Meltwater
Outside Insight

Scalability testing of a production Kubernetes cluster

Federico Hernandez & Simone Sciarrati

Meltwater
Outside Insight

30000+ customers

1500+ employees

55 global offices

Founded 2001 in Oslo

Meltwater
Engineering

9+ engineering offices (7+ countries)

350+ engineers in 50+ teams

AWS with 170+ accounts

	December 2018	May 2019	October 2019
	71	146	265
	1251	2102	3954
	283	561	944
	150	435	818
	25	77	103

Kubernetes Failure Stories

A compiled list of links to public failure stories related to Kubernetes. Most recent publications on top.

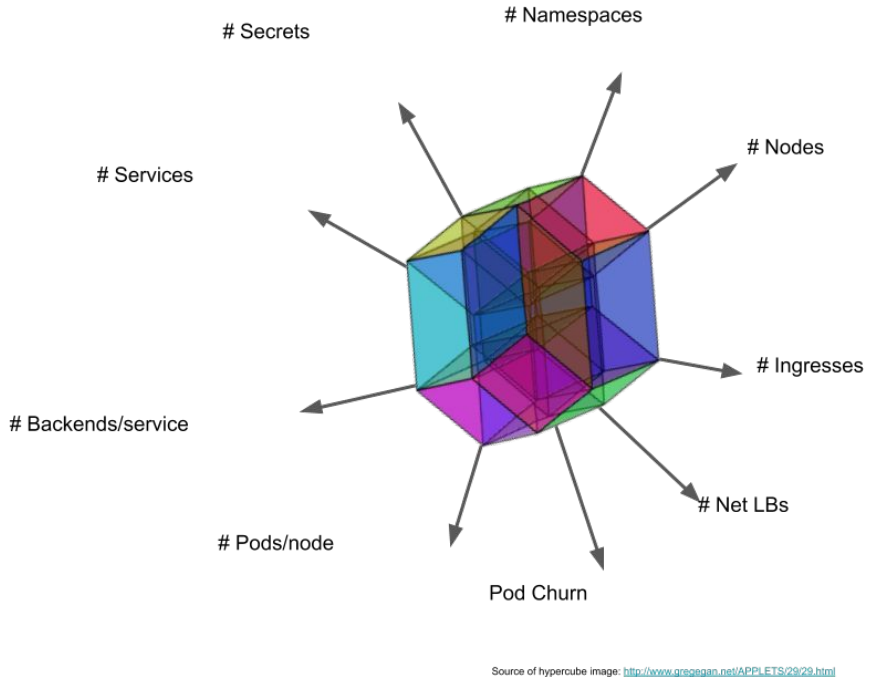
- A Kubernetes crime story - Prezi - blog post 2019
  - involved: AWS EKS, SNAT, conntrack, Amazon VPC CNI plugin
  - impact: delay of 1-3 seconds for outgoing TCP connections
- Postmortem: New K8s workers unable to join cluster - FREE NOW - postmortem 2019
  - involved: AWS spot instances, kops, CentOS, container-selinux
  - impact: insufficient cluster capacity in testing environments, failed deployments in production and staging environments
- How a simple admission webhook lead to a cluster outage - Jetstack - blog post 2019
  - involved: ValidatingWebhookConfiguration, GKE node auto-repair
  - impact: prolonged downtime of non-prod environment, nodes lost, failed master upgrade
- Post Mortem: Kubernetes Node OOM - Blue Matador - blog post 2019
  - involved: AWS, SystemOOM, EBS, fluentd-sumologic, no resource requests/limits
  - impact: unknown, Pods killed
- Kubernetes' dirty endpoint secret and Ingress - Ravelin - blog post 2019
  - involved: GKE, Ingress, replication controller, SIGTERM, "graceful shutdown"
  - impact: occasional 502 errors
- How a Production Outage Was Caused Using Kubernetes Pod Priorities - Grafana Labs 2019
  - involved: Pod priorities
  - impact: cascading Pod evictions
- Moving to Kubernetes: the Bad and the Ugly - Xing - ContainerDays EU 2019
  - involved: nginx Ingress, network interrupts, conntrack, frozen CronJob, PLEG, stuck controllers
  - impact: lost requests, response time jumps, not ready nodes
- Kubernetes Failure Stories, or: How to Crash Your Cluster - Zalando - ContainerDays EU 2019
  - involved: AWS IAM, Kubelet, `--kube-api-qps`, Skipper-Ingress, AWS, `OOMKill`, CronJob, CoreDNS, CPU throttling
  - impact: build errors, production outages
- Build Errors of Continuous Delivery Platform - Zalando - postmortem 2019

https://k8s.af



Kubernetes official thresholds

Quantity	Threshold: scope=namespace	Threshold: scope=cluster
#Nodes	n/a	5000
#Namespaces	n/a	10000
#Pods	3000	150000
#Pods per node	min(110, 10*#cores)	min(110, 10*#cores)
#Services	5000	10000
#All service endpoints	TBD	TBD
#Endpoints per service	250	n/a
#Secrets	TBD	TBD
#ConfigMaps	TBD	TBD
#Deployments	2000	TBD
#DaemonSets	TBD	TBD
#Jobs	TBD	TBD
#StatefulSets	TBD	TBD



https://github.com/kubernetes/community/blob/master/sig-scalability/configs-and-limits/thresholds.md

Kubernetes official SLOs

Latency of mutating API calls, 99th percentile:	`< 1s`
Latency of non-streaming read-only API class, 99th percentile:	`< 1s`
Startup latency of schedulable stateless pods, 99th percentile:	`< 5s`

Startup latency of schedulable stateful pods, 99th percentile:	`< ?s`
Latency of programming DNS, 99th percentile:	`< ?s`
In-cluster network latency from a single pod, 99th percentile:	`< ?s`

https://github.com/kubernetes/community/blob/master/sig-scalability/slos/slos.md

Under the condition that

Runs a single or more appropriately sized master machines

Events are stored in a separate etcd instance (or cluster)

All etcd instances are running on master machine(s)

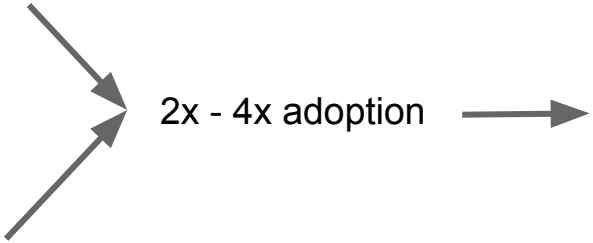
Cluster churn is <= 20, where churn is defined as:

```
Churn = #(Pod spec creations/updates/deletions)
      + #(user originated requests) in a given second
```

Defining a real-world scalability test

- Current cluster size
- Architecture
- Utilization

- Thresholds
- SLOs



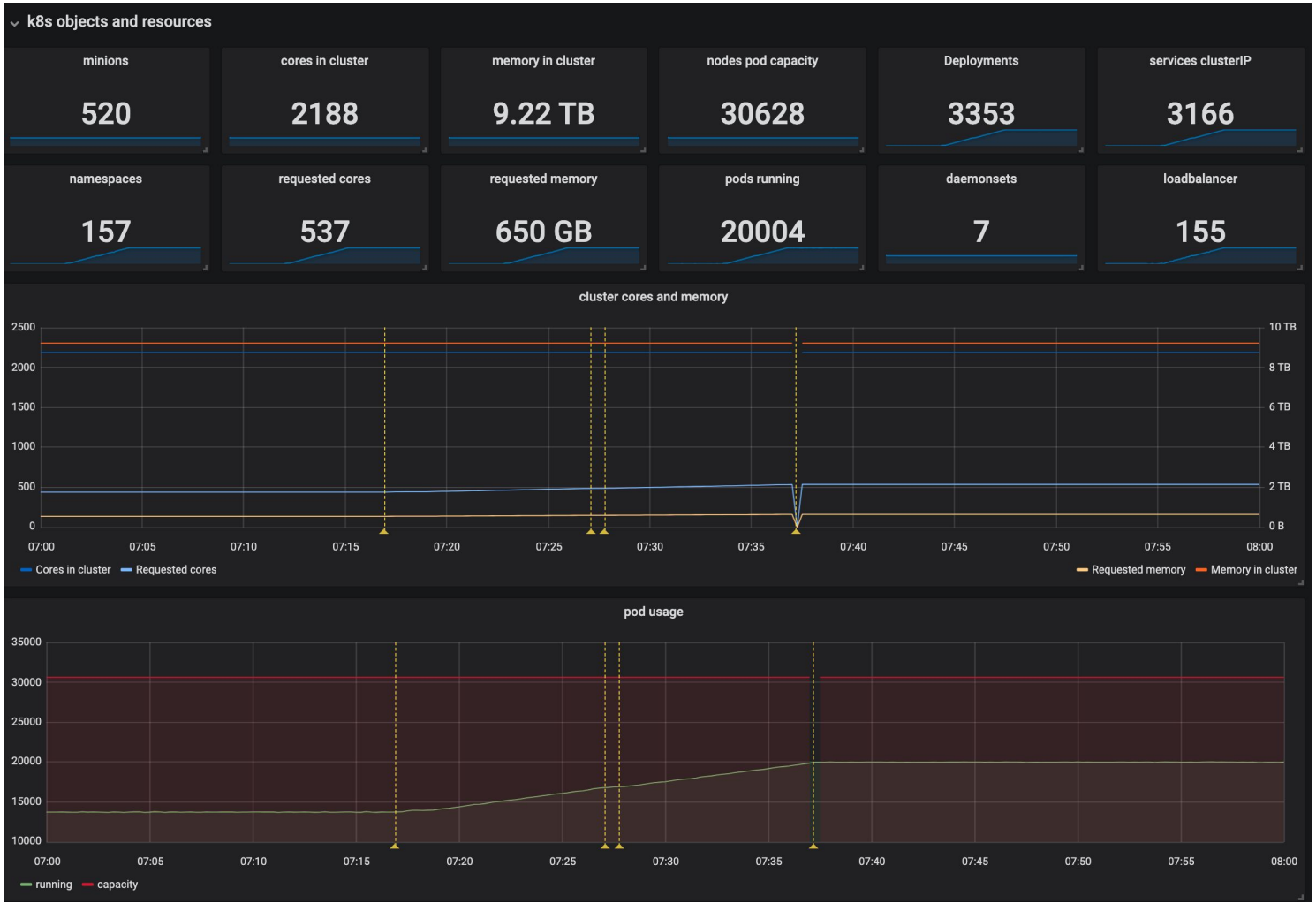
2x - 4x adoption

Nodes	NS	Dep/NS	Pods/Dep
2x	=	=	=
=	2x	2x	=
2x	2x	2x	=
4x	2x	2x	=

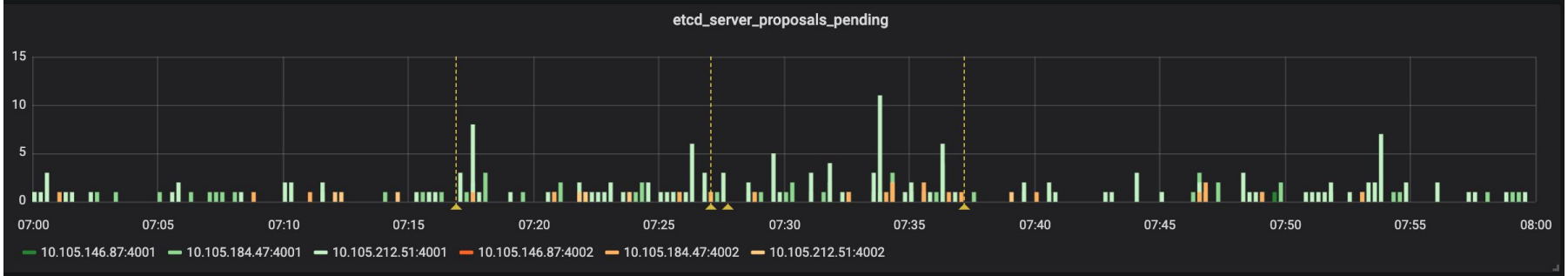
Known parameters ***Hypothesis*** ***Validation***


N446SP
SMOKING PROHIBITED

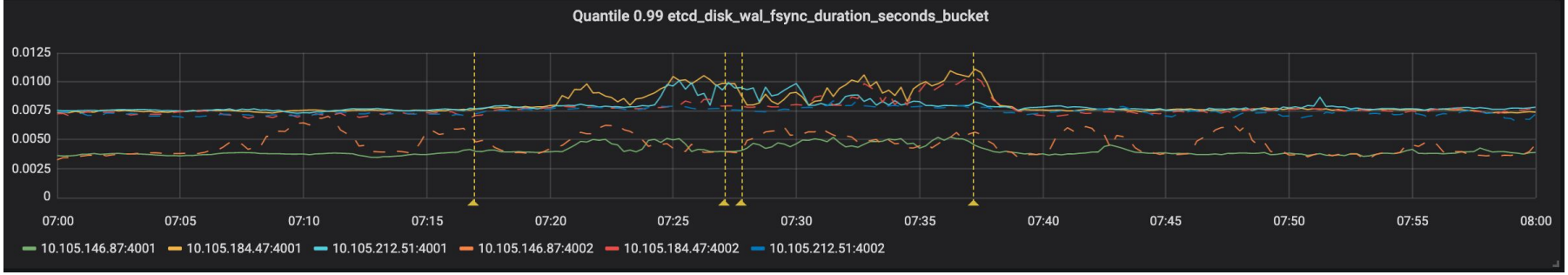
- **k8s objects and resources** *(14 panels)*
- **pod and node state** *(2 panels)*
- **aws cni** *(32 panels)*
- **etcd** *(16 panels)*
- **kube-dns** *(2 panels)*
- **masters cpu memory** *(6 panels)*
- **k8s slo** *(5 panels)*

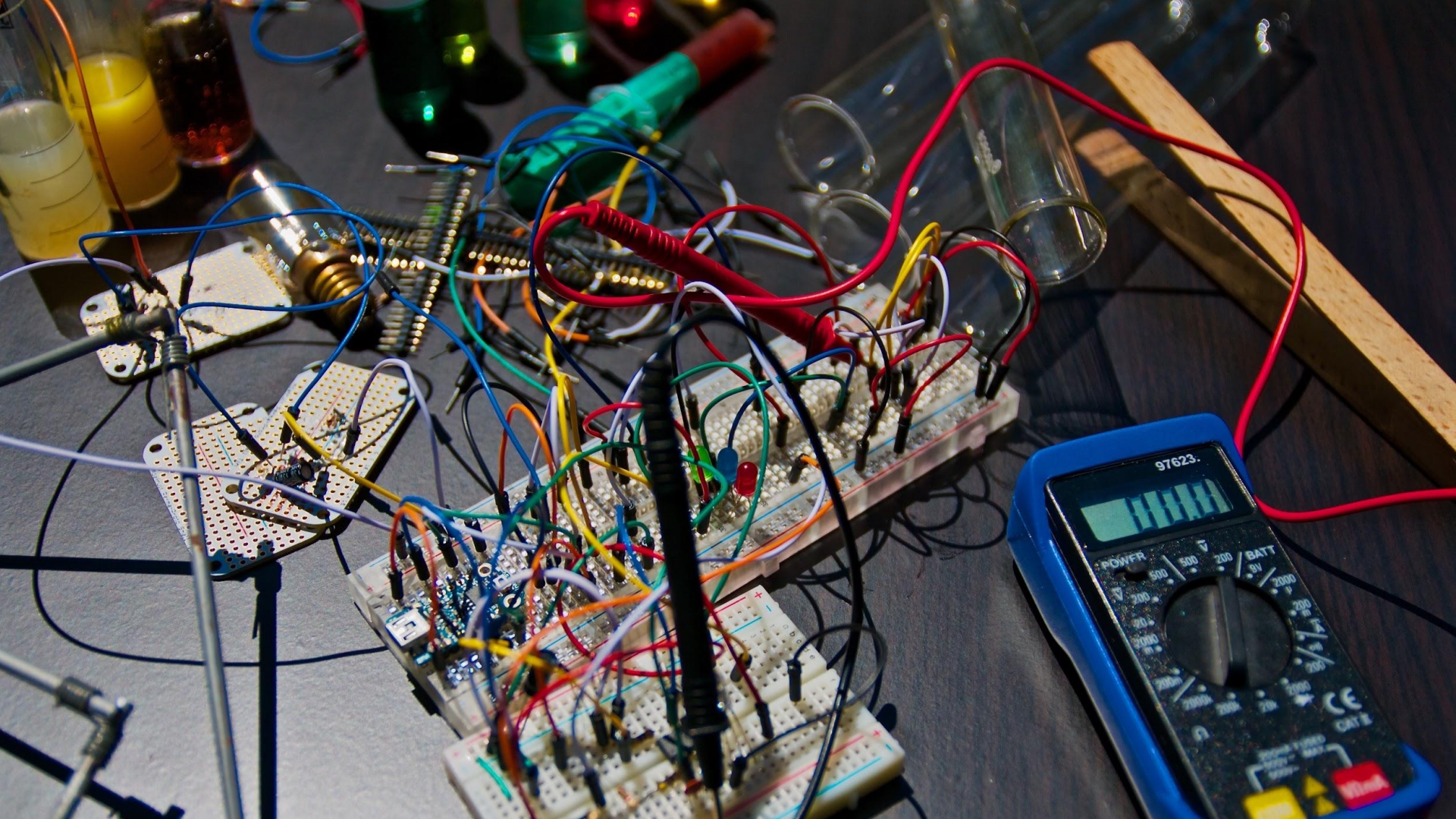

k8s objects and resources
minions
520
cores in cluster
2188
memory in cluster
9.22 TB
nodes pod capacity
30628
Deployments
3353
services clusterIP
3166
namespaces
157
requested cores
537
requested memory
650 GB
pods running
20004
daemonsets
7
loadbalancer
155
cluster cores and memory
2500
2000
1500
1000
500
0
10 TB
8 TB
6 TB
4 TB
2 TB
0 B
07:00
07:05
07:10
07:15
07:20
07:25
07:30
07:35
07:40
07:45
07:50
07:55
08:00
Cores in cluster
Requested cores
Requested memory
Memory in cluster
pod usage
35000
30000
25000
20000
15000
10000
07:00
07:05
07:10
07:15
07:20
07:25
07:30
07:35
07:40
07:45
07:50
07:55
08:00
running
capacity

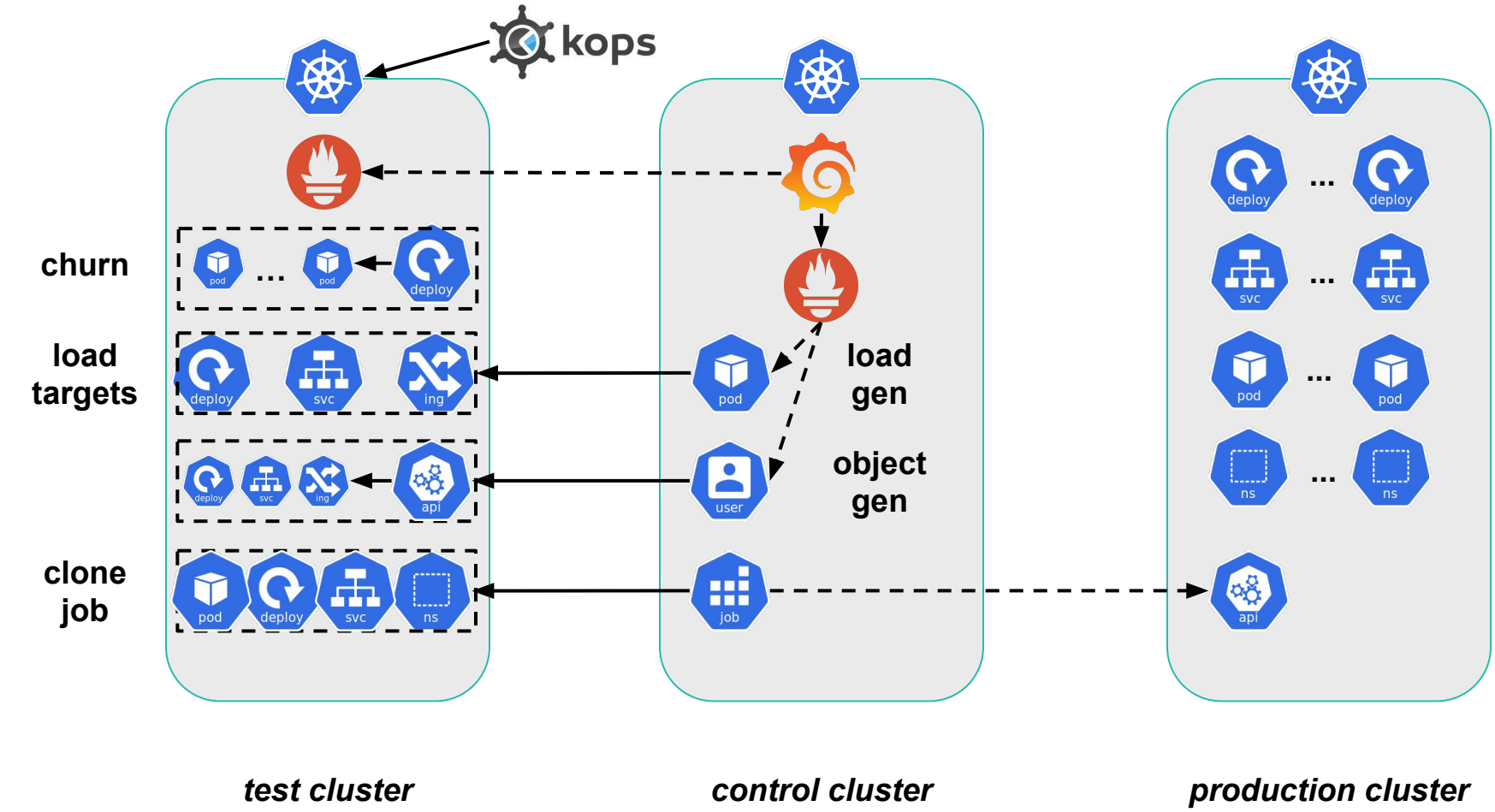
etcd

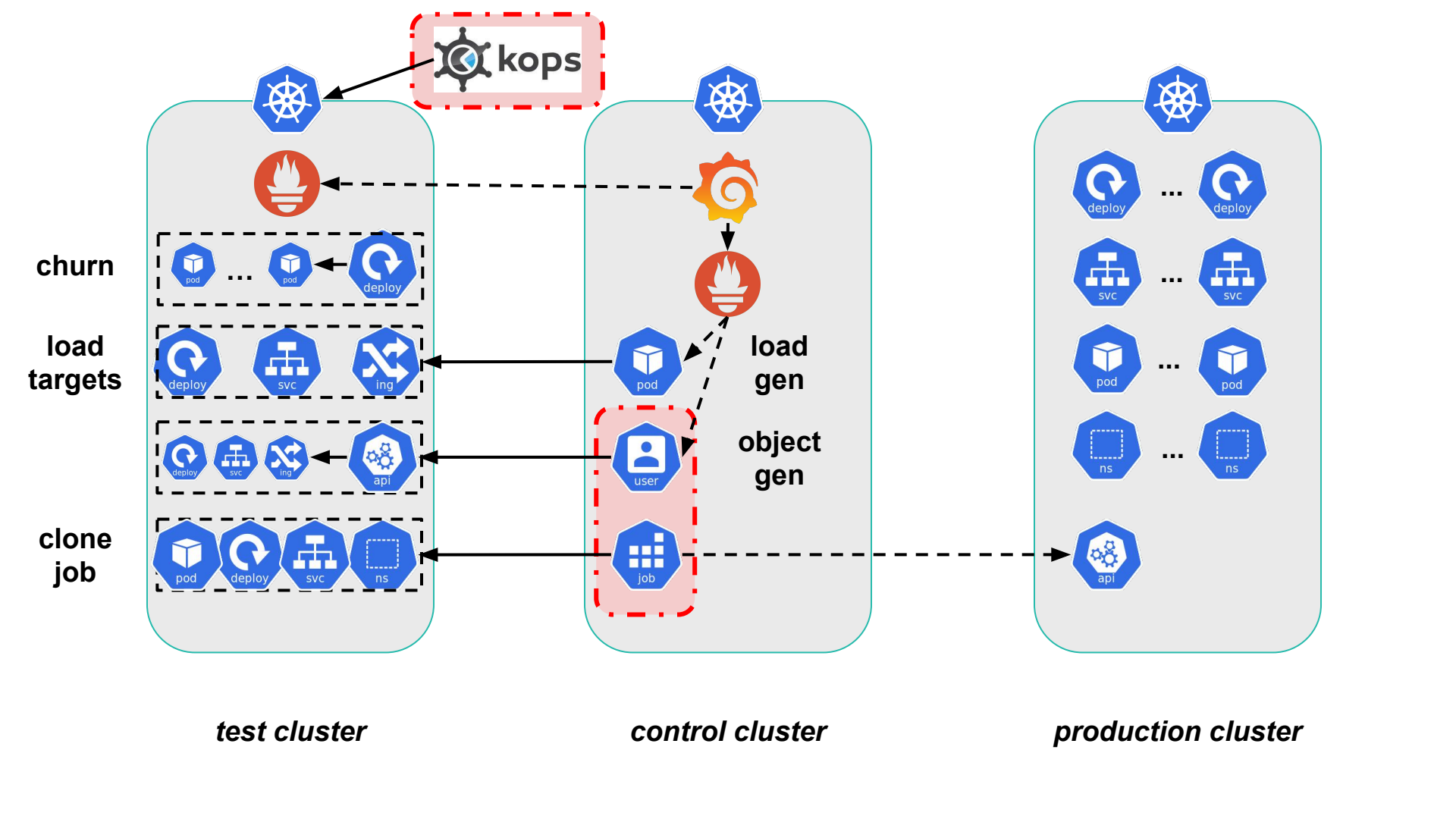

etcd_server_proposals_pending
15
10
5
0
07:00
07:05
07:10
07:15
07:20
07:25
07:30
07:35
07:40
07:45
07:50
07:55
08:00
10.105.146.87:4001
10.105.184.47:4001
10.105.212.51:4001
10.105.146.87:4002
10.105.184.47:4002
10.105.212.51:4002


etcd_server_proposals_committed per second
200
150
100
50
0
07:00
07:05
07:10
07:15
07:20
07:25
07:30
07:35
07:40
07:45
07:50
07:55
08:00
10.105.146.87:4001
10.105.184.47:4001
10.105.212.51:4001
10.105.146.87:4002
10.105.184.47:4002
10.105.212.51:4002


Quantile 0.99 etcd_disk_wal_fsync_duration_seconds_bucket
0.0125
0.0100
0.0075
0.0050
0.0025
0
07:00
07:05
07:10
07:15
07:20
07:25
07:30
07:35
07:40
07:45
07:50
07:55
08:00
10.105.146.87:4001
10.105.184.47:4001
10.105.212.51:4001
10.105.146.87:4002
10.105.184.47:4002
10.105.212.51:4002


97623.
POWER
BATT


kops
churn
load targets
clone job
pod
deploy
svc
ing
api
ns
job
user
load gen
object gen
test cluster
control cluster
production cluster


kops
churn
load targets
clone job
pod
deploy
svc
ing
api
ns
job
user
load gen
object gen
test cluster
control cluster
production cluster

Results and Learnings

Results and Learnings

Kubernetes itself is quite robust.

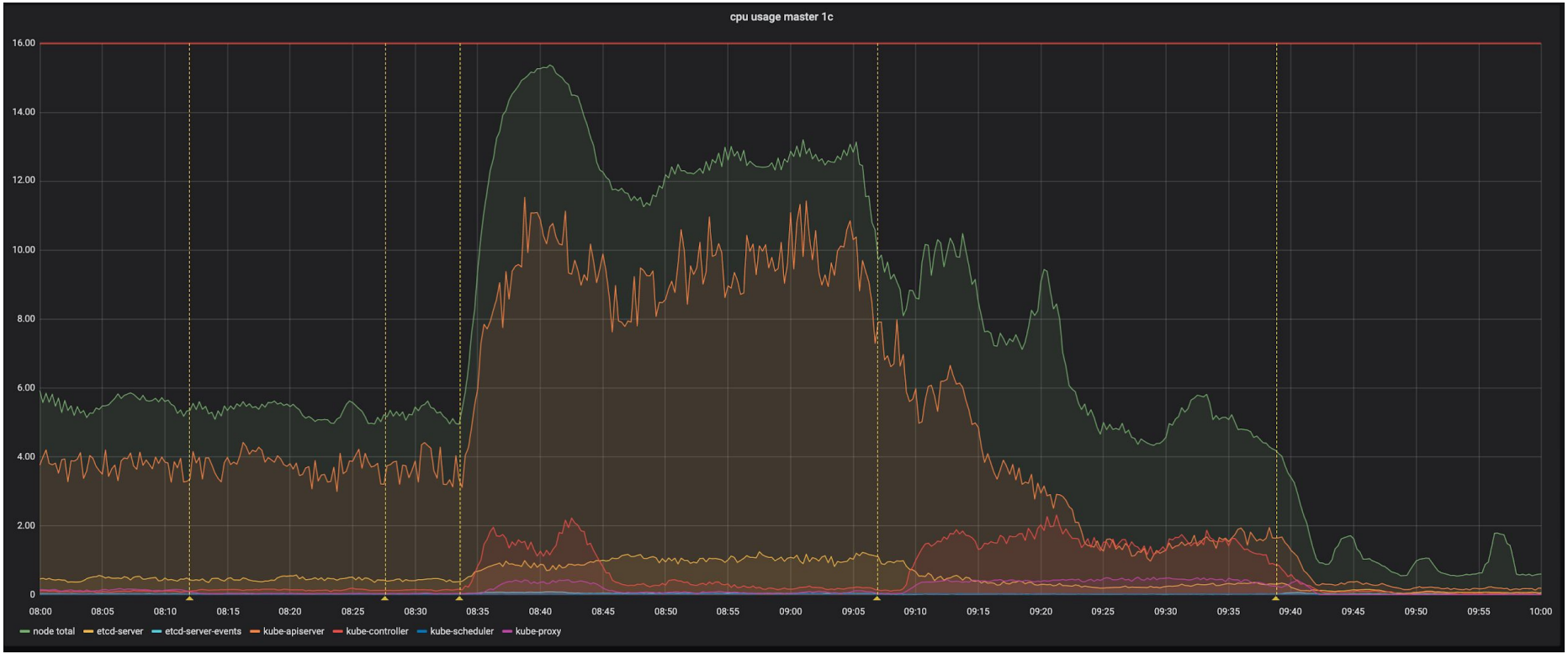
We could scale up to 4x the number of nodes and workloads of our current production cluster.

But…

User experience starts to degregate without performance tuning.

Some components for monitoring and add-on functionality needed resource tuning and different setup to cope with the large cluster.

Going from 500 to 900 nodes



https://kubernetes.io/docs/setup/best-practices/cluster-large/

Federico Hernandez
@recollir

Simone Sciarrati
@dezmodue