Designing scalable Kubernetes clusters on AWS

NICK YOUNG | PRINCIPAL ENGINEER | @YOUNGNICK

Who am I?

2 yrs on Kubernetes team

4.5 years at Atlassian
20 years as a Sysadmin

Worked on OnDemand

Scheduling 150,000 customers worth of Jira, Confluence, and Bamboo JVMs is hard!

Kubernetes was exciting

Google’s way of scheduling workloads across clusters seemed like a good idea.

Who am I?

2 yrs on Kubernetes team

4.5 years at Atlassian
20 years as a Sysadmin

Worked on OnDemand

Scheduling 150,000 customers worth of Jira, Confluence, and Bamboo JVMs is hard!

Kubernetes was exciting

Google’s way of scheduling workloads across clusters seemed like a good idea.

Who am I?

2 yrs on Kubernetes team

4.5 years at Atlassian
20 years as a Sysadmin

Worked on OnDemand

Scheduling 150,000 customers worth of Jira, Confluence, and Bamboo JVMs is hard!

Kubernetes was exciting

Google's way of scheduling workloads across clusters seemed like a good idea.

WHAT WERE WE BUILDING?

A set of clusters that could run 95% or more of compute workloads in Atlassian

NICK'S RULE #0 OF DESIGNING SCALABLE STUFF

Design out the biggest problems you know about, so you can find new and interesting ones later.

The problems we decided to solve

Manage blast radius

We build a layer cake with strong isolation between layers, and clearly define what a cluster means to us.

Cattle, not pets

We embrace immutable infrastructure as much as possible.

Manage dependencies

Eventually, lots of things will be running on us - we can only depend on AWS things.

The problems we decided to solve

Manage blast radius

We build a layer cake with strong isolation between layers, and clearly define what a cluster means to us.

Cattle, not pets

We embrace immutable infrastructure as much as possible.

Manage dependencies

Eventually, lots of things will be running on us - we can only depend on AWS things.

The problems we decided to solve

Manage blast radius

We build a layer cake with strong isolation between layers, and clearly define what a cluster means to us.

Cattle, not pets

We embrace immutable infrastructure as much as possible.

Manage dependencies

Eventually, lots of things will be running on us - we can only depend on AWS things.

The layer cake

The layer cake

FLAG

Base AWS configuration, including VPCs, subnets, VGWs, security groups, and so on.

The layer cake

KARR

All the compute, control plane and etcd pieces. Stands up an apiserver endpoint, nothing else.

FLAG

Base AWS configuration, including VPCs, subnets, VGWs, security groups, and so on.

The layer cake

Goliath

All configuration that runs inside Kubernetes. Importantly, includes RBAC, PSPs, etc.

KARR

All the compute, control plane and etcd pieces. Stands up an apiserver endpoint, nothing else.

FLAG

Base AWS configuration, including VPCs, subnets, VGWs, security groups, and so on.

Cattle, not pets

Controllers and nodes

Created in ASGs, cycled automatically or scaled by autoscaler

etcd servers

Like milk cows you know the name of.

Rebuilding

We can burn a cluster down to the FLAG and rebuild in <30min.

Managing Dependencies

Secrets

Wherever possible, secrets are stored in private S3 buckets only accessible to the nodes.

Image storage - ECR

We can’t depend on any other container registry being up.

So how did we do?

Clusters scale pretty well

Biggest size so far is about 300 m4.10xlarge
That’s 12,000 vCPUs and 48TB of RAM.

Mainly batch (for now)

Batch workloads are the easiest to get working on Kubernetes. We currently run about 15k-20k builds per day.

Evaluating Service Meshes

Our service workloads are coming, we are looking at service meshes at the moment.

So how did we do?

Clusters scale pretty well

Biggest size so far is about 300 m4.10xlarge
That’s 12,000 vCPUs and 48TB of RAM.

Mainly batch (for now)

Batch workloads are the easiest to get working on Kubernetes. We currently run about 15k-20k builds per day.

Evaluating Service Meshes

Our service workloads are coming, we are looking at service meshes at the moment.

So how did we do?

Clusters scale pretty well

Biggest size so far is about 300 m4.10xlarge
That’s 12,000 vCPUs and 48TB of RAM.

Mainly batch (for now)

Batch workloads are the easiest to get working on Kubernetes. We currently run about 15k-20k builds per day.

Evaluating Service Meshes

Our service workloads are coming, we are looking at service meshes at the moment.

Thanks!

NICK YOUNG | PRINCIPAL ENGINEER | @YOUNGNICK