Container-based Architectures on AWS

Sascha Möllering
Solutions Architect, @sascha242, Amazon Web Services Germany GmbH

Steffen Grunwald
Solutions Architect, @steffeng, Amazon Web Services Germany GmbH
Amazon EKS – Now Generally Available

by Jeff Barr | on 05 JUN 2018 | in Amazon Elastic Container Service for Kubernetes, Launch, News | Permalink | Comments | Share

We announced Amazon Elastic Container Service for Kubernetes and invited customers to take a look at a preview during re:Invent 2017. Today I am pleased to be able to let you know that Amazon EKS is available for use in production form. It has been certified as Kubernetes conformant, and is ready to run your existing Kubernetes workloads.

Based on the most recent data from the Cloud Native Computing Foundation, we know that AWS is the leading environment for Kubernetes, with 57% of all companies who run Kubernetes choosing to do so on AWS. Customers tell us that Kubernetes is core to their IT strategy, and are already running hundreds of millions of containers on AWS every week. Amazon EKS simplifies the process of building, securing, operating, and maintaining Kubernetes clusters, and brings the benefits of container-based computing to organizations that want to focus on building applications instead of setting up a Kubernetes cluster from scratch.
We started in 2014
Customers running Docker on EC2 from the very beginning...
But there were pain points.

Things like scheduling, placing, managing and deploying containers were difficult.

They wanted something to make those pain points better.
So we built ECS

Amazon Elastic Container Service

Highly scalable, high performance container management system

A managed platform

Cluster management

Container orchestration

Deep AWS integration
So we built ECS

- Highly scalable, high performance container management system
- AWS VPC networking mode
- Advanced task placement
- Deep integration with AWS platform
- Global footprint
- Powerful scheduling engines
- Auto scaling
- CloudWatch metrics
- Load balancers

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Amazon ECS

Internet

EC2 INSTANCES

TASK
Containers

TAS
Containers

ECS AGENT

AGENT COMMUNICATION SERVICE

API

CLUSTER MANAGEMENT ENGINE

KEY/VALUE STORE
Lightweight agent on each host
API for launching containers on the cluster
Container task is placed on a host

Internet

EC2 INSTANCES

TAS

Containers

EC2 INSTANCES

TAS

Containers

EC2 INSTANCES

TAS

Containers

Amazon ECS

AGENT COMMUNICATION SERVICE

API

CLUSTER MANAGEMENT ENGINE

KEY/VALUE STORE

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Traffic is sent to your host
ECS Optimized Amazon Machine Images (AMIs)

- Optimized AMIs available for Linux & Windows
- Bring your own images based on it
- Expects ECS cluster name in user-data
- Update images on SNS update notifications

Amazon ECS Support for Windows Server Containers is Generally Available

Posted On: Dec 5, 2017

Amazon Elastic Container Service (Amazon ECS) now supports running Windows Server containers for production workloads.

Previously, Windows containers were supported by Amazon ECS as a public beta.

Now, Amazon ECS provides an ECS-Optimized Windows Server Amazon Machine Image (AMI), which is based on the EC2 Windows Server 2016 AMI and includes Docker 17.06 Enterprise Edition and the ECS Agent 1.16. This AMI provides improved instance and container launch time performance. The ECS Agent runs as a Windows Service, which provides an improved ECS Agent lifecycle management experience for elastic load balancing.
Automatic Service Scaling

Add/Remove ECS tasks

Scaling Policies

Amazon ECS

Publish metrics

Amazon CloudWatch

Auto Scaling ECS service

Availability Zone A

TASK A

Availability Zone B

TASK C

TASK B

Application Load Balancer
IAM Roles For Tasks

ECS Cluster

EC2 Instance

TASK A

TASK B

EC2 Instance

TASK B

Amazon DynamoDB

Amazon S3

Amazon ECS
Secrets Management

EC2 Instance

TASK A

EC2 Instance

TASK B

EC2 Instance

TASK B

ECS Cluster

Amazon ECS

System Manager – Parameter Store

• prod.app1.db-pass
• general.license-code
• prod.app2.user-name
Amazon Elastic Container Registry (Amazon ECR)

- Cloud-based Docker image registry
- Fully managed
- Secure – images encrypted at rest, integrated with IAM
- Scalable and Highly Available
- Integrated with Amazon ECS and the Docker CLI
AWS Fargate

Underlying technology for container management

No cluster or infrastructure to manage or scale

Everything is handled at the container level

Scale seamlessly on demand

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
What does Fargate mean?

Not worrying about scaling, underlying infrastructure, cluster resources, capacity, setup.

Just give it a task definition or pod (in 2018), set some resource limits, and away you go.
Task Definitions Repository on GitHub

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Continuous Deployment in Amazon ECS

1. Developers
2. AWS CodePipeline
3. AWS CodeBuild
4. Amazon ECR
5. Amazon ECS
6. AWS Fargate
7. Amazon EC2

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Europe's leading app for ordering taxis
Majority of services on ECS
10+ million users with 45,000+ taxis across 40+ cities.

With the microservice architecture (140+ services) built on AWS, mytaxi can provide new features to users faster than ever before.

Running entirely on Spot.
„In November 2015 we moved our Docker container architecture to Amazon ECS, and for the first time ever in December we were able to celebrate a new year in which our system could handle the huge number of requests without any crashes or interruptions.“

—Sebastian Herzberg, System Engineer
“Run Kubernetes for me.”
Amazon Elastic Container Service for Kubernetes: EKS

Managed Kubernetes on AWS

Managed Kubernetes control plane

Highly available

Automated version upgrades

Integration with other AWS services

CloudTrail, CloudWatch, ELB, IAM, VPC, PrivateLink
Elastic Container Service for Kubernetes

- Platform for enterprises to run production grade workloads
- Provides a native and upstream Kubernetes experience
- Not forced to use additional AWS services, but offer seamless integration
- EKS team actively contributes to the Kubernetes project
Elastic Container Service for Kubernetes
EKS Architecture

- Kubectl
- mycluster.eks.amazonaws.com
- Availability Zone 1
- Availability Zone 2
- Availability Zone 3
```yaml
node {
  stage 'Checkout'
  git 'https://github.com/omarlari/aws-container-sample-app.git'

  stage 'Build Dockerfile'
  docker.build('hello')

  stage 'Push to ECR'
  sh "\$(docker run awscli aws ecr get-login --region \${REGION} --no-include-email) sed 's|\$\{ECR_REPO\}\]|\$\{ECR_REPO\}\}|' docker.withRegistry('https://\$\{ECR_REPO\}') {
    docker.image('hello').push('\$\{BUILD_NUMBER\}')
  }

  stage 'update application'
  kubernetes: {
    node {
      docker.image('kubectl').inside("--volume=/home/ec2-user/.kube:/config/.kube"){
        sh 'kubectl describe deployment \$\{APP\}
        sh 'kubectl set image deployment/\$\{APP\} hello=\$\{ECR_REPO\}/hello:\$\{BUILD_NUMBER\}
        sh 'kubectl describe deployment \$\{APP\}
      }
    }
  }
}
```
AWS CodePipeline – CI/CD with Kubernetes

1. Developer
2. AWS CodeCommit
3. AWS CodeBuild
4. AWS Lambda
5. Amazon ECR
6. Kubernetes

AWS CodePipeline
Batch Processing with Containers
What is batch computing?

Run jobs asynchronously and automatically across one or more computers.

Jobs may have dependencies, making the sequencing and scheduling of multiple jobs complex and challenging.
Cloud makes Sense for Batch

• Scalable
• Reliable
• Choice:
  • Compute resources (GPUs, RAM- or CPU-bound)
  • Storage resources (fileshares, performance characteristics)
  • Downstream services (e.g. databases, streaming services)
  • Pricing models
• Pay as you go (per second)
Anatomy of a Batch

Event

- Changes in data state
- Scheduled triggers
- Requests to endpoints

Job Queue

Compute

Your Code

Services (anything)

Execution

Auto Scaling

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Options for Batch Workloads on AWS

Amazon ECS

AWS Batch

AWS Lambda
(Amazon SQS as built-in event source coming soon)
Containers make Sense for Batch

- Benefits from container development model
- Polyglot
- Do one thing well
- Black Box – and easy to model
Basic Batch Workflow with ECS

Input is put to S3 Bucket

Event is put to Amazon SQS

Queue depth is used for scaling

ECS provisions clusters and schedules tasks

Workers poll queue

Containerized batch workers process files

Output is put to S3 Bucket
Introducing AWS Batch

• Fully managed batch primitives

• Focus on your applications (shell scripts, Linux executables, Docker images) and their resource requirements

• We take care of the rest!
What you need to take care of ...

Event
- Changes in data state
- Scheduled triggers
- Requests to endpoints

Job Queue
- Your Code

Compute
- Execution
- Auto Scaling

Services (anything)
Typical AWS Batch Job Architecture

- Input is put to S3 Bucket
- Job Queue with runnable jobs
- Scheduler
- AWS Batch Compute Environment
- Output is put to S3 Bucket

Job definition:
- Application Image
  - + config
- IAM role
FROM amazonlinux:latest

RUN yum update -y
RUN yum install ImageMagick aws-cli -y

ADD flip.sh /usr/local/bin/flip.sh

WORKDIR /tmp
USER nobody

ENTRYPOINT ["/usr/local/bin/flip.sh"]
flip.sh

#!/bin/bash
SRCBKT=$1 # SOURCE BUCKET
OBJ=$2 # OBJECT KEY
TRGBKT=$3 # TARGET BUCKET

error_exit() { echo "${1}" >&2; exit 1; }

tmpfile=$(mktemp /tmp/image.XXXXXX)
aws s3 cp "s3://$SRCBKT/$OBJ" "$tmpfile" \
  || error_exit "Download failed $SRCBKT/$OBJ"
convert -flip "$tmpfile" "$tmpfile" \
  || error_exit "Failed to flip file"
aws s3 cp "$tmpfile" "s3://$TRGBKT/$OBJ" \
  || error_exit "Upload failed $TRGBKT/$OBJ"
AWS Batch Concepts

- Job definitions
- Jobs
- Job queue
- Compute environments
Job Definitions

AWS Batch job definitions specify how jobs are to be run.

Some attributes in a job definition:

- Container Image
- IAM role associated with the job
- vCPU and memory requirements
- Mount points
- Environment variables
- Retry strategy
Jobs

**Jobs** are the unit of work executed by AWS Batch. Set/overwrite **Job Definition** attributes, e.g.:

- Command
- Parameters
- Dependencies

**E.g. job’s Command:**

```
["batch-demo-input","steffeng.jpg","batch-demo-output"]
```

This JSON array is passed to the job as your command to execute.
Easily run massively parallel jobs

Start up to 10,000 copies of an application with a single call using **Array Jobs**.

Efficient way to run:

- Parametric sweeps
- Monte Carlo simulations
- Processing a large collection of objects
Job Retries

AWS Batch supports **up to 10 attempts** per job:

- errors in the AWS Batch job
- termination of the Spot Instance

The `AWS_BATCH_JOB_ATTEMPT` environment variable is set to the container's corresponding job attempt number.
Compute Environments

**Managed**
AWS scales and configures your instances for you.

Optional choice:
- On demand/ Spot
- Instance Types/ Mix
- Amazon Machine Image

**Unmanaged**
You control and manage the instance configuration, provisioning, and scaling.

Full control over scaling and instance provisioning for the ECS cluster used by AWS Batch.
Bring your own AMIs

Customer Provided AMIs let you set the AMI that is launched as part of a managed compute environment.

Makes it possible to configure Docker settings, mount EBS/ EFS volumes, and configure drivers for GPU jobs.

AMIs must be Linux-based, HVM and have a working ECS agent installation.
Job Queues

Jobs are submitted to **Job Queues**.

Job queues are assigned to one or more compute environments.

Each job queue has a priority assigned. Jobs in queues with higher priority take precedence.
Use Spot Instances

Get your jobs done faster or cheaper.

AWS Batch retries jobs on instance terminations and selects from multiple instance types.
Pricing

No additional charge for AWS Batch or Amazon ECS.
You only pay for the underlying resources that you consume!
Let’s build!
Questions? Ask these guys at the Ask an Architect booth:
Please **complete the session survey** in the summit mobile app.

Next session in this room:

14:00 - Kubernetes Running on AWS
Thank you!