**Setup AWS CodeDeploy for you application :**

1. **Create AWS IAM roles**

The first step towards setting up codedeploy is to setup two IAM roles. One for codedeploy to talk to EC2 instances and other for EC2 instance to access s3.

Let’s create the first role for codedeploy

* Go to IAM -> roles -> create new Role
* Give a name for the role “code-deploy” and Goto Next Step
* In Role Type -> select -> Amazon EC2 below AWS service Roles
* In Attach Policy select -> AWSCodeDeployRole
* Create Role
* Edit the Trust relationship and update the content to the following one

{

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

"Statement": [

{

"Sid": "",

"Effect": "Allow",

"Principal": {

"Service": "codedeploy.amazonaws.com"

},

"Action": "sts:AssumeRole"

}

]

}

Now we can create the second role to give EC2 instance to access s3.

* Go to IAM -> roles -> create new Role
* Give a name for the role “CodeDeploy-EC2” and Goto Next Step
* In Role Type -> select -> Amazon EC2 below AWS service Roles
* In Attach Policy select -> AmazonS3ReadOnlyAccess
* Create Role

**2. Create AWS instance**

Next step is to Goto EC2 Instances and launch a new instance. While creating an instance you can choose any instance type but make sure to choose CodeDeploy-EC2 as IAM role in Configure instance.

In Add tags section add a tag with Name as key and Value as codedeploy-demo (You can name the instance as per your need)

**3. Install code deploy**

Once the instance is booted up we can install the code deploy agent that instance

wget https://aws-codedeploy-us-east-1.s3.amazonaws.com/latest/install

chmod +x ./install

sudo ./install auto

sudo service codedeploy-agent status

**4. Prepare the application**

Next is to add the **appspec.yml** file to the application, appspec.yml file will have information on what to install on to instances and what lifecycle events to run.

The format for appspec.yml file i

version: 0.0

os: linux

files:

- source: /index.html

destination: /var/www/html/

hooks:

BeforeInstall:

- location: deploy/before\_install

timeout: 300

runas: root

AfterInstall:

- location: deploy/restart\_server

timeout: 300

runas: root

The beforeInstall hook will be

*# deploy/before\_install*

*#!/bin/bash*

sudo rm -f /var/www/html/index.html

and AfterInstall hook

*# deploy/after\_install*

*#!/bin/bash*

sudo service httpd restart

# 5. Setup CodeDeploy on AWS Console:

* Now its time to create a deployment. On aws console navigate to AWS CodeDeploy and create a new application.
* Fill in the name of application and instances using the tag and value
* choose the deployment configuration,
* Now add the the IAM role, which we create before as the service role
* Once the application is created, we can deploy new revision.
* For the first time, the codedeploy app will ask to connect to Github.
* Once the github connection is setup, You can provide the repo name along with github username,
* Now click on Deploy Now, which will deploy to all the instance configured for the deployment application.

**6. Setup Autodeploy from Github**

First we will create a IAM policy which give access to register and create a new deployment, also to create new revision for a deployment group.

Choose Create Your own policy from **Create Policy** and give some name codedeploy-github-access and for policy document use the below template.

{

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

"Statement": [

{

"Effect": "Allow",

"Action": "codedeploy:GetDeploymentConfig",

"Resource": "arn:aws:codedeploy:ACCOUNT\_REGION:ACCOUNT\_ID:deploymentconfig:\*"

},

{

"Effect": "Allow",

"Action": "codedeploy:RegisterApplicationRevision",

"Resource": "arn:aws:codedeploy:ACCOUNT\_REGION:ACCOUNT\_ID:application:APPLICATION\_NAME"

},

{

"Effect": "Allow",

"Action": "codedeploy:GetApplicationRevision",

"Resource": "arn:aws:codedeploy:ACCOUNT\_REGION:ACCOUNT\_ID:application:APPLICATION\_NAME"

},

{

"Effect": "Allow",

"Action": "codedeploy:CreateDeployment",

"Resource":

"arn:aws:codedeploy:ACCOUNT\_REGION:ACCOUNT\_ID:deploymentgroup:APPLICATION\_NAME/DEPLOYMENT\_GROUP"

}

]

}

Please make sure you replace the ACCOUNT\_REGION, ACCOUNT\_ID, APPLICATION\_NAME and DEPLOYMENT\_GROUP according to your application.

### 7. Github Integration

To invoke AWS codedeploy from github, we need to configure two integrations on Github. Before we configure we need to [generate new token](https://github.com/settings/tokens/new) with access to repo status and repo\_deployments.

##### 1) AWS CodeDeploy

We can add AWS CodeDeploy integration by navigating to Project Settings -> Integrations and services. Then from the Add service dropdown choose AWS CodeDeploy

##### 2) GitHub Auto-Deployment

From the same Project Settings -> Integrations and services, this time we can choose GitHub Auto-Deployment from the Add service dropdown.

Now when we edit file and commit on master branch or merge any Pull request a new deployment will be created on AWS CodeDeploy.

**8. Jenkins Integration**