

How to cost optimize Jenkins jobs on Kubernetes with EC2 Spot Instances

I choose this project because of my familiarity with standalone Jenkins setup, configuration of free-style and pipelines projects. While preparing for CS 79C, I became familiar with Kubernetes, EC2 instances, etc. Hence combining all these areas would help me learn migration Jenkins to EC2 instances on Cloud9 and enhance my knowledge in these areas.

I will launch an Amazon EKS cluster with a managed node group running On-Demand instances for a Jenkins server. Then, I will set up another EKS managed node group running Spot Instances, and run a sample Jenkins build on those Spot Instances. Later I will also configure the build to retry in case Spot Instances are interrupted, when EC2 needs the capacity back

I will use Cloud9, Amazon EKS Cluster, EC2 Spot and On-Demand instances

Definitions:

Cloud9: AWS Cloud9 allows you to write, run, and debug your code with just a browser.

Amazon EKS Cluster: An **Amazon EKS cluster** consists of a control plane and the **Amazon EC2** or **AWS Fargate** compute that you run pods on.

EC2 Spot and On-Demand instances: A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price.

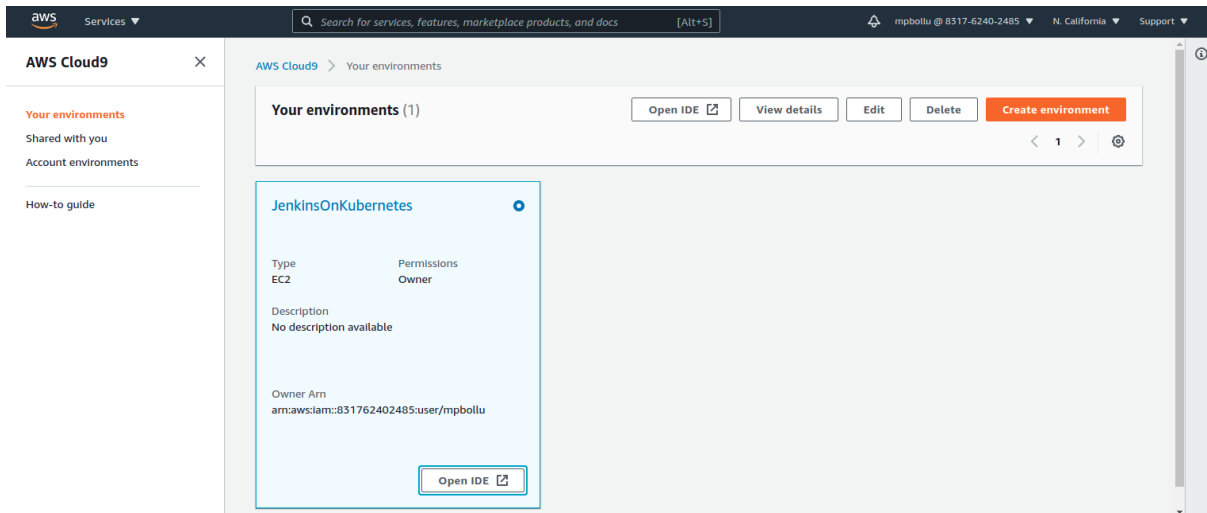
Jenkins: Automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery.

Referred to Documentation:

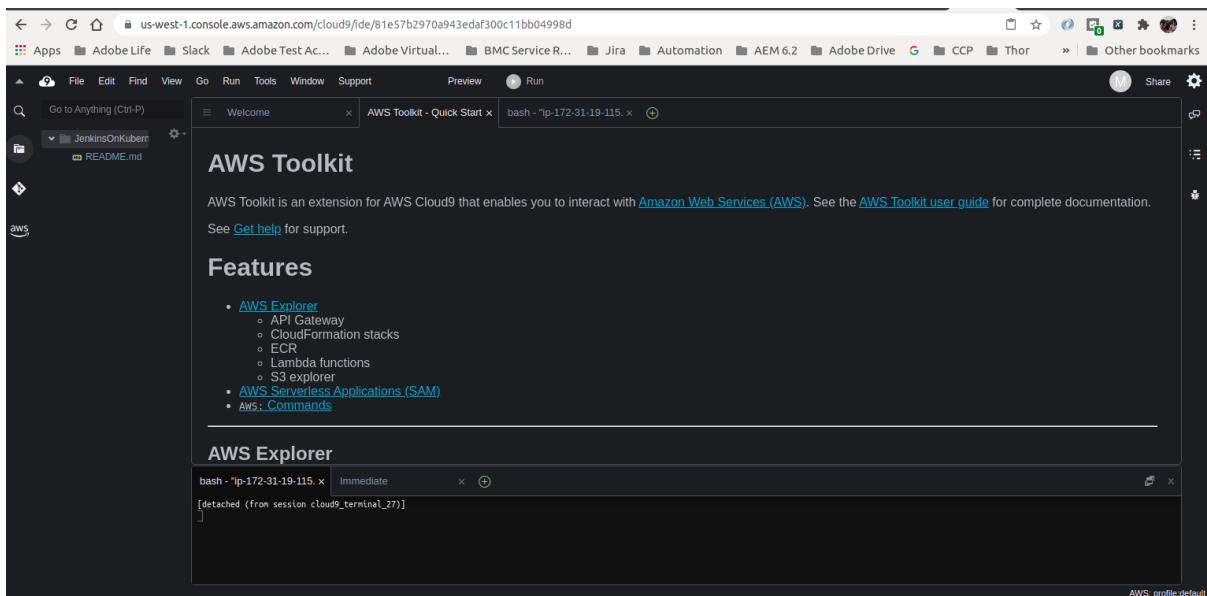
https://aws.amazon.com/getting-started/hands-on/cost-optimize-jenkins/?trk=gs_card

Step 1: Setup Cloud9

AWS Cloud9 environment created/running:



Open Cloud9 IDE:



Step 2: Setup Prerequisites

```
inflat: aws/dist/cryptography-3.3.2-py3.8.egg-info/LICENSE.APACHE
inflat: aws/dist/cryptography-3.3.2-py3.8.egg-info/LICENSE.PSF
inflat: aws/dist/cryptography-3.3.2-py3.8.egg-info/LICENSE
inflat: aws/dist/cryptography-3.3.2-py3.8.egg-info/LICENSE.BSD
inflat: aws/dist/cryptography-3.3.2-py3.8.egg-info/AUTHORS.rst
creating: aws/dist/include/python3.8/
inflat: aws/dist/include/python3.8/pyconfig.h
creating: aws/dist/lib/cpython-38-x86_64-linux-gnu/
inflat: aws/dist/lib/cpython-38-x86_64-linux-gnu/soib.cpython-38-x86_64-linux-gnu.so
You can now run: /usr/local/bin/aws --version
mpollu:/environment $ sudo curl --silent --location -o /usr/local/bin/kubectl https://storage.googleapis.com/kubernetes-release/release/v1.19.6/bin/linux/amd64/kubectl
mpollu:/environment $ sudo chmod +x /usr/local/bin/kubectl
mpollu:/environment $ sudo curl --silent --location https://github.com/weaveworks/eksctl/releases/download/v0.38.0/eksctl_linux_amd64.tar.gz | tar xz -C /tmp
mpollu:/environment $ sudo mv -v /tmp/eksctl /usr/local/bin
mpollu:/environment $ curl -sS https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
Downloading https://get.helm.sh/helm-v3.5.4-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
mpollu:/environment $ helm repo add jenkins https://charts.jenkins.io
"jenkins" has been added to your repositories
mpollu:/environment $ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "jenkins" chart repository
Update Complete. Happy Helming!*
```

Step 3: Create an EKS cluster

Attach Admin Role to Cloud9 EC2 instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs
aws-cloud9-jenkinsOnKuber...	i-0cd8f9ad58433ef4	Running	t2.micro	2/2 checks passed	No alarms	us-west-1c	ec2-54-215-102-38.us-...	54.215.102.38	-	-

New KeyPair created and added IAM Role:

Name	Fingerprint	ID
jenkinsKeyPair	e0:97:22:d2:ea:7c:70:bc:8e:d9:66:93:d...	key-0c6f1625467496fcc

Step 4: Install Jenkins server on an On Demand nodegroup

Running Jenkins from UI:



Welcome to Jenkins!

Sign in

☐ Keep me signed in

Jenking running on CLI:

intent=jenkins-server indicating that the Jenkins is running on *jenkins-server-ng* nodegroup.

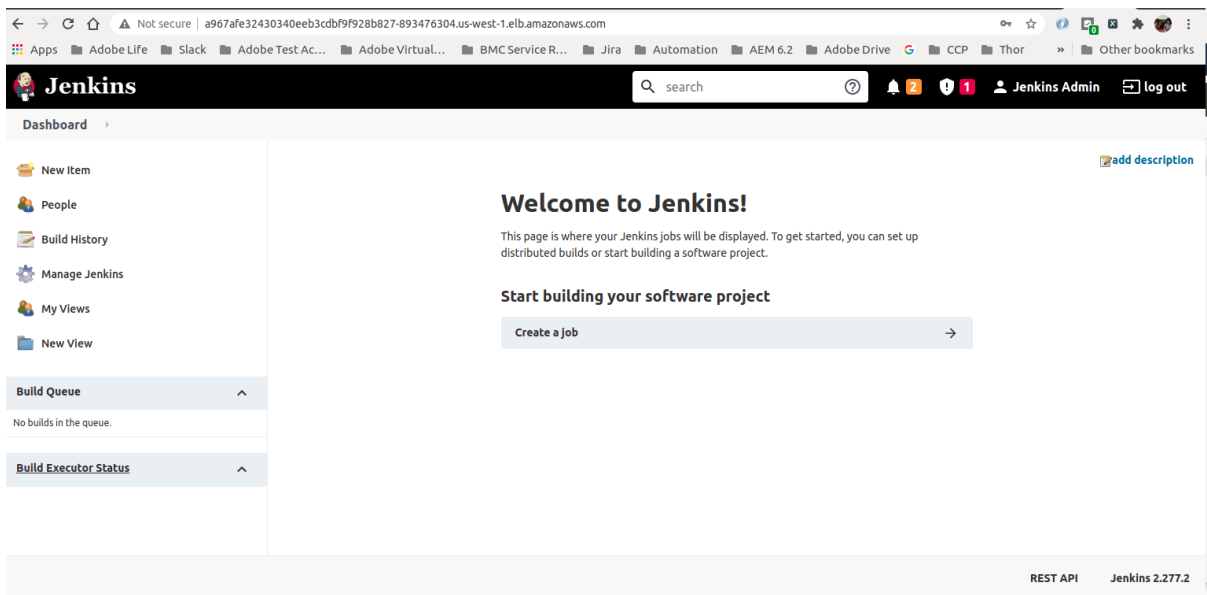
```
mpollu:~/environment $ kubectl get nodes -o wide --show-labels $(kubectl get pod cicd-jenkins-0 -o jsonpath='{.spec.nodeName}') | grep intent
ip-192-168-49-241.us-west-1.compute.internal Ready <none> 8m34s v1.19.6-eks-49a6c0 192.168.49.241 54.67.58.16 Amazon Linux 2 5.4.105-48.177.anzn2.x86_64 docker://19.3.13 alpha.eksctl
.io/cluster-name=jenkins-cluster,alpha.eksctl.io/nodegroup-name=jenkins-server-ng,beta.kubernetes.io/arch=amd64,beta.kubernetes.io/instance-type=m5.large,beta.kubernetes.io/os=linux,eks.amazonaws.com/capacityType=ON_DEMAND,eks.amazonaws.com/nodegroup-image=ami-021a6978cb08d37f5,eks.amazonaws.com/nodegroup=jenkins-server-ng,eks.amazonaws.com/sourceLaunchTemplateId=lt-005a9a76064e893aa,eks.amazonaws.com/sourceLaunchTemplateVersion=1,failure-domain.beta.kubernetes.io/region=us-west-1,failure-domain.beta.kubernetes.io/zone=us-west-1b,intent=jenkins-server,kubernetes.io/arch=amd64,kubernetes.io/hostname=ip-192-168-49-241.us-west-1.compute.internal,kubernetes.io/os=linux,lifecycle=OnDemand,node.kubernetes.io/instance-type=m5.large,topology.kubernetes.io/region=us-west-1,topology.kubernetes.io/zone=us-west-1b
mpollu:~/environment $
```

bash - 1p-172-31-19-115. x Immediate x +

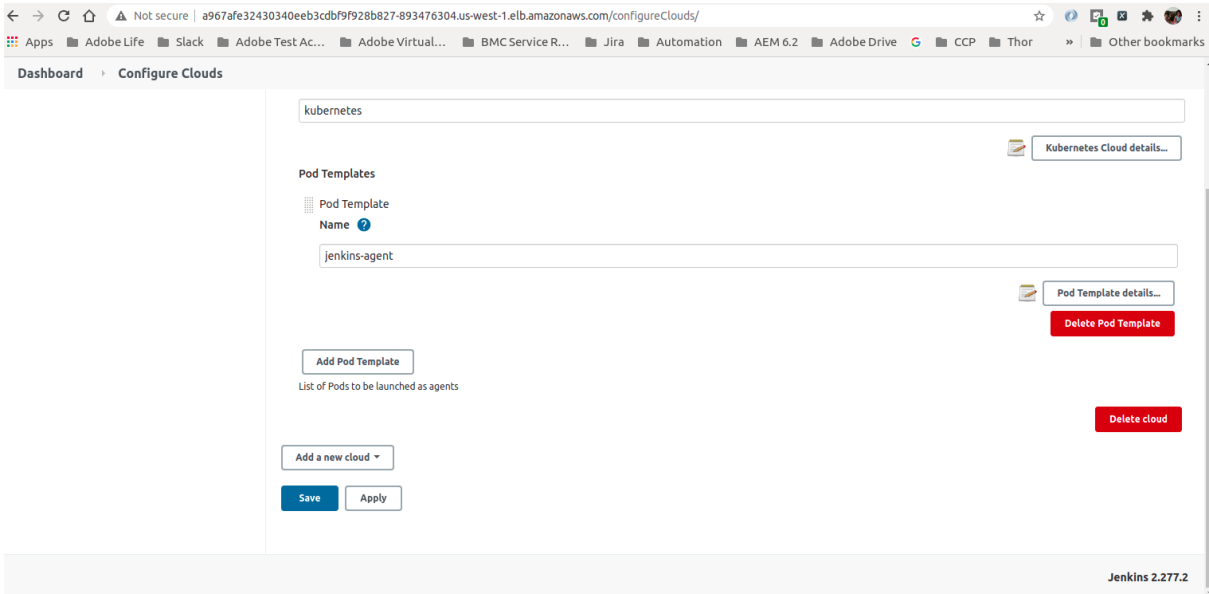
[detached (from session cloud9_terminal_27)]

Step 5: Install Jenkins agents on a EC2 Spot nodegroup

Login into Jenkins as Admin:

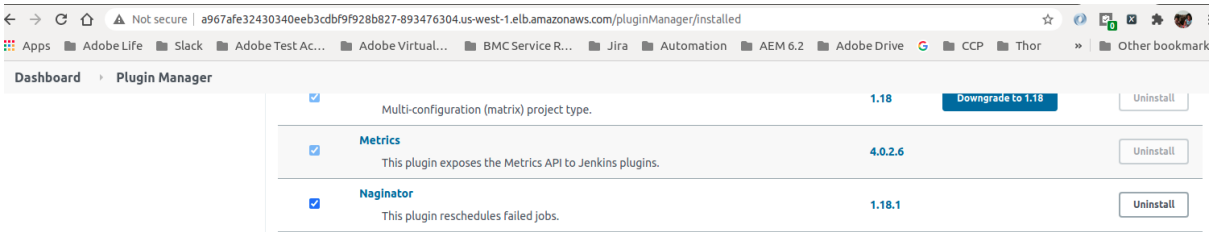


Pod Template:



Step 6: Setup Jenkins plugins for retry

Naginator:



Step 7: Setup a sample Continuous Integration (CI) pipeline in Jenkins

Free-style Build-Sample Project created:

Dashboard > Build-Sample >

Back to Dashboard

Status

Changes

Workspace

Build Now

Configure

Delete Project

Rename

Build History trend ^

find

Atom feed for all Atom feed for failures

Project Build-Sample

[add description](#)

[Disable Project](#)

Workspace

Recent Changes

Permalinks

Step 8: Run the sample CI pipeline in Jenkins

Build Now Build-Sample:

Dashboard >

New Item

People

Build History

Manage Jenkins

My Views

Lockable Resources

New View

Build Queue

No builds in the queue.

Build Executor Status

default-n52qq

1 Build-Sample #1

[add description](#)

S	W	Name	Last Success	Last Failure	Last Duration
		Build-Sample	N/A	N/A	N/A

Icon: S M L

[Legend](#) [Atom feed for all](#) [Atom feed for failures](#) [Atom feed for just latest builds](#)

Jenkins agent in Cloud9 IDE:

```
mpbollu:~/environment $ kubectl get pod -w
NAME          READY   STATUS    RESTARTS   AGE
cicd-jenkins-0 2/2     Running   0           38m
default-n52qq 1/1     Running   0           2m14s
^Cmpbollu:~/environment $
```

Build-Sample Console Output:

Dashboard › Build-Sample › #1

```
name: "jnlp"
resources:
  limits:
    memory: "512Mi"
    cpu: "512m"
  requests:
    memory: "512Mi"
    cpu: "512m"
tty: false
volumeMounts:
- mountPath: "/home/jenkins"
  name: "workspace-volume"
  readOnly: false
workingDir: "/home/jenkins"
nodeSelector:
  kubernetes.io/os: "linux"
restartPolicy: "Never"
serviceAccountName: "default"
volumes:
- emptyDir:
    medium: ""
  name: "workspace-volume"
```

```
Building remotely on default-n52qq (cid-jenkins-agent) in workspace /home/jenkins/workspace/Build-Sample
[Build-Sample] $ /bin/sh -xe /tmp/jenkins6029157892916056446.sh
+ sleep 3m
+ echo Job completed successfully
Job completed successfully
Finished: SUCCESS
```

Step 9: Cleanup

The screenshot shows a terminal window with the following output:

```

2021-04-16 19:01:26 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:27 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-xvccc, kube-system/kube-proxy-7u7zh
2021-04-16 19:01:27 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:27 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-xvccc, kube-system/kube-proxy-7u7zh
2021-04-16 19:01:28 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:28 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-xvccc, kube-system/kube-proxy-7u7zh
2021-04-16 19:01:29 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:29 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-xvccc, kube-system/kube-proxy-7u7zh
2021-04-16 19:01:29 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:30 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-xvccc, kube-system/kube-proxy-7u7zh
2021-04-16 19:01:30 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:31 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-xvccc, kube-system/kube-proxy-7u7zh
2021-04-16 19:01:31 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:31 [I] pod eviction error ("pods"/"coredns-974b5cfd-wannh": "not found") on node ip-192-168-49-241.us-west-1.compute.internal
2021-04-16 19:01:36 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-xvccc, kube-system/kube-proxy-7u7zh
2021-04-16 19:01:36 [I] Ignoring DaemonSet-managed Pods: kube-system/aws-node-gp4kt, kube-system/kube-proxy-rf8g
2021-04-16 19:01:36 [✓] drained all nodes (ip-192-168-49-241.us-west-1.compute.internal ip-192-168-93-140.us-west-1.compute.internal)
2021-04-16 19:01:36 [I] will delete 1 nodegroups from cluster "jenkins-cluster"
2021-04-16 19:01:36 [I] 1 task: { delete nodegroup "jenkins-cluster" [async] }
2021-04-16 19:01:37 [I] will delete stack "eksctl-jenkins-cluster-nodegroup-jenkins-server-ng"
2021-04-16 19:01:37 [I] will delete 0 nodegroups from auth configmap in cluster "jenkins-cluster"
2021-04-16 19:01:37 [✓] deleted 1 nodegroup(s) from cluster "jenkins-cluster"
npboller:~$ kubectl delete cluster --name=jenkins-cluster
2021-04-16 19:01:55 [I] eksctl version 0.38.0
2021-04-16 19:01:55 [I] using region us-west-2
2021-04-16 19:01:55 [I] deleting eks cluster "jenkins-cluster"
error: cluster "jenkins-cluster" does not exist
npboller:~$

```

