Code Deploy

QUALITY THOUGHT

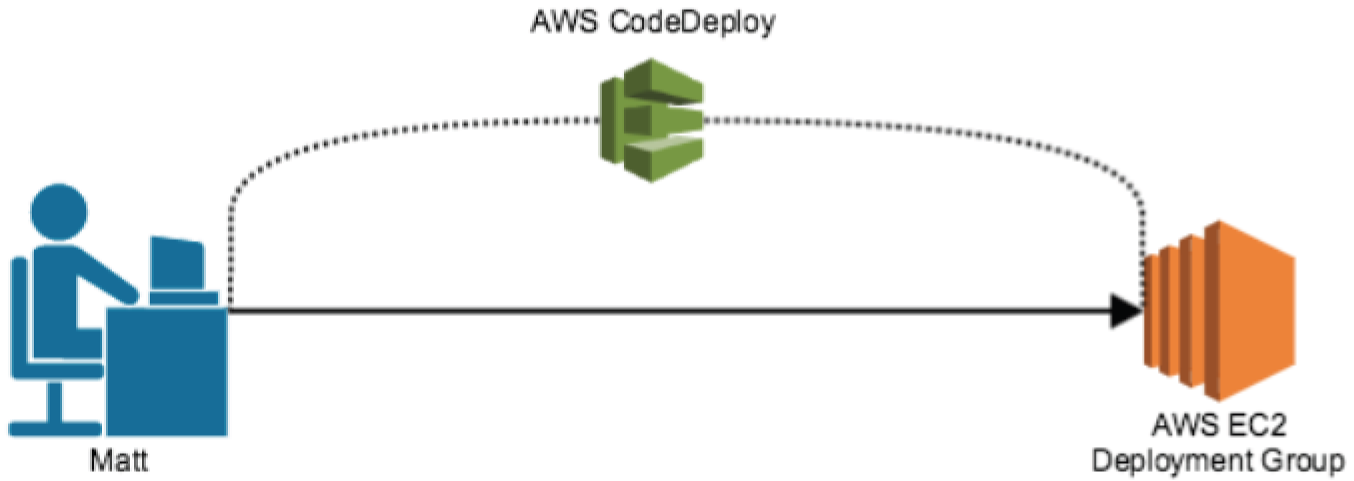
INFO SYSTEMS (INDIA) PVT. LTD

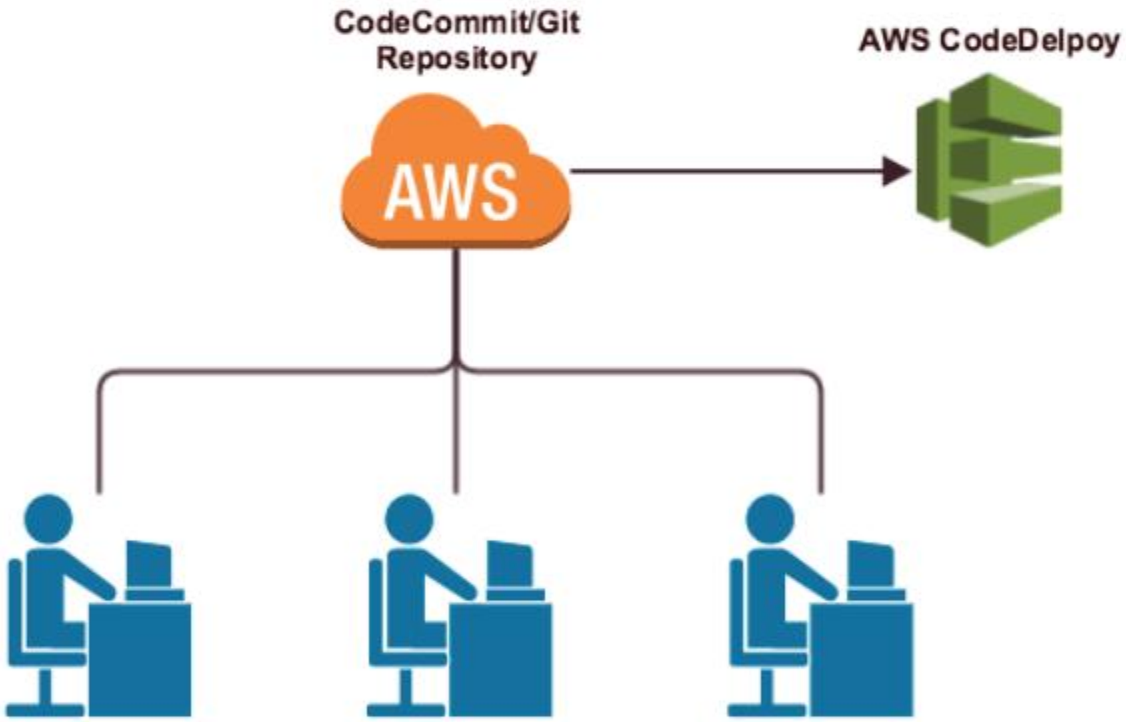
What is CodeDeploy?

- CodeDeploy is an AWS service for automating the deployment process of your applications, from Git-based version control systems or S3 buckets to Amazon EC2 instances, on-premise instances, or both.

- ***Automation:***
  - You can easily automate your code deployment to development, test, and production environments.
- ***Scale:***
  - Deploy your code to one or thousands of instances at once.
- ***Reduced Downtime:***
  - Rolling updates allow you to decrease downtime by allowing you to track application heath and stop/rollback deployments if there are errors.
- ***Control:***
  - Easily keep track of your deployments by receiving reports that list when and where each of your application revisions is deployed.
- ***Platform-agnostic:***
  - CodeDeploy is built to work with any application.

CodeDeploy: Workflow




CodeCommit/Git Repository
AWS
AWS CodeDelpoy

Setup and Configuration

CodeDeploy: Setup & Configuration

1) Provision an IAM user with a custom CodeDeploy Policy
   - *Gives a non-admin user the rights to manage all the elements needed to use CodeDeploy.*

2) Create an Instance Profile
   - *This allows you to launch EC2 instances that are configured for use with CodeDeploy.*

3) Create a Service Role
   - *This will allow CodeDeploy to communicate and interact with other AWS Services.*

4) Install the AWS Command Line Interface (CLI)
   - *For Windows: (2) Windows: Git & AWS CLI Installation*
   - *For OSX/Linux: (7) OSX/Linux: AWS CLI Installation*

******These videos are located under the CodeCommit section of this course***

Steps

Create a custom own policy with following content

https://s3-us-west-2.amazonaws.com/qt-test-s3-1/DevOps/codedeploy-codedeploycustomuser-policy_1468694436.txt

Attach policy to user

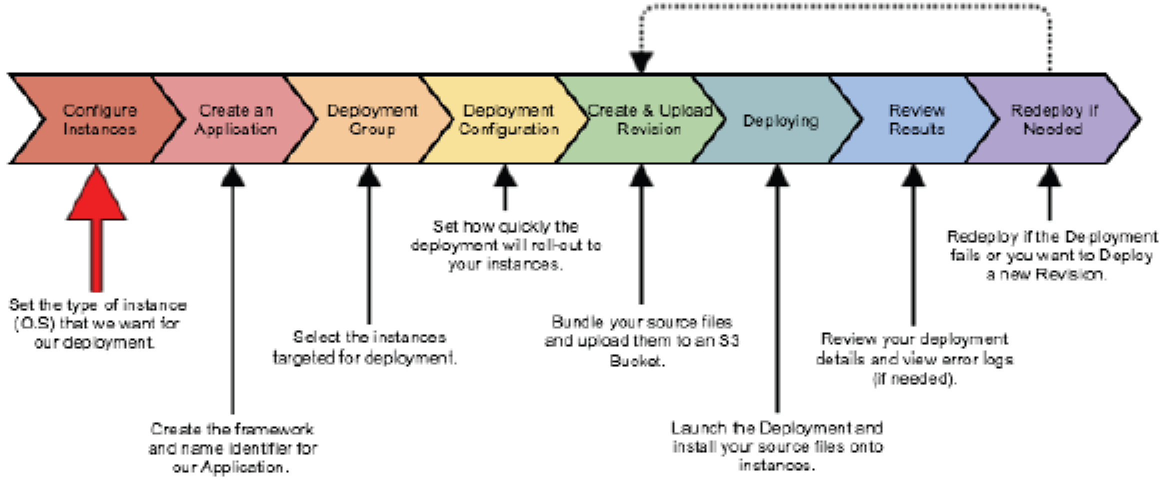
Create an instance profile by creating own policy @ https://s3-us-west-2.amazonaws.com/qt-test-s3-1/DevOps/codedeploy-codedeploydemo-ec2-permissions-instance-role-policy-s3-access_1468694804.txt

Create a Role to apply above created policy

Steps contd..

Create a Role for CodeDeploy and name it as CodeDeployServiceRole

CodeDeploy Workflow:


Configure Instances
Create an Application
Deployment Group
Deployment Configuration
Create & Upload Revision
Deploying
Review Results
Redeploy if Needed
Set the type of instance (O.S) that we want for our deployment.
Create the framework and name identifier for our Application.
Select the instances targeted for deployment.
Set how quickly the deployment will roll-out to your instances.
Bundle your source files and upload them to an S3 Bucket.
Launch the Deployment and install your source files onto instances.
Review your deployment details and view error logs (if needed).
Redeploy if the Deployment fails or you want to Deploy a new Revision.

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

Setting up Deployment Instances

- What type of instances does CodeDeploy support?
  - *Amazon Linux*
  - *Ubuntu Server*
  - *Red Hat Enterprise Linux (RHEL)*
  - *Windows Server*

- What methods can we use to launch & configure instances?
  - EC2 Instances:
    - *Manually create & configure in the AWS Console or CLI*
    - *CloudFormation template*
  - On-Premise:

- For the purpose of this lesson, we are going to review how to manually set up and configure an EC2 instance for CodeDeploy via the AWS console.

Configuration steps

1) Launch a new Amazon Linux AMI

2) Select appropriate instance Type

3) Set IAM role to the Instance Profile we created in the Setup & Configuration lesson

Configuration Steps CONTD...

4) Open ‘Advanced Details’ and add the following bash

Script:

```
#!/bin/bash

yum -y update

yum install -y ruby

yum install -y aws-cli

cd /home/ec2-user

aws s3 cp s3://bucket-name/latest/install . --region region-name

chmod +x ./install

./install auto
```

CONFIGURATION STEPS CONTD..

5) Add storage

6) Add a ‘Name’ Tag to the instance

7) Configure a Security Group

8) Review and Launch

9) Select a key pair

10) Check to see if the AWS CodeDeploy agent has been successfully installed

AWS CodeDeploy Agent:

- The AWS CodeDeploy agent is a custom software package that must be installed on all instances that will be part of a deployment group. The agent specifies many of the settings that are needed for the instance to interact with CodeDepoy – like directory paths, log files, and deployment polling time intervals. The agent is also customizable, so you can alter it to fit your deployment needs.

- For Amazon, Ubuntu, and RHEL instances:
  - *Name:* *codedeployagent.yml*
  - *Location:* */etc/codedeploy-agent/conf*

- For Window Servers:
  - *Name:* *conf.yml*
  - *Location:* *C:\ProgramData\Amazon\CodeDeploy*

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

AWS CodeDeploy Agent:

Linux server

S3 Bucket Name	Region Name
aws-codedeploy-us-east-1	us-east-1
aws-codedeploy-us-west-1	us-west-1
aws-codedeploy-us-west-2	us-west-2
aws-codedeploy-eu-west-1	eu-west-1
aws-codedeploy-eu-central-1	eu-central-1
aws-codedeploy-ap-northeast-1	ap-northeast-1
aws-codedeploy-ap-northeast-2	ap-northeast-2
aws-codedeploy-ap-southeast-1	ap-southeast-1
aws-codedeploy-ap-southeast-2	ap-southeast-2
aws-codedeploy-sa-east-1	sa-east-1

Windows server

S3 Bucket Name
aws-codedeploy-us-east-1
aws-codedeploy-us-west-1
aws-codedeploy-us-west-2
aws-codedeploy-eu-west-1
aws-codedeploy-eu-central-1
aws-codedeploy-ap-northeast-1
aws-codedeploy-ap-northeast-2
aws-codedeploy-ap-southeast-1
aws-codedeploy-ap-southeast-2
aws-codedeploy-sa-east-1

CodeDeploy Agent Status:

- Command:

  sudo service codedeploy-agent status

- Good Response:

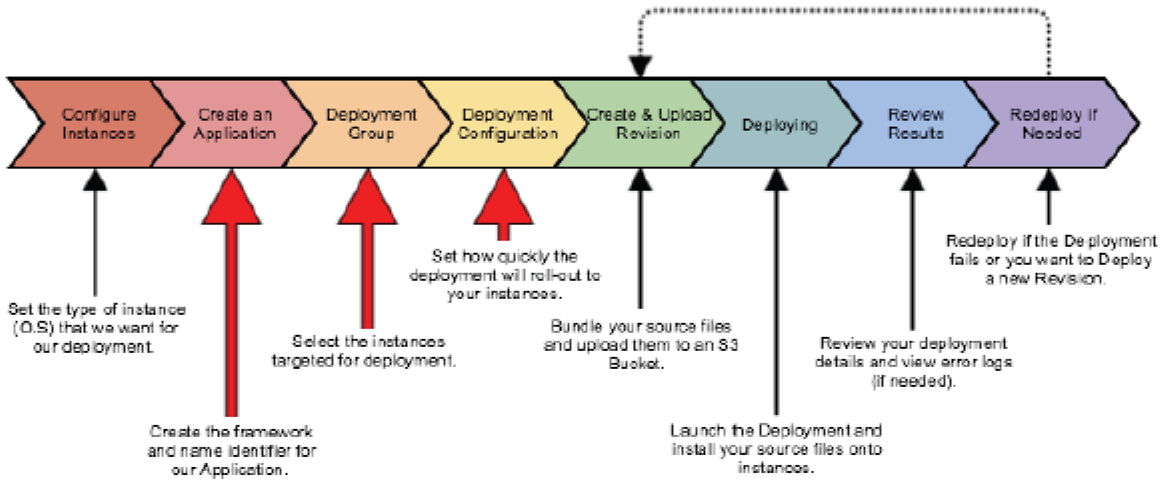
  The AWS CodeDeploy agent is running.

- Error Responses:

  Error: ***No AWS CodeDeploy agent running***
  Fix: **sudo service codedeploy-agent start**

  Error: ***codedeploy-agent: unrecognized service***
  Fix: ***Launch a new instance and double check the bash script for errors, OR double check the permissions policy attached to the Instance Profile (make sure it allows access to S3)***

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD


Configure Instances
Create an Application
Deployment Group
Deployment Configuration
Create & Upload Revision
Deploying
Review Results
Redeploy if Needed
Set the type of instance (O.S) that we want for our deployment.
Create the framework and name identifier for our Application.
Select the instances targeted for deployment.
Set how quickly the deployment will roll-out to your instances.
Bundle your source files and upload them to an S3 Bucket.
Launch the Deployment and install your source files onto instances.
Review your deployment details and view error logs (if needed).
Redeploy if the Deployment fails or you want to Deploy a new Revision.

Understanding the Terminology:

- CodeDeploy **Application**:
  - *The application is simply a name identifier used to reference your deployment settings*
- CodeDeploy **Deployment Group**:
  - *The deployment group is the instance (or instances) you want to target and deploy your code to*
- CodeDeploy **Deployment Configuration**:
  - *Select how many instances in your deployment group your code will deploy to at any given time*

Managing an Application from the AWS Console:

Creating an Application, Deployment Group & Configuration:

1) *Navigate to CodeDeploy*
2) *Choose Create a New Application*
3) *Give the Application a name*

The next set of steps is to set up the <u>Deployment Group</u> within the Application

4) *Give the deployment group a name*
5) *Select the tag type, Key & value for your instance*

The next step is to setup the <u>Deployment Configuration</u> within the Application

6) *Select deployment configuration*
   - *OneAtATime*
   - *AllAtOnce*
   - *HalfAtATime*

Optional

7) *Create a trigger*

Set permissions

8) *Select the CodeDeploy Service role we create*

9) *Finalize & create the Application*

Managing an Application from the AWS CLI:

Creating an Application, Deployment Group & Configuration:

Base CLI Command:

```
aws deploy
```

1) *Create a new application*

```
aws deploy create-application --application-name <NAME>
```

2) *Create the deployment group, configuration & other options*

In one command we will specify:

- *Deployment group (tag, key, value)*
- *Deployment Configuration (AllAtOnce, OneAtATime, HalfAtATime)*
- *Trigger (optional)*
- *Service role (permissions)*

```
aws deploy create-deployment-group --application-name <NAME> --
deployment-group-name <NAME> --ec2-tag-filters
Key=Name,Value=<EC2VALUE>,Type=KEY_AND_VALUE --
deployment-config-name CodeDeployDefault.<SELECTOPTION> --
service-role-arn <SERVICE-ROLE_ARN>
```

Edit & Delete a CodeDeploy Application:

- Changing the Application Name:
  - *AWS Console: N/A*
  - *AWS CLI Commands:*

    aws deploy list-applications *(list all our applications)*

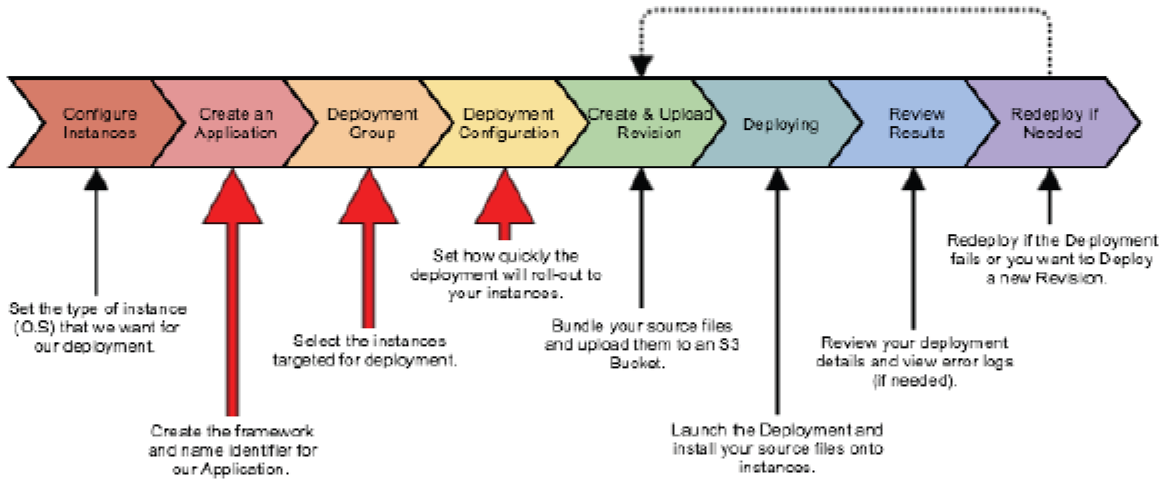
    aws deploy update-application --application-name <NAME> --new-application-name <NEWNAME>
    (rename our application)

- Deleting an Application:
  - AWS CLI Commands:

    aws deploy delete-application --application-name <NAME>
    (delete an application)
  - AWS Console:

    1) Click on the application
    2) Scroll to the bottom and click 'Delete Application'

CodeDeploy Workflow:



Configure Instances
Create an Application
Deployment Group
Deployment Configuration
Create & Upload Revision
Deploying
Review Results
Redeploy if Needed
Set the type of instance (O.S) that we want for our deployment.
Create the framework and name identifier for our Application.
Select the instances targeted for deployment.
Set how quickly the deployment will roll-out to your instances.
Bundle your source files and upload them to an S3 Bucket.
Launch the Deployment and install your source files onto instances.
Review your deployment details and view error logs (if needed).
Redeploy if the Deployment fails or you want to Deploy a new Revision.

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

Edit, Add & Delete Deployment Groups & Configurations (in the AWS Console):

- Editing and Adding Groups & Configurations:
  1) *Navigate to CodeDeploy*
  2) *Click on one of your applications*
  3) *Select a deployment group and under actions, select "Edit"*
  4) *Here you can:*
     - *Change the deployment group name*
     - *Add/change/delete instances to deploy to*
     - *Change the deployment configurations*
     - *Add/edit/remove SNS Triggers*
     - *Change the service role*

- Adding an Additional Deployment Group:
  1) Click on an application
  2) Under Deployment Groups, click 'Create Deployment Group'
  3) Fill in the form with new information

- Deleting an Deployment Group:
  1) *Click on an application*
  2) Under Deployment Groups, select a deployment group
  3) Under Actions, click 'Delete'

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

Edit, Add & Delete Deployment Groups & Configurations (using the AWS CLI):

- Base CLI Command:

  aws deploy

- Edit a deployment group/configuration elements:

  required arguments:

  aws deploy update-deployment-group --application-name <NAME> --current-deployment-group-name <NAME>

  Optional (elements to change):

  --new-deployment-group-name <NAME>
  --ec2-tag-filters
  --on-premises-instance-tag-filters
  Key=Name,Value=<VALUE>,Type=KEY_AND_VALUE
  --auto-scaling-groups <NAME>
  --deployment-config-name CodeDeployDefault.<SELECTOPTION>
  --service-role-arn <SERVICE-ROLE_ARN>

- Delete a deployment group/configuration:

  aws deploy delete-deployment-group --application-name <NAME> --deployment-group-name <NAME>

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

Creating & Deleting Custom Deployment Configurations:

- Default AWS Deployment Configurations:
  - *OneAtATime*
  - *AllAtOnce*
  - *HalfAtATime*

- Creating a Custom Deployment Configuration:
  - *You can create a custom configuration via:*
    - *The AWS CLI*
    - *The AWS APIs*
    - *AWS CloudFormation Template*
  - *CLI Command:*

    aws deploy create-deployment-config

    Options

    --deployment-config-name <NAME>

    --minimum-healthy-hosts type=<OPTION>,value=<# or %>
    - *HOST_COUNT*
    - *FLEET_PERCENT*

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

Creating & Deleting a Custom Deployment Configuration:

- Viewing a Custom Deployment Configuration:
  - *CLI Commands:*

    To list all Deployment Configurations
    aws deploy list-deployment-configs

    To view detailed information on a specific Configuration
    aws deploy get-deployment-config --deployment-config-name <NAME>

- Deleting a Custom Deployment Configuration:
  - CLI Command:

    aws deploy delete-deployment-config --deployment-config-name <NAME>

What is the AppSpec File?

- AppSpec (short for Application Specification):
  - *Is YAML-formatted file used to specify the:*
    - *Source and target location of the files we want to deploy*
    - *Permissions given to your files once at their target location*
    - *Lifecycle event hooks available to run specific scripts against*
  - *The AppSpec file MUST be named **"appspec.yml"***

- YAML (YAML Ain't Markup Language):
  - *Is a human friendly data serialization standard for all programming languages*

- Source/Target location (required):
  - *The location of the directory (or specific file) we want to deploy from, and the target directory (or specific file location) we want to deploy to.*

- *Permissions (optional)*
  - *Specifies what (if any) special permissions you want your file or directories to have once deployed in the instance*

  Only applies to Amazon Linux, Ubuntu Server & RHEL

- *LifeCycle Event Hooks (optional):*
  - *Events in the deployment lifecycle that can trigger specific scripts to run*

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

AppSpec File “Header” Section:

- ***version**:*

  This number is completely arbitrary and can be used by you to keep track of versions/revisions.
- ***os**:*

  This is where you will specify the operating system of the instances you are deploying to. There are only two options, and you must choose only one of them.

  1) linux

  2) windows

- Example:

  ***version:** 1.0*

  ***os:** linux*

  ***version:** 2.0*

  ***os:** windows*

AppSpec File “Files” Section:

- Executes during the deployments “Install” lifecycle event.
- Tells CodeDeploy which files from your application revision should be installed during deployment.

- Base Format:

```
files:
  - source: <source-file-location>
    destination: <destination-file-location>
```

- Example:

```
files:
  - source: /html/wonderwidgets.html
    destination: /mywebsitefiles          (c:\mywebsitefiles)
  - source: /executables
    destination: /exe                     (c:\exe)
```

- Source Options:
  - *If ‘source’ refers to a file, ONLY that file will be installed*
  - *If ‘source’ refers to a directory, ALL directory content will be installed*
  - *If ‘source’ is just a single ‘ / ’, ALL files in the Revision will be installed*

AppSpec File "Permissions" Section:

- Specifies and assigns any special permissions you would like to assign to a file, files, or directory after installation.
- This section is ***optional*** and only applies to Amazon Linux, Ubuntu Server, and RHEL instances.
- This section MUST be removed for windows servers.

- Base Format:

```
permissions:
 - object: <object-specification>
   pattern: <pattern-specification>
   except: <exception-specification>
   owner: <owner-account-name>
   group: <group-name>
   mode: <mode-specification>
   acls:
     - <acls-specification>
   context:
     user: <user-specification>
     type: <type-specification>
     range: <range-specification>
   type:
     - <object-type>
```

AppSpec File “Permissions” Section:

- ***object*** *(required):*
  The target directory or file(s) you want to set permissions on.
- ***pattern****:*
  Specify a pattern to identify certain type of files to apply permissions to.
- ***except****:*
  Specify exceptions to the above pattern
- ***owner****:*
  Set the owner for the object (source settings if blank)
- ***group****:*
  Set the group for the object (source settings if blank)
- ***mode****:*
  Set the permissions value (think chmod command)
- ***acls****:*
  Access Control List entries applied to the object.
- ***context****:*
  For Security-Enhanced Linux (SELinux)-enabled instances.
  - *user: The SELinux user*
  - *type: The SELinux type name*
  - *range: The SELinux range specifier*
- ***type****:*
  Specify if the object is a file or a directory
  - *file: Permissions will be applied only to object’s files*
  - *directory: Permissions will be applied recursively to all directories, and files in the object.*

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

AppSpec File “Permissions” Section:

- Example:

```
permissions:
  - object: /wonderwidgets/html
    pattern: “*.html”
    except: “index.html”
    mode: 400
    type:
      - file
```

- *acls & context:*

```
acls:
  - u:matt:rw
  - u:kelly:rwx
context:
  user: unconfined_u
  type: httpd_sys_content_t
  range: s0
```

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

AppSpec File “Hook” Section:

- Allows you to link deployment lifecycle event hooks to scripts.

- Deployment Lifecycle Event Hooks:

ApplicationStop	*(Before app revision is downloaded, can’t be used to on the first revision upload)*
DownloadBundle	*(CodeDeploy Agent copies files to a temp loc.)*
BeforeInstall	*(Time to run pre-install tasks, i.e. decrypting files or backing up the current version)*
Install	*(CodeDeploy Agent installs files from temp loc. to your destination folder)*
AfterInstall	*(Time to configure your application or change permissions on files)*
ApplicationStart	*(Restart services that were stopped during ApplicationStop)*
ValidateService	*(Verify deployment was competed successfully)*

****highlighted hooks are ones you can run scripts against***

AppSpec File “Hook” Section:

- Base Format:

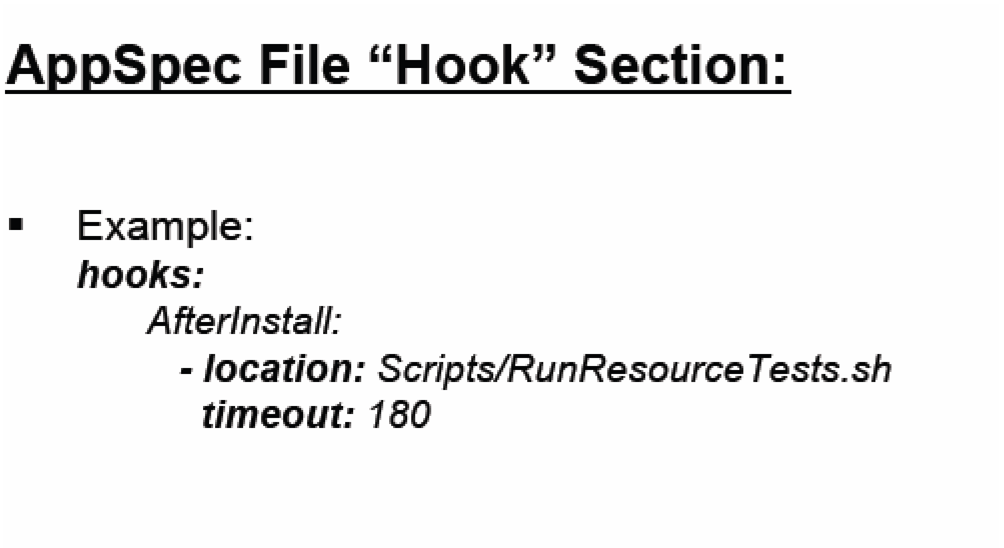
```
hooks:
    <deployment-lifecycle-event-name>
      - location: <script-location>
        timeout: <timeout-in-seconds>
        runas: <user-name>
```

- ***location*** *(required):*
  The location of the script file that you want to run.
- ***timeout****:*
  The amount of time you want to “allow” the script to run before it is considered to have failed (if not fully competed). Default timeout is set to 3600 second (1 hour).
- ***runas****:*
  The user to “assume” when running the script

AppSpec File “Hook” Section:

- The scripts that run during the deployment lifecycle can also access the following environment variables:

- ***APPLICTION_NAME***:
  The current CodeDeploy Application name.
- ***DEPLOYMENT_ID***:
  An ID number that CodeDeploy assigned to the current deployment.
- ***DEPLOYMENT_GROUP_NAME***:
  The name of the current CodeDeploy Deployment Group.
- ***DEPLOYMENT_GROUP_ID***:
  An ID number that CodeDeploy assigned to the current Deployment Group.
- ***LIFECYCLE_EVENT***:
  The name of the current deployment lifecycle event.
  These environment variables are local to each lifecycle event.



AppSpec File “Hook” Section:

- Example:

```
hooks:
    AfterInstall:
      - location: Scripts/RunResourceTests.sh
        timeout: 180
```

https://s3-us-west-2.amazonaws.com/qt-test-s3-1/DevOps/codedeploy-codedeploydemo-ec2-permissions-instance-role-policy-s3-access_1468694804.txt

Formatting your AppSpec File:

- Specific spacing is required for the AppSpec file (YAML):

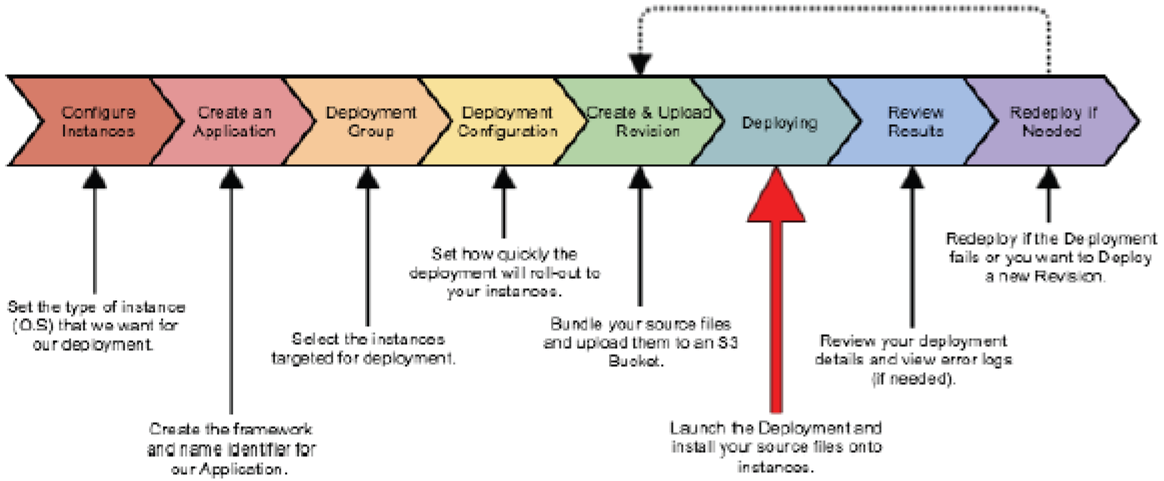
```
version:[1]version-number
os:[1]operating-system-name
files:
[2]-[1]source:[1]source-files-location
[4]destination:[1]destination-files-location
permissions:
[2]-[1]object:[1]object-specification
[4]pattern:[1]pattern-specification
[4]except:[1]exception-specification
[4]owner:[1]owner-account-name
[4]group:[1]group-name
[4]mode:[1]mode-specification
[4]acls:
[6]-[1]acls-specification
[4]context:
[6]user:[1]user-specification
[6]type:[1]type-specification
[6]range:[1]range-specification
[4]type:
[6]-[1]object-type
hooks:
[2]deployment-lifecycle-event-name:
[4]-[1]location:[1]script-location
[6]timeout:[1]timeout-in-seconds
[6]runas:[1]user-name
```

Validating your AppSpec File:

- You use a YAML validator to validate the AppSpec File.
- The validator will check to make sure that you have the proper amount of spaces per line and correctly formatted sections.

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

CodeDeploy Workflow:


Configure Instances
Create an Application
Deployment Group
Deployment Configuration
Create & Upload Revision
Deploying
Review Results
Redeploy if Needed
Set the type of instance (O.S) that we want for our deployment.
Create the framework and name identifier for our Application.
Select the instances targeted for deployment.
Set how quickly the deployment will roll-out to your instances.
Bundle your source files and upload them to an S3 Bucket.
Launch the Deployment and install your source files onto instances.
Review your deployment details and view error logs (if needed).
Redeploy if the Deployment fails or you want to Deploy a new Revision.

Deploying a Revision:

- You can deploy a revision to an instance via:
  - *AWS Console.*
  - *AWS CLI*
  - *AWS API*

Deploying a Revision via the AWS Console:

1) Navigate to CodeDeploy

2) From the dropdown at the top select “Deployments”

3) Click “Create New Deployment”

4) Fill out the form
   - *Application Name*
   - *Deployment Group Name*
   - *Revision Type (Where you revision is stored: S3/Github)*
   - *Revision Location*
   - *Deployment Description*
   - *Deployment Config*

5) Click “Deploy Now”

6) Monitor deployment “status”

7) Done!

Deploying a Revision via the AWS CLI:

1) Copy and paste the response given when we pushed the revision to our S3 bucket

2) Fill in the required areas inside the <> brakets:
   - *Deployment Group Name*
   - *Deployment Configuration Name:*
     - ***CodeDeployDefault.OneAtATime***
     - ***CodeDeployDefault.AllAtOnce***
     - ***CodeDeployDefault.HalfAtATime***
     - ***Custom (if you created one)***
   - *Description*

3) Execute the command

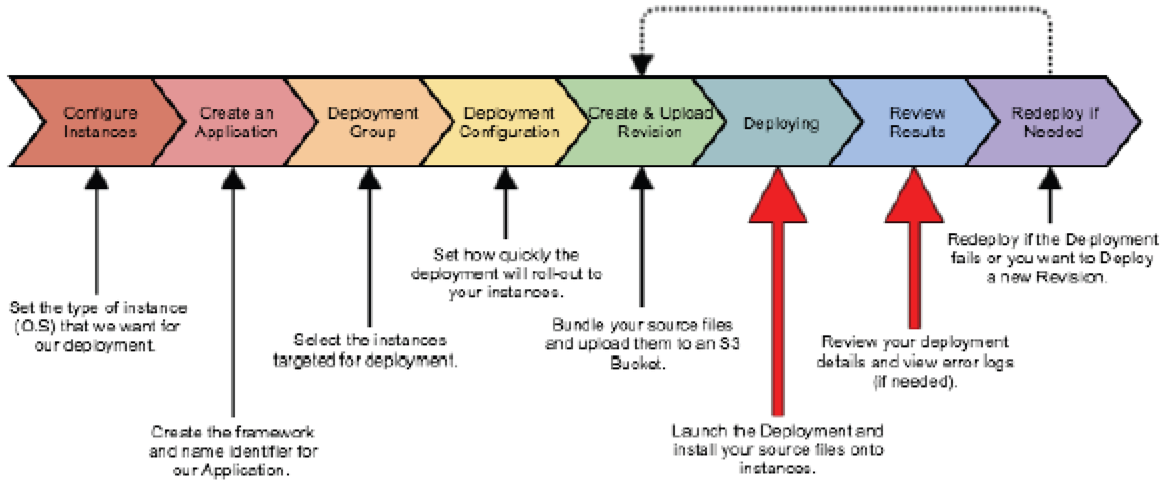
4) Check the "status" of the deployment:
   - Run the command:

```
aws deploy get-deployment --deployment-id <ID>
```

5) Done!

QUALITY THOUGHT
INFO SYSTEMS (INDIA) PVT. LTD

CodeDeploy Workflow:



Configure Instances
Create an Application
Deployment Group
Deployment Configuration
Create & Upload Revision
Deploying
Review Results
Redeploy if Needed
Set the type of instance (O.S) that we want for our deployment.
Create the framework and name identifier for our Application.
Select the instances targeted for deployment.
Set how quickly the deployment will roll-out to your instances.
Bundle your source files and upload them to an S3 Bucket.
Launch the Deployment and install your source files onto instances.
Review your deployment details and view error logs (if needed).
Redeploy if the Deployment fails or you want to Deploy a new Revision.

QUALITY THOUGHT INFO SYSTEMS (INDIA) PVT. LTD

CodeDeploy SNS Triggers:

- SNS Triggers must be setup pre-deployment
- SNS Triggers are great for automating the monitoring process. Allowing you to get instant notifications if deployments or instances fail or succeed.
- SNS Triggers can be setup via:
  - *AWS Console*
  - *AWS CLI*
- SNS Triggers:
  - *Deployment Status:*
    - ***Deployment Starts***
    - ***Deployment succeeds***
    - ***Deployment fails***
    - ***Deployment Stops***
  - *Instance Status:*
    - ***Instance Starts***
    - ***Instance Succeeds***
    - ***Instance Fails***

Add SNS Triggers via the AWS Console:

1) Navigate to CodeDeploy

2) Click on the Application you want to add a trigger to

3) Click on the "arrow" on the Deployment Group that you want to add the trigger to (this will display the Deployment Group setting)

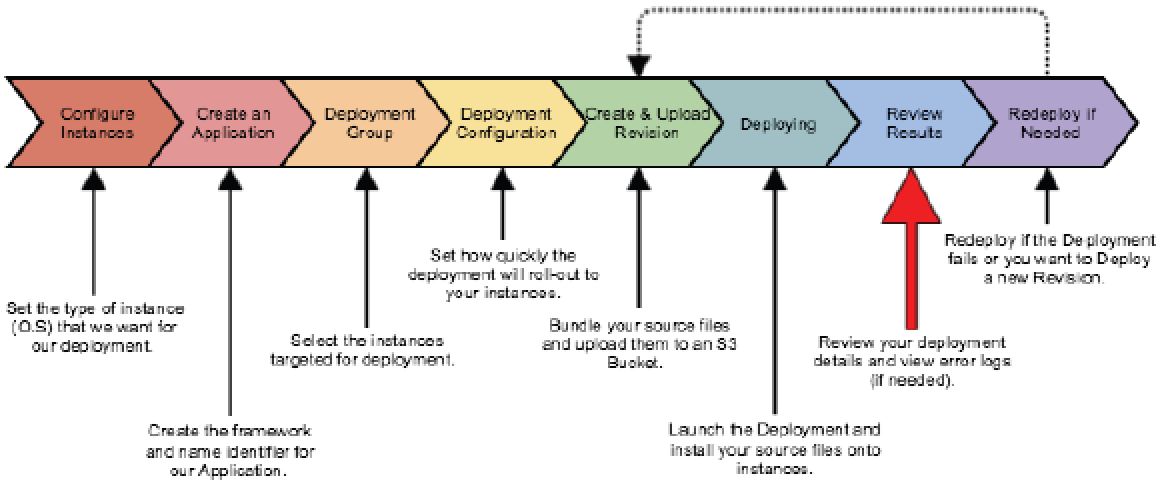
4) Click on "Create Triggers"

5) Fill our the form, setting or selecting:
   - *Trigger Name*
   - *The event(s) you want to invoke the trigger*
   - *The SNS topic you want to execute*

   ***You must already have a SNS topic setup**

6) Click "Create Trigger"

CodeDeploy Workflow:



Configure Instances
Create an Application
Deployment Group
Deployment Configuration
Create & Upload Revision
Deploying
Review Results
Redeploy if Needed
Set the type of instance (O.S) that we want for our deployment.
Create the framework and name identifier for our Application.
Select the instances targeted for deployment.
Set how quickly the deployment will roll-out to your instances.
Bundle your source files and upload them to an S3 Bucket.
Launch the Deployment and install your source files onto instances.
Review your deployment details and view error logs (if needed).
Redeploy if the Deployment fails or you want to Deploy a new Revision.

Viewing Deployment Details via the Console:

- What can we view?
  - *Deployment details*
  - *Instance details*
  - *Application details*
  - *Deployment Group details*
  - *Application Revision details*
  - *Deployment Configuration details*

- After you make your first deployment, or multiple deployment, I suggest you take some time to click around CodeDeploy in the AWS Console to get familiar with all the various views and information that is available.