

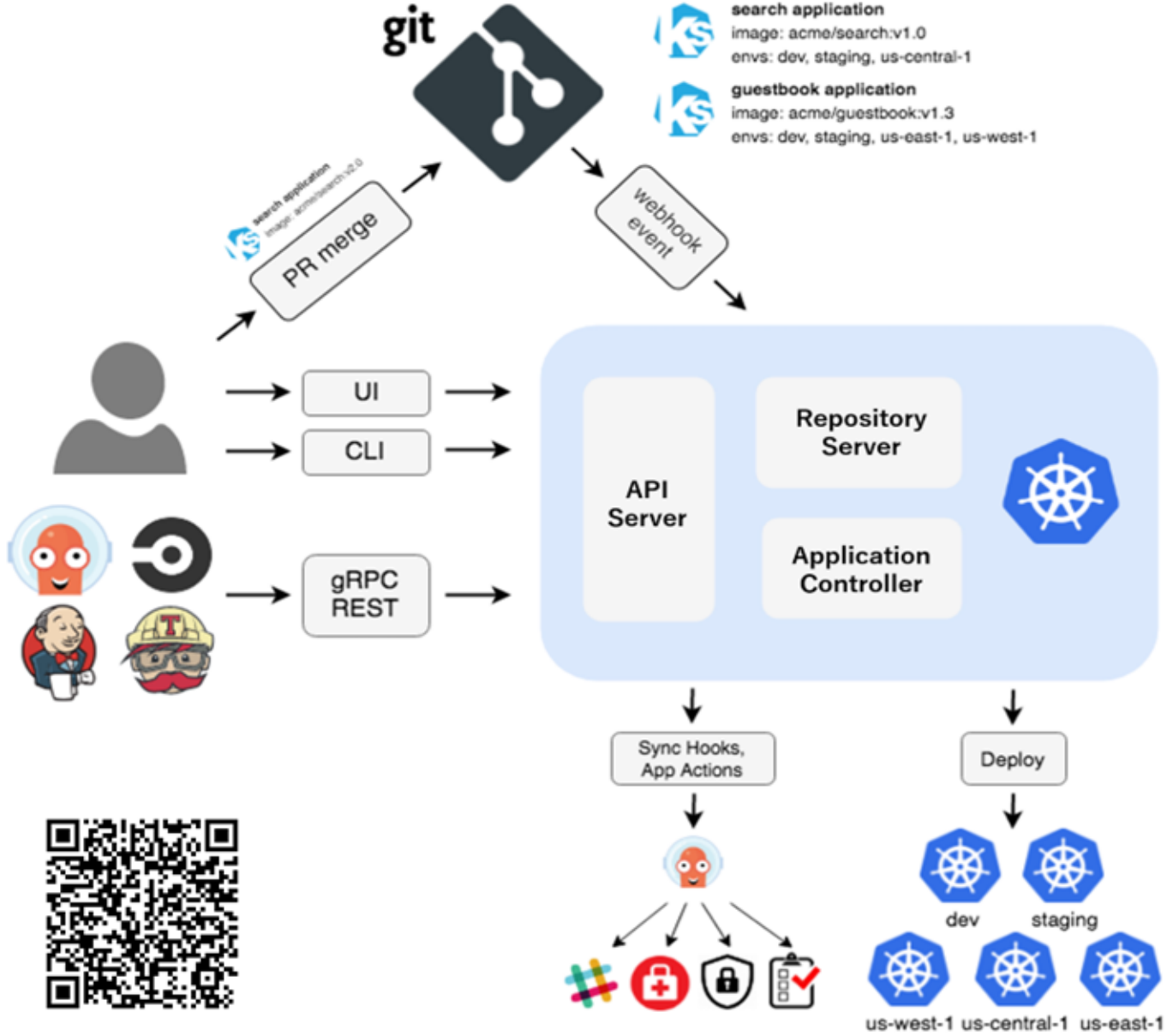
ArgoCon
NORTH AMERICA

Drop Your Memory

Do You Really Need Redis for Argo CD?

Netanel Kadosh, Kaltura
Adi Ziv, AWS

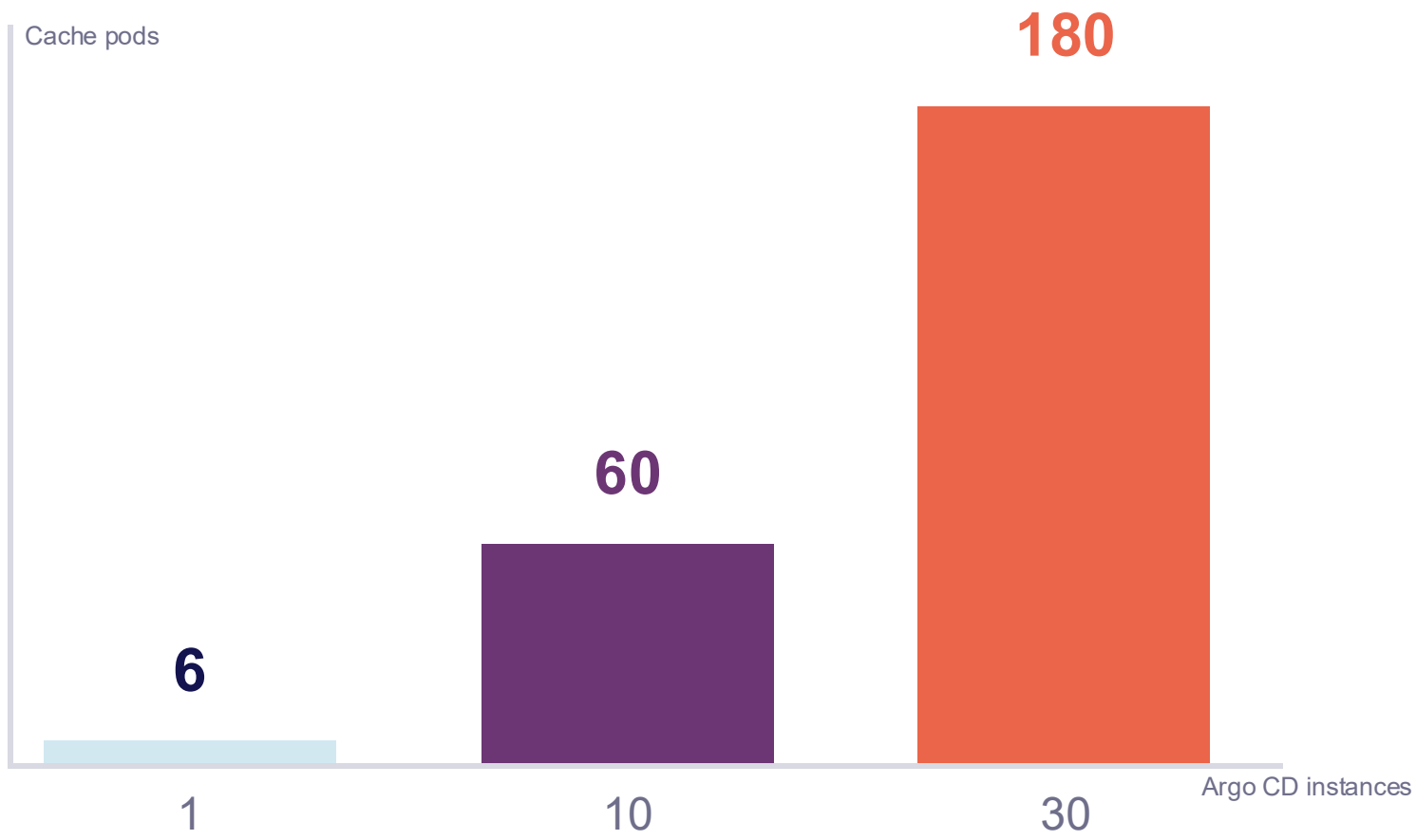
Argo CD Architecture



Where's Redis?



The Hitchhikers



180

cache pods

Kaltura's Argo CD Fleet



Dozens

Argo CD instances



Hundreds

Kubernetes clusters



Thousands

apps synced daily



ArgoCon
NORTH AMERICA

After years with the default setup we asked,

Do we really need Redis?

Drop Your Memory: Do You Really Need Redis for Argo CD?



ArgoCon
NORTH AMERICA

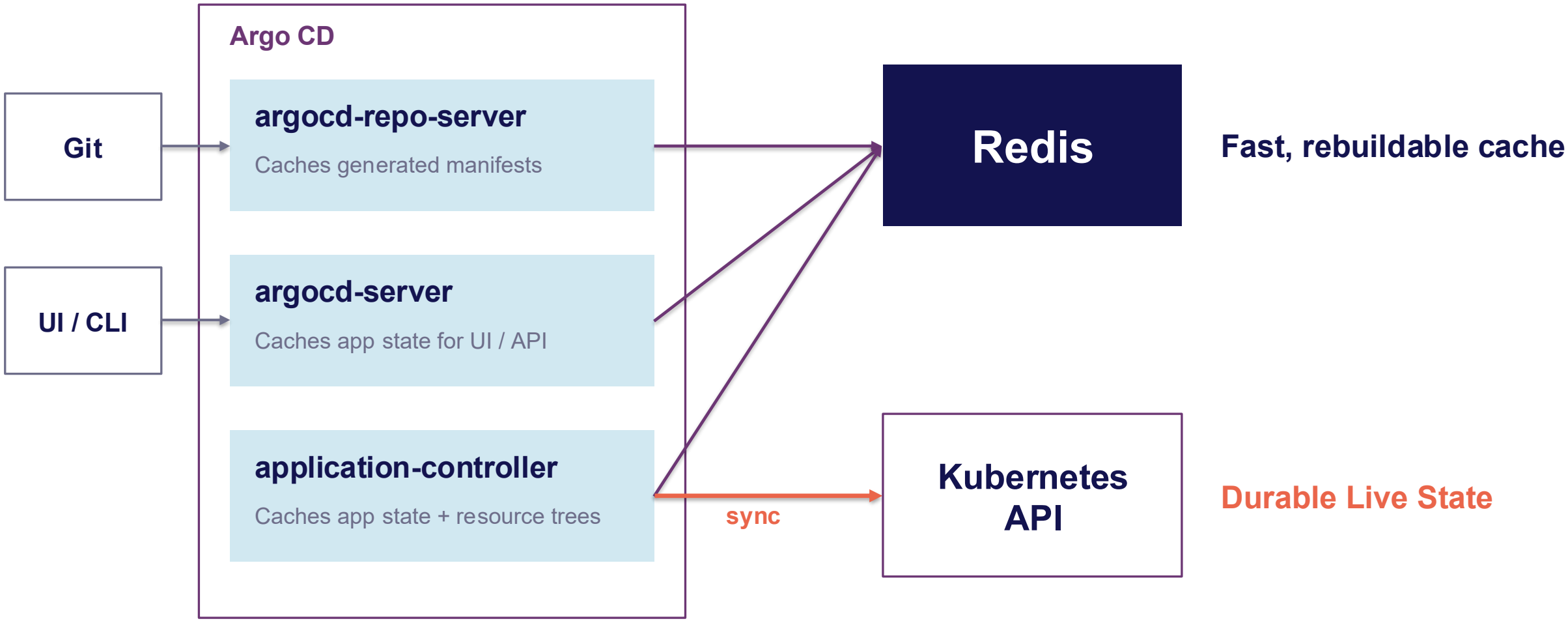


Adi Ziv
Solutions Architect
AWS

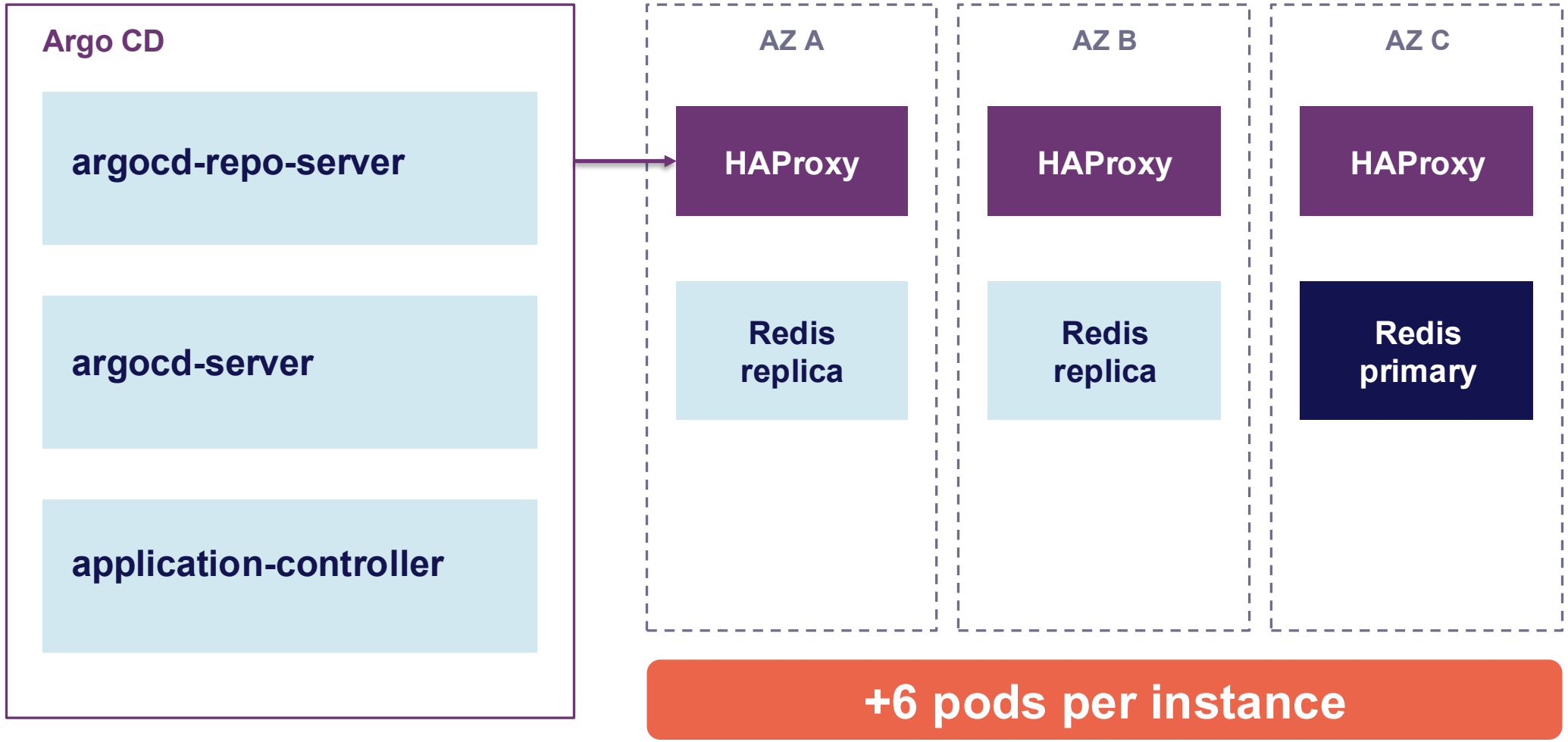


Netanel Kadosh
Senior DevOps Engineer
Kaltura

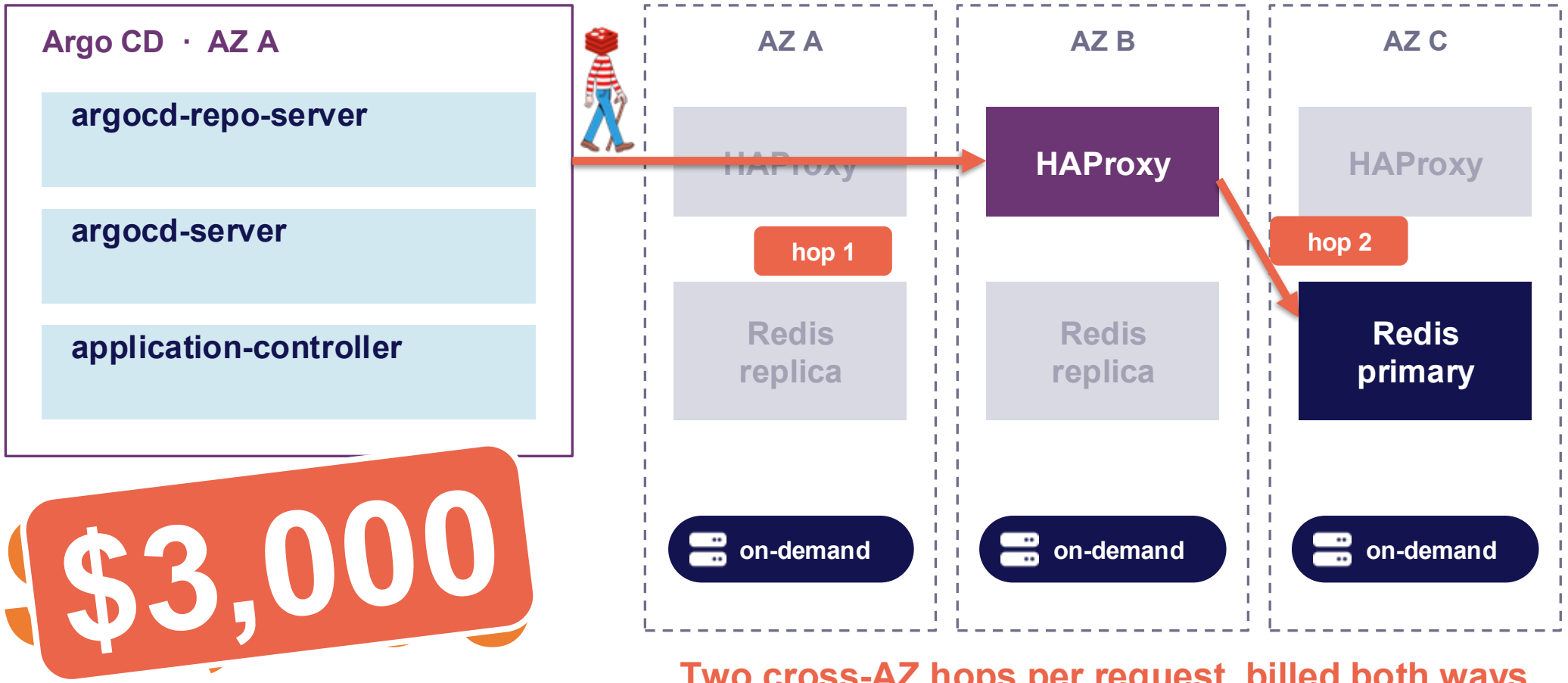
How Argo CD works with Redis



HA Under the Hood



Where the Money Went



$3,000

/month

$2,000 cross-AZ + **~$1,000 on-demand nodes**

Two cross-AZ hops per request, billed both ways

Beyond the Bill



Scale

Nodes sized for bulk syncs



Ops Overhead

Upgrades, failover, CVEs



Performance

Cache trouble slows the UI



ArgoCon
NORTH AMERICA

We never chose to run Redis.
We inherited it.

1 **State?** Disposable

2 **Business Value?** No

3 **Better Alternatives?** Yes

Offload it.

What Do We Really Need?

ArgoCon NORTH AMERICA



Redis compatible



Scalable



Cost effective



Fast

Let's go with a managed service

- Scaling
- Multi-AZ
- Failover
- Patching



Introducing Valkey





Open source

BSD license, Linux Foundation governance



Drop-in compatible

Forked from Redis 7.2; Argo CD's client just works



Cheaper on AWS

33% lower Serverless unit price than Redis OSS



Blazing fast

70% lower average latency

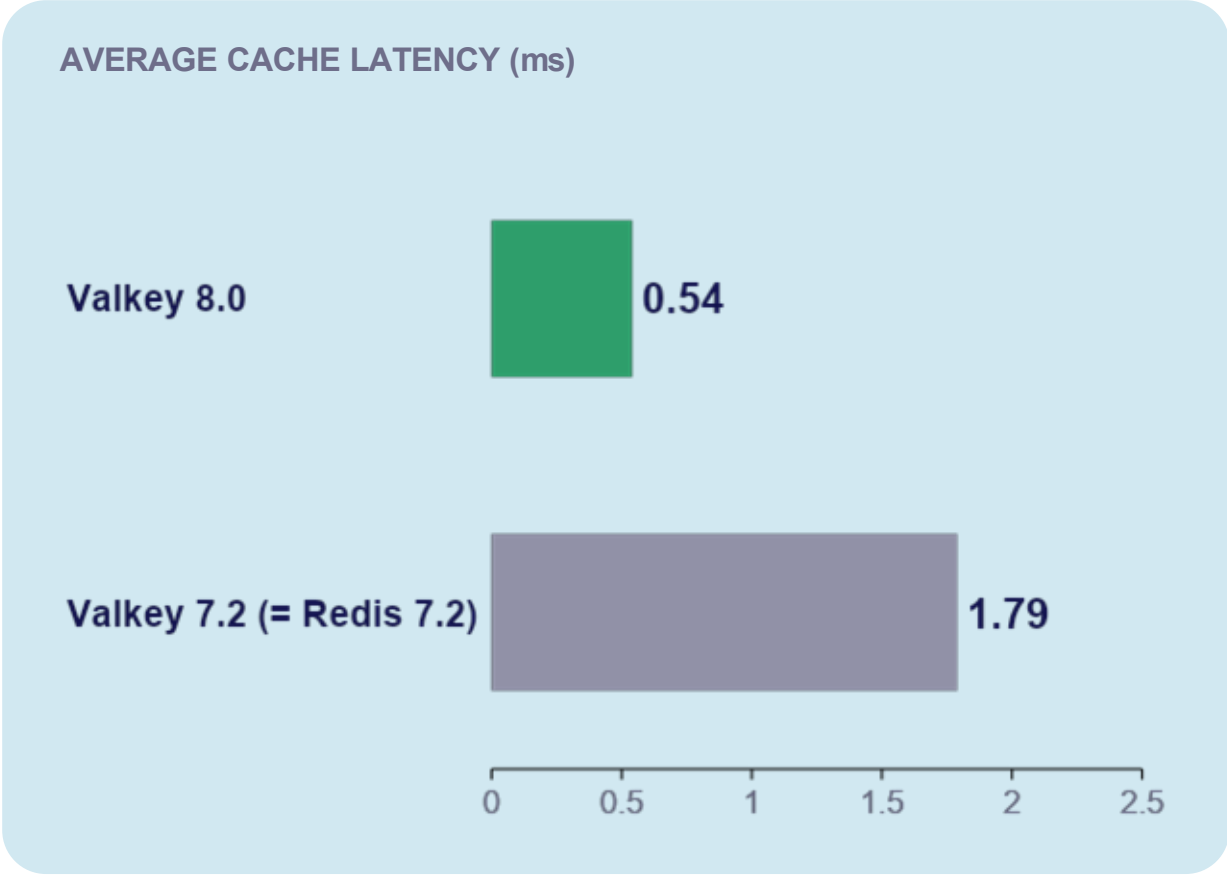
Lower Latency Under Load

ArgoCon NORTH AMERICA

−70%

lower average cache latency

~~1.79 ms~~ → **0.54 ms**



Source: valkey.io, 'Unlock 1 Million RPS', Aug 2024

Managed Valkey is Everywhere



AWS

Kaltura's env

ElastiCache

Google Cloud

Memorystore

DigitalOcean

Managed Caching

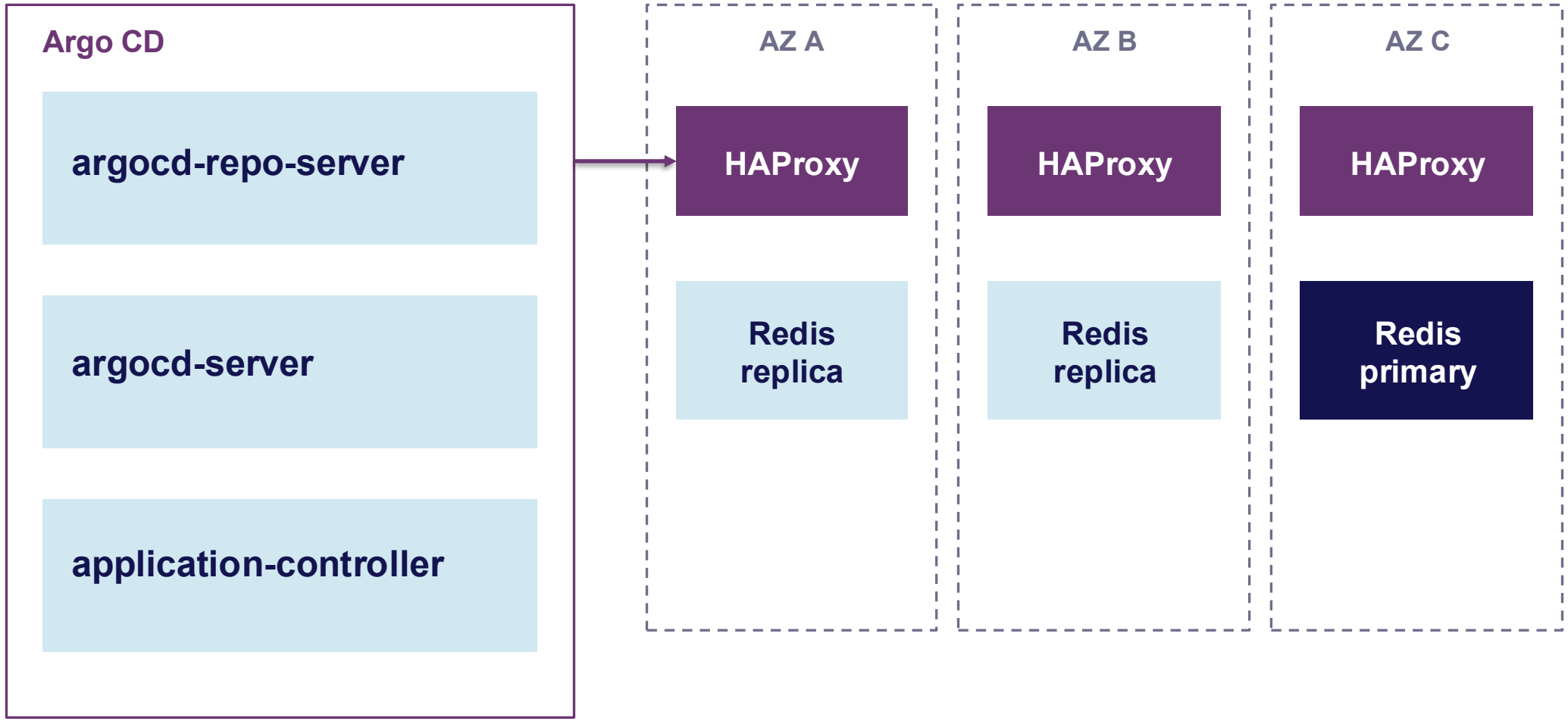
Aiven

Aiven for Valkey

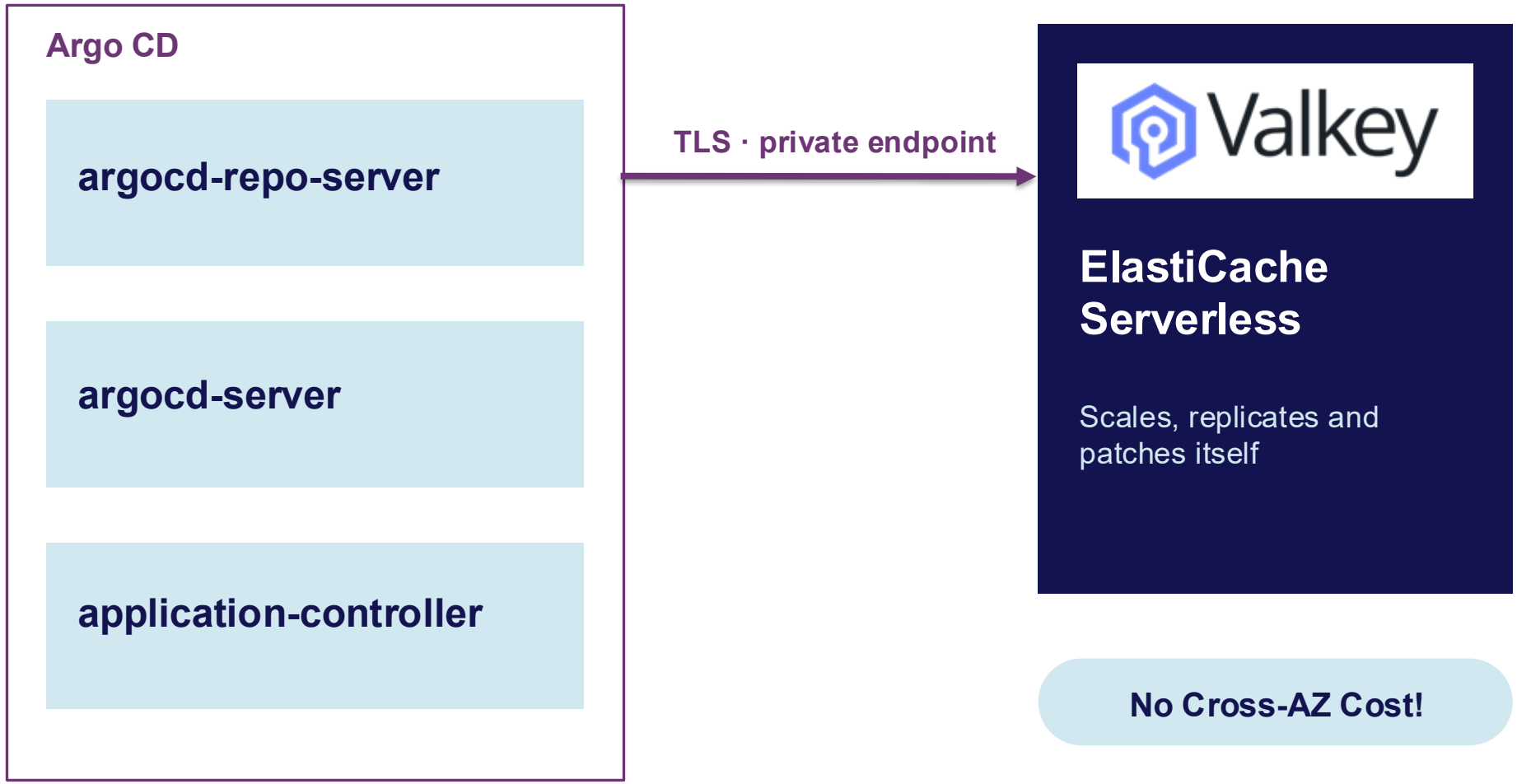
Oracle Cloud

OCI Cache

New Architecture (AWS example)



New Architecture (AWS example)



Migration: Get a Valkey Endpoint



Provision it, or use the one you already have

- Private endpoint
- TLS on
- Password in a K8s Secret

Migration: Point Argo CD to Valkey

ArgoCon NORTH AMERICA

```yaml
redis:
  enabled: false
redis-ha:
  enabled: false
```

```yaml
externalRedis:
  host: "<cache-endpoint>"
  port: 6379
  existingSecret: argocd-cache
```

```yaml
controller:
  extraArgs: ["--redis-use-tls"]
server:
  extraArgs: ["--redis-use-tls"]
repoServer:
  extraArgs: ["--redis-use-tls"]
```

1 Disable bundled Redis

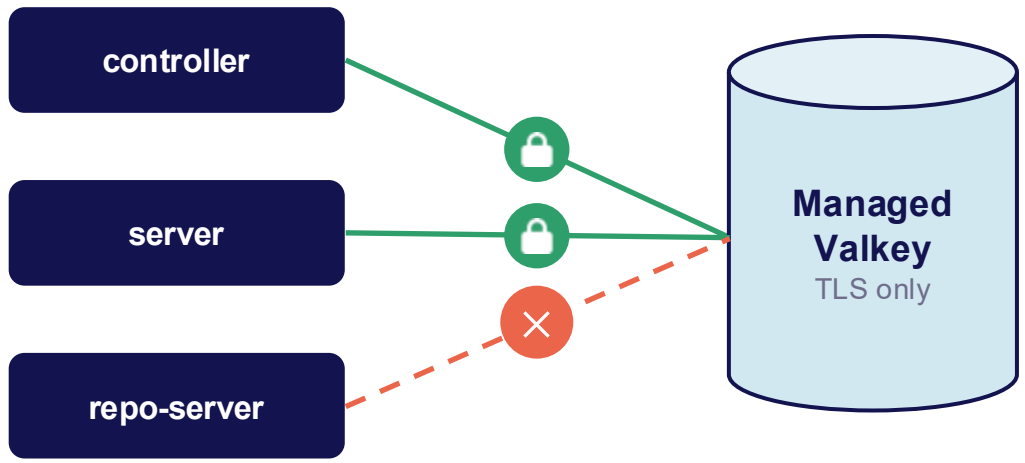
2 Endpoint + Secret

3 TLS on all three

What Tripped Us Up

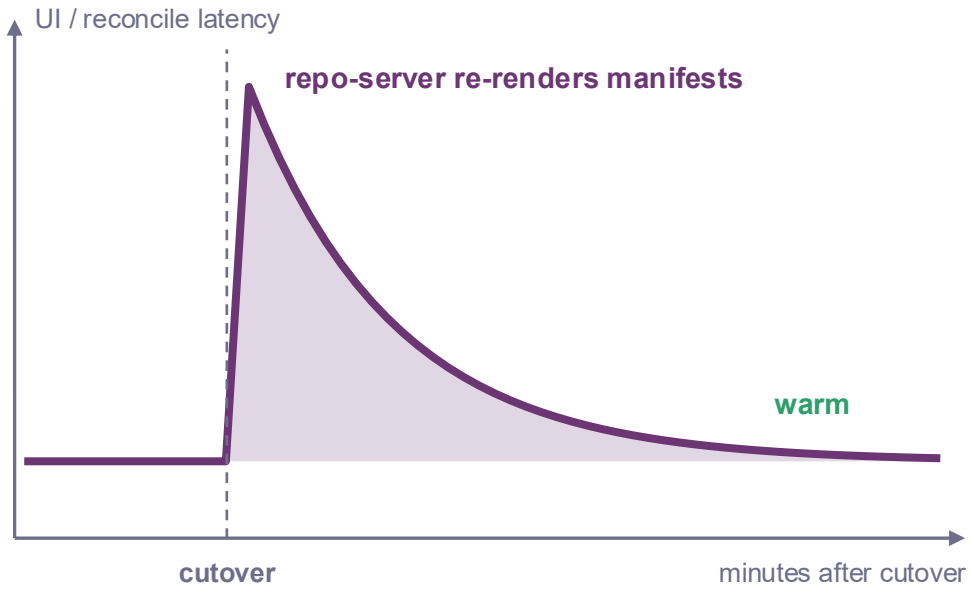
GOTCHA 01

TLS: all or nothing



GOTCHA 02

Cold cache

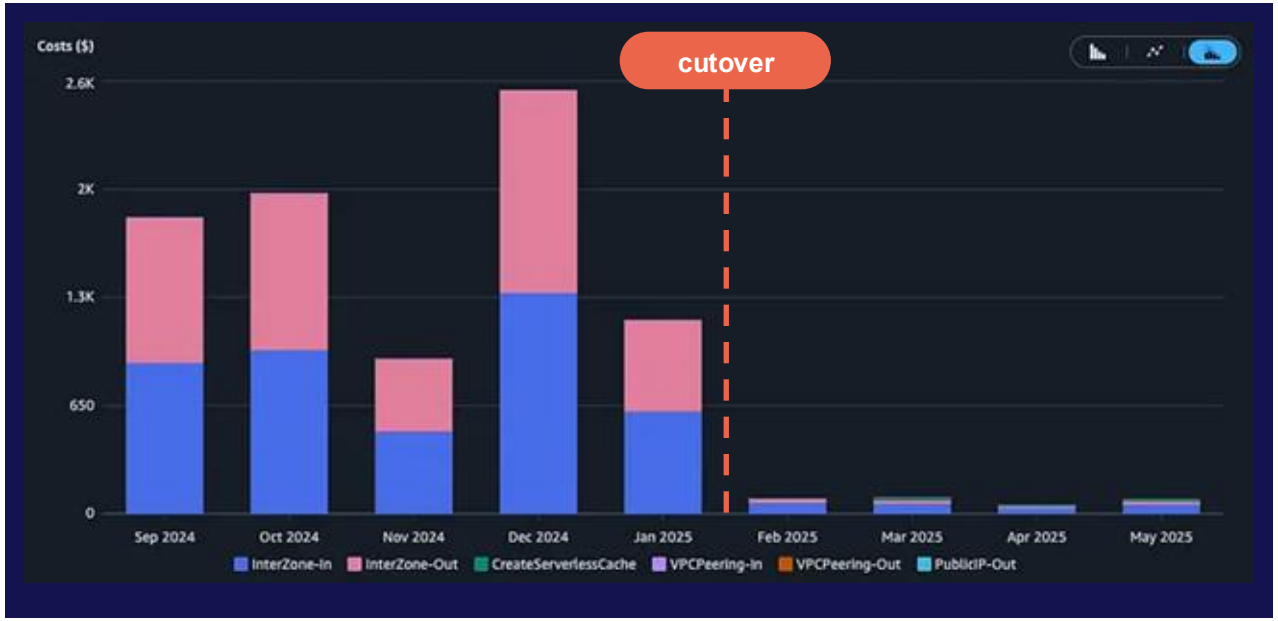


Results: The Bill

ArgoCon NORTH AMERICA

~95%

lower Redis-related costs



Monthly Redis-related costs, AWS Cost Explorer (Kaltura)

Beyond the Bill



Scale

Auto-scales with demand



Ops Overhead

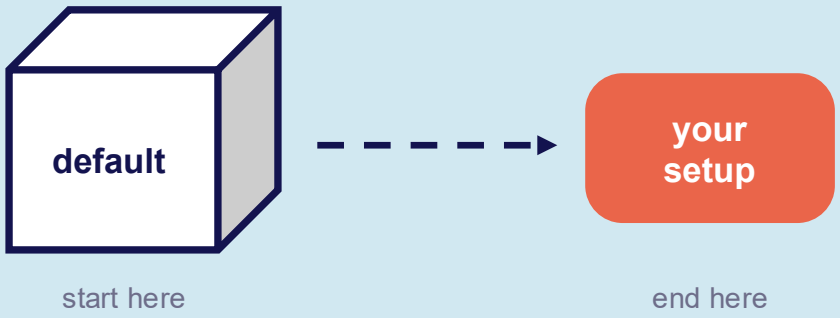
Provider owns patching



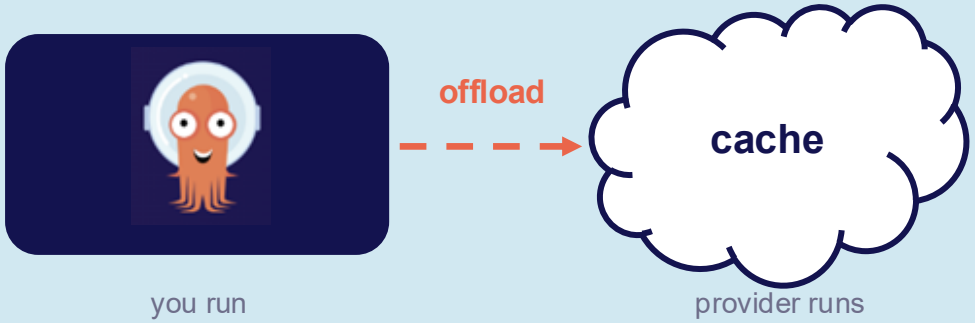
Performance

70% better latency

Takeaways



Defaults are a starting point, not a destination



Run what makes you better. Offload the rest.

Thank You!



Talk to us!

Netanel

Adi