Containers

Packaging

Distribution

Immutable infrastructure
What is Kubernetes?

- Open source container management platform
- Helps you run containers at scale
- Gives you primitives for building modern applications
A single extensible API

SCALE

PERFORMANCE

BREADTH
Vibrant and growing community of users and contributors
Kubernetes can be run anywhere!
Cloud-native applications

MICROSERVICE TOOLING → NATIVE APPLICATIONS
“Run Kubernetes for me.”

“Native AWS integrations”

“An open source Kubernetes experience.”
ELASTIC CONTAINER SERVICE FOR KUBERNETES

GA yesterday 6/5!
Amazon Container Services

Amazon ECS  
Amazon EKS
AWS Fargate  
Amazon ECR
EKS is Kubernetes Certified
Open Source Kubernetes Community

Kubernetes
https://github.com/kubernetes/kubernetes

CNI plugin
https://github.com/aws/amazon-vpc-cni-k8s

Heptio AWS Authenticator
https://github.com/heptio/authenticator

Virtual Kubelet
https://github.com/virtual-kubelet/virtual-kubelet/

Cloud Provider Working Group
https://github.com/kubernetes/community/tree/master/wg-cloud-provider

External-DNS
https://github.com/kubernetes-incubator/external-dns

CoreOS ALB Ingress
https://github.com/coreos/alb-ingress-controller

SIG AWS
@CHRISTOPH_K@TIFFANYFAYJ
EKS - Customers

1. Create EKS cluster
2. Provision worker nodes
3. Launch add-ons
4. Launch workloads
EKS - Kubernetes Control Plane

Create cluster

Create HA Control Plane

IAM integration

Certificate Management

Setup LB
EKS Architecture
IAM Authentication
IAM Authentication + kubectl

1) Passes AWS Identity

2) Verifies AWS Identity

3) Authorizes AWS Identity with RBAC

4) K8s action allowed/denied

https://github.com/heptiolabs/kubernetes-aws-authenticator
EKS Worker Nodes
Worker provisioning

- kubectl
- config map
- Role
- Workers
- AWS Auth
- config map & RBAC
Metrics

Visualizer
Grafana, Kibana, Dashboard

Alerting
AlertManager, Kapacitor

Cluster-wide Aggregator
Prometheus, Heapster

Nodes
Node exporter

Pod/Container
Kube-state-metrics
cAdvisor

Application
/metrics
JMX

Data Model
InfluxDB, Graphite
Networking
Native VPC networking with CNI plugin

Pods have the same VPC address inside the pod as on the VPC

Simple, secure networking

Open source and on Github

https://github.com/aws/amazon-vpc-cni-k8s
How do I configure network security with EKS?
Kubernetes Network Policies enforce network security rules

Calico is the leading implementation of the network policy API

Open source, active development (>100 contributors)

Commercial support available from Tigera

https://www.projectcalico.org/
STAGE SEPARATION
Isolate dev, test, and prod

“TENANT” SEPARATION
Namespaces – without network policy, they are not network isolated

FINE-GRAINED FIREWALLS
Reduce attack surface within microservice-based applications

COMPLIANCE
E.g., PCI, HIPAA

Namespaces – without network policy, they are not network isolated

Reduce attack surface within microservice-based applications

E.g., PCI, HIPAA
What version of Kubernetes does EKS support?

1.10.3 currently
Kubernetes Autoscaling with Amazon EKS
Auto Scaling Cluster

Two options
- AWS AutoScaling
- k8s Cluster AutoScaler

Cluster Autoscaler
- Reactive
- Aware of Pod / Cluster state
- Utilizes AWS AutoScaling

AWS AutoScaling
- Scaling on CloudWatch Metrics

Pods

Horizontal Pod Autoscaler
- Scales pods in response to k8s generated metrics (CPU)
Package manager that allows you to bundle up deployment resources and publish them

- `helm search mysql`

<table>
<thead>
<tr>
<th>NAME</th>
<th>CHART VERSION</th>
<th>APP VERSION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>stable/mysql</td>
<td>0.6.0</td>
<td>5.7.14</td>
<td>Fast, reliable, sc</td>
</tr>
<tr>
<td>stable/prometheus-mysql-exporter</td>
<td>0.1.0</td>
<td>v0.10.0</td>
<td>A Helm chart for p</td>
</tr>
<tr>
<td>stable/percona</td>
<td>0.3.2</td>
<td>5.7.17</td>
<td>free, fully compat</td>
</tr>
</tbody>
</table>

- `helm install install stable/mysql`
  [displays README + information about deployment]

- `helm list`

<table>
<thead>
<tr>
<th>NAME</th>
<th>REVISION</th>
<th>UPDATED</th>
<th>STATUS</th>
<th>CHART</th>
<th>NAMESPACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>nobby-cow</td>
<td>1</td>
<td>Wed Jun 6 12:54:00 2018</td>
<td>DEPLOYED</td>
<td>mysql-0.6.0</td>
<td>default</td>
</tr>
</tbody>
</table>
Hosting Helm repositories

• Anywhere that serves HTTP can host a helm repo
• Host private Helm Repo with Chartmuseum
  https://github.com/kubernetes-helm/chartmuseum

• There’s also a handy plugin for S3!
• This means IAM Role = auth for your repo 😊

• https://github.com/hypnoglow/helm-s3
Deploying Helm on EKS

Helm 2.9+ works with EKS
RBAC permissions required

```bash
kubectl -n kube-system create serviceaccount tiller

kubectl create clusterrolebinding tiller --clusterrole cluster-admin --serviceaccount=kube-system:tiller

helm init --service-account tiller
```
Know-how & Tools
Load Balancing
Service Type - LoadBalancer (NLB)

Request to NGINX Pod {NLB}:443

Internet

NLB

NLB Forwards to the node {node:32001}

EC2 instances

Nginx Pods

k8s service ClusterIP receives request

nginx-service:32001

kube-proxy:32002

kube-proxy load balances to pods

10001:8080

10002:8080

10003:8080

10001:8080

10002:8080

10003:8080
Network Load Balancer

apiVersion: v1
kind: Service
metadata:
  name: nginx
  namespace: default
labels:
  app: nginx
annotations:
  service.beta.kubernetes.io/aws-load-balancer-type: "nlb"

spec:
  type: LoadBalancer
  externalTrafficPolicy: Local
  ports:
    - name: http
      port: 80
      protocol: TCP
      targetPort: 80
  selector:
    app: nginx

More options:
- Draining
- Logging
- SSL Certs
- Tagging
- Security groups
- Health checks

https://github.com/kubernetes/kubernetes/blob/master/pkg/cloudprovider/providers/aws/aws.go
Ingress Type - CoreOS ALB Ingress

Request to NGINX Pod (ALB):443

Internet

ALB

ALB Routes based on the path.

/api

/home

EC2 instances

Nginx Pods

10002:8080

kube-proxy:32001

nginx-service:32003

Webapp Pods

10002:8080

kube-proxy:32002

webapp-service:32004

Proxies request to the k8s service ClusterIP

Load Balances to pods

Installation: https://github.com/pahud/eks-alb-ingress

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
DNS
Automatic Route53 DNS creation for services

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
  annotations:
    # Uses https://github.com/kubernetes-incubator/external-dns
    external-dns.alpha.kubernetes.io/hostname: nginx.highlyavailable.systems.
spec:
  type: LoadBalancer
  ports:
    - port: 80
      name: http
      targetPort: 80
  selector:
    app: nginx
```
Automatic Route53 DNS creation for Ingress

```yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: nginx
  annotations:
    kubernetes.io/ingress.class: "nginx"
spec:
rules:
- host: nginx.highlyavailable.systems
  http:
    paths:
    - backend:
        serviceName: nginx
        servicePort: 80
```
Scheduling
Controlling scheduling
Resource requirements

Volume filters

Resource filters

Topology filters

Prioritization
Limit resource usage

Container A
- limit: 600m
- request: 600m

Container B
- limit: 800m
- request: 400m

∑ Pod CPU and memory resources
Resource Quotas

Applied per Namespace

```yaml
apiVersion: v1
class: ResourceQuota
metadata:
  name: production
spec:
  hard:
    requests.cpu: "1"
    requests.memory: 1Gi
    limits.cpu: "2"
    limits.memory: 2Gi
```

Pod Resource Request

```yaml
apiVersion: v1
class: Pod
metadata:
  name: production
spec:
  containers: 
    - name: nginx-pod
      image: nginx
      resources:
        limits:
          memory: "800Mi"
          cpu: "800m" # 0.8 vCPU
        requests:
          memory: "600Mi"
          cpu: "400m" # 0.4 vCPU
```

ResourceQuota defined both, so Pod must define both
Controlling scheduling

Resource requirements

Constraints
- Taints
- Tolerations

Node-level
Pod-level

Volume filters
Resource filters
Topology filters
Prioritization
Taints and Tolerations

# Taint node

$ kubectl taint nodes ip-10-0-32-12.us-west-2.compute.internal \ 
  skynet=false:NoSchedule

# Tolerations

custom_object

kind: Pod

spec:
  tolerations:
    - key: skynet
      operator: Equal
      value: "false"
      effect: NoSchedule

[...]

Match taint to schedule onto tainted node
Controlling scheduling

Resource requirements

Constraints
- Taints  Node-level
- Tolerations  Pod-level

Affinity/Anti-Affinity
Affinity / Anti-Affinity

- Control scheduling onto nodes
  - Combine with Taints & Tolerations
- Distribute Pods across cluster

```yaml
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
        - matchExpressions:
            - key: "beta.kubernetes.io/instance-type"
              operator: In
              values: ["r4.large","r4.xlarge"]
```
Deployment Strategies
Rolling Update

apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: my-app
labels:
  app: my-app
spec:
  replicas: 10
strategy:
  type: RollingUpdate
  rollingUpdate:
    maxSurge: 1  # Numeric or percentage based value
    maxUnavailable: 0

[...]

@CHRISTOPH_K@TIFFANYFAYJ © 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Blue / Green Deployment

Blue

```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: my-app-blue
labels:
  app: my-app
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: my-app
        version: blue
[...]
```

Green

```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: my-app-green
labels:
  app: my-app
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: my-app
        version: green
[...]
```
Blue / Green Deployment

```bash
kubectl patch service my-app -p '{"spec":{"selector":{"version":"green"}}}'
```

<table>
<thead>
<tr>
<th>Blue</th>
<th>Green</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>kind</strong>: Service</td>
<td><strong>kind</strong>: Service</td>
</tr>
<tr>
<td><strong>metadata</strong>:</td>
<td><strong>metadata</strong>:</td>
</tr>
<tr>
<td>name: my-app</td>
<td>name: my-app</td>
</tr>
<tr>
<td>labels:</td>
<td>labels:</td>
</tr>
<tr>
<td>app: my-app</td>
<td>app: my-app</td>
</tr>
<tr>
<td><strong>spec</strong>:</td>
<td><strong>spec</strong>:</td>
</tr>
<tr>
<td>type: LoadBalancer</td>
<td>type: NodePort</td>
</tr>
<tr>
<td>ports:</td>
<td>ports:</td>
</tr>
<tr>
<td>name: http</td>
<td>name: http</td>
</tr>
<tr>
<td>port: 80</td>
<td>port: 80</td>
</tr>
<tr>
<td>targetPort: http</td>
<td>targetPort: http</td>
</tr>
<tr>
<td><strong>selector</strong>:</td>
<td><strong>selector</strong>:</td>
</tr>
<tr>
<td>app: my-app</td>
<td>app: my-app</td>
</tr>
<tr>
<td><strong>version</strong>: blue</td>
<td><strong>version</strong>: green</td>
</tr>
</tbody>
</table>
Canary Deployment

Production

```
apiVersion: extensions/v1beta1
type: Deployment
metadata:
  name: my-app-prod
  labels:
    app: my-app
spec:
  replicas: 9
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: images/container:v1
[...]
```

Canary

```
apiVersion: extensions/v1beta1
type: Deployment
metadata:
  name: my-app-canary
  labels:
    app: my-app
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: images/container:v2
[...]
```

More examples at https://container-solutions.com/kubernetes-deployment-strategies/
Network Policies
Network Policy

kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: web-allow-prod
spec:
  podSelector:
    matchLabels:
      app: web
  ingress:
    - from:
      namespaceSelector:
        matchLabels:
          purpose: production
Want to learn more?
Tooling and Ecosystem

https://github.com/ramitsurana/awesome-kubernetes
https://discuss.kubernetes.io/
http://slack.k8s.io/

TGIK Playlist:
https://www.youtube.com/playlist?list=PLvmPtYZtoXOENHJiAQc6HmV2jmuexKfrJ
EKS - Getting started
https://aws.amazon.com/eks


https://aws.amazon.com/blogs/compute/

https://medium.com/containers-on-aws
Questions?
Please complete the session survey in the summit mobile app.
Thank You

@christoph_k
@tiffanyfayj

Special thanks to:
Paul Maddox, Abby Fuller, Nishi Davidson, Brandon Chavis, Arun Gupta, Chris Hein, Omar Lari, and many more...

https://aws.amazon.com/containers