Batch: When You Only Care It Runs Not When or Where or How

Jon Reiter 23 August 2018 DataFinnovation

Batch: The Basics

- System for batch processing (!)
- Official tag line: plans, schedules, and executes your batch computing workloads across the full range of AWS compute services and features
- What it really does:
 - Provisions resources for jobs
 - Manages job queueing
 - Arranges logging
 - Gives a dashboard

What About Jobs?

- Batch has it's own
 - Job definition scheme
 - Compute resource definition scheme
- It's related to ECS
 - You can tell it uses ECS inside
 - But the specs are non-ECS
- Doesn't do anything about scheduling jobs

Job Model

- A job is a container
 - When the container exits the job is done
 - It doesn't time out!
 - Full range of cpu and memory available
- You arrange kicking off your own jobs
 - CloudWatch + Lambda is your scheduler
 - After you put it in a queue Batch takes over
- Basic dependency controls

Scheduling Model

- Latency is high: 5-10 minutes to start is normal
- Shutdown is not immediate
 - Makes an attempt to minimize resource wastage
- Can use spot instances
 - Even has "optimal" instance type which finds the smallest/cheapest workable choice
- If you care deeply about these things do not use batch!
 - Also, reconsider your priorities

Resource Model

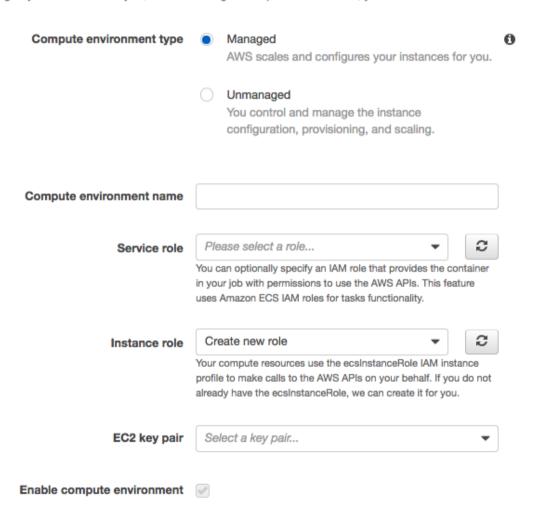
- Two parts
- Compute environments:
 - Instance type(s)
 - Size limits and some settings, but less than ECS
- Job queues:
 - On top of compute environment(s)
 - Basic priority control

The Workflow

- Job in a container
- Container in ECR
- Compute environment in Batch
- Job queue in Batch
- Job definition in Batch
- CloudWatch trigger to start
- Lambda from trigger to submit to queue
- Check the dashboard every so often for failures

Create a compute environment

A compute environment is a collection of AWS Batch compute resources that can be associated with a job queue. In a managed compute environment, AWS Batch handles the scaling and provisioning of your instances for you; in an unmanaged compute environment, you control the instances.



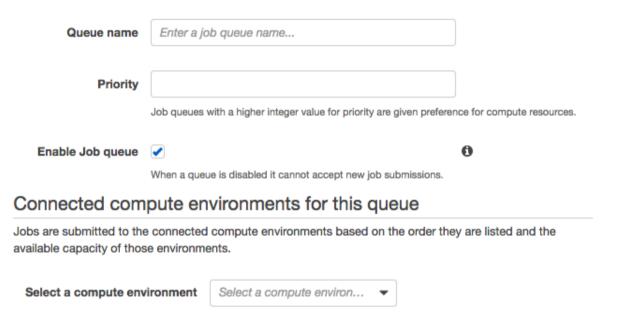
Configure your compute resources

Compute resources are the Amazon EC2 instances that jobs are placed on. There are multiple instance types that are available to use in AWS Batch as either ON-Demand or SpoT instances.

Provisioning model	 On-Demand EC2 instances that you pay by the hour for.
	Spot Save money by using Spot instances but be aware your instances can be interrupted with a two minute notification when EC2 needs the capacity back.
Allowed instance types	optimal • • •
	Optimal chooses the best fit of M4, C4, and R4 instance types available in the region.
	Jobs define their vCPU and memory requirements at submission and that information is used to match the job with the most appropriately sized instance.
Minimum vCPUs	0
	By keeping this set to 0 you will not have instance time wasted when there is no work to be run. If you set this above zero you will maintain that number of vCPUs at all times.
Desired vCPUs	0
Maximum vCPUs	256
	EC2 instance type limits will determine the maximum vCPUs you can have. Learn more

Create a job queue

A job queue accepts job submissions and places them on an appropriate compute environment.

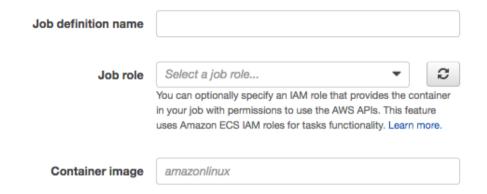


Cancel

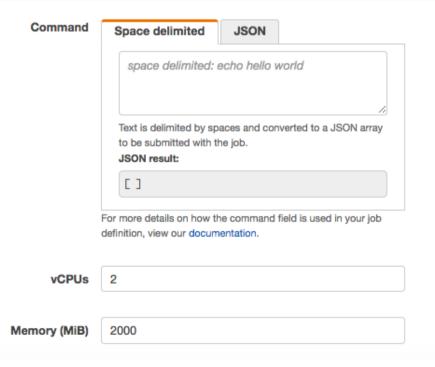
Create

Create a job definition

Job definitions allow you to save the values for a job so it can be used as a template later.



Environment



And Now Kick It Off...

import boto3

My (The) Use Case

- Some code needs to get run every so often
- As long as it finishes we want to spend as little money, time and effort as possible
- Batch
 - "optimal" instances running on spot
 - Push Docker container to ECR
 - Maintain lambda function
 - Store everything in S3/RDS/Dynamo/whatever
 - Check dashboard every so often