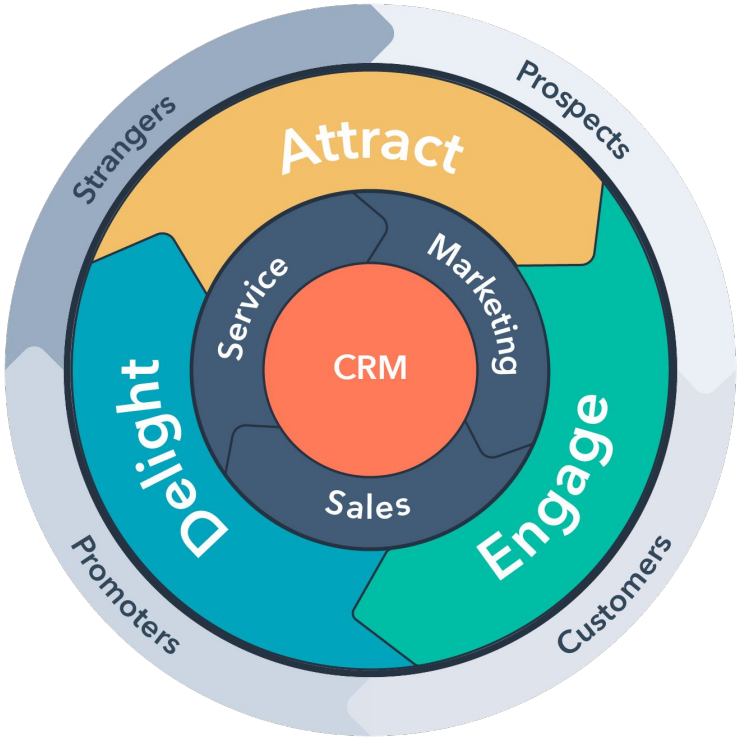
Vitess at HubSpot

How We Moved Hundreds of MySQL Databases into Kubernetes

Tom Petr, Alex Charis
KubeCon 2018

What is HubSpot?



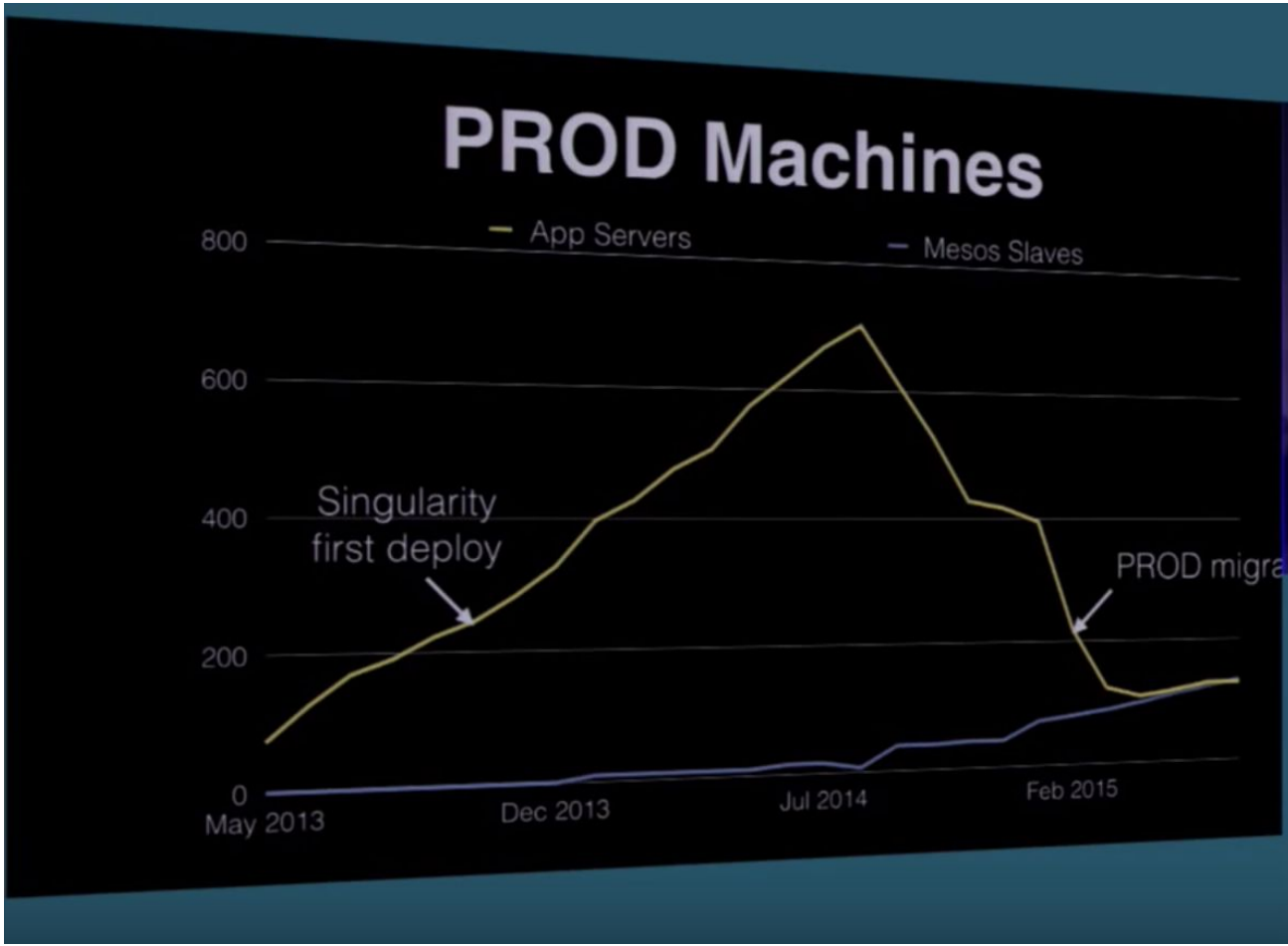
- Thousands of microservices
- Hundreds of deploys each day
- Many small, autonomous teams

2013...

- Clunky CI pipeline
- Flaky ssh-based deployments
- No elasticity
- Zero automation

MESOS

Singularity

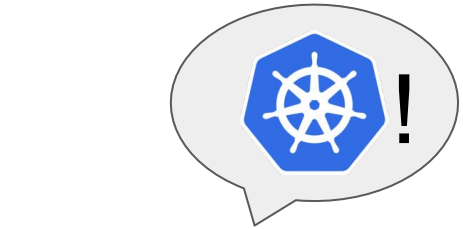


#MesosCon

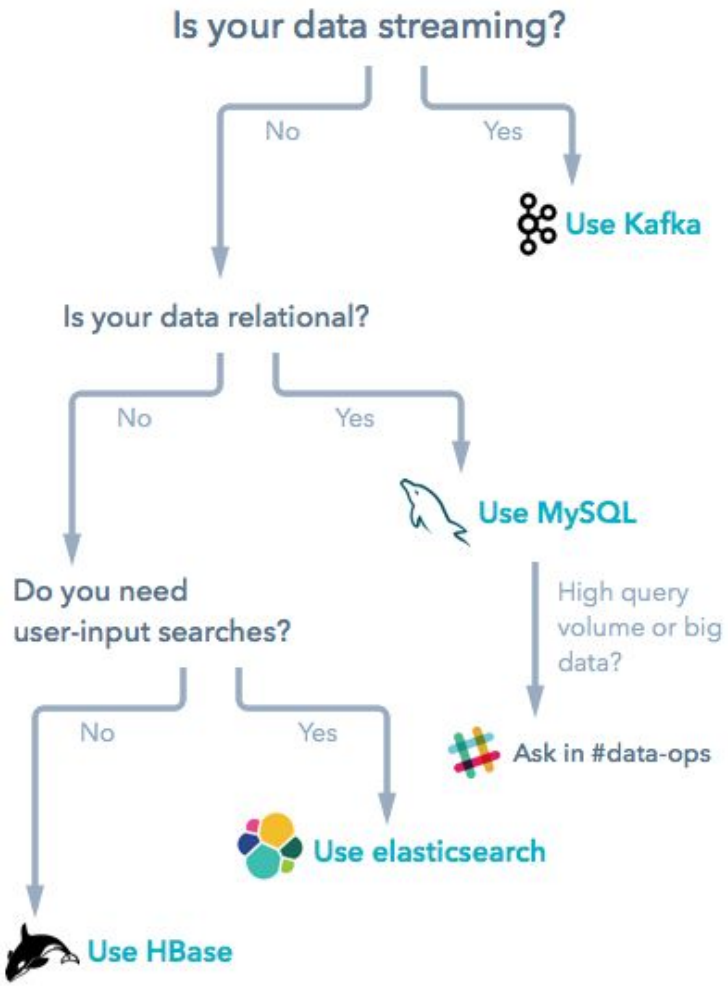
AUGUST 20-21, 2015

SEATTLE, WASHINGTON

...2016: What about infrastructure?



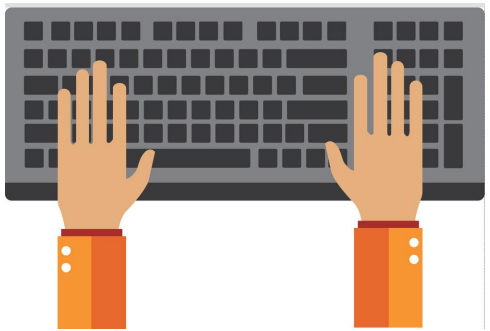
Data at HubSpot in 2016



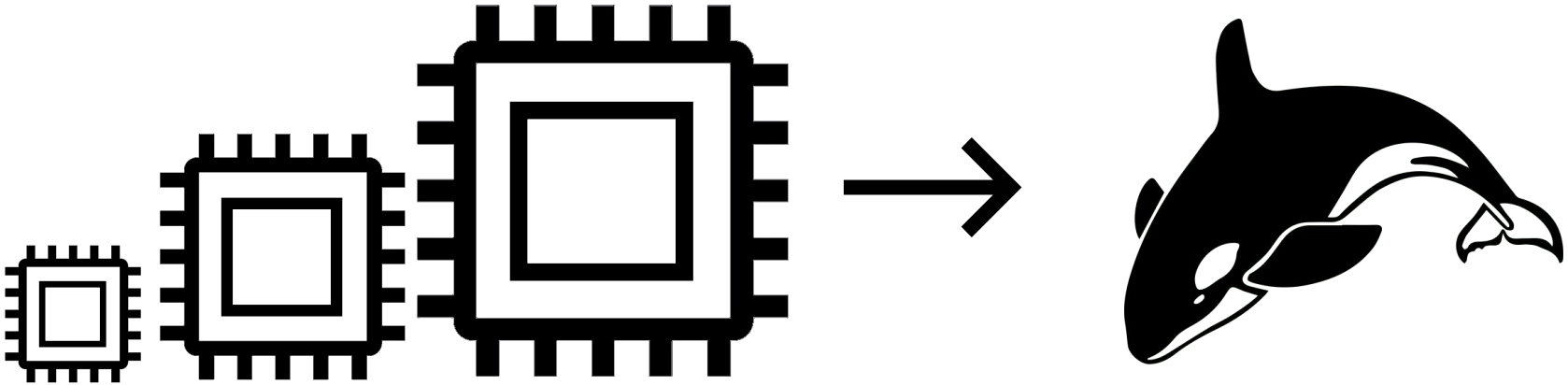
MySQL Scalability Woes

MySQL Scalability Woes

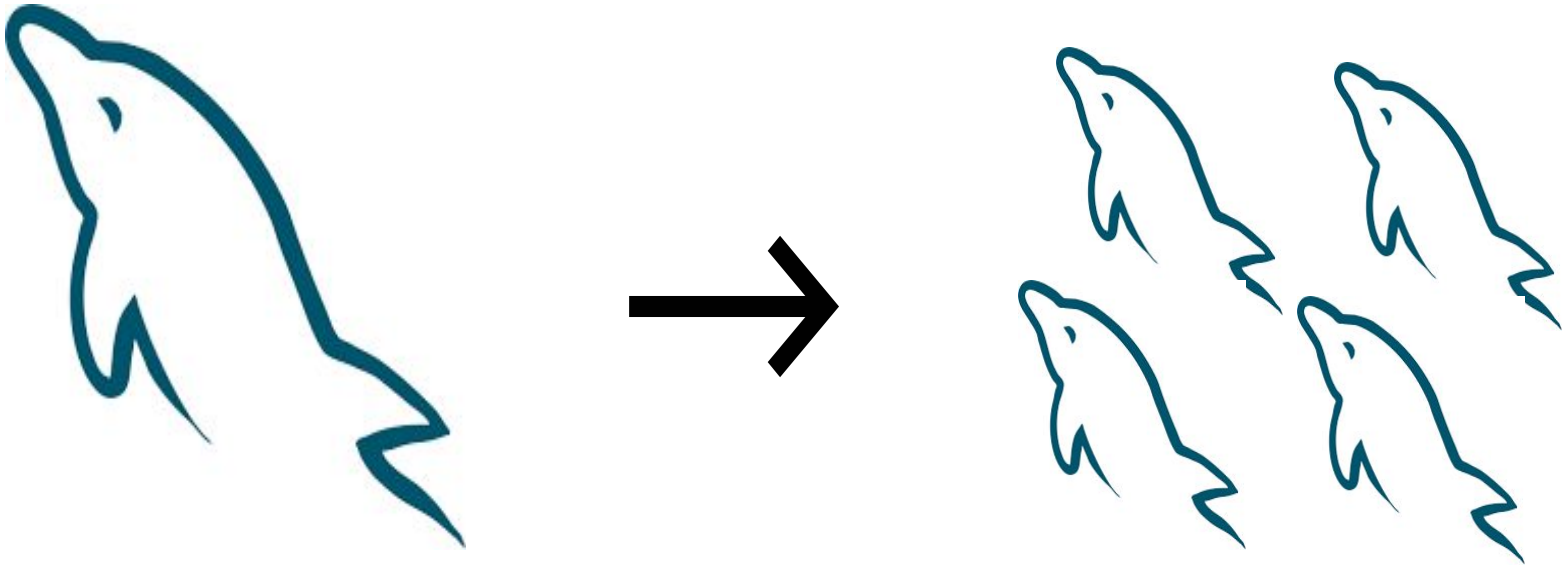
amazon →



MySQL Scalability Woes

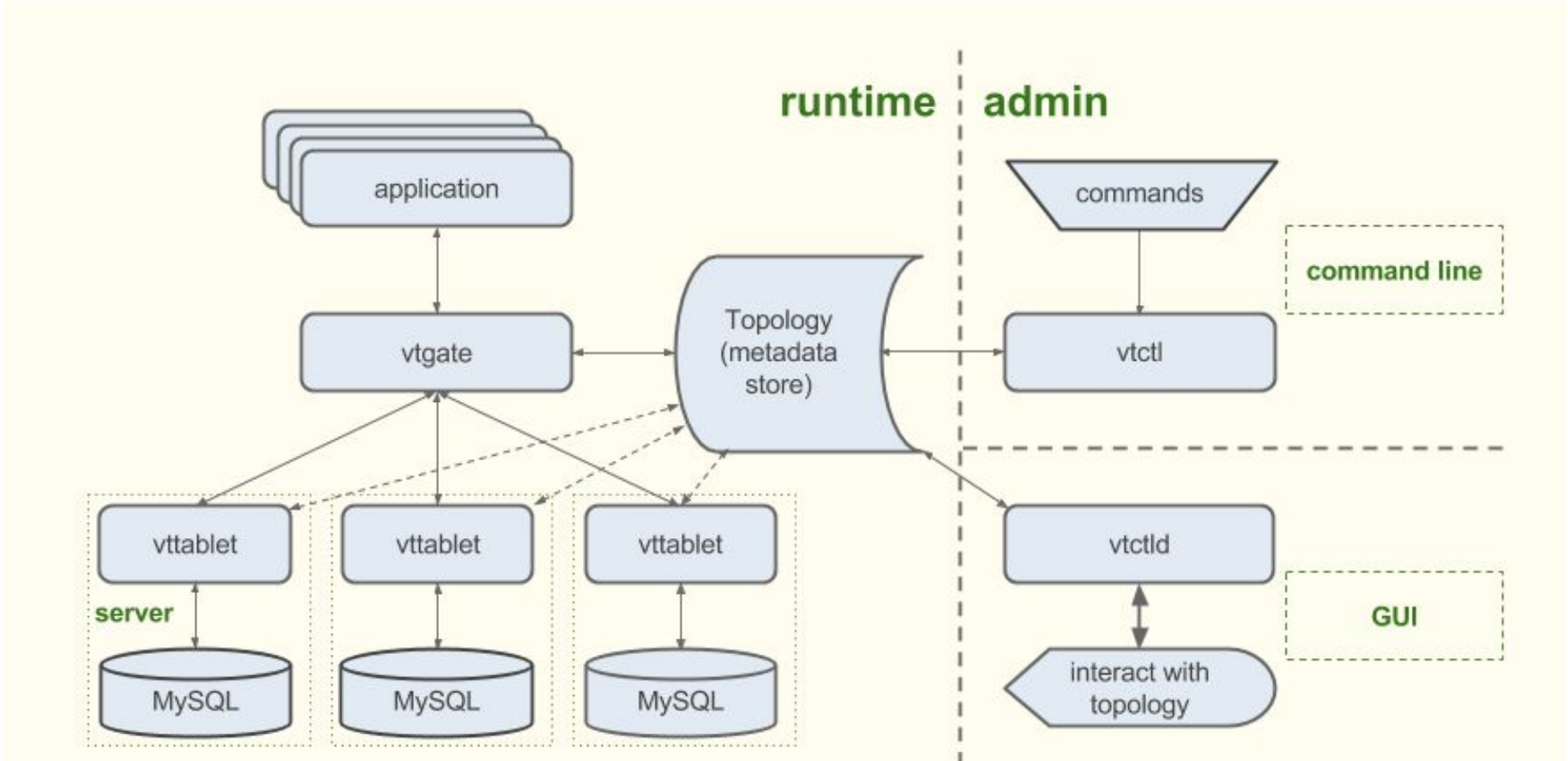


MySQL Scalability Woes



So what now?

What is Vitess?



But how?

But how?

CoreOS Blog

All CoreOS Posts | Technical Posts | Announcements

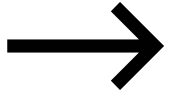
← Back to All Blogs

Introducing Operators: Putting Operational Knowledge into Software

November 03, 2016 • By Brandon Philips

Tags: announcements Operators

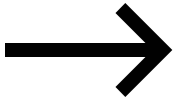
What's an operator?



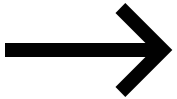
StatefulSet, PDB,
CronJob,
Deployment,
Service

Custom Resources

```
apiVersion: vitess.hubspot.com/v1beta1
kind: Keyspace
metadata:
  name: memes-0
  ...
spec:
  keyspace: Memes
  performanceClass: medium
  replicas: 3
  sensitive: true
  ...
status:
  phase: Running
  reason: ""
```



```
apiVersion: policy/v1beta1
kind: PodDisruptionBudget
metadata:
  ...
spec:
  maxUnavailable: 1
  ...
```



```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: memes-0-backup
  ...
spec:
  concurrencyPolicy: Forbid
  jobTemplate:
    spec:
      parallelism: 1
      template:
        spec:
          affinity:
            podAntiAffinity:
              ...
          restartPolicy: Never
          terminationGracePeriodSeconds: 86400
          volumes:
          - flexVolume:
  schedule: 32 */09 * * *
  ...
```

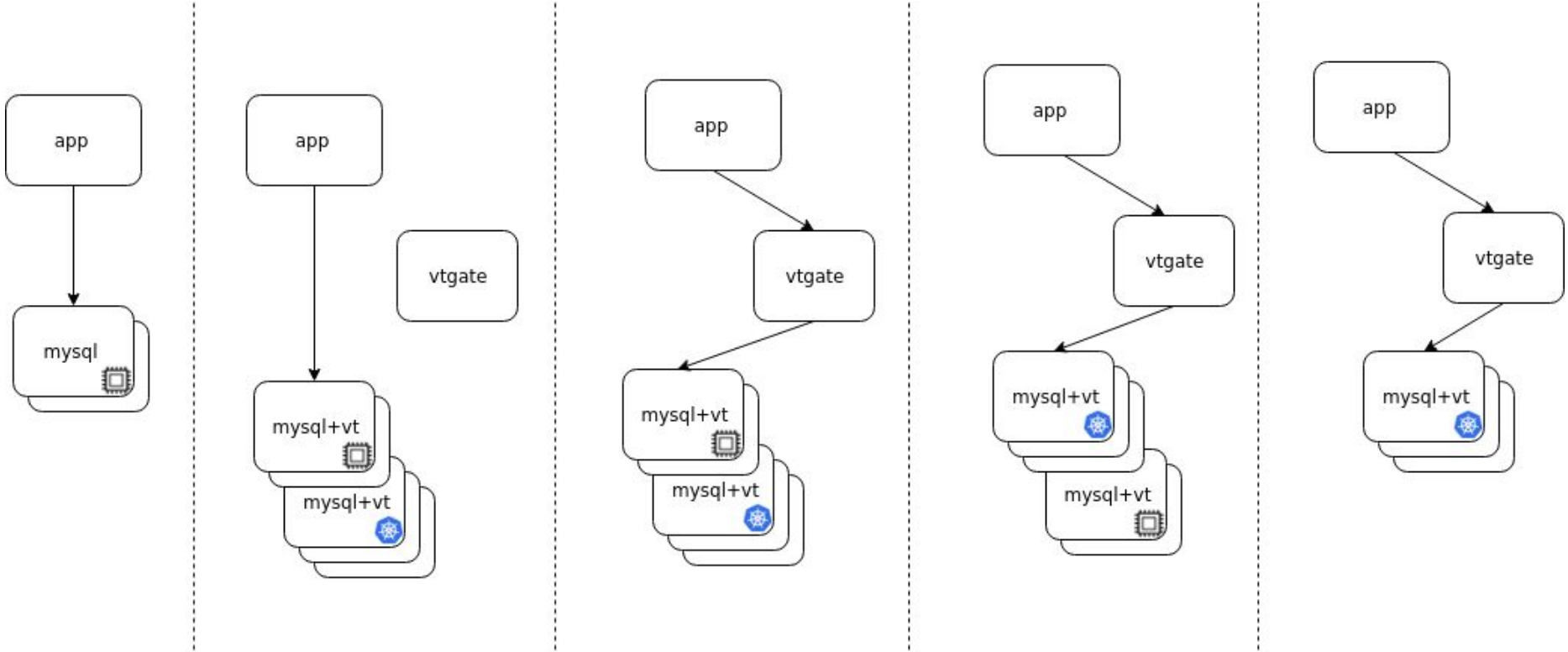
```
apiVersion: apps/v1
kind: StatefulSet
▶ metadata:
spec:
  podManagementPolicy: Parallel
  replicas: 3
  revisionHistoryLimit: 10
▶ selector:
  serviceName: vttablet
  template:
  ▶ metadata:
    spec:
      containers:
      -
      ▶ env:
        image: docker.hubteam.com/vitess-internal/tablet:2421
        imagePullPolicy: IfNotPresent
        lifecycle:
        ▶ preStop:
      ▶ livenessProbe:
        name: vttablet
      ▶ ports:
      ▶ readinessProbe:
      ▶ resources:
      ▶ volumeMounts:
      -
      ▶ env:
        image: docker.hubteam.com/vitess-internal/tablet:2421
        imagePullPolicy: IfNotPresent
        lifecycle:
        ▶ preStop:
        name: mysql
      ▶ ports:
      ▶ resources:
      ▶ volumeMounts:
      -
      ▶ env:
        image: docker.hubteam.com/vitess-internal/cron:2425
        imagePullPolicy: IfNotPresent
        name: cron
      ▶ resources:
      ▶ volumeMounts:
      -
      ▶ env:
        image: docker.hubteam.com/fluentd/mysql-forwarder:39
        imagePullPolicy: IfNotPresent
        name: fluentd
      ▶ volumeMounts:
      -
      ▶ env:
        image: docker.hubteam.com/vitess-internal/collectd:2409
        imagePullPolicy: IfNotPresent
        name: collectd
      ▶ resources:
      ▶ volumeMounts:
      dnsPolicy: ClusterFirst
      initContainers:
      -
      ▶ command:
      ▶ env:
        image: docker.hubteam.com/vitess-internal/tablet:2421
        imagePullPolicy: IfNotPresent
        name: init-vtroot
      ▶ resources:
      ▶ volumeMounts:
      restartPolicy: Always
      schedulerName: default-scheduler
      serviceAccount: vitess
      serviceAccountName: vitess
      terminationGracePeriodSeconds: 86400
    ▶ volumes:
  updateStrategy:
    type: OnDelete
  ▶ volumeClaimTemplates:
▶ status:
```

Lots of other work!

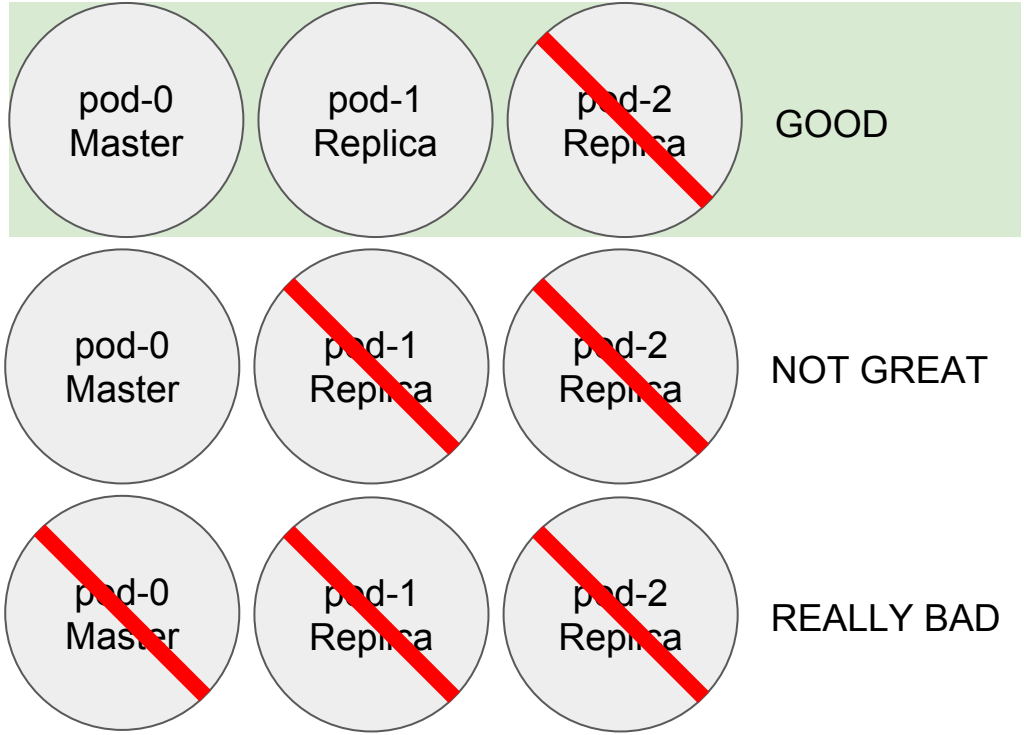
```
apiVersion: vitess.hubspot.com/v1beta1
kind: VtgatePool
▶ metadata:
spec:
  ▶ availabilityZones:
    defaultReplicasPerZone: 4
    enableSSL: true
    ...
▶ status:
```

```
select_statement:
  base_select order_by_opt limit_opt lock
  {
    sel := $1.(*Select)
    sel.OrderBy = $2
    sel.Limit = $3
    sel.Lock = $4
    $$ = sel
  }
| union_lhs union_op union_rhs order_by_o
  {
    $$ = &Union{Type: $2, Left: $1, Right
  }
| SELECT comment_opt cache_opt NEXT num_v
  {
    $$ = &Select{Comments: Comments($2),
```

Migration

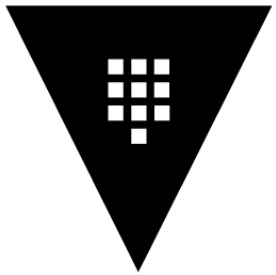

app
mysql
app
vtgate
mysql+vt
mysql+vt
app
vtgate
mysql+vt
mysql+vt
app
vtgate
mysql+vt
mysql+vt
app
vtgate
mysql+vt

Protecting against cluster maintenance




```
apiVersion: policy/v1beta1
kind: PodDisruptionBudget
spec:
  maxUnavailable: 1
  selector:
    matchLabels:
      app: vitess
      component: vttablet
      keyspace: Memes
      role: serving
      shard: "0"
```

Injecting Vault secrets into Pods



Vault

Secret Updater

```
apiVersion: v1
kind: Secret
metadata:
  name: thegoods
data:
  aws_access_key: aGVsbG8K
```

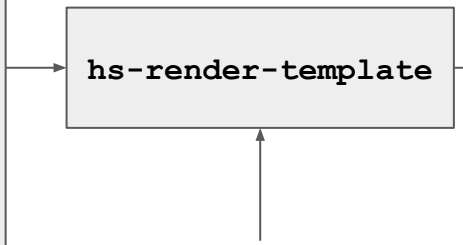
```
apiVersion: hubspot.com/v1
kind: SecretTemplate
metadata:
  name: thegoods
data:
  aws_access_key:
    vaultKey: mykey
vaultMount: hs-engineering
```

Parameterizing configuration in Docker images

```
# Exporting here vs including as part of
# the container env is necessary
# because cron does not inherit
# the root environment
export AWS_ACCESS_KEY="{{dirs.vault.access_key}}"
export AWS_SECRET_KEY="{{dirs.vault.secret_key}}"

# Always flush logs
mysql -e "FLUSH BINARY LOGS"

exec binlog-backup
```



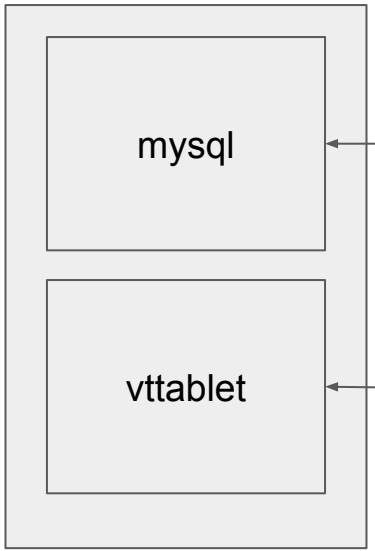
Downward API
Node metadata
YAML files
Env vars

```
# Exporting here vs including as part of
# the container env is necessary
# because cron does not inherit
# the root environment
export AWS_ACCESS_KEY="XXXX"
export AWS_SECRET_KEY="XXXX"

# Always flush logs
mysql -e "FLUSH BINARY LOGS"

exec binlog-backup
```

Terminating Pods gracefully




```
while tablet-is-running; do
    sleep 5
done
```

```
if [[ $TABLET_TYPE == “master” ]]; then
  vtctl PlannedReparentShard ...
fi
```

(Nearly) infinite resources

```
NAME                                  READY   STATUS    RESTARTS   AGE
important-service-74d75fcd54-xai2g    0/1     Pending   0          2d
important-service-74d75fcd54-z35cf    0/1     Pending   0          2d
important-service-74d75fcd54-9aolm    0/1     Pending   0          2d
important-service-74d75fcd54-iu3hx    0/1     Pending   0          2d
important-service-74d75fcd54-2y2h9    0/1     Pending   0          2d
important-service-74d75fcd54-zqrjm    0/1     Pending   0          2d
important-service-74d75fcd54-i0lpy    0/1     Pending   0          2d
important-service-74d75fcd54-ihl83    0/1     Pending   0          2d
important-service-74d75fcd54-eyuvd    0/1     Pending   0          2d
important-service-74d75fcd54-al2cz    0/1     Pending   0          2d
```

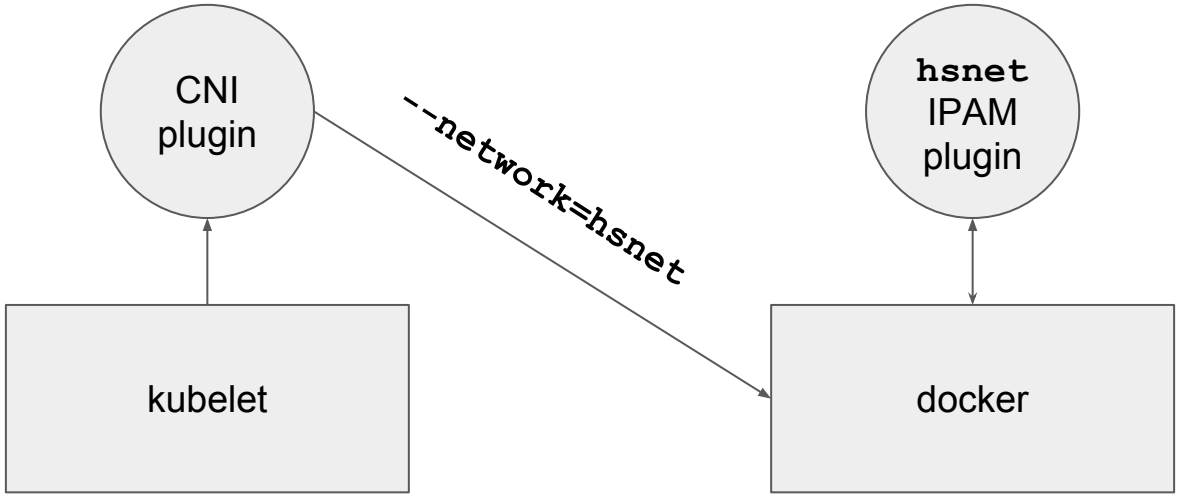
Hacking around bugs

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gp2-xfs
provisioner: kubernetes.io/aws-ebs
parameters:
  type: gp2
  fsType: xfs
```

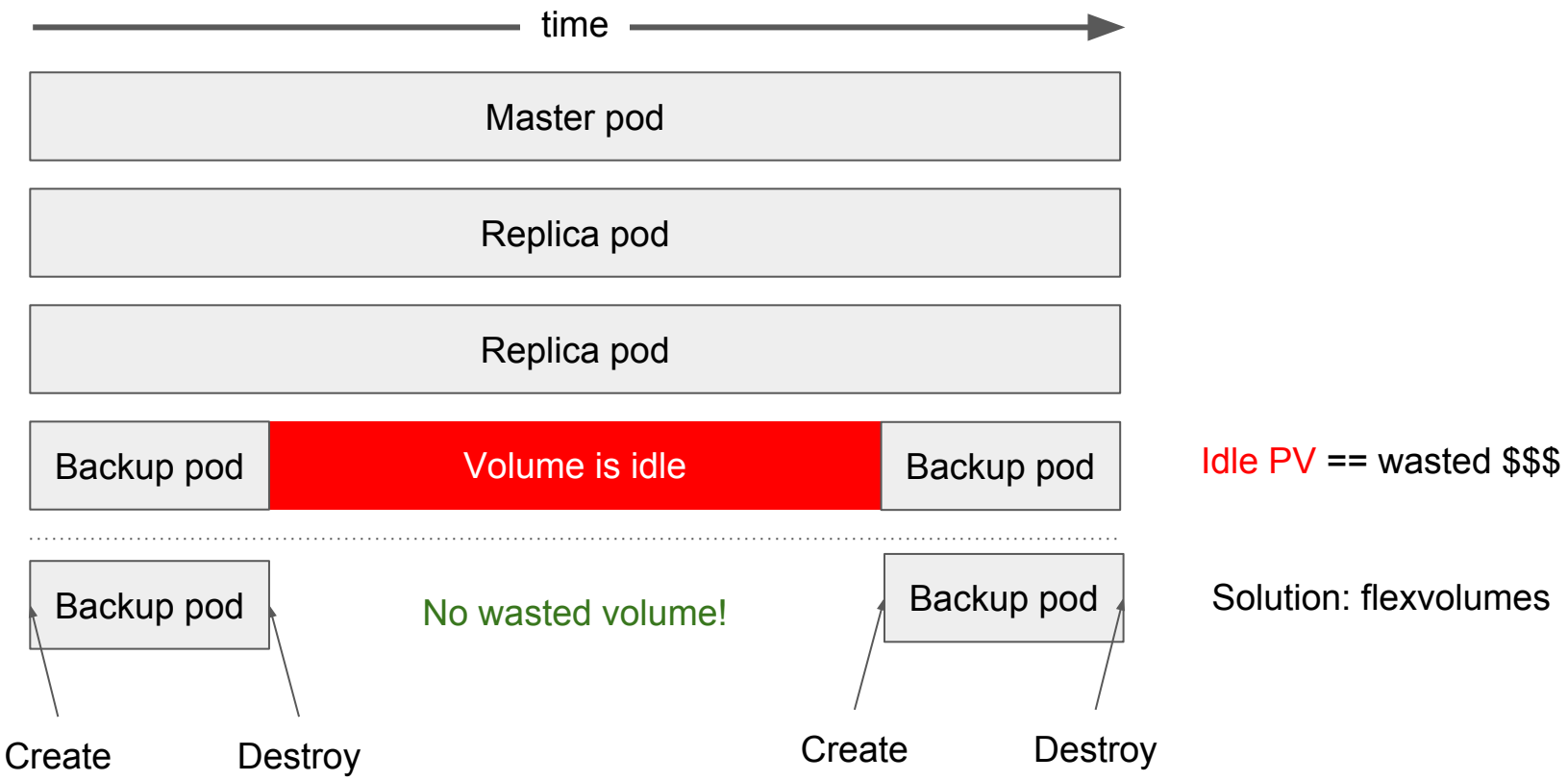
Fixed in #45345

Protecting against rapid IP address reuse

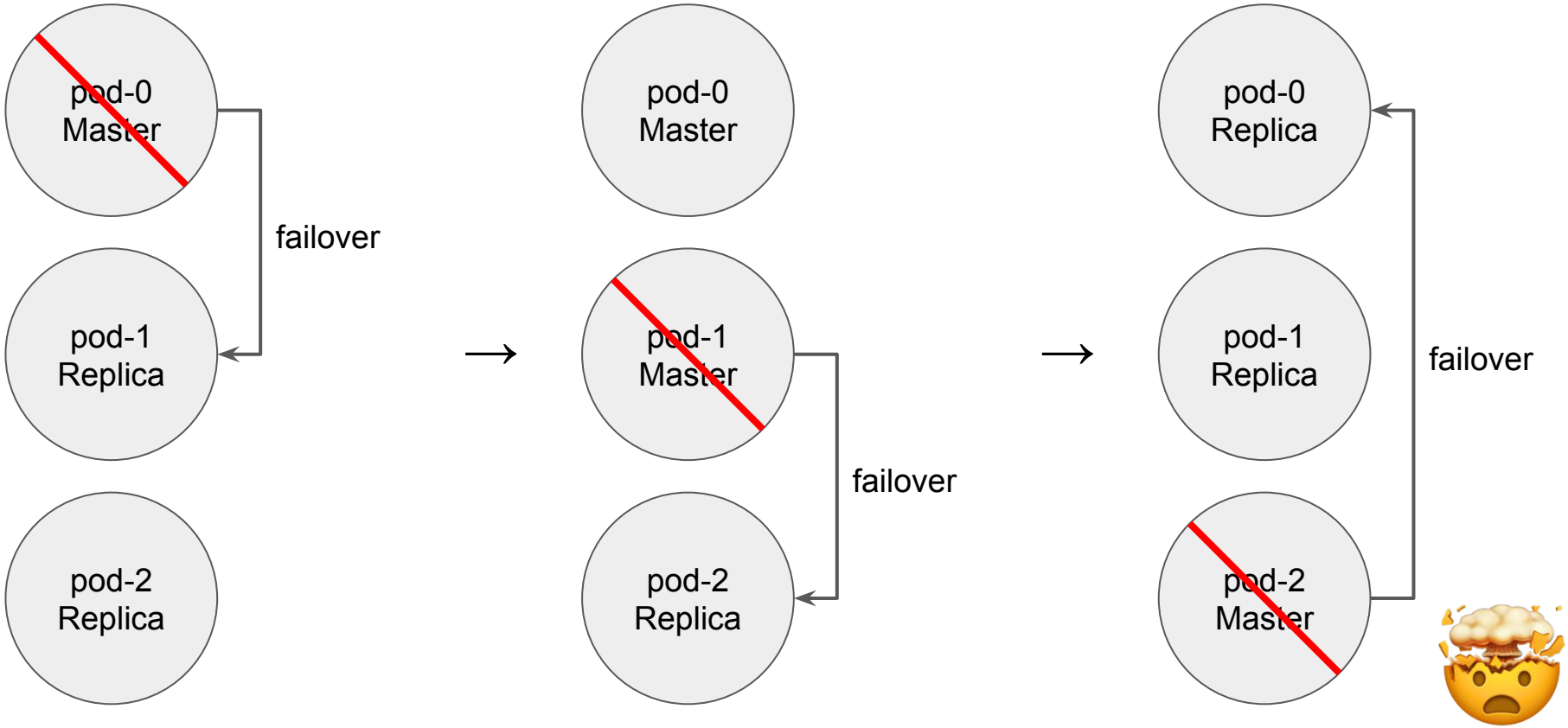
- Docker allocates IPs lowest-first by default
- Created our own CNI+IPAM plugin to allocate FIFO instead



Optimizing Vitess Backups

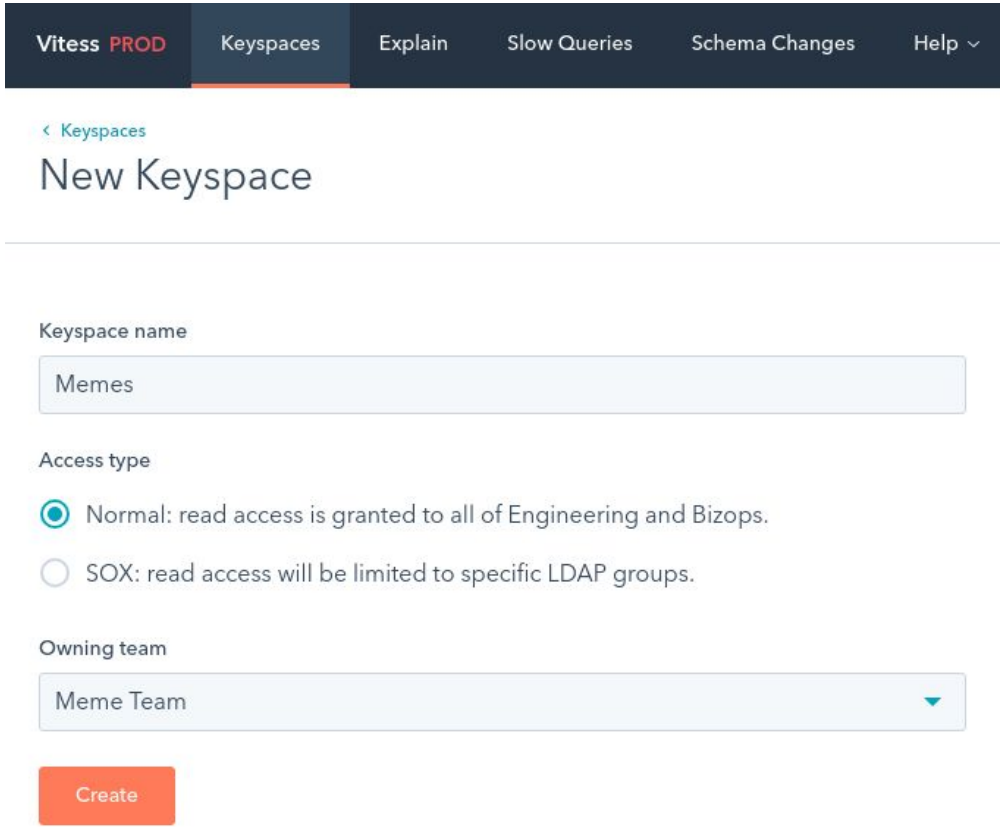


Controlling rollouts



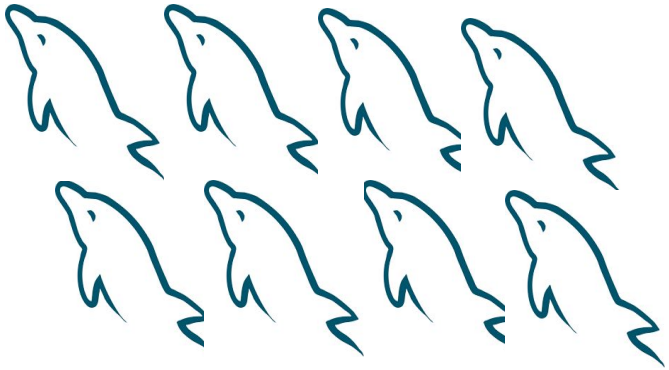
Lessons learned

Results

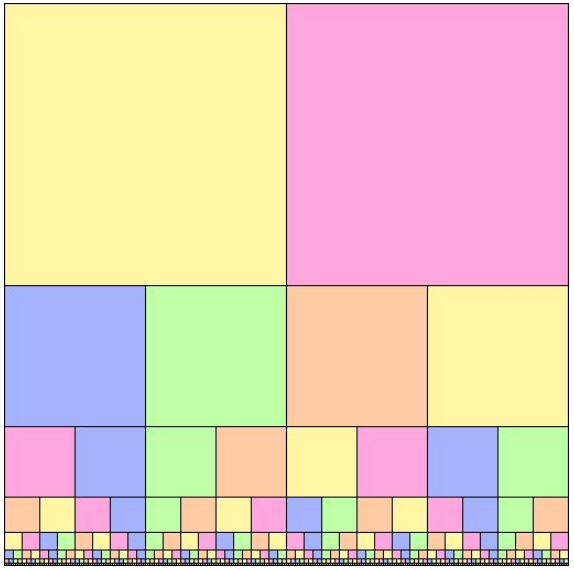



```
apiVersion: vitess.hubspot.com/v1beta1
kind: Keyspace
metadata:
  name: memes-0
  ...
spec:
  keyspace: Memes
  performanceClass: medium
  replicas: 3
  sensitive: true
  ...
status:
  phase: Running
  reason: ""
```

Results



Results



Thanks and Q&A

vitess.io

hubspot.com