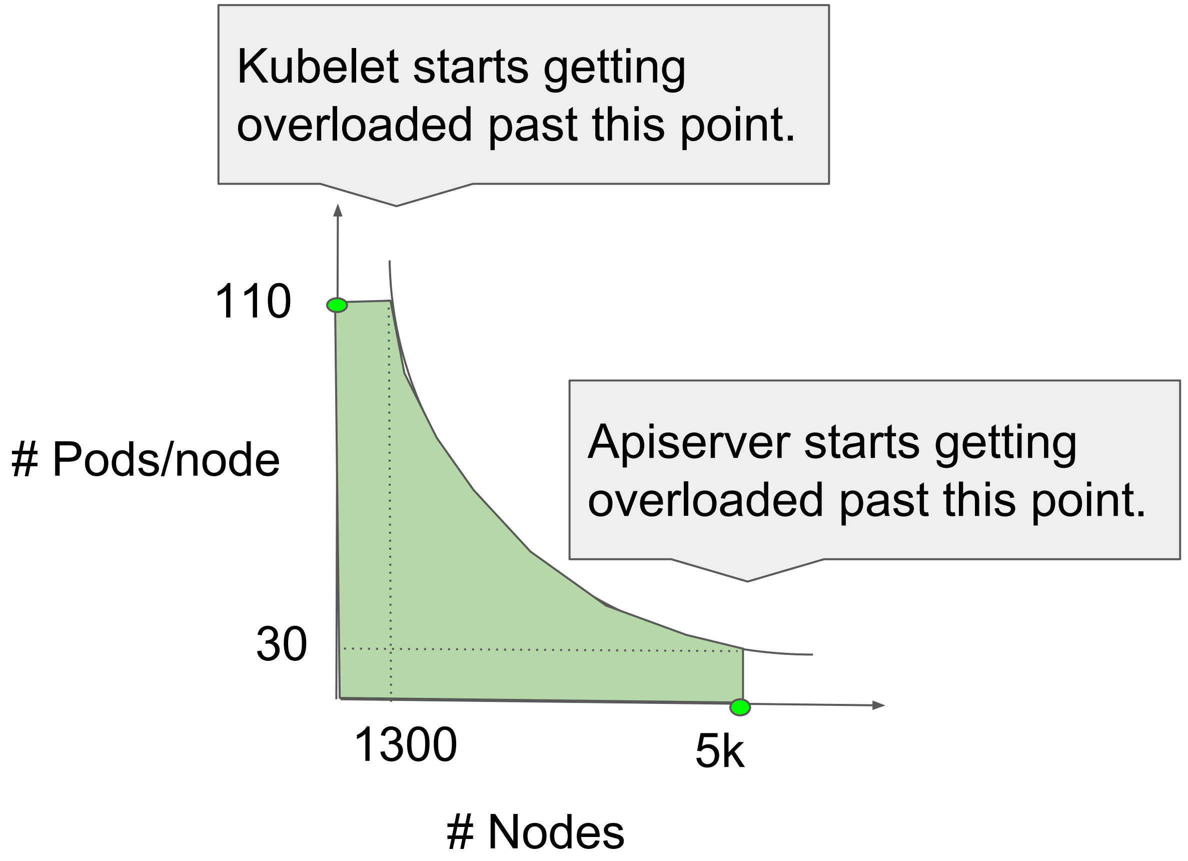
Few notes...

The scalability limits we're about to discuss are:

- For k8s control-plane in general and NOT specific to any cloud provider
- Don't form an exhaustive list, but just the known ones
- Form a rough sketch of what we believe are *safe* configurations based on historical evidence. So in practice you may be able to:
  - push outside these limits to *some extent*
  - screw up even within the limits in *some ways*

In general, use discretion or consult SIG scalability if in doubt.

#Nodes vs #Pods/node



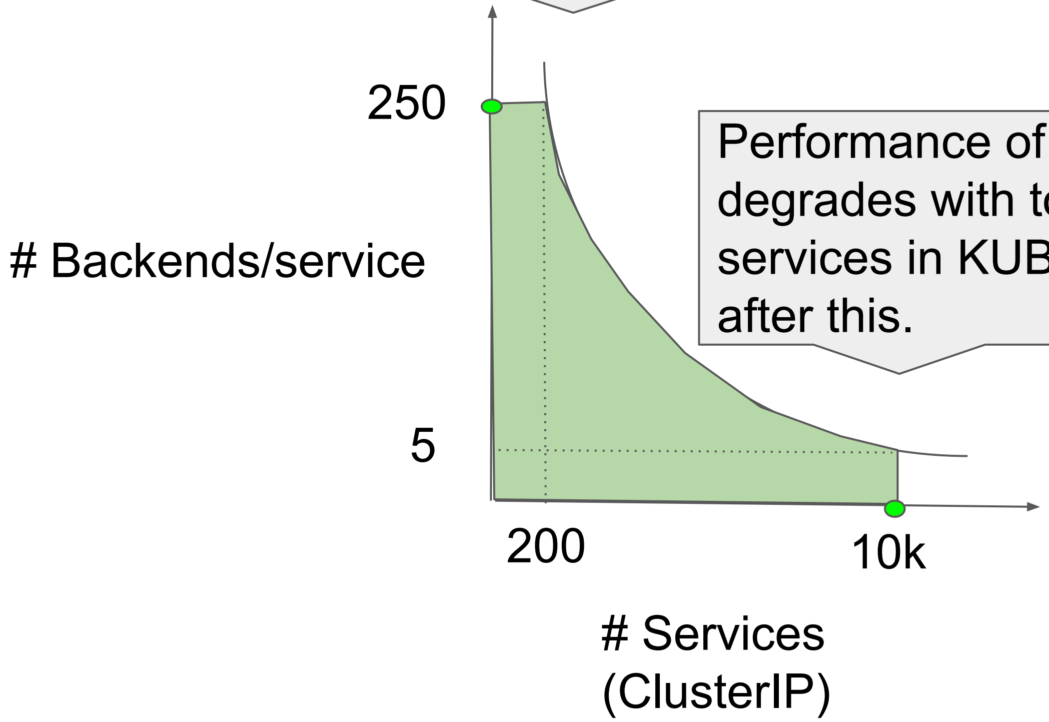
#Pods <= 150k
&
#Nodes <= 5k
&
#Pods/node <= 110

We assume the average #containers/pod is not too high (<= 2).

Having too many containers might reduce the limit of 110 because some resources are allocated per container.

#Services vs #Backends/service

Endpoints traffic becomes larger after this (due to being quadratic in #backends).



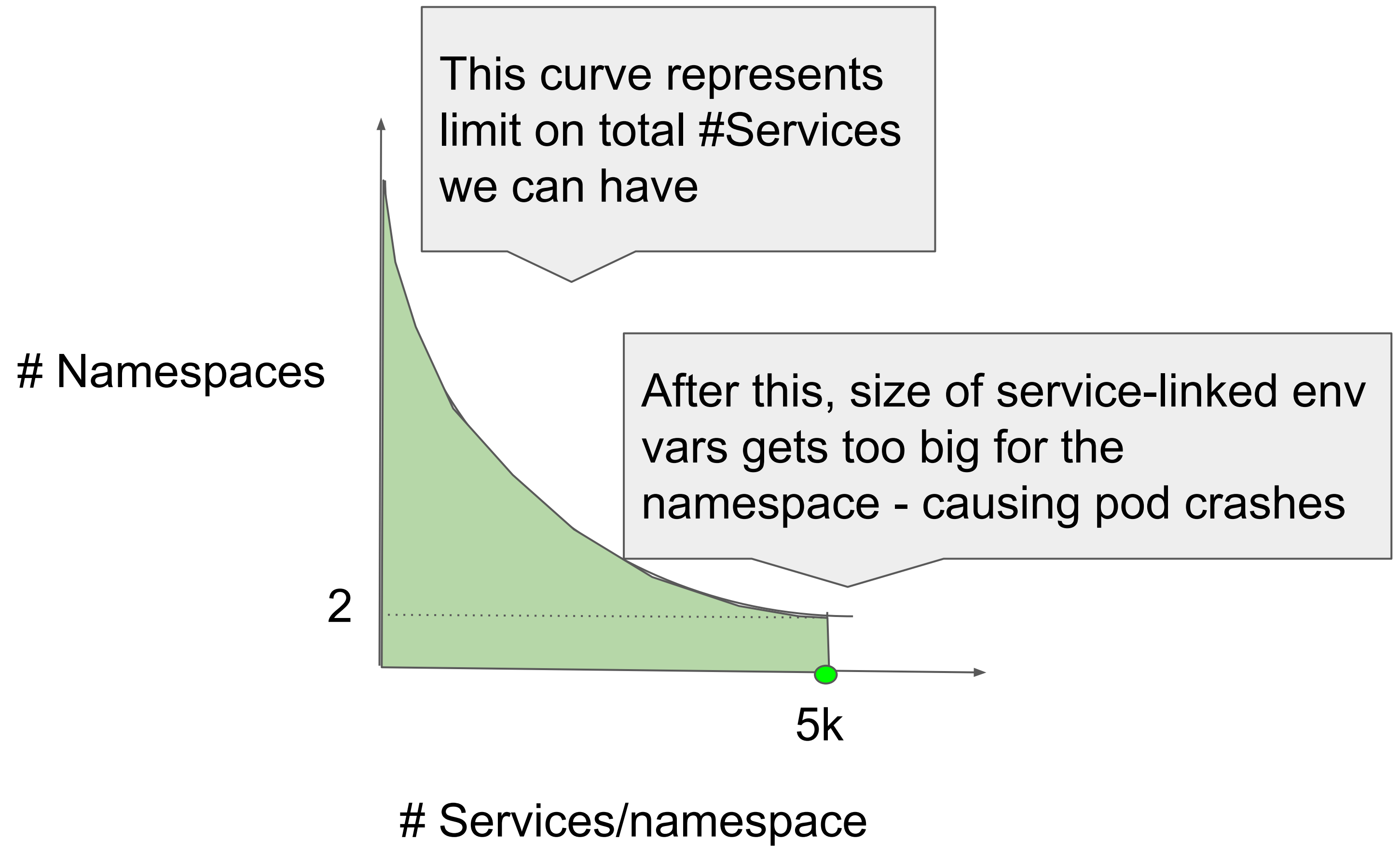
Performance of iptables degrades with too many services in KUBE_SVC chain after this.

#Backends <= 50k
&
#Services <= 10k
&
#Backends/service <= 250

Note: You can have more backends if majority of them belong to small services. For e.g we tested with 75k backends comprising of:

- 7500 services of size 5
- 600 services of size 30
- 75 services of size 250

#Services/namespace

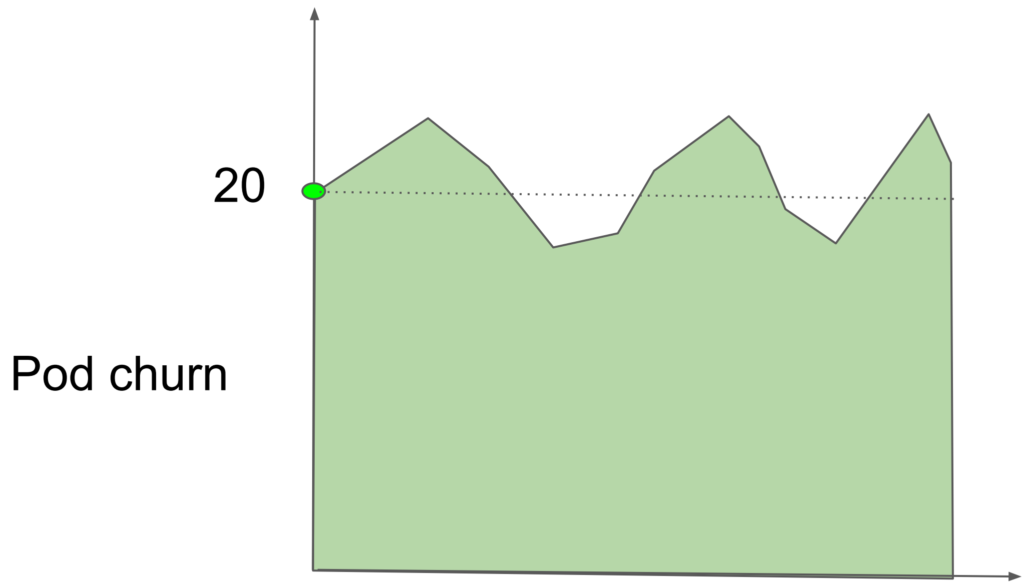


#Services <= 10k
&
#Services/namespace <= 5k

Pod Churn

“ Pod churn = (#Pod-creates|updates|deletes) per second”

Pod churn <= 20/s



<some caveats>

Some caveats:

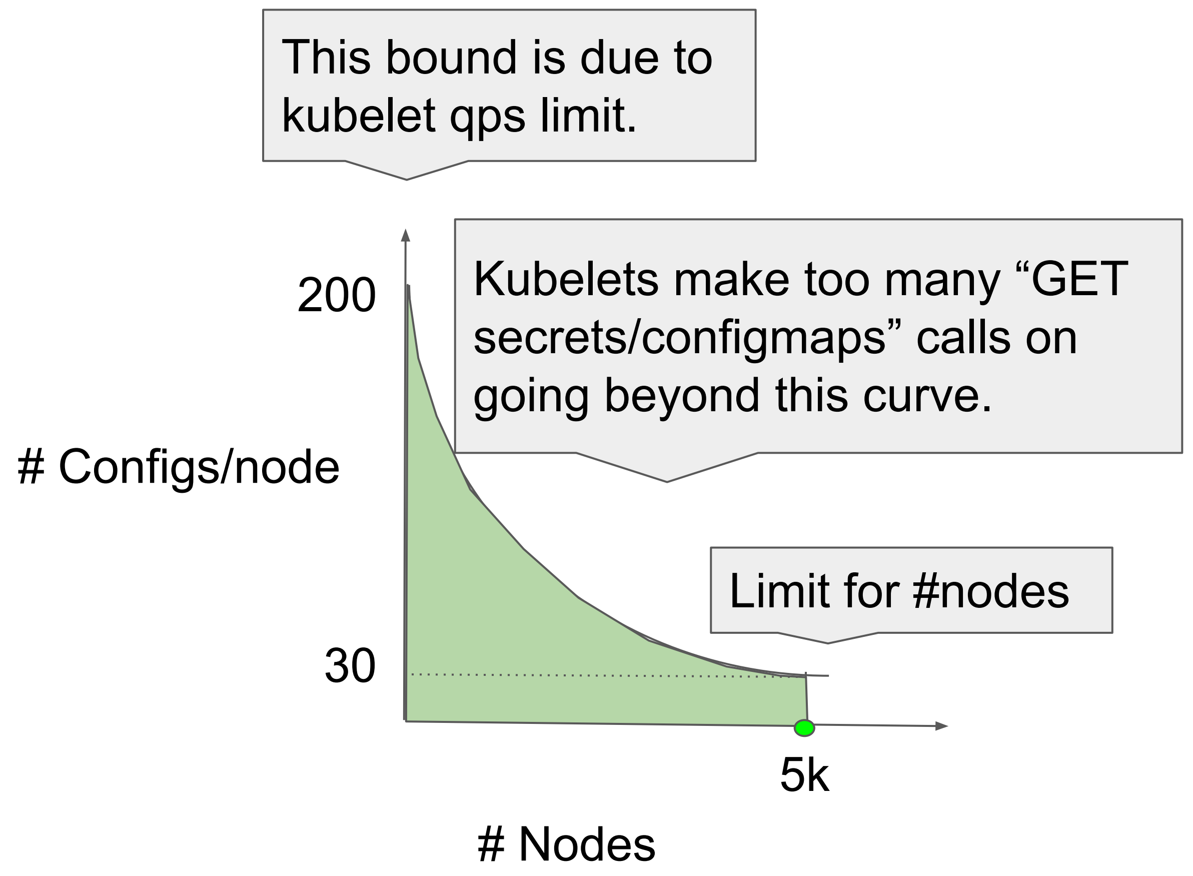
- You can go above 20 only if you’re manually changing pods, as controller-manager has default qps limit of 20

- For deletions through GC, only a throughput of 10/s can be achieved currently as each delete uses 2 API calls

- If pods belong to huge services, higher churn can affect control plane due to endpoints traffic

#Nodes vs #Configs/node

“#Configs/Node = Avg (# Unique secrets + # Unique configmaps) needed per node”



Σ_{nodes} #Configs <= 150k
&
#Nodes <= 5k

We got rid of this limitation in k8s 1.12 after moving kubelets to watch secrets.

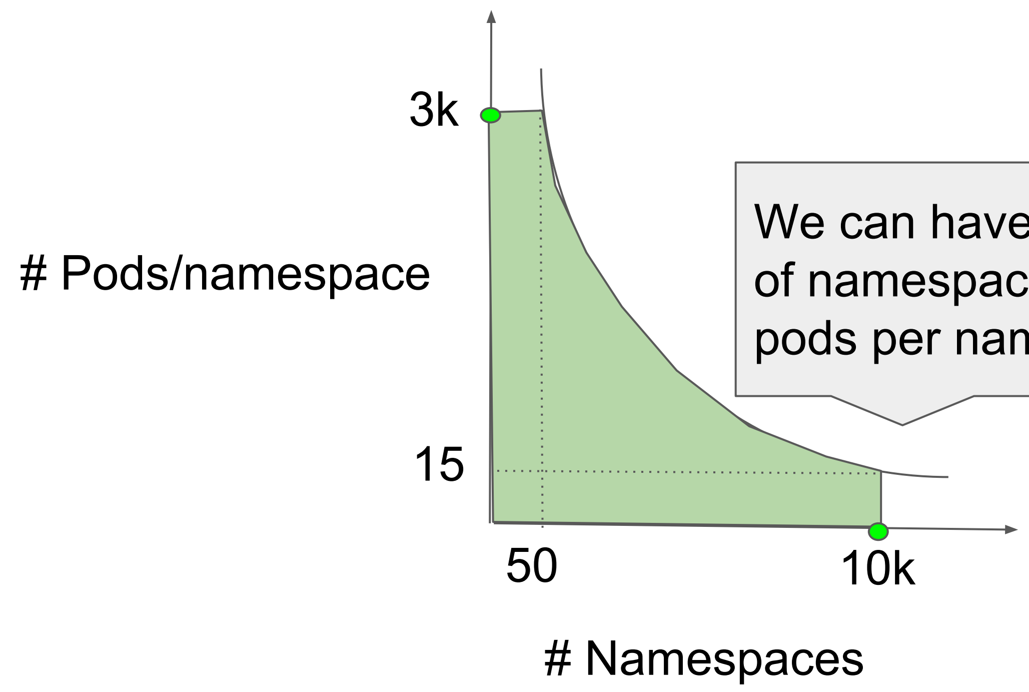
Few ways to mitigate it for versions < 1.12:

- Colocate pods needing same set of secrets on fewer nodes
- Don’t mount the default serviceAccount secret if your pods don’t need API access or namespace-based identity

#Namespaces vs #Pods/namespace

KubeCon | CloudNativeCon
North America 2018

Controllers may start seeing a performance drop as we increase #pods per namespace



We can have a large no. of namespaces with few pods per namespace

#Pods <= 150k
&
#Namespaces <= 10k
&
#Pods/namespace <= 3k

We got rid of the limitation on x-axis in k8s 1.12 after moving kubelets to watch secrets.

KubeCon | CloudNativeCon

North America 2018

Scalability: Next Steps

Knowing our bounds better

SIG scalability:

- tests '*plain vanilla*' configs, to find core k8s bounds
- doesn't test features from individual verticals, as then we can't scale horizontally.

So…

If you're a k8s developer:

- scale test your features, stressing/adding axes as relevant (use scale presubmits!)
- make the resulting envelopes you discover common knowledge (tell us!)

If you're a k8s user:

- let us know limits you've discovered/faced

Where to find us?

KubeCon | CloudNativeCon
North America 2018

SIG Scalability is happy to receive any feedback/questions through:

- Mailing list: kubernetes-sig-scale@googlegroups.com
- Slack channel: https://kubernetes.slack.com/messages/C09QZTRH7
- SIG meetings: https://zoom.us/j/989573207 (Thursdays 16:30 UTC, bi-weekly)
- SIG page: https://github.com/kubernetes/community/tree/master/sig-scalability

Tweet **#SIGScalability** or **#K8sScalability** with questions/feedback!

KubeCon | CloudNativeCon

North America 2018

Thank you!