KubeCon | CloudNativeCon

Europe 2025

KubeCon | CloudNativeCon Europe 2025

Mastering Efficiency in Argo CD

Scaling Smarter, Not Costlier

Intro

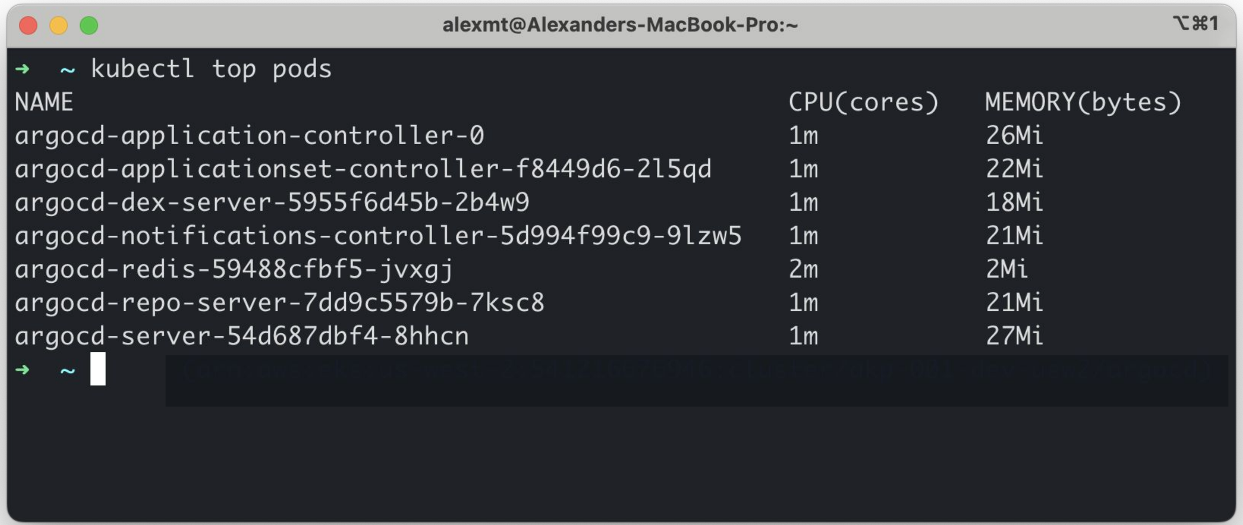
Alexander Matyushentsev

Argo Project co-creator
Co-founder and Chief Architect at Akuity

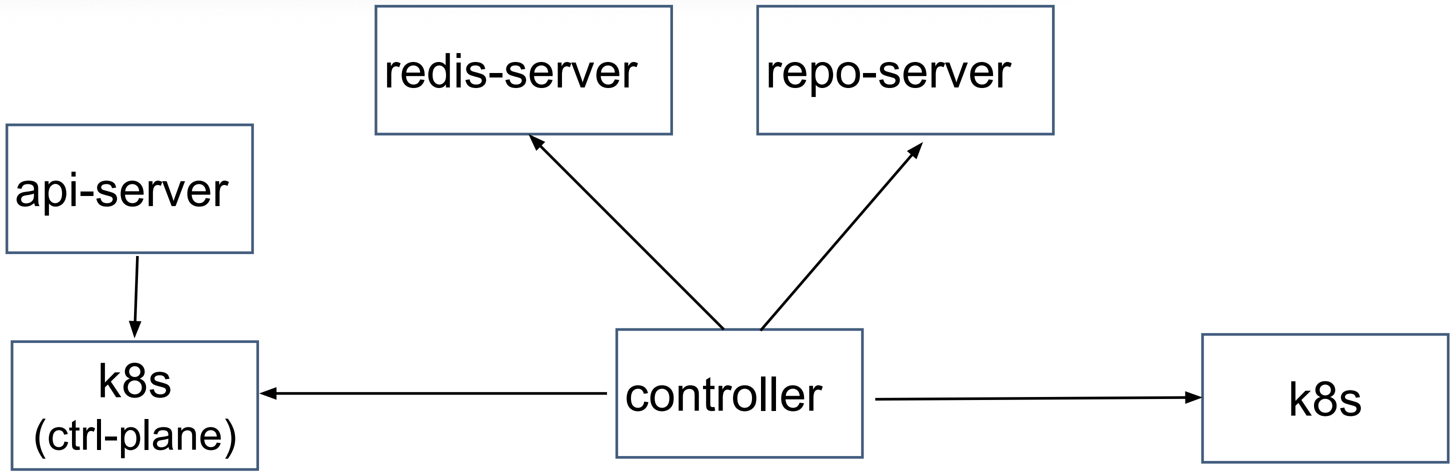
Agenda

- Why worry about it?
  - Argo CD is efficient and cheap!
  - When can it really get expensive?
- How much is expensive?
  - How much does it really cost and when to start worrying about the price?
- How to make it cheaper?

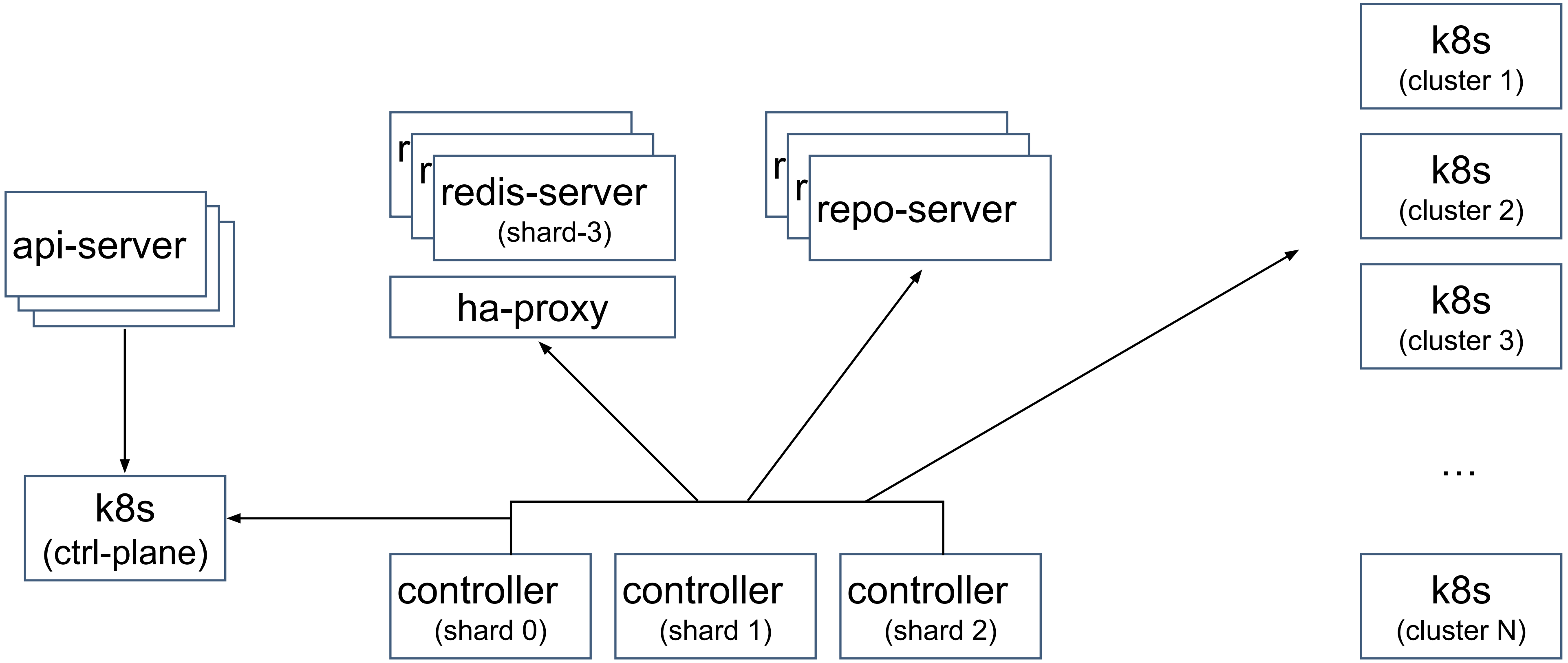
Argo CD is efficient and cheap!



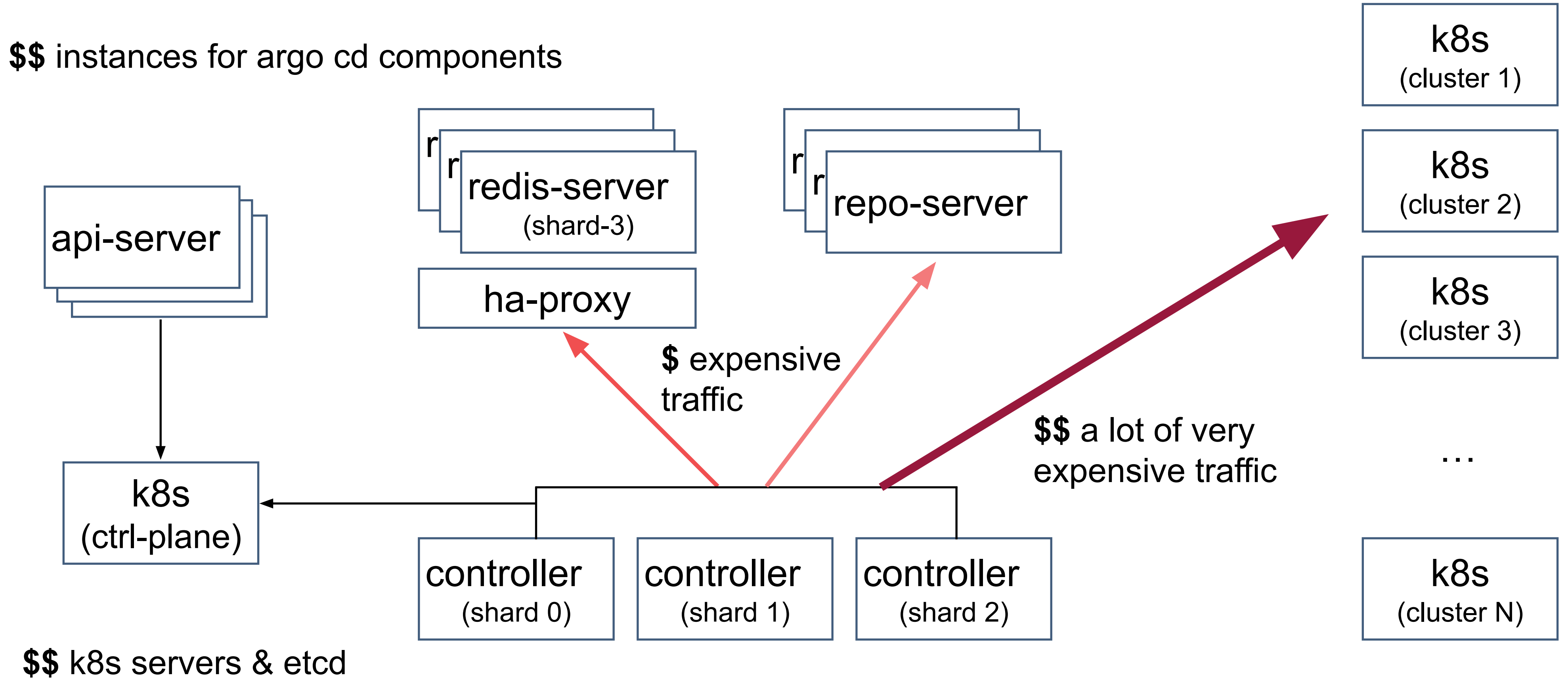
```
alexmt@Alexanders-MacBook-Pro:~                                    ⌥⌘1
➜  ~ kubectl top pods
NAME                                              CPU(cores)   MEMORY(bytes)
argocd-application-controller-0                   1m           26Mi
argocd-applicationset-controller-f8449d6-215qd    1m           22Mi
argocd-dex-server-5955f6d45b-2b4w9                1m           18Mi
argocd-notifications-controller-5d994f99c9-9lzw5  1m           21Mi
argocd-redis-59488cfbf5-jvxgj                     2m           2Mi
argocd-repo-server-7dd9c5579b-7ksc8               1m           21Mi
argocd-server-54d687dbf4-8hhcn                    1m           27Mi
➜  ~
```



Centralized control planes are complicated



… and require costly infrastructure



How expensive? We've measured it for you!

Cluster Configuration

- Cloud: GCP
- Node Count: 3
- Node Type: n2-standard-8 (8vCPU, 32GB per node)
- Location: US Central (Multi-Zonal)

Argo CD Configuration

- HA Argo CD installation
- 3 controller shards

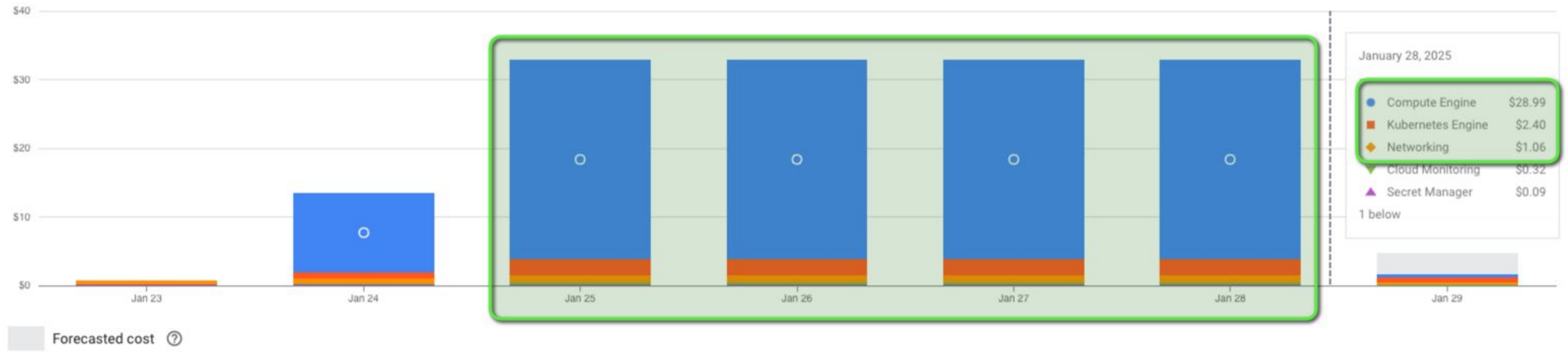
Applications

- 800 deployed Applications

Managed Clusters

- 3 clusters managed by Argo CD
- Accessed over internet
- Manually sharded in Argo CD for even distribution

Practical Results



January 28, 2025

●	Compute Engine	$28.99
■	Kubernetes Engine	$2.40
◆	Networking	$1.06
▼	Cloud Monitoring	$0.32
▲	Secret Manager	$0.09

1 below

Download CSV

	Service	Cost	Discounts	Promotions & others	↓ Subtotal	% Change
●	Compute Engine	$127.77	–	–	$127.77	New
■	Kubernetes Engine	$11.38	–	–	$11.38	New
◆	Networking	$5.85	–	–	$5.85	↑ 39%

Result Extrapolation

Cluster Cost	Network Cost	Total Infra Cost
$31.40/Day	$7.07/Day	$38.47/Day
$973.40/Month	$219.17/Month	$1,192.57/Month
$11,461.00/Year	$2,580.62/Year	$14,041.62/Year

Result Extrapolation

$1,192.57/800 apps = $1.49/App/Month

Applications	Estimated Monthly Cost	Estimated Annual Cost
800	$1,192	$14,041
1,500	$2,235	$26,327
3,000	$4,470	$52,655
6,000	$8,940	😵

Let’s make it cheaper!

Networking

- Cross AZ control plane K8S traffic
- Cross AZ Redis traffic
- Over internet controller traffic

Compute Cost

- Over-provisioned Redis
- Over-provisioned repo-server

Control Plane K8S

- Dedicated K8S server

Cross AZ control plane K8S traffic

Problem

- Controller “watches” Application CRDs which causes a LOT of cross AZ traffic - hundreds MBs per second
- Might cost **hundreds of $ per day** assuming price 0.01$ per 1Gb

Reason 1: Large Applications

- Engineers create Argo CD Application CRD with very large specs: up to 1MB of JSON
  - Inlined Helm values
- Controller makes frequent small patch requests
- each patch request triggers 2 (1 PATCH + 1 WATCH) responses (2 * 1MB) with full Application definition

Solution:

- Reduce application revision history limit using spec.revisionHistoryLimit
- Encourage moving values into values.yaml file & limit number of sources

Reason 2: Resource Health in Application CRD

- Controller stores detailed resource health information in Application CRD
- Resource health changes frequently which causes very frequent Application patches

Solution:

- Disable storing health information in Application CRD
- Add `controller.resource.health.persist: “true”` to `argocd-cmd-params-cm` ConfigMap

Cross AZ Redis traffic

Problem

- Controller sends thousands of Redis SET requests causing terabytes of cross AZ traffic per day
- Might cost **hundreds of $ per day** assuming price 0.01$ per 1Gb

Reason: Applications with large number (dozens of hundreds) of K8S resources

- Controller stores application resources tree in Redis (for visualization in UI)
- Whole tree is being updated every time when any resource changes
- Amount of traffic grows exponentially: tree with higher resource number causes more and heavier requests

Solution:

- Be careful with orphaned resource monitoring feature
- Enable resource tree sharing
- Add ARGOCD_APPLICATION_TREE_SHARD_SIZE=50 env variable to controller
- Expected Redis traffic reduction is up to 10x times

Over internet controller traffic

Problem

- Controller receives gigabytes of data from managed K8S clusters
- Traffic is usually external and cost 10x more than cross AZ
- Might **cost thousands of $ per month**

Reason: Controller "watches" all resources in managed K8S clusters

- By default controller attempts to manage all resources in K8S cluster
- Uses discovery API to find all resources in a cluster and perform list and watch request on each resource

Solution:

- Exclude resources that you never intend to manage
- Use `resource.exclusions` in `argocd-cm` ConfigMap to specify exclusion list
- Examples: Endpoint, EndpointSlice

Over-provisioned Redis

Problem

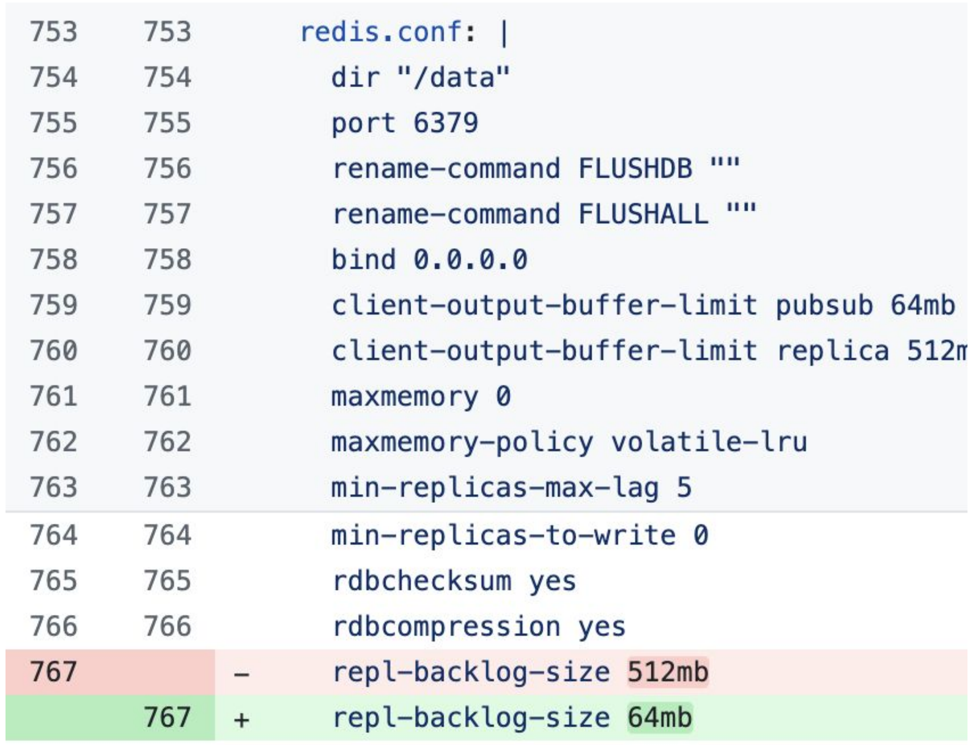
- HA Redis consists of 5 Pods
- Redis memory usage is growing with time and requires ~2 Gb of RAM

Reason: Unnecessary high replication cache

- Argo CD bundles Redis with unnecessary high replication cache (512Mb)
- Cache get’s feels with time causing high memory usage

Solution:

- Reduce cache size to 64Mb
- Reduce Redis container memory requests to 128Mb

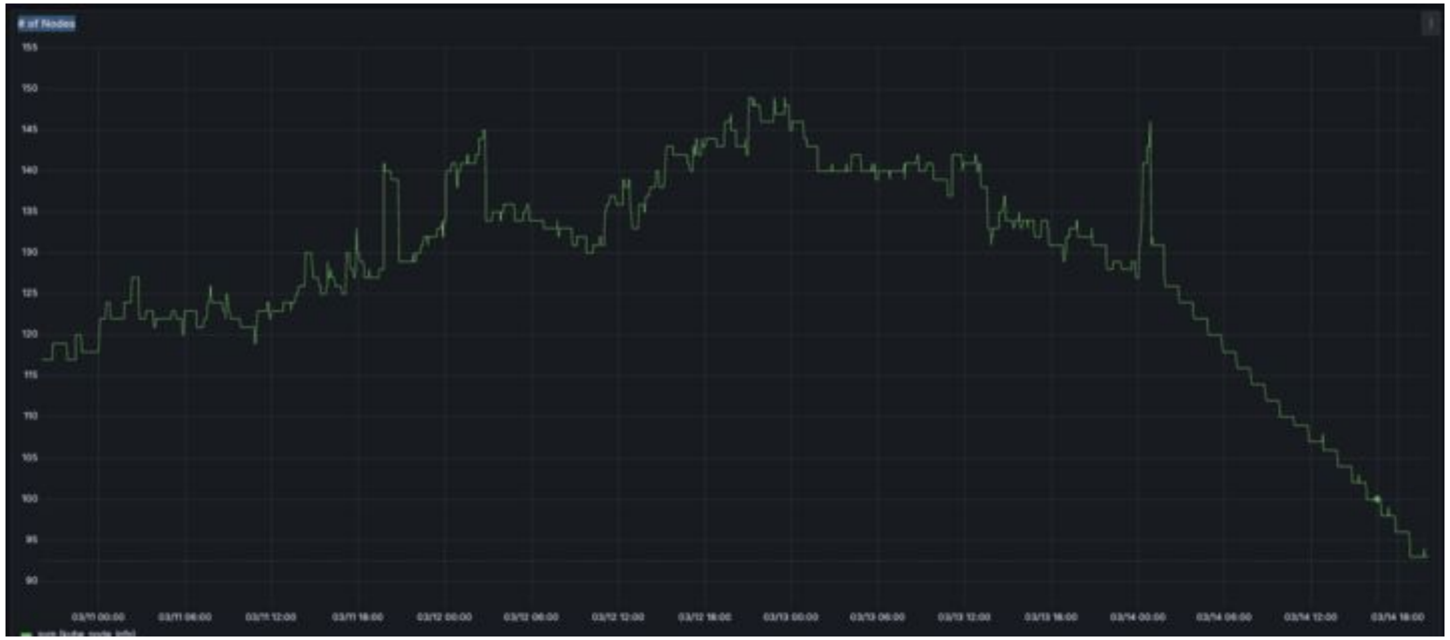


```
    redis.conf: |
      dir "/data"
      port 6379
      rename-command FLUSHDB ""
      rename-command FLUSHALL ""
      bind 0.0.0.0
      client-output-buffer-limit pubsub 64mb
      client-output-buffer-limit replica 512
      maxmemory 0
      maxmemory-policy volatile-lru
      min-replicas-max-lag 5
      min-replicas-to-write 0
      rdbchecksum yes
      rdbcompression yes
-     repl-backlog-size 512mb
+     repl-backlog-size 64mb
```

Over-provisioned Redis

Akuity Internal Results

- Saved 548 Gb in memory requests for 100+ Argo CD instances
- Number of nodes dropped from 148 to 80
- Saving us 200 ~ 300$ daily



Over-provisioned repo-server

Problem

- Repo Server generates manifests very slow
- The better performance requires running very large number of replicas

Reason: Sequential manifests generation for one Git repository

- Repo server by default process only one request at a time per Git repository
- Mono repos (repos with multiple Argo CD app manifests) are handled sequentially
- Single commit to mono repo causes huge spike of pending requests

Solution:

- Support concurrent manifest generation
  - Helm: enable manifest generation by adding `ARGOCD_HELM_ALLOW_CONCURRENCY=true` env variable
  - Kustomize: avoid overriding images using Argo CD Application `.spec.source.kustomization` field
- Fine tune resource utilization by setting `--parallelismlimit` flag.
  - The limit value = min(CPUs, memory GBs)

Dedicated K8S server

Problem

- Dedicated control plane K8S is a major overhead
- For HA requires running 3 master nodes costing hundreds of $ per month

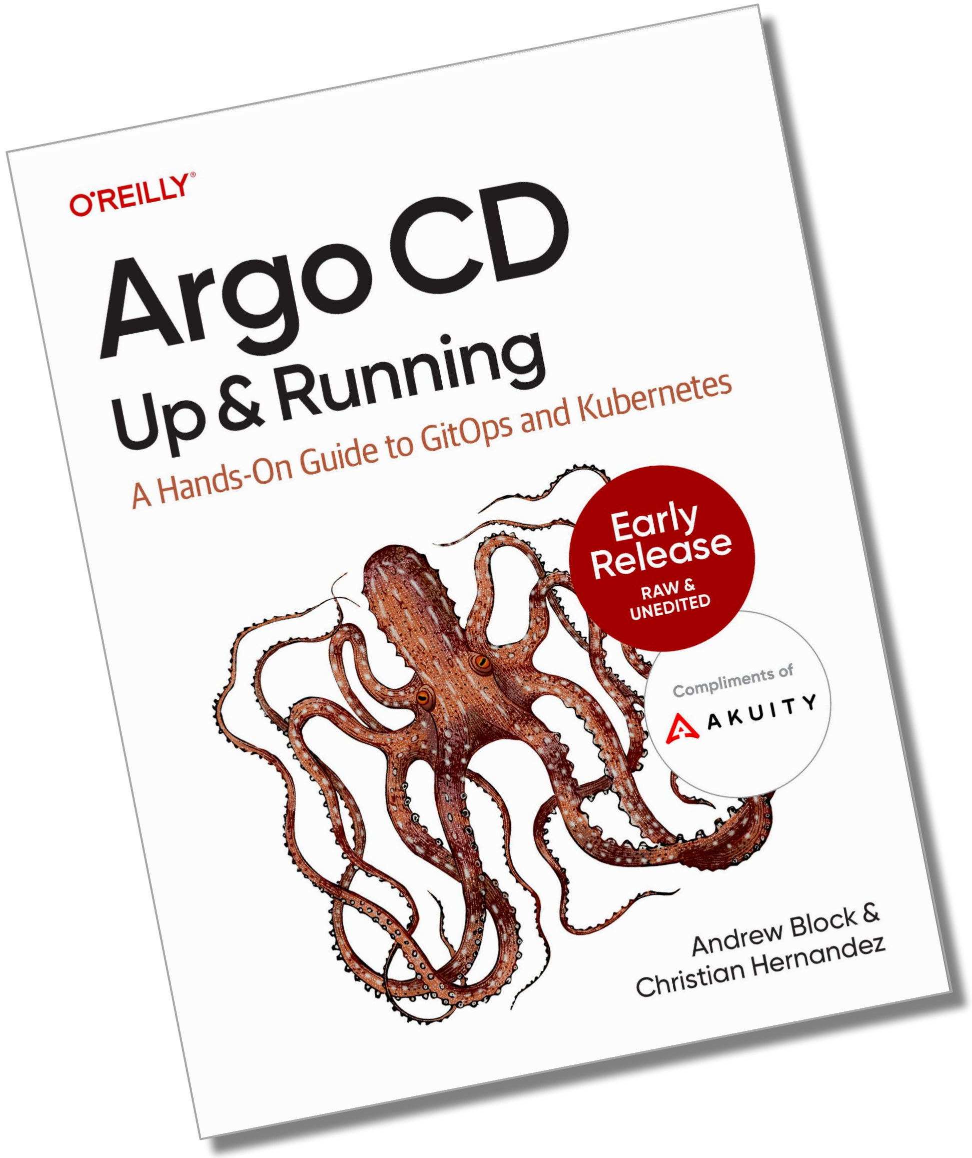
Reason: Argo CD itself requires K8S cluster

- Argo CD uses K8S API and cannot installed outside of K8S

Solution:

- Share control plane K8S cluster with other platform services
- Consider using K3S/vCluster to store Argo CD metadata (Applications)
  - Argo CD needs K8S only to manage collection of Argo CD Applications and access Secrets/ConfigMaps
  - Using K3S gives more flexibility (such as running 2 Argo CD version on same cluster) and improves availability

EARLY RELEASE



O'REILLY

Argo CD

Up & Running

A Hands-On Guide to GitOps and Kubernetes

Early Release
RAW & UNEDITED

Compliments of AKUITY

Andrew Block &
Christian Hernandez

Download Here!

KubeCon CloudNativeCon Europe 2025

THANK YOU!