**Case Study**

Problem Statement : We need to automate the CI/CD process for the organization so that every developer in the org can deploy their code swiftly to production servers but testing their changes in a stage environment. The app should be deployed to stage environment before promoted to production. This is an internet facing NodeJS web application. End users will access the application through an URL like https://node-example.com. As a DevOps engineer you are tasked to automate the deployment pipeline for this application with AWS services and other open source tools.

Design and implement a robust CI/CD process so that the developers can deploy the code anytime they wish on their own easily. The application should be deployed in a highly available, fault tolerant environment. In the event of any abnormality like service outage or deployment failure the developers should be notified of the incident. You are open to use any of the AWS services and technologies needed to implement this solution.

The application source code is hosted here. https://github.com/opsworkshop/node-hello

Solution :

One of the ways that we can implement this scenario by using the following steps :-

* Design of the solution :
  + 1. CI/CD - We can leverage the AWS DevOps pipeline services like CodeCommit,CodeBuild and CodeDeploy for implementing the CI/CD process.
    2. Hosting - Apart from that we can also use the EC2 to run the nodejs applications that can be configured using EBS & with Cloudfront to regsiter our host with the DNS.
    3. Notifications : For notifications related to any such changes/faliure we can use the SNS services which will give us notifications/alerts on our mail.
    4. HA/Fault tolerant : We can have a back-up of the existing code repository in the S3 services.And can use Lambda services for AWS Serverless Automation to have another secondary CI/CD pipeline ready.
* Complexity, (less complex scores more) :
* Automation level :
* Maintenance required :
* Security : We can secure the CI/CD process using IAM & CloudTrail for logging services for the users.

CI/CD

Initial steps : At first we can signup for the AWS console from here [https://portal.aws.amazon.com/billing/signup#/start](https://portal.aws.amazon.com/billing/signup" \l "/start) . After creating the root account we will enable all the neccessary security features(Enable MFA,delete root access keys,create individual IAM users,IAM Password Policy etc) and billing details and then following the AWS best practices we will create a new IAM users with the administration level access for all the AWS services and start using that account for our configuration and other needs.From here after we can start creating the groups and users for the projects members.We can share them the credentials inidividually using the one time generated credentials.csv file depending upon the AWS management console access or programatic access using access key id.

Step-1: At first since the code is hosted on the external repository services we can migrate our code from the Github to CodeCommit in AWS.

* The followings steps will clone the demo application hosted on GitHub git\_url="https://github.com/opsworkshop/node-hello.git" to a local repo in a directory named node-hello.

Commands :

**git clone --mirror https://github.com/opsworkshop/node-hello.git node-hello**

//Change directories to the directory where you made the clone.

**cd node-hello**

**//**creating the repostiory in the codecommit

**aws codecommit create-repository --repository-name node-hello –repository- description "GOGO Hello NodeJS App" --tags Team=DevOps**

//output

{

"repositoryMetadata": {

"accountId": "597733473342",

"repositoryId": "53408346-25a5-405d-802f-1c67404bcab7",

"repositoryName": "node-hello",

"repositoryDescription": "GOGO Hello NodeJS App",

"lastModifiedDate": 1572265625.823,

"creationDate": 1572265625.823,

"cloneUrlHttp": "<https://git-codecommit.us-west-2.amazonaws.com/v1/repos/node-> hello",

"cloneUrlSsh": "ssh://git-codecommit.us-west-2.amazonaws.com/v1/repos/node- hello",

"Arn": "arn:aws:codecommit:us-west-2:597733473342:node-hello"

}

}

* Run the git push command, specifying the URL and name of the destination CodeCommit repository and the --all option. (This is the URL we copied in Step 1: Create a CodeCommit Repository).For example, if we named our repository Node-Hello and we are set up to use HTTPS, we would run the following command:

**git push https://git-codecommit.us-west-2.amazonaws.com/v1/repos/node-hello --all**

Step 2 : After this we can share the CodeCommit Repository. After we create a repository in CodeCommit, two endpoints are generated: one for HTTPS connections and one for SSH connections. Both provide secure connections over a network. Our team members can use either protocol. Both endpoints remain active no matter which protocol we recommend to our users. Before we can share our repository with others, we must create IAM policies that allow other users access to your repository. Provide those access instructions to our team members.