AWS re:Invent

NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

CMP305-R

Data analysis with Amazon EKS and AWS Batch

Angel Pizarro

Principal Developer Advocate
AWS

Maxime Hugues

Principal HPC Specialist Solutions Architect
AWS

Agenda

Presentation: High-level overview of the workshop and how to participate in the workshop

Hands on: Start on the initial environment bootstrap steps

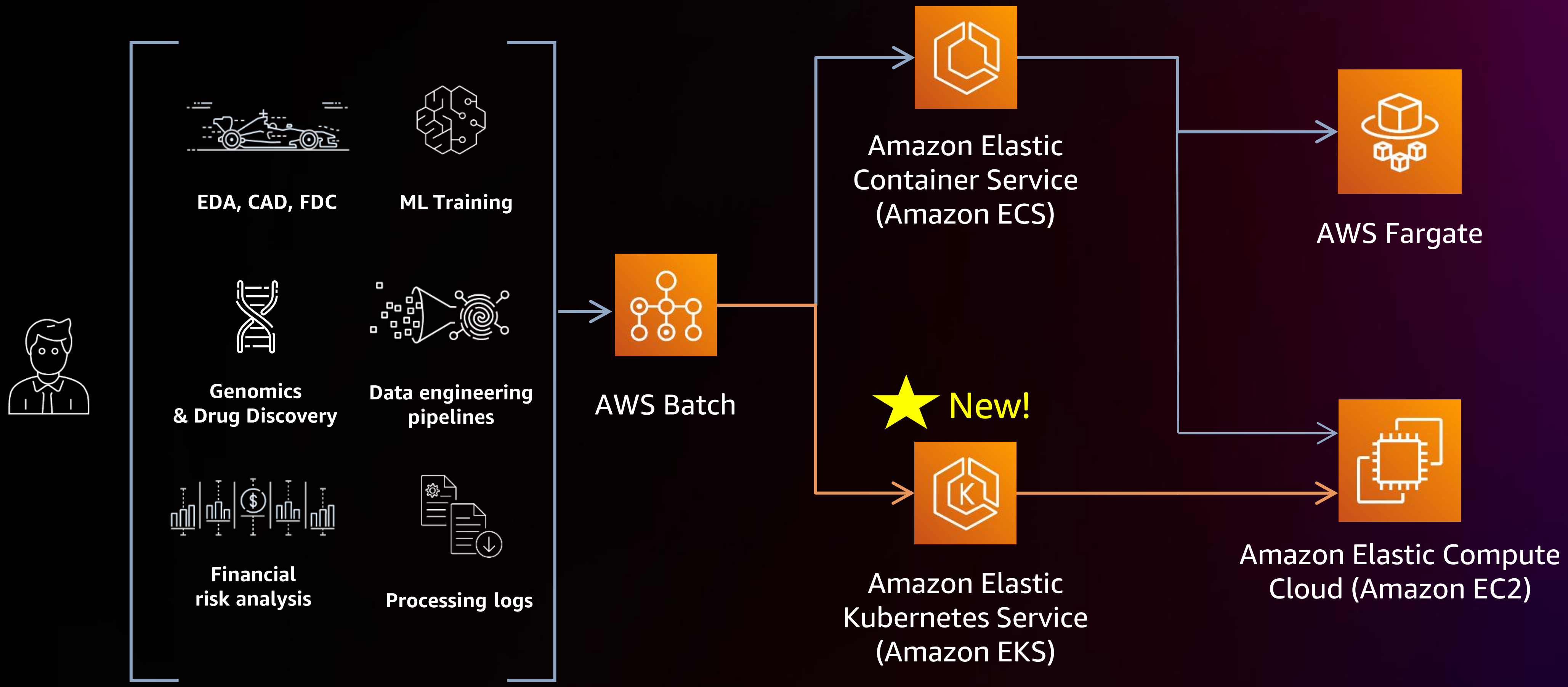
Presentation: Deep dive into AWS Batch for Amazon EKS

Hands-on: Workshop exercises

AWS Batch for Amazon EKS

AWS Batch

FULLY MANAGED BATCH SCHEDULER FOR CONTAINERIZED WORKLOADS


EDA, CAD, FDC
ML Training
Genomics & Drug Discovery
Data engineering pipelines
Financial risk analysis
Processing logs
AWS Batch
Amazon Elastic Container Service (Amazon ECS)
AWS Fargate
★ New!
Amazon Elastic Kubernetes Service (Amazon EKS)
Amazon Elastic Compute Cloud (Amazon EC2)

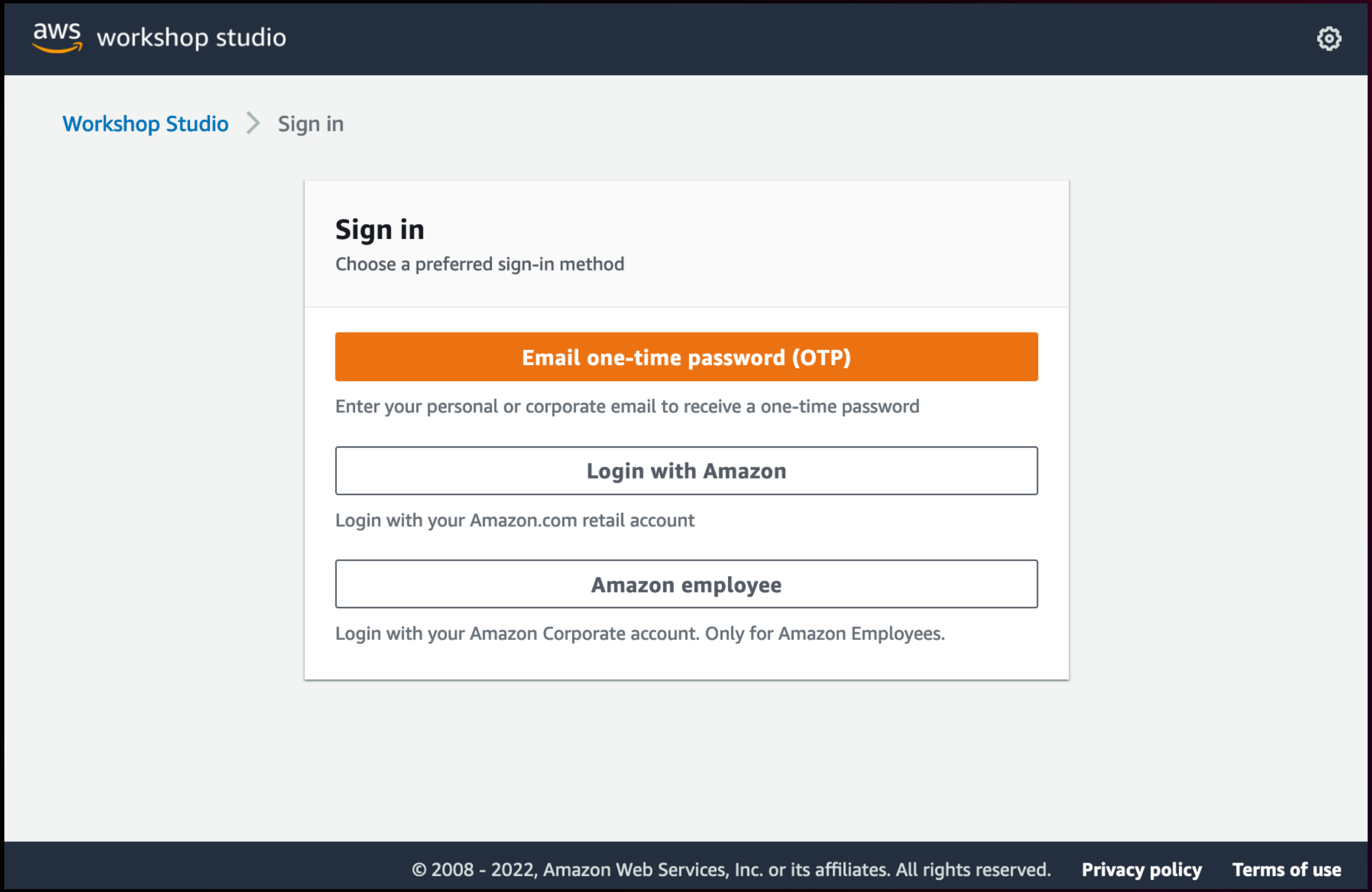
Getting started with this workshop

Getting started with this workshop

- As a participant, you will have access to an AWS account with any optional pre-provisioned infrastructure and IAM policies needed to complete this workshop.
- The AWS account will only be available for the duration of this workshop. You will lose access to the account thereafter.
- The optional pre-provisioned infrastructure will be deployed to a specific region. Check your workshop content to determine whether other regions will be used.
- Be sure to review the terms and conditions of the event. Do not upload any personal or confidential information in the account.

Step 1: Sign in via your preferred method

CMP335-R https://hpc.news/curie577 CMP335-R1 https://hpc.news/darwin47



Sign in

Choose a preferred sign-in method

Enter your personal or corporate email to receive a one-time password

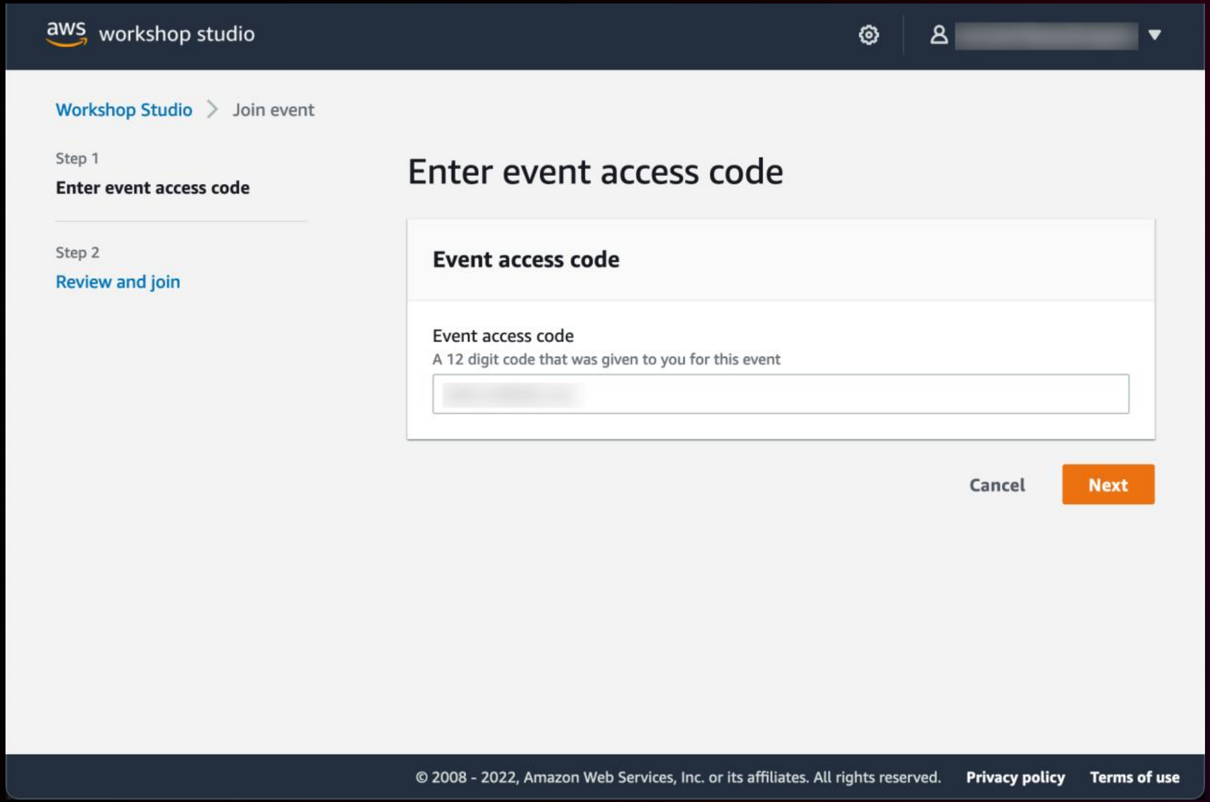
Login with your Amazon.com retail account

Login with your Amazon Corporate account. Only for Amazon Employees.

© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy policy Terms of use

Step 2: Enter event access code

Your access code should be pre-populated; choose **Next**



[redacted]

Workshop Studio > Join event

Step 1
Enter event access code

Step 2
Review and join

Enter event access code

Event access code

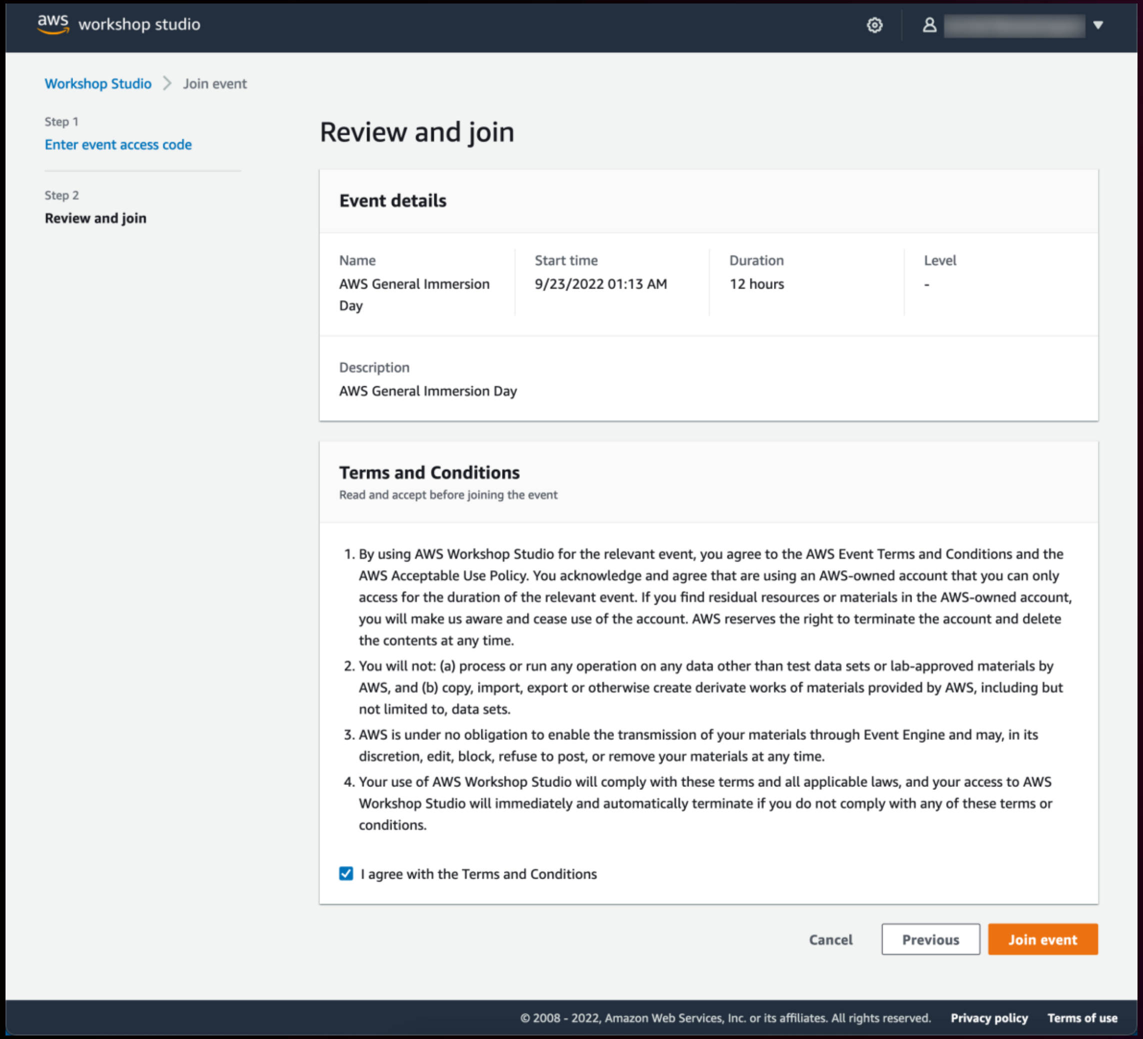
Event access code
A 12 digit code that was given to you for this event

[redacted]

Cancel | Next

Privacy policy **Terms of use**

Step 3: Review terms and join event



Workshop Studio > Join event

Step 1
Enter event access code

Step 2
Review and join

Review and join

Event details

Name	Start time	Duration	Level
AWS General Immersion Day	9/23/2022 01:13 AM	12 hours	-

Description
AWS General Immersion Day

Terms and Conditions

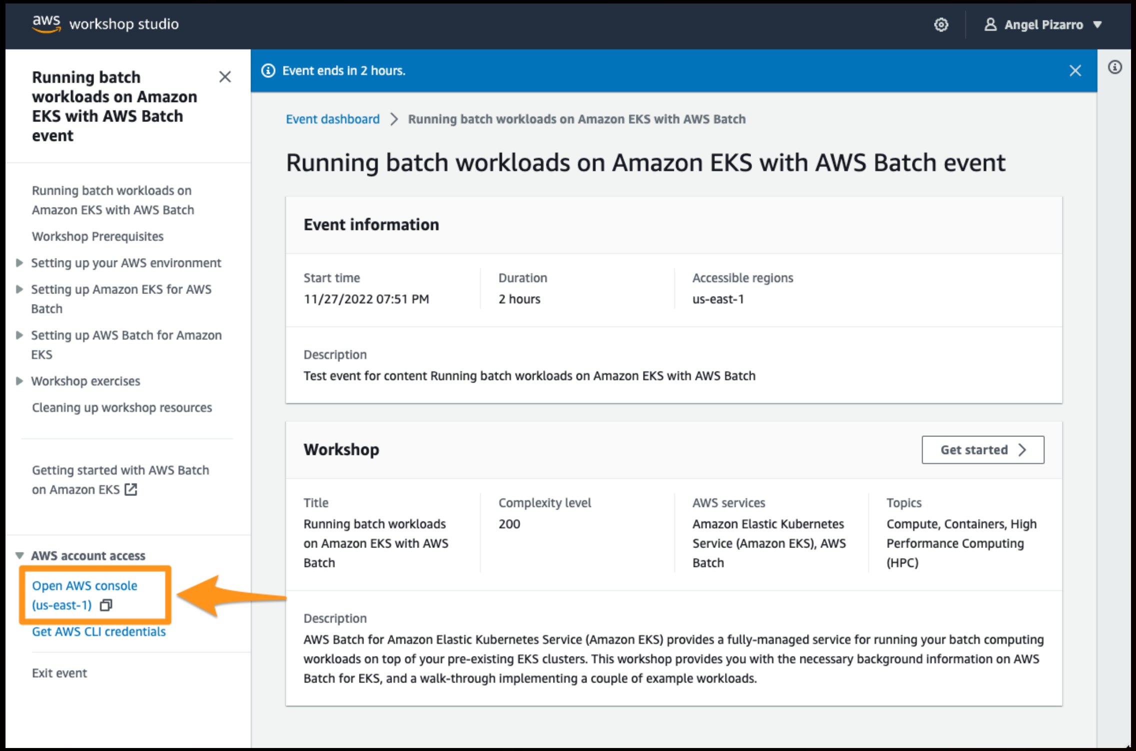
Read and accept before joining the event

1. By using AWS Workshop Studio for the relevant event, you agree to the AWS Event Terms and Conditions and the AWS Acceptable Use Policy. You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivate works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through Event Engine and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of AWS Workshop Studio will comply with these terms and all applicable laws, and your access to AWS Workshop Studio will immediately and automatically terminate if you do not comply with any of these terms or conditions.

☑ I agree with the Terms and Conditions

Step 4: Access AWS account

Access the AWS Console in a new browser tab



aws workshop studio Angel Pizarro ▼

Running batch workloads on Amazon EKS with AWS Batch event ✕

- Running batch workloads on Amazon EKS with AWS Batch
- Workshop Prerequisites
- ▸ Setting up your AWS environment
- ▸ Setting up Amazon EKS for AWS Batch
- ▸ Setting up AWS Batch for Amazon EKS
- ▸ Workshop exercises
- Cleaning up workshop resources
- Getting started with AWS Batch on Amazon EKS

▾ AWS account access

- Open AWS console (us-east-1)
- Get AWS CLI credentials

Exit event

ⓘ Event ends in 2 hours. ✕

Event dashboard > Running batch workloads on Amazon EKS with AWS Batch

Running batch workloads on Amazon EKS with AWS Batch event

Event information

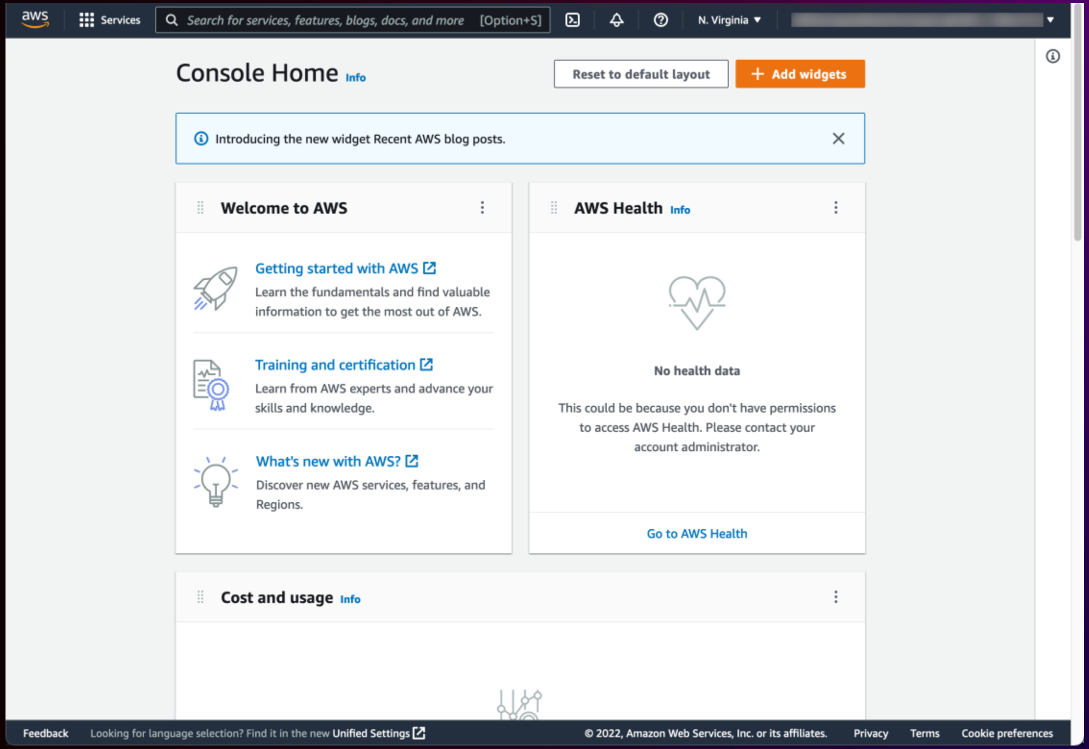
Start time	Duration	Accessible regions
11/27/2022 07:51 PM	2 hours	us-east-1

Description
Test event for content Running batch workloads on Amazon EKS with AWS Batch

Workshop Get started >

Title	Complexity level	AWS services	Topics
Running batch workloads on Amazon EKS with AWS Batch	200	Amazon Elastic Kubernetes Service (Amazon EKS), AWS Batch	Compute, Containers, High Performance Computing (HPC)

Description
AWS Batch for Amazon Elastic Kubernetes Service (Amazon EKS) provides a fully-managed service for running your batch computing workloads on top of your pre-existing EKS clusters. This workshop provides you with the necessary background information on AWS Batch for EKS, and a walk-through implementing a couple of example workloads.



aws Services Search for services, features, blogs, docs, and more [Option+S] N. Virginia ▼ [redacted]

Console Home Info Reset to default layout + Add widgets

ⓘ Introducing the new widget Recent AWS blog posts. ✕

Welcome to AWS

- Getting started with AWS — Learn the fundamentals and find valuable information to get the most out of AWS.
- Training and certification — Learn from AWS experts and advance your skills and knowledge.
- What's new with AWS? — Discover new AWS services, features, and Regions.

AWS Health Info

No health data

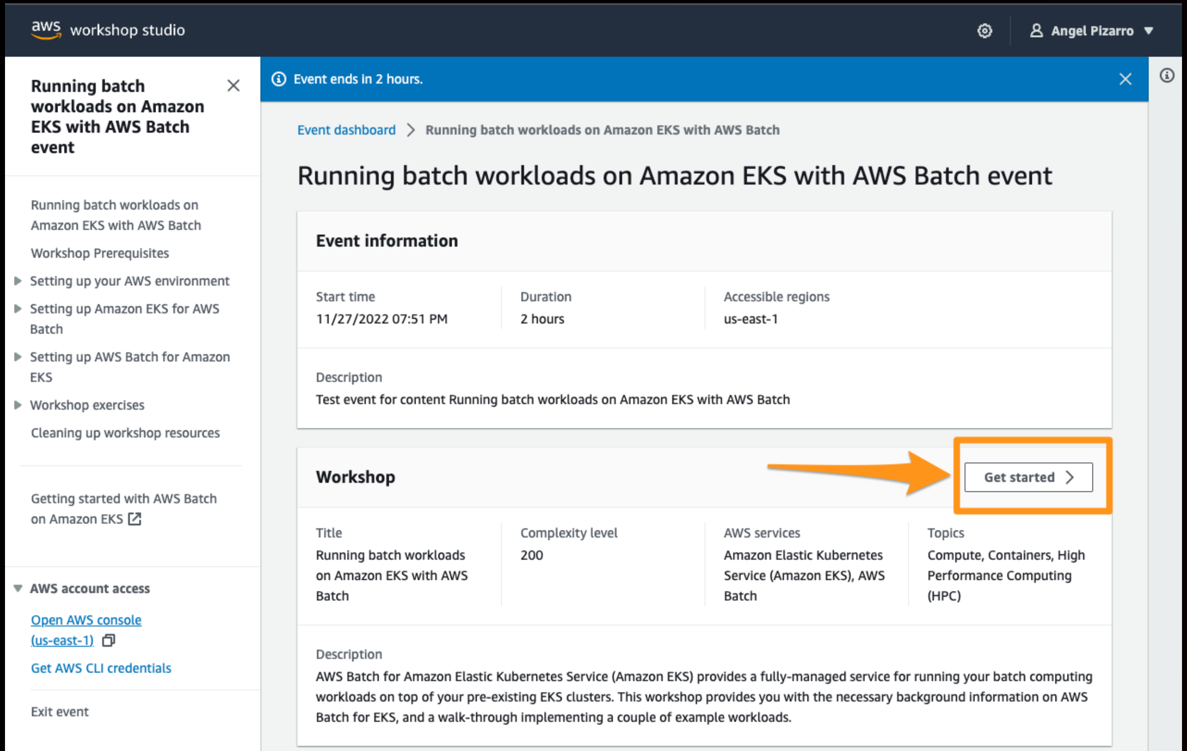
This could be because you don't have permissions to access AWS Health. Please contact your account administrator.

Go to AWS Health

Cost and usage Info

Feedback Looking for language selection? Find it in the new Unified Settings © 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 5: Get started with the workshop



Running batch workloads on Amazon EKS with AWS Batch event

- Running batch workloads on Amazon EKS with AWS Batch
- Workshop Prerequisites
- Setting up your AWS environment
- Setting up Amazon EKS for AWS Batch
- Setting up AWS Batch for Amazon EKS
- Workshop exercises
- Cleaning up workshop resources

Getting started with AWS Batch on Amazon EKS

AWS account access

- Open AWS console (us-east-1)
- Get AWS CLI credentials

Exit event

Event ends in 2 hours.

Event dashboard > Running batch workloads on Amazon EKS with AWS Batch

Running batch workloads on Amazon EKS with AWS Batch event

Event information

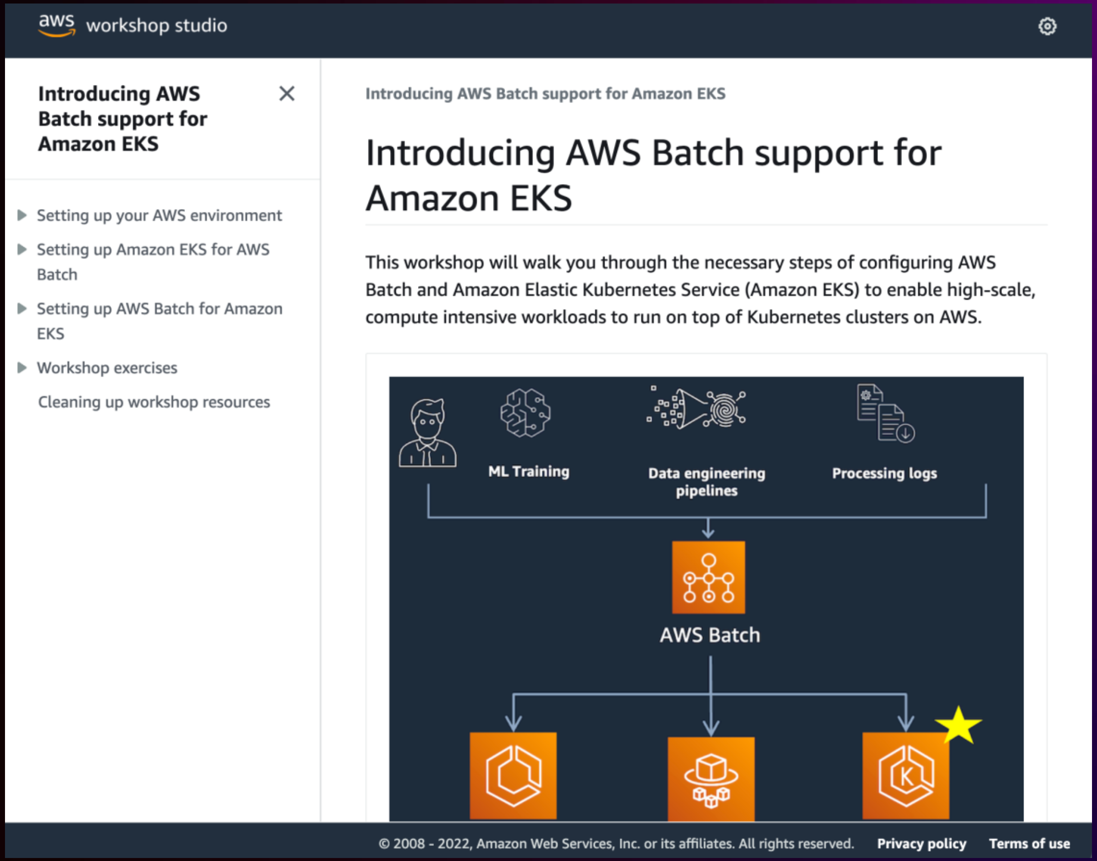
Start time	Duration	Accessible regions
11/27/2022 07:51 PM	2 hours	us-east-1

Description
Test event for content Running batch workloads on Amazon EKS with AWS Batch

Workshop

Title	Complexity level	AWS services	Topics
Running batch workloads on Amazon EKS with AWS Batch	200	Amazon Elastic Kubernetes Service (Amazon EKS), AWS Batch	Compute, Containers, High Performance Computing (HPC)

Description
AWS Batch for Amazon Elastic Kubernetes Service (Amazon EKS) provides a fully-managed service for running your batch computing workloads on top of your pre-existing EKS clusters. This workshop provides you with the necessary background information on AWS Batch for EKS, and a walk-through implementing a couple of example workloads.



Introducing AWS Batch support for Amazon EKS

- Setting up your AWS environment
- Setting up Amazon EKS for AWS Batch
- Setting up AWS Batch for Amazon EKS
- Workshop exercises
- Cleaning up workshop resources

Introducing AWS Batch support for Amazon EKS

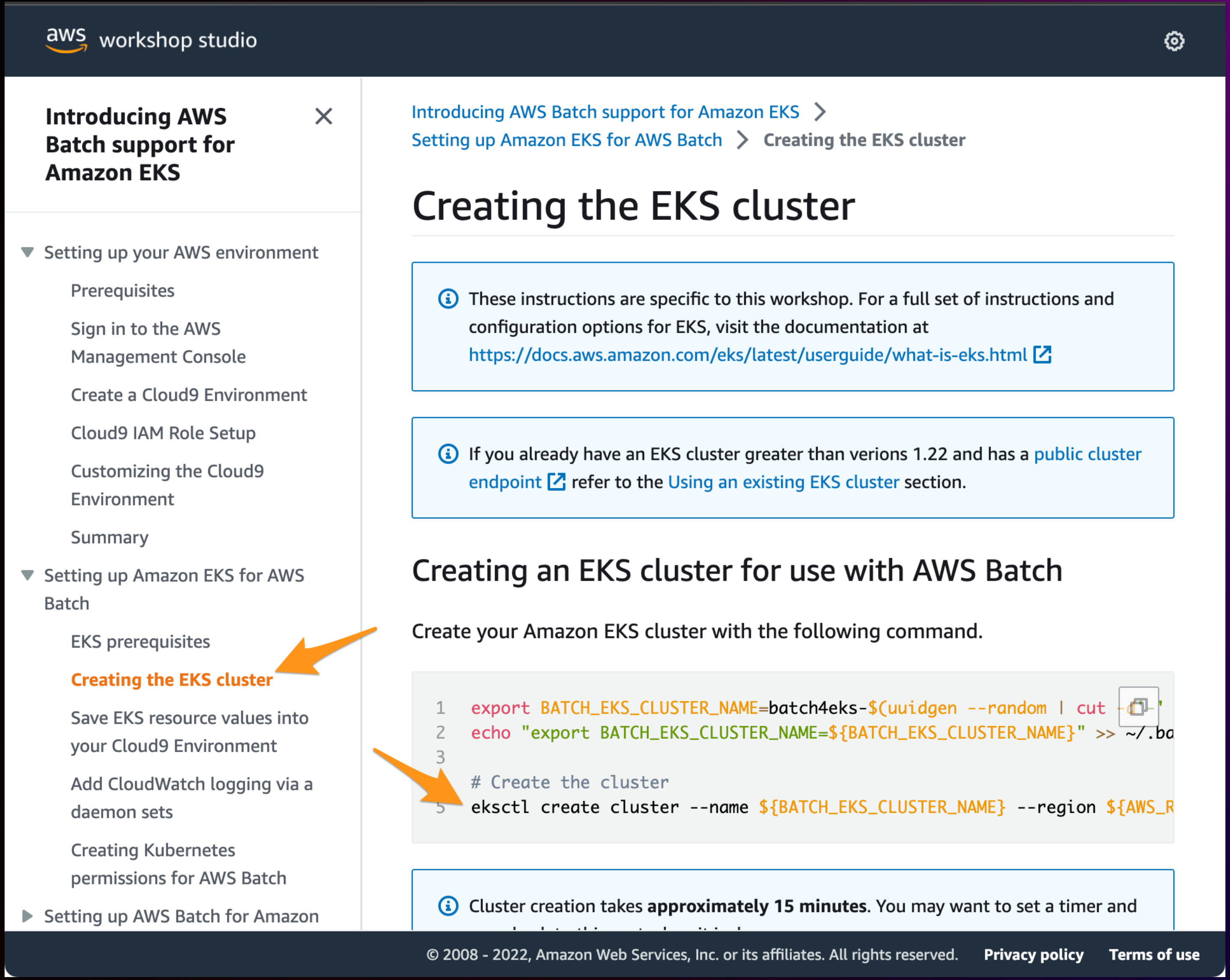
Introducing AWS Batch support for Amazon EKS

This workshop will walk you through the necessary steps of configuring AWS Batch and Amazon Elastic Kubernetes Service (Amazon EKS) to enable high-scale, compute intensive workloads to run on top of Kubernetes clusters on AWS.

© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy policy Terms of use

Step 6: Work until you start your EKS cluster

Complete all steps up through starting a Kubernetes cluster using `eksctl`



aws workshop studio

Introducing AWS Batch support for Amazon EKS > Setting up Amazon EKS for AWS Batch > Creating the EKS cluster

Creating the EKS cluster

These instructions are specific to this workshop. For a full set of instructions and configuration options for EKS, visit the documentation at https://docs.aws.amazon.com/eks/latest/userguide/what-is-eks.html

If you already have an EKS cluster greater than verions 1.22 and has a public cluster endpoint refer to the Using an existing EKS cluster section.

Creating an EKS cluster for use with AWS Batch

Create your Amazon EKS cluster with the following command.

```
export BATCH_EKS_CLUSTER_NAME=batch4eks-$(uuidgen --random | cut -d'
echo "export BATCH_EKS_CLUSTER_NAME=${BATCH_EKS_CLUSTER_NAME}" >> ~/.ba

# Create the cluster
eksctl create cluster --name ${BATCH_EKS_CLUSTER_NAME} --region ${AWS_R
```

Cluster creation takes **approximately 15 minutes**. You may want to set a timer and

AWS Batch for Amazon EKS

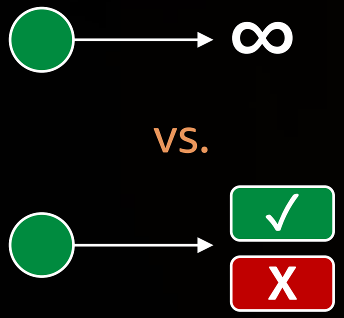
Kubernetes excels at managing microservices

Batch workloads, like ML training, differ from microservices in important ways

Stable vs. finite

Microservices are assumed to not stop once started.

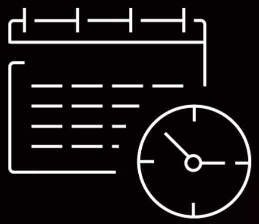
Batch workloads by definition run to completion (succeed or fail).



Response times

Microservices expect millisecond response times.

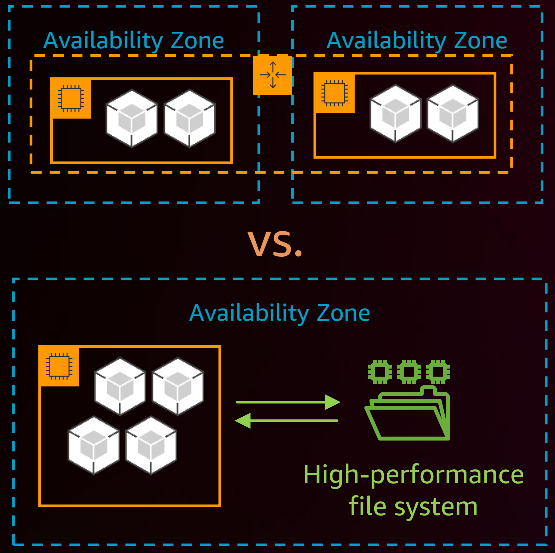
Batch results return of result expectations are at least minutes, sometimes days.



Replicated vs. not

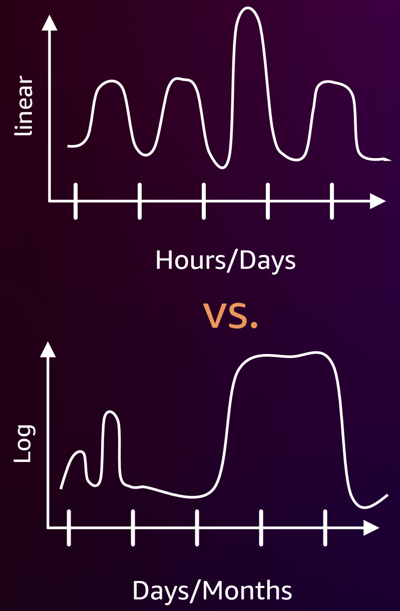
High-availability is not a thing for batch.

Fast access to Zone-level resources



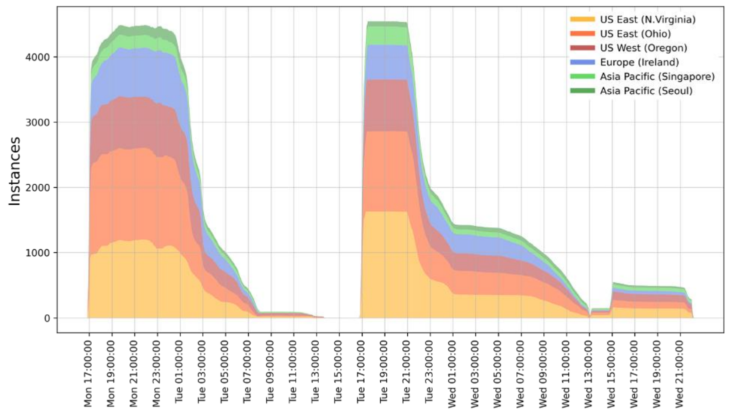
Cyclical vs. episodic

Both time and capacity scales diverge



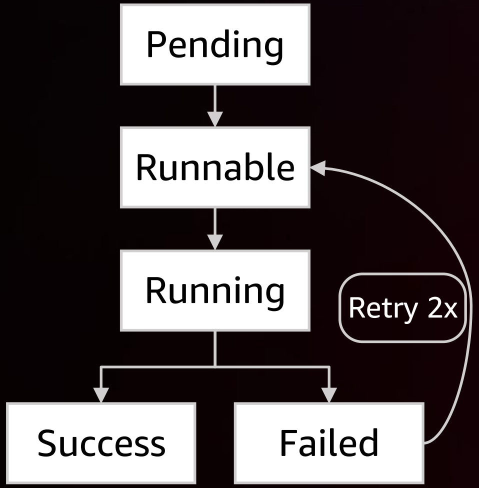
Additional batch workload requirements

Extreme scaling!



Launch 4500+ instances for a few hours, then scale to zero for months.

Job lifecycle, retry logic



Pending

Runnable

Running

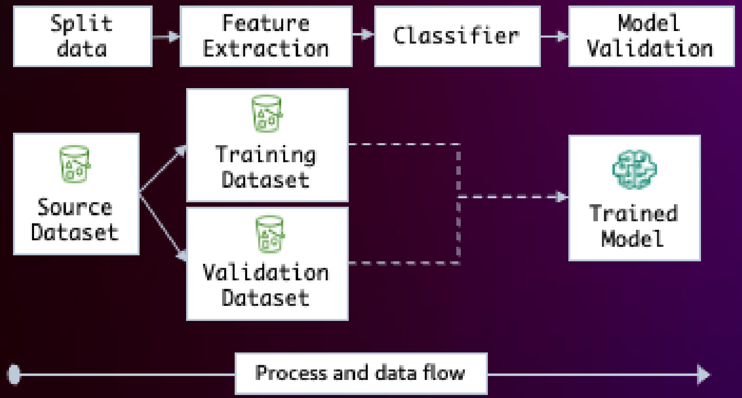
Retry 2x

Success

Failed

Retry logic can be modified by the exit status. E.g. only retry in the case of Spot reclaim

Job dependencies



Split data

Feature Extraction

Classifier

Model Validation

Source Dataset

Training Dataset

Validation Dataset

Trained Model

Process and data flow

Dependency across jobs can be defined *a priori* or at runtime.

If there is a failure early, *it makes no sense to keep going.*

Solutions for batch computing on Kubernetes

- Several open-source frameworks exists for batch computing
  - Apache YuniKorn, Volcano, Kueue
- Ability to creat your own solution with general purpose scheduler + proper batch-style node scaling
  - Karpenter, Hashicorp Nomad
  - Leverage the Kubernetes Jobs API

All of these solutions strive to meet the challenges of batch workloads on Kubernetes, but . . .

We heard from customers that running high-scale, compute-intensive workloads on Kubernetes can be challenging

Running and scaling batch workloads on Kubernetes is difficult and requires significant investment

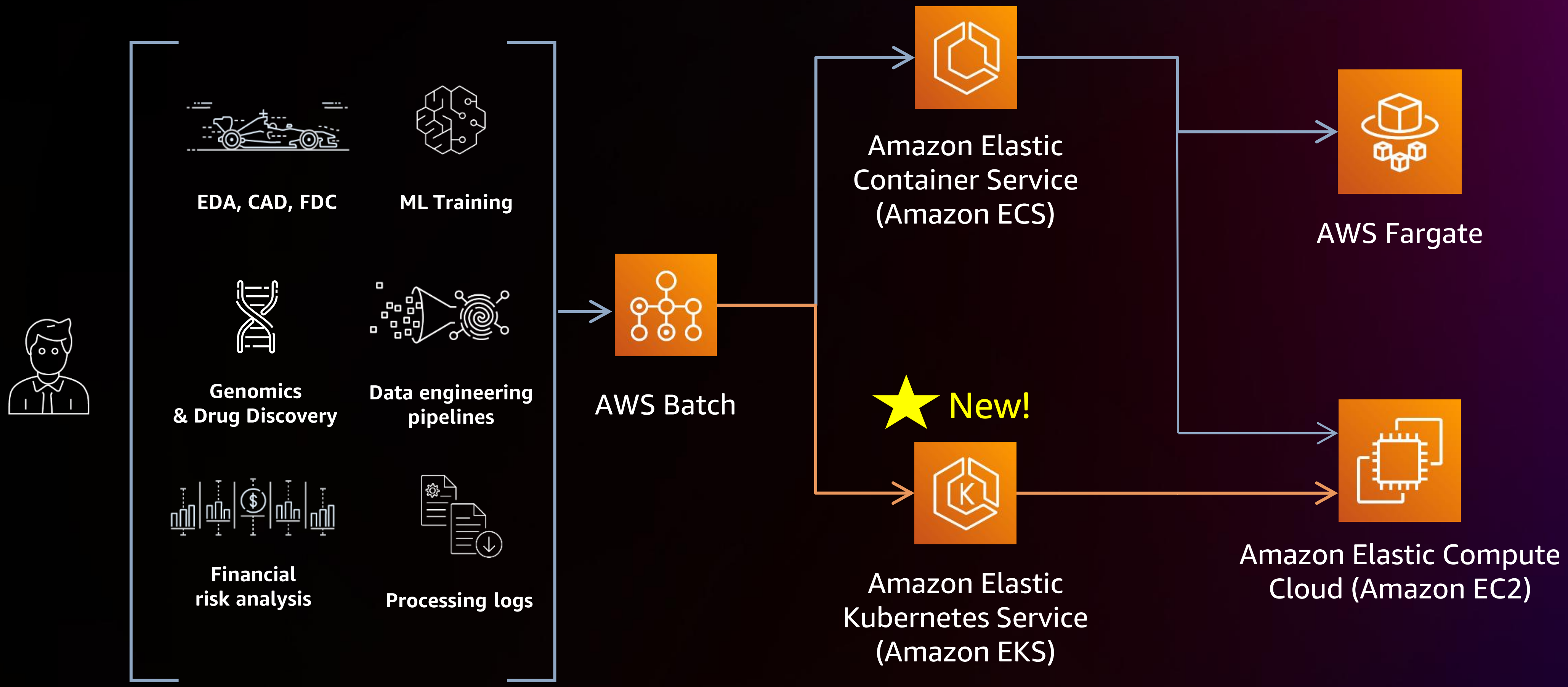
Operational and cost optimizations are different for batch workloads

Scheduling pods quickly and efficiently is a challenge for the k8s native scheduler

All of this extra work was leading to a lot of undifferentiated heavy lifting

AWS Batch

FULLY MANAGED BATCH SCHEDULER FOR CONTAINERIZED WORKLOADS


EDA, CAD, FDC
ML Training
Genomics & Drug Discovery
Data engineering pipelines
Financial risk analysis
Processing logs
AWS Batch
Amazon Elastic Container Service (Amazon ECS)
AWS Fargate
New!
Amazon Elastic Kubernetes Service (Amazon EKS)
Amazon Elastic Compute Cloud (Amazon EC2)

What is AWS Batch?

Job scheduler

- Schedules and runs jobs asynchronously
- Manages dependencies

Resource orchestrator

- Manages and optimizes compute resources
- Scales up/down as needed
- Utilizes the right compute resources for the job

Fully managed

Integrated with AWS services

Massive scalability

Optimized resource provisioning

Cost-efficient

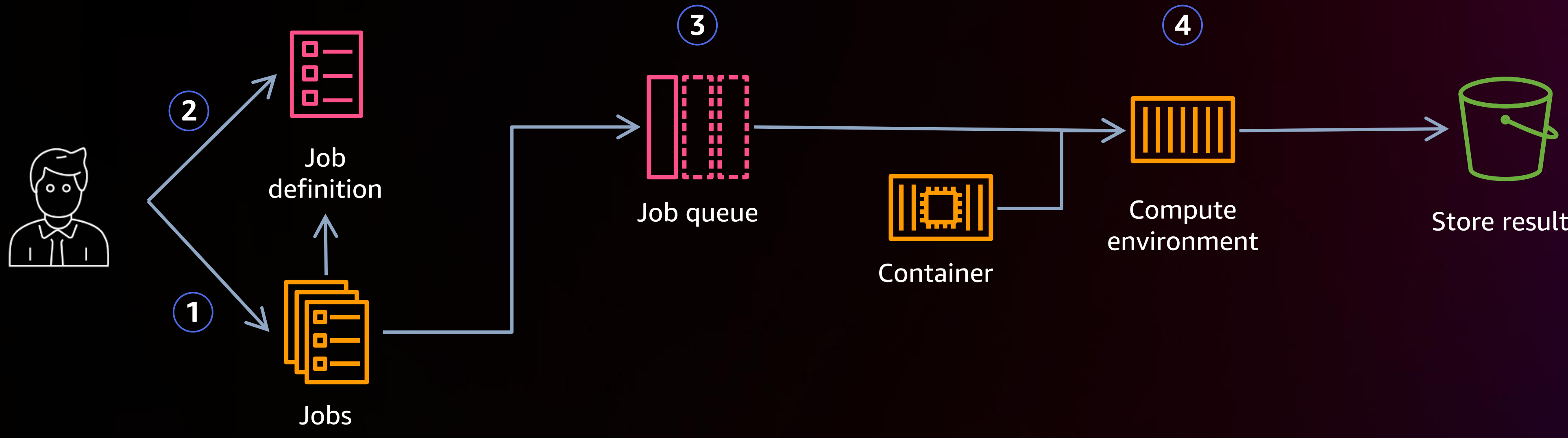
AWS Batch overview

1 **Job**
Actual work request. Each job must reference a job definition, but many parameters may be overridden when submitted

2 **Job definition**
Template that has common attributes (such as container image, IAM role, vCPU and memory requirements)

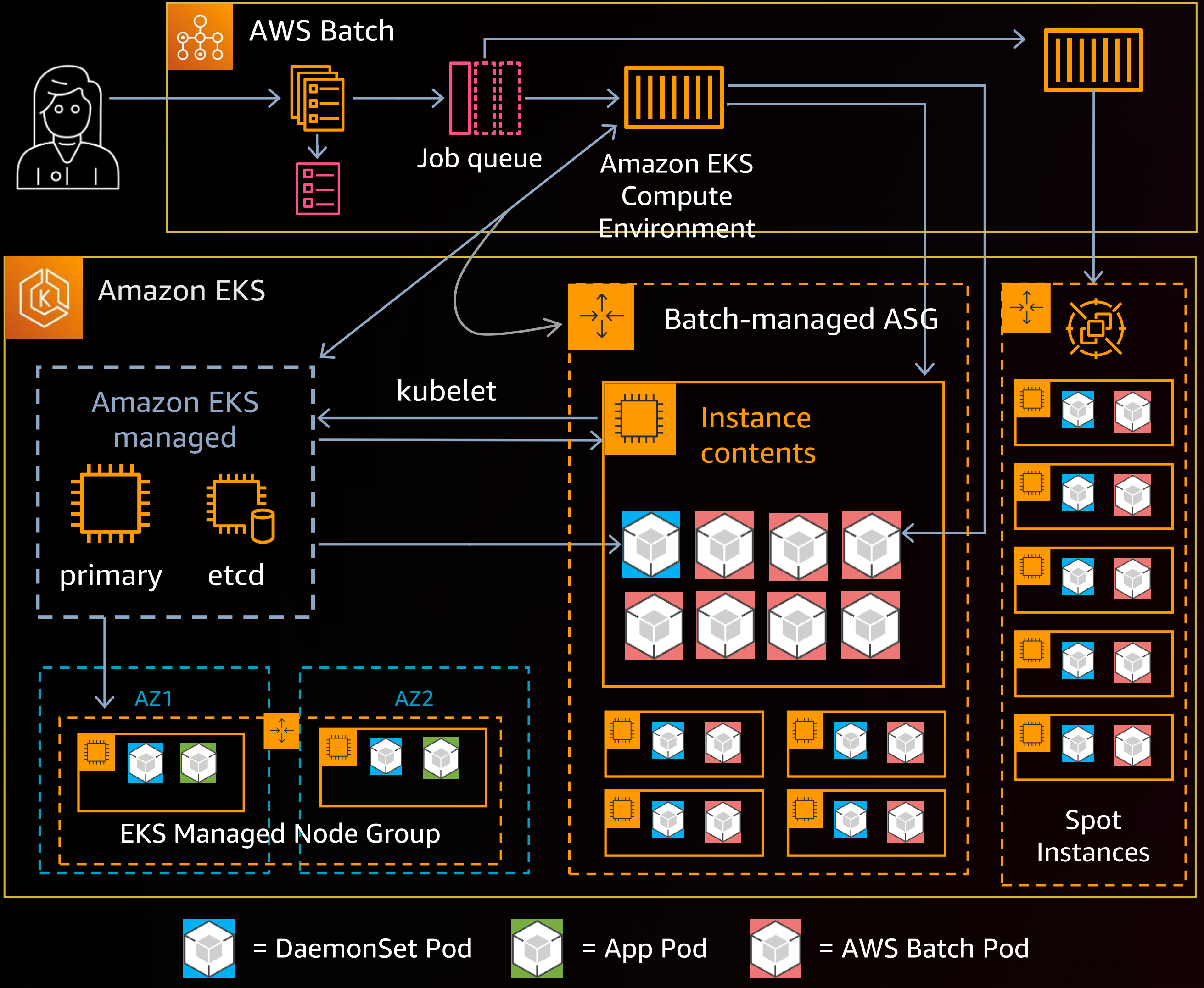
3 **Job queue (JQ)**
Queue determines priorities. Each JQ is connected to one or more CE

4 **Compute environment (CE)**
Resource mix (defines on-demand vs. Spot and other instance types); CE can be connected to more than one JQ



Diving into the details

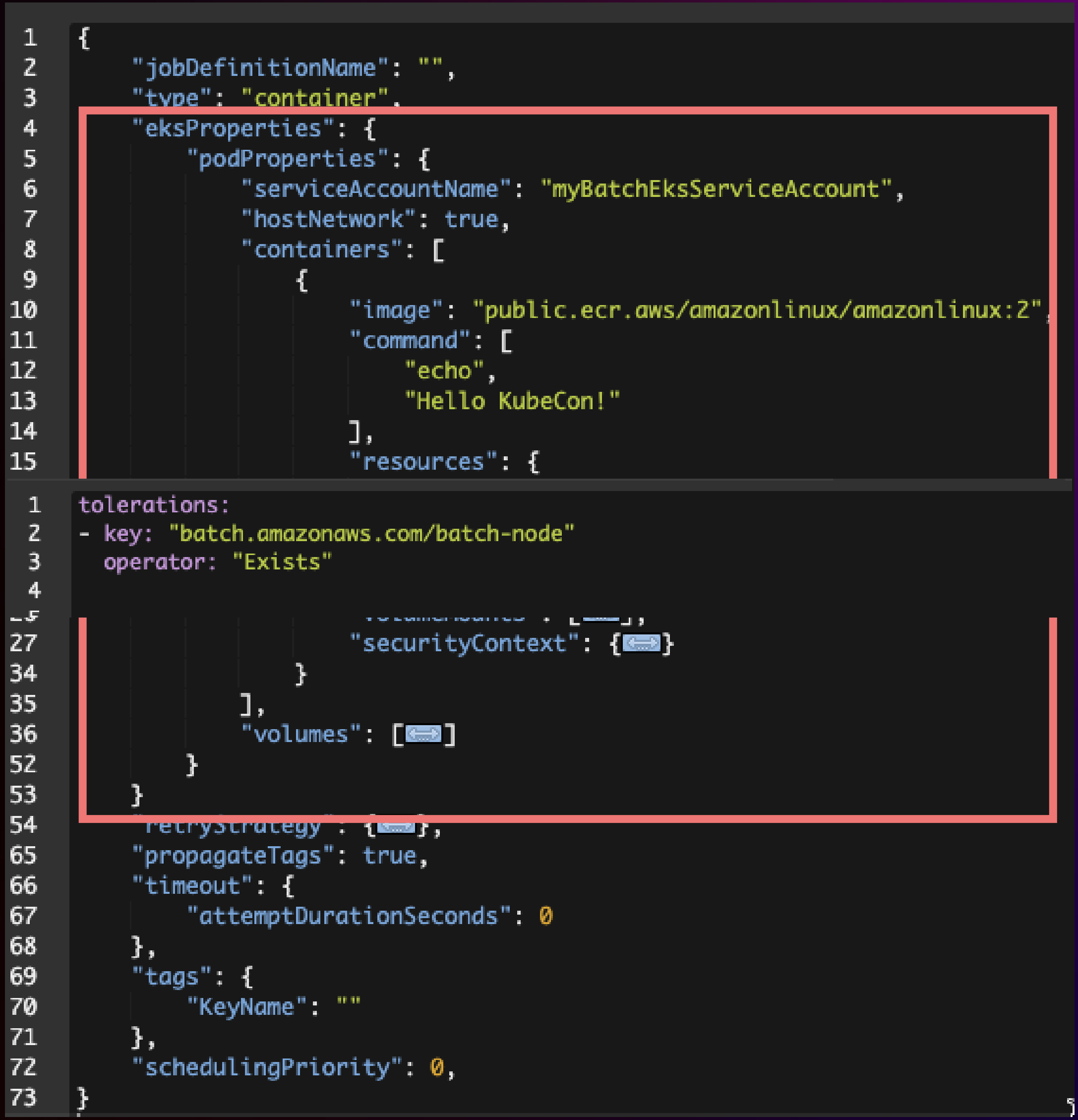
AWS Batch for Amazon EKS – Under the hood



- Batch is the main entry point for batch workload requests
- Batch launches worker nodes based on the jobs queued
- Batch nodes are separate from other EKS node groups
  - Using Batch's scaling knowledge for capacity pools and Spot interruption likelihood
  - Taints prevent placement of other pods on Batch-managed nodes
- Batch handles the pod placements via nodename
- Multiple Compute Environments per cluster

Amazon EKS specific API parameters

- Amazon EKS compute environments and job definitions have **separate parameters from Amazon ECS**
- AWS Batch supports EKS versions **1.21+, but 1.22+ is recommended**
- You can scope a DaemonSet to Batch nodes using **tolerations**



```
{
    "jobDefinitionName": "",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "serviceAccountName": "myBatchEksServiceAccount",
            "hostNetwork": true,
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": [
                        "echo",
                        "Hello KubeCon!"
                    ],
                    "resources": {
```

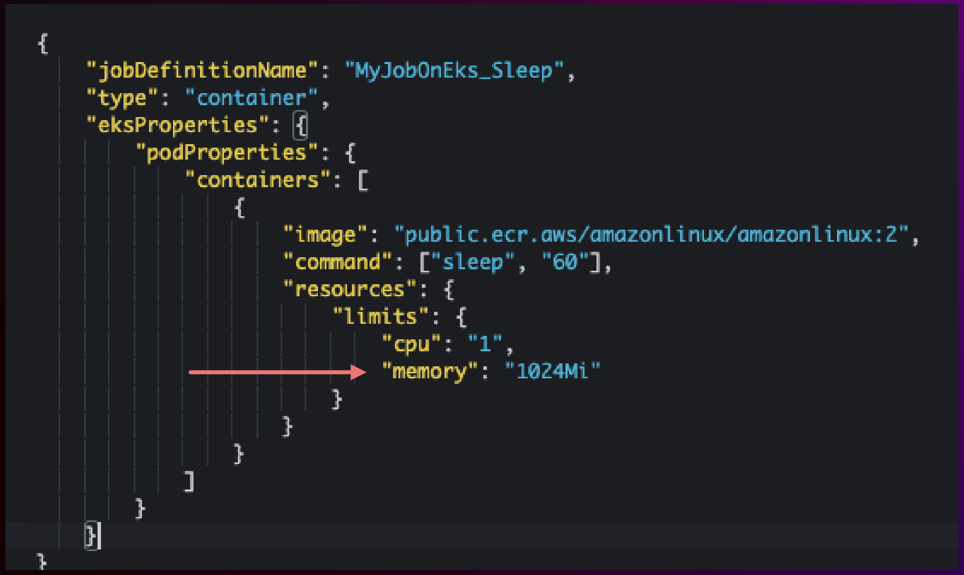
```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"

```

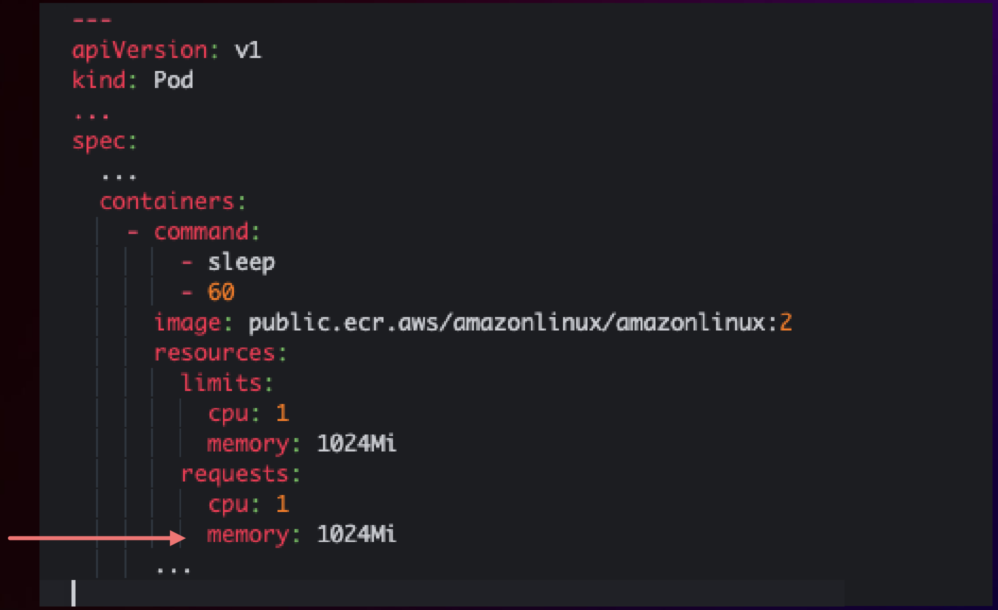
```
                    "volumeMounts": [⟺],
                    "securityContext": {⟺}
                }
            ],
            "volumes": [⟺]
        }
    }
    "retryStrategy": {⟺},
    "propagateTags": true,
    "timeout": {
        "attemptDurationSeconds": 0
    },
    "tags": {
        "KeyName": ""
    },
    "schedulingPriority": 0,
}
```

vCPU and memory considerations for EKS jobs

- Batch on EKS only supports setting the **limits** configuration in container resources for jobs
- Batch also has a unique set of constraints for `cpu` and `memory` specifications
  - `cpu` can be specified only in whole cpu values (ie., 2) or in fraction forms
  - When cpu is specified in fraction forms, it must be in increments of 0.25
  - cpu cannot be specified in millCPU form, i.e. 100m
  - cpu must be >= 0.25 (the smallest cpu size of a job)
- `memory` can only be specified in Mebibytes (Mi) form
- When Batch translates an EKS Job into a pod, it sets `request` equal to `limits`



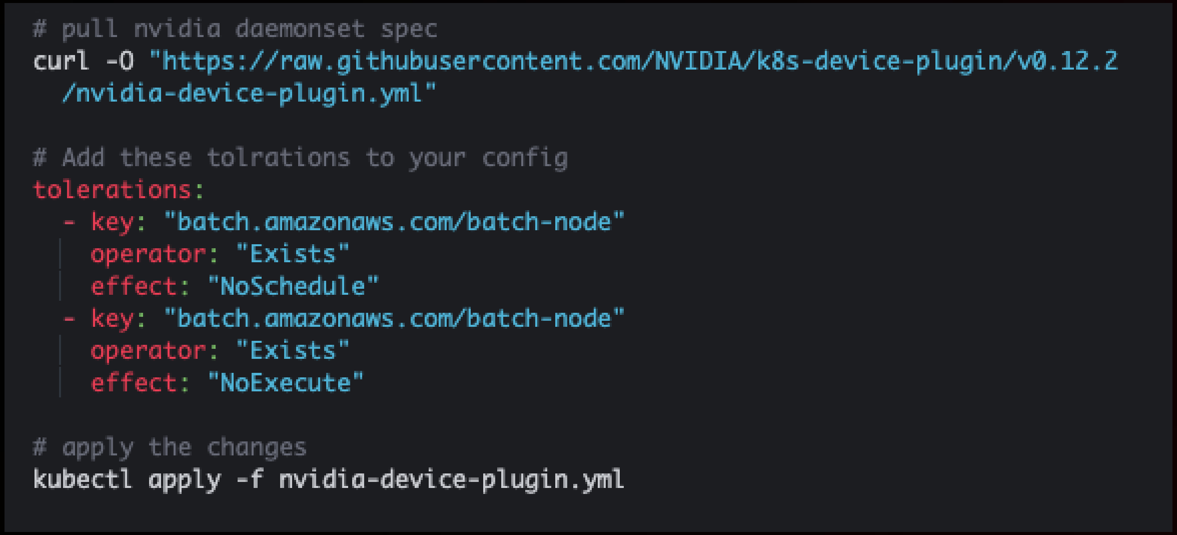
```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi"
                        }
                    }
                }
            ]
        }
    }
}
```



```
---
apiVersion: v1
kind: Pod
...
spec:
  ...
  containers:
    - command:
        - sleep
        - 60
      image: public.ecr.aws/amazonlinux/amazonlinux:2
      resources:
        limits:
          cpu: 1
          memory: 1024Mi
        requests:
          cpu: 1
          memory: 1024Mi
      ...
```

GPU workloads

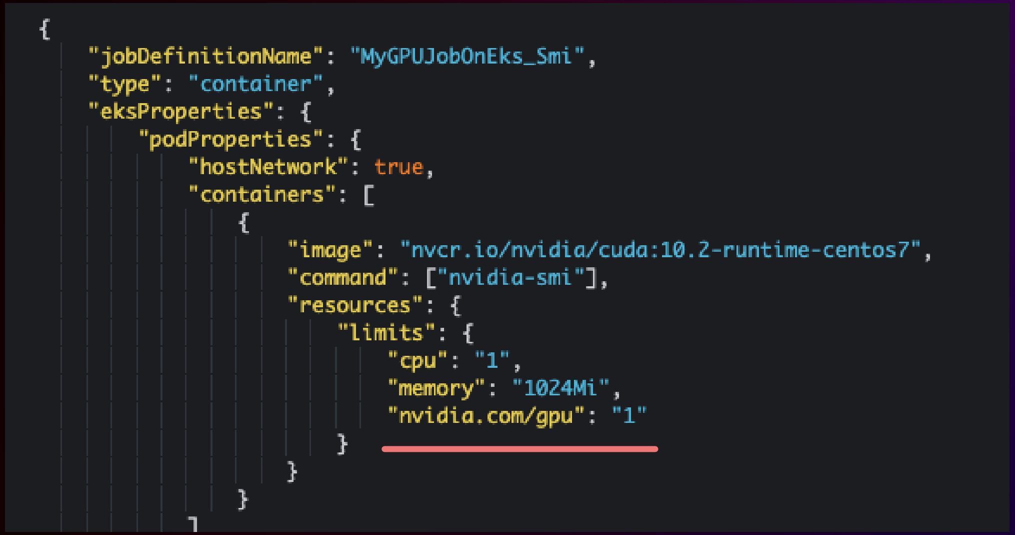
- Batch supports P2, P3, P4, G3 and G4 instance families
- By default, Batch uses the Amazon EKS Accelerated AMI with Kubernetes version matching your Amazon EKS cluster control-plane version
- Batch **does not** manage the NVIDIA GPU device plugin on your behalf. You must install this plugin into your Amazon EKS cluster as a DaemonSet!



```
# pull nvidia daemonset spec
curl -O "https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/v0.12.2
  /nvidia-device-plugin.yml"

# Add these tolrations to your config
tolerations:
  - key: "batch.amazonaws.com/batch-node"
    operator: "Exists"
    effect: "NoSchedule"
  - key: "batch.amazonaws.com/batch-node"
    operator: "Exists"
    effect: "NoExecute"

# apply the changes
kubectl apply -f nvidia-device-plugin.yml
```



```
{
    "jobDefinitionName": "MyGPUJobOnEks_Smi",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "hostNetwork": true,
            "containers": [
                {
                    "image": "nvcr.io/nvidia/cuda:10.2-runtime-centos7",
                    "command": ["nvidia-smi"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi",
                            "nvidia.com/gpu": "1"
                        }
                    }
                }
            ]
```

Shared responsibility – The infrastructure edition

This is your cluster! AWS Batch is a good tenant, but will need some permissions defined.

OIDC, RBAC, namespace, AWS Identity and Access Management (IAM) service account identity mapping

Register the Amazon EKS cluster with AWS Batch as a CE

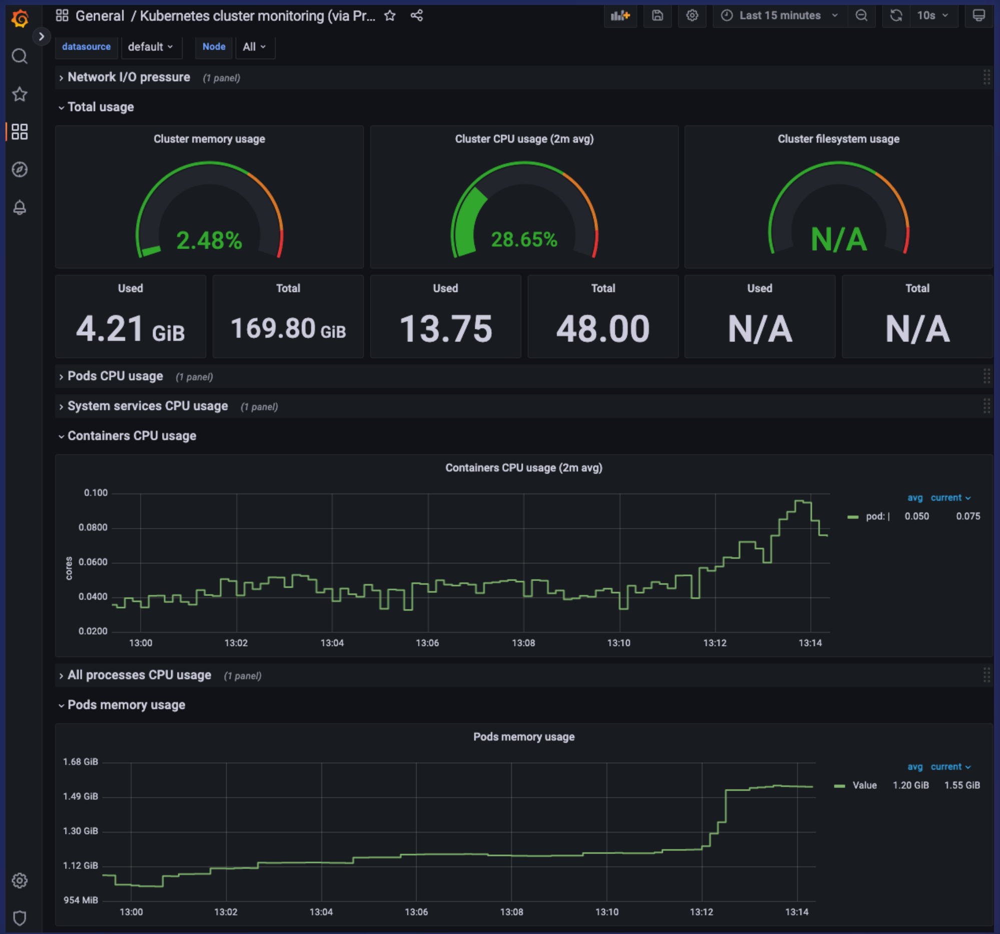
CE validate cluster for the proper permissions and version

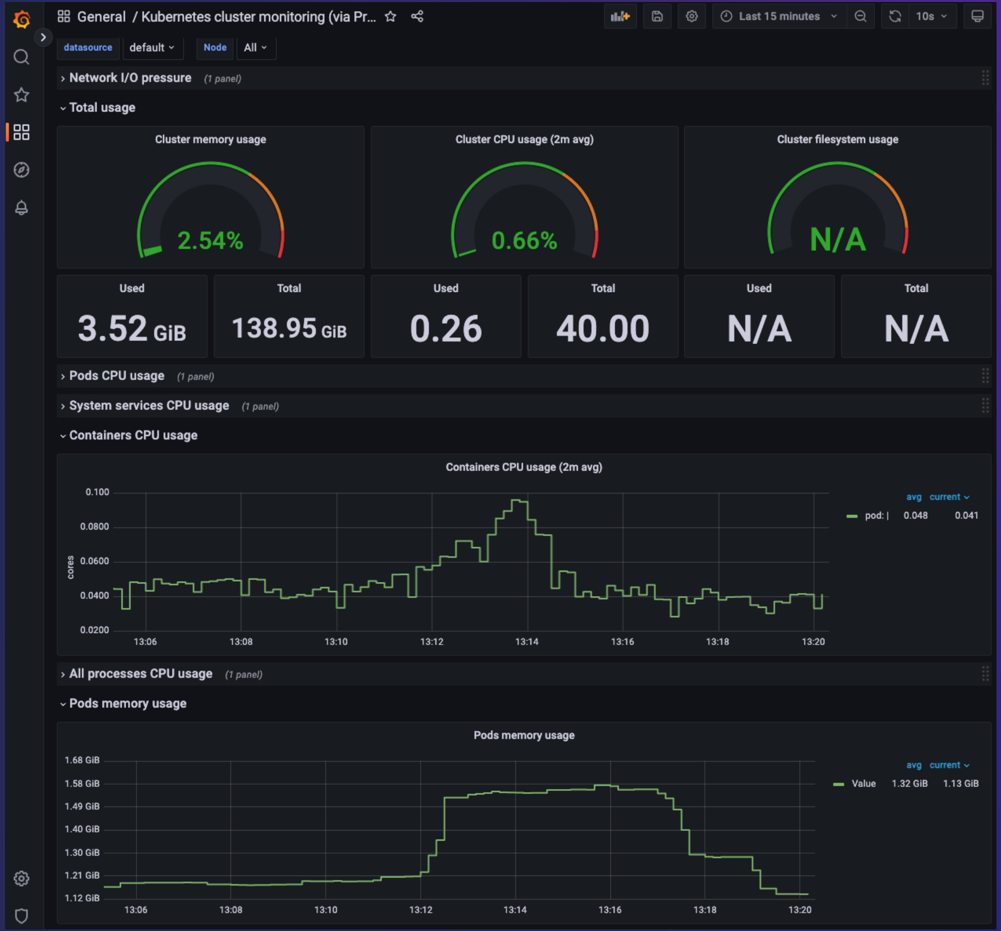
If the check fails, the CE is marked as INVALID

Why leverage AWS Batch for workloads on Amazon EKS?

- Managed services **reduce your operational and optimization overhead.** Lets you focus on business requirements
  - Benefit from improved operations and cost optimizations as they are developed.
- **Advanced batch features** like Fair-Share scheduling, job dependencies, and compute allocation strategies
- **Integration** with other AWS services (e.g. IAM, Amazon EventBridge, AWS Step Functions)
- Take advantage of the **Kubernetes ecosystem** of partners and tools

Monitoring and visualization with Prometheus and Grafana





Workshop exercises

Finalize environment setup

- Now that you have an Amazno EKS cluster started, you will
- Set up necessary permissions for Kubernetes, Amazon EKS, IAM and AWS Batch
- Set up Amazon EKS specific AWS Batch resources, including the compute environment and job queue

Exercise 1: Hello AWS Batch and Amazon EKS!

- Basic shell echo example that shows
- Setting up Amazon EKS specific AWS Batch job definitions
- Executing the AWS Batch job
- Review the results from the pod in Amazon CloudWatch Logs

Exercise 2: Process images in Amazon S3

- Create a customer Docker container image and push it to an (Amazon ECR
- Setting up IAM service accounts to access private Amazon S3 and Amazon ECR resources
- Define and run the AWS Batch jobs
- Review the result in Amazon S3 (optional)

Exercise 2: Calculate Pi and watch the cluster scale using Prometheus and Grafana

- Install and configure Prometheus and Grafana on the Amazon EKS Cluster
- Create a customer Docker image and push the image to Amazon ECR
- Create a Job Definition and submit an Array Job to stress the cluster
- Watch the cluster metrics scale as AWS Batch managed nodes are added and pods are running

Thank you!

Angel Pizarro
pizarroa@amazon.com

Maxime Hugues
maxhaws@amazon.com