Continuous Deployment with Github, GitHub Actions, and AWS CodeDeploy

Continuous Deployment

Continuous Deployment is a software development practice in which every code change goes through the entire pipeline and is put into production, automatically, resulting in many production deployments every day.

Continuous Deployment Workflow

Web Application

1. Developer commits code changes to GitHub repository.
2. GitHub Actions will trigger a new build on pull request merge.
3. GitHub Actions will run the build steps from the GitHub Actions workflow. Build steps should do the following:
   1. Run the unit test.
   2. Validate Packer Template
   3. Build Application Artifact(s)
   4. Build AMI (Note: New AMI image is not used in the CI/CD pipeline in this assignment)
      1. Upgrade OS packages
      2. Install dependencies (python, node.js, etc.)
      3. Install application dependencies (pip install for python)
      4. Copy application artifact from step 3.
   5. Zip the artifacts and upload the zip archive to the CodeDeploy's S3 bucket.
   6. Trigger a new CodeDeploy deployment with the latest revision of your artifact.

IAM Setup

Replace * with appropriate ARN name to create security policies.

CodeDeploy-EC2-S3 Policy for the Server (EC2)

The **CodeDeploy-EC2-S3** policy allows EC2 instances to read data from S3 buckets. This policy is required for EC2 instances to download the latest application revision.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Effect": "Allow",
            "Resource": [
                "",
                ""
            ]
        }
    ]
}
```

GH-Upload-To-S3 Policy for GitHub Actions to Upload to AWS S3

The GH-Upload-To-S3 policy allows GitHub Actions to upload artifacts from the latest successful build to the dedicated S3 bucket used by CodeDeploy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                ""
            ]
        }
    ]
}
```

GH-Code-Deploy Policy for GitHub Actions to Call CodeDeploy

The **GH-Code-Deploy** policy allows GitHub Actions to call CodeDeploy APIs to initiate application deployment on EC2 instances.

Note:

1. Replace `AWS_REGION` with AWS region.
2. Replace `AWS_ACCOUNT_ID` with your account id.
3. Replace `CODE_DEPLOY_APPLICATION_NAME` with your CodeDeploy Application Name. We haven't created CodeDeploy Application yet. So note down the application name for later steps.

```
{
    "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codedeploy:RegisterApplicationRevision",
        "codedeploy:GetApplicationRevision"
      ],
      "Resource": [
        "arn:aws:codedeploy:AWS_REGION:AWS_ACCOUNT_ID:application:CODE_DEPLOY_APPLICATION_NAME"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "codedeploy:CreateDeployment",
        "codedeploy:GetDeployment"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "codedeploy:GetDeploymentConfig"
      ],
      "Resource": [
        "arn:aws:codedeploy:AWS_REGION:AWS_ACCOUNT_ID:deploymentconfig:CodeDeployDefault.OneAtATime",
        "arn:aws:codedeploy:AWS_REGION:AWS_ACCOUNT_ID:deploymentconfig:CodeDeployDefault.HalfAtATime",
        "arn:aws:codedeploy:AWS_REGION:AWS_ACCOUNT_ID:deploymentconfig:CodeDeployDefault.AllAtOnce"
      ]
    }
  ]
}
```

GitHub Actions IAM User Policies

Attach following IAM policies to the `ghactions` IAM user:

1. GH-Upload-To-S3
2. GH-Code-Deploy
3. **gh-ec2-ami** **(https://www.packer.io/docs/builders/amazon.html#iam-task-or-instance-role)**

Create CodeDeployEC2ServiceRole IAM Role for EC2 Instance(s)

Create a new role `CodeDeployEC2ServiceRole` for EC2 instances that will be used to host your web application.

Identity and Access Management (IAM)

Dashboard

Groups

Users

Roles

Policies

Identity providers

Account settings

Credential report

Encryption keys

🔍 Search IAM

▼ AWS Organizations

Organization activity

Service control policies (SCPs)

Create role	Delete role

🔍 Search

Role name ▼	Description	Trusted entities

Create role

Select type of trusted entity

AWS service	Another AWS account	Web identity	SAML 2.0 federation
AWS service EC2, Lambda and others	**Another AWS account** Belonging to you or 3rd party	**Web identity** Cognito or any OpenID provider	**SAML 2.0 federation** Your corporate directory

Allows AWS services to perform actions on your behalf. Learn more

Choose the service that will use this role

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

API Gateway	Comprehend	EMR	Kinesis	S3
AWS Backup	Config	ElastiCache	Lambda	SMS
AWS Support	Connect	Elastic Beanstalk	Lex	SNS
Amplify	DMS	Elastic Container Service	License Manager	SWF
AppSync	Data Lifecycle Manager	Elastic Transcoder	Machine Learning	SageMaker
Application Auto Scaling	Data Pipeline	ElasticLoadBalancing	Macie	Security Hub
Application Discovery	DataSync	Forecast	MediaConvert	Service Catalog

* Required

Cancel   **Next: Permissions**

Create role

▾ Attach permissions policies

Choose one or more policies to attach to your new role.

[Create policy]   [⟳]

Filter policies ⌄   [🔍 Search   ]   **Showing 16 results**

Reset filters

	Used as	Description
POLICY TYPE		
☑ Customer managed (16) ⬅	None	Provides full access to AWS services an...
☐ AWS managed (524)	None	
☐ AWS managed - job function (10)	None	
	Permissions policy (2)	
POLICY USE		
☐ Used for permissions (20)	None	
☐ Used for boundary (0)	None	
☐ Not used (530)	None	

▸ Set permissions boundary

* Required   Cancel   [Previous]   [**Next: Tags**]

Create role

▼ Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy

Filter policies ∨	🔍 Search		Showing 16 results

		Policy name ▼	Used as	Description
☐	▶	AdministratorAccess-201702021852	*None*	Provides full access to AWS services an...
☐	▶	circleci-ec2-ami	*None*	
☐	▶	CircleCI-Upload-To-S3	*None*	
☐	▶	CirlceCI-Code-Deploy	*None*	
☑	▶	CodeDeploy-EC2-S3	Permissions policy (2)	
☐	▶	CodeDeployDemo-EC2-Permissions	*None*	
☐	▶	EC2-S3-Webapp	*None*	
☐	▶	fall2018.csye6225.cloud	*None*	

▸ Set permissions boundary

* Required

Cancel   Previous   **Next: Tags**

Create role

Add tags (optional)

IAM tags are key-value pairs you can add to your role. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this role. Learn more

Key	Value (optional)	Remove
Add new key		

You can add 50 more tags.

Cancel   Previous   **Next: Review**

Create role

Review

Provide the required information below and review this role before you create it.

Role name*

CodeDeployEC2ServiceRole

Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.

Role description

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Trusted entities   AWS service: ec2.amazonaws.com

Policies   CodeDeploy-EC2-S3 ☑

Permissions boundary   Permissions boundary is not set

No tags were added.

* Required   Cancel   Previous   Create role

Create CodeDeployServiceRole IAM Role for CodeDeploy

Create a new role `CodeDeployServiceRole` for EC2 instances that will be used to host your web application.

Create role

Select type of trusted entity

AWS service EC2, Lambda and others	**Another AWS account** Belonging to you or 3rd party	**Web identity** Cognito or any OpenID provider	**SAML 2.0 federation** Your corporate directory

Allows AWS services to perform actions on your behalf. Learn more

Choose the service that will use this role

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

API Gateway	Comprehend	EMR	Kinesis	S3
AWS Backup	Config	ElastiCache	Lambda	SMS
AWS Support	Connect	Elastic Beanstalk	Lex	SNS
Amplify	DMS	Elastic Container Service	License Manager	SWF
AppSync	Data Lifecycle Manager	Elastic Transcoder	Machine Learning	SageMaker
Application Auto Scaling	Data Pipeline	ElasticLoadBalancing	Macie	Security Hub
Application Discovery Service	DataSync	Forecast	MediaConvert	Service Catalog
Batch	DeepLens	Glue	OpsWorks	Step Functions
CloudFormation	Directory Service	Greengrass	Personalize	Storage Gateway
CloudHSM	DynamoDB	GuardDuty	RAM	Transfer
CloudTrail	EC2	Inspector	RDS	Trusted Advisor
CloudWatch Application Insights	EC2 - Fleet	IoT	Redshift	VPC
CloudWatch Events	EC2 Auto Scaling	IoT Things Graph	Rekognition	WorkLink
CodeBuild	EKS	KMS	RoboMaker	WorkMail
CodeDeploy				

Select your use case

CodeDeploy
Allows CodeDeploy to call AWS services such as Auto Scaling on your behalf.

CodeDeploy - ECS
Allows CodeDeploy to read S3 objects, invoke Lambda functions, publish to SNS topics, and update ECS services on your behalf.

CodeDeploy for Lambda
Allows CodeDeploy to route traffic to a new version of an AWS Lambda function version on your behalf.

* Required   Cancel   **Next: Permissions**

Create role

▾ **Attached permissions policies**

The type of role that you selected requires the following policy.

	Policy name ▾	Used as	Description
▸	📦 AWSCodeDeployRole	Permissions policy (1)	Provides CodeDeploy service access to expan…

Filter policies ⌄   🔍 Search   Showing 1 result

▸ Set permissions boundary

* Required

Cancel   Previous   **Next: Tags**

Create role

Add tags (optional)

IAM tags are key-value pairs you can add to your role. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this role. Learn more

Key	Value (optional)	Remove
Add new key		

You can add 50 more tags.

Cancel   Previous   Next: Review

Create role

Review

Provide the required information below and review this role before you create it.

Role name* | CodeDeployServiceRole

Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.

Role description | Allows CodeDeploy to call AWS services such as Auto Scaling on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Trusted entities | AWS service: codedeploy.amazonaws.com

Policies | 📦 AWSCodeDeployRole ↗

Permissions boundary | Permissions boundary is not set

No tags were added.

* Required   Cancel   Previous   **Create role**

Create S3 bucket for CodeDeploy

1. Create a S3 bucket in same region as your EC2 instance.
2. Bucket name should be `codedeploy.yourdomain.tld` where `yourdomain.tld` should be replaced with your domain name.

Create EC2 Instance to Host Application

Create one or more EC2 instance to host your application. This EC2 instance must have the IAM Role **CodeDeployEC2ServiceRole**. Tag the instance with `KEY` and `VALUE` of your choice. You will need the `KEY` and `VALUE` later when creating CodeDeploy application.

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instan role to the instance, and more.

Number of instances ⓘ	1	Launch into Auto Scaling Group ⓘ
Purchasing option ⓘ	☐ Request Spot instances	
Network ⓘ	vpc-cdf372ab ⬍	↻ Create ne
Subnet ⓘ	subnet-e12853cc | us-east-1a ⬍ 250 IP Addresses available	Create ne
Auto-assign Public IP ⓘ	Use subnet setting (Enable) ⬍	
IAM role ⓘ	CodeDeployEC2ServiceRole ⬍	↻ Create ne
Shutdown behavior ⓘ	Stop ⬍	
Enable termination protection ⓘ	☐ Protect against accidental termination	
Monitoring ⓘ	☐ Enable CloudWatch detailed monitoring Additional charges apply.	
Tenancy ⓘ	Shared - Run a shared hardware instance ⬍ Additional charges will apply for dedicated tenancy.	

Install CodeDeploy Agent

Install (https://docs.aws.amazon.com/codedeploy/latest/userguide/codedeploy-agent-operations-install.html) CodeDeploy Agent from the region your EC2 instance is running.

CodeDeploy App Spec

Create AWS CodeDeploy App Spec file. The AppSpec file is used to manage each deployment as a series of lifecycle events. The **appspec.yml** file should be at the root of your repository. See App Spec documentation **here** (http://docs.aws.amazon.com/codedeploy/latest/userguide/writing-app-spec.html).

```
version: 0.0
os: linux

files:
  - source: ./build/libs/ROOT.war
    destination: /var/lib/tomcat7/webapps

hooks:
  AfterInstall:
  - location: ./restartTomcat.sh
    timeout: 180
    runas: centos
```

Create CodeDeploy Application

1. Application Name - **csye6225-webapp**
2. Compute Platform - **EC2/On-premises**

Create CodeDeploy Deployment Group

1. Deployment group name - **csye6225-webapp-deployment**
2. Service role - **CodeDeployServiceRole**
3. Deployment type - **In-place**
4. Environment Configuration - **Amazon EC2 Instances**
    1. Provide the tag group key and values.
5. Deployment settings - **CodeDeployDefault.AllAtOnce**
6. Load Balancer - **disabled**
7. Rollback - **Rollback when a deployment fails**
8. Everything else can be left to default values.