National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Ganesh Patil
Student ID: x20193009

School of Computing
National College of Ireland

Supervisor: Sean Heeney

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Ganesh Patil
Student ID:	x20193009
Programme:	Cloud Computing
Year:	2021
Module:	MSc Research Project
Supervisor:	Sean Heeney
Submission Due Date:	15/08/2022
Project Title:	Configuration Manual
Word Count:	952
Page Count:	11

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Ganesh Patil
Date:	15th August 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Ganesh Patil
x20193009

1 Introduction

There are two main sections in the project. One would be the deployment and creation of pipeline while second being the introduction of vulnerabilities inside the application to test the pipeline. This configuration manual will help you to re-create the deployment of simple java based application following the DevSecOps best practices with the help of DCVS which will scan for static and dynamic vulnerabilities.

1.1 Prerequisites for conducting the research project

To conduct this research project, make sure you have AWS account set-up and small cost which would require to use these services. There are very few pre-requisites as main aim of the research project was to make it simpler and automated following the best practices of security. Hence, we will be doing the most out of AWS services and integrating a few third party tools such as sonarqube, OWASP ZAP and Clair.

- ### 1.1.1 Configuring the environment

  In the AWS dashboard go the search bar and look up for Cloud9. Create the environment as highlighted in the image below. In the configuration setting select the "t2.micro" as it is free and we won't need much memory. An EC2 instance will be launched in background for the Cloud9 environment as indicated in Image 2.

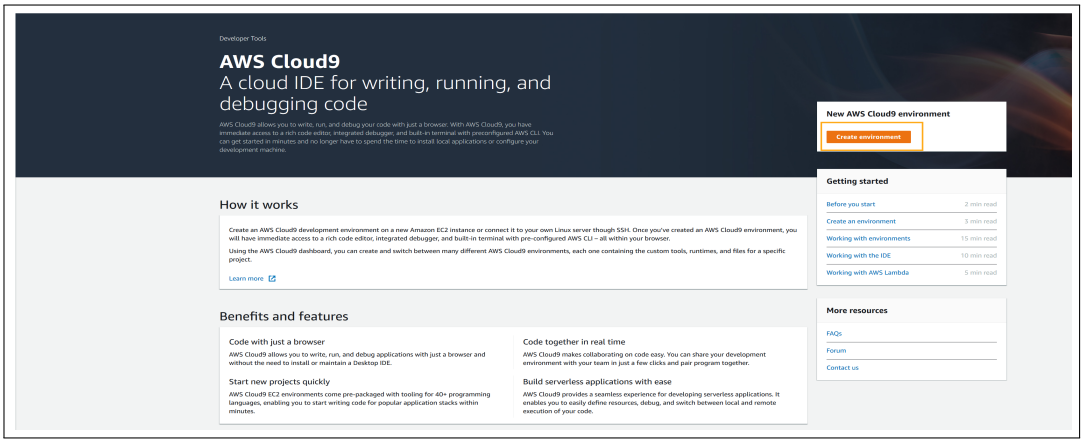


Figure 1: Cloud9 Front Page

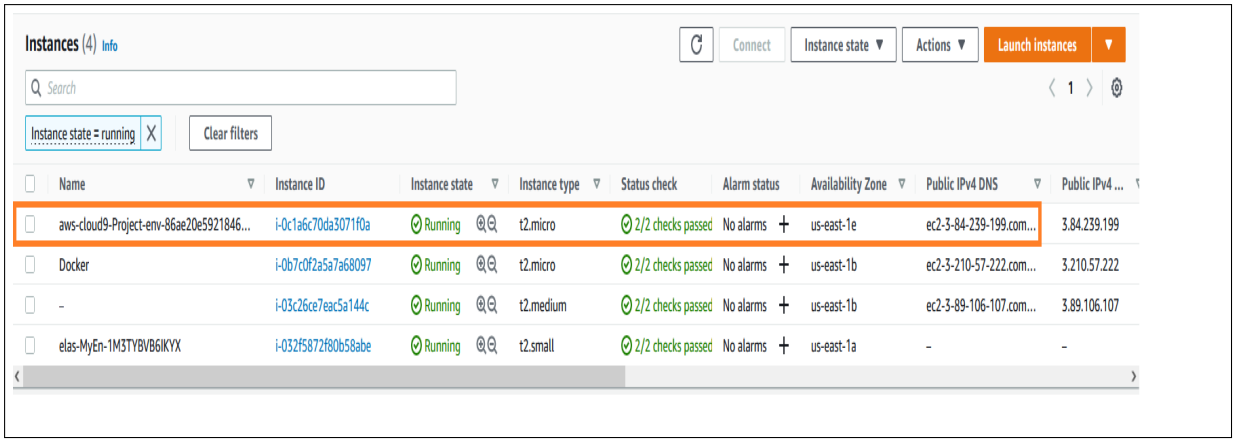


Figure 2: EC2 Instance

Below steps need to be done to provide suitable access to the EC2 environment.

- **1.1.2 IAM Role**
  - Search for IAM in the search bar. Then navigate to Roles, in that select the option for create role.
  - Select AWS service as trusted entity and EC2 as Common usecase.
  - On the next page search for AdministratorAccess as the Policy and provide a suitable name for the role and finish by creating the role.
  - Navigate to EC2 console to attach the role to IAM console.
  - Select the EC2 instance and then click Actions and to choose the Security option and modify IAM role.
  - A drop down will appear where select the appropriate role created in the IAM console and click on save.

CLI needs to installed so that configuration of environment takes place in no time.

- **1.1.3 CLI Installation steps**
  - First we should ensure that we remove if there are any associated present in the file which needs to be removed. TO ensure that use the command.

```
rm -vf ${HOME}/.aws/credentials
```

  - Installing of jq

```
sudo yum install jq -y
```

– CLI needs to be configured with respect to the current region

```
export ACCOUNT_ID=$(aws sts get-caller-identity --output text --query Account)
export AWS_REGION=$(curl -s 169.254.169.254/latest/dynamic/instance-identity/document | jq -r '.region')
```

– The bash profile needs to be saved to keep the configurations intact.

```
echo "export ACCOUNT_ID=${ACCOUNT_ID}" | tee -a ~/.bash_profile
echo "export AWS_REGION=${AWS_REGION}" | tee -a ~/.bash_profile
aws configure set default.region ${AWS_REGION}
aws configure get default.region
```

Before starting the experiment let's check if the CLI has been configured properly. If not then just configure it.

```
aws --version
aws configure get region
```

2 Setting up the Platform

- For this Research project we will utilize the functionality of CodeCommit Repository.

```
aws codecommit create-repository --repository-name docker-repo-${ACCOUNT_ID} --repository-description "DevSecOps Research Project"
```

- Transfer the code to CodeCommit repo by cloning the github repo. Use the follwoing github link for cloning. `https://github.com/ganeshpatil97/Thesis_project`
- The git needs to setup for the codecommit repo for tracking the changes

```
git init
git checkout -b main
git add .
git commit -m "Initial Commit"
git status
```

- Provide the link of remote so that the remote and the local version are at same point.

```
git remote add codecommit https://git-codecommit.us-east-1.amazonaws.com/v1/repos/docker-repo-${ACCOUNT_ID}
```

3 Setting Up S3

We need to upload the lambda function and sample java app onto the S3 bucket. The do that follow the steps below.

- Now lets start by creation of S3 bucket in the AWS using the below commands.

```
aws s3 mb s3://dsop-bucket-${ACCOUNT_ID}
export S3_BUCKET="dsop-bucket-${ACCOUNT_ID}"
```

- The sample app needs to be copied in the S3 bucket along with corretto which is the jdk required for the java app.

```
cd ~/environment/devsecops-cicd/application
aws s3 cp corretto.zip s3://dsop-bucket-${ACCOUNT_ID}
```

- Then paste the lambda function into the bucket.

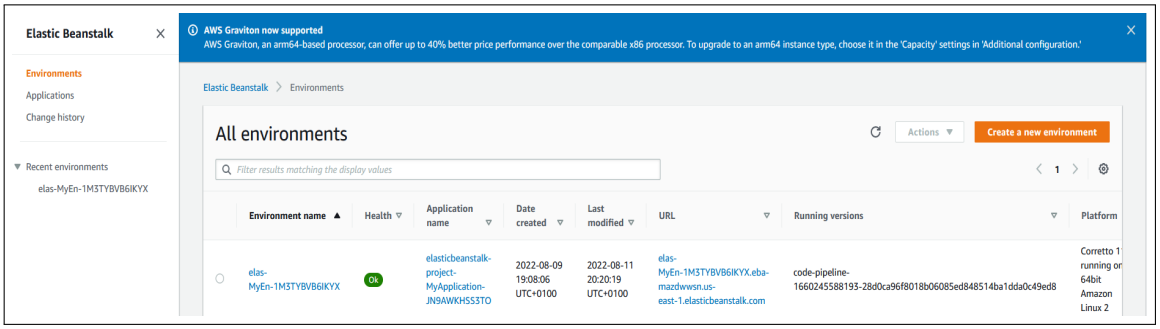
```
cd lambda-functions
aws s3 cp import_findings_security_hub.zip s3://dsop-bucket-${ACCOUNT_ID}
```

4 Setting up Elastic Beanstalk

- Using the below command create the CloudFormation template for the purpose of deployment.The template can be found on `https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-cli-creating-stack.html`

```
aws cloudformation create-stack --stack-name elasticbeanstalk-project --template-body
file://$HOME/environment/devsecops-cicd/application/templates/elasticbeanstalk-infrastructure-java.yaml --parameters ParameterKey=EBSolutionStack,ParameterValue="${EB_SolutionStack}"
ParameterKey=S3BucketName,ParameterValue=${S3_BUCKET} --capabilities CAPABILITY_NAMED_IAM
```

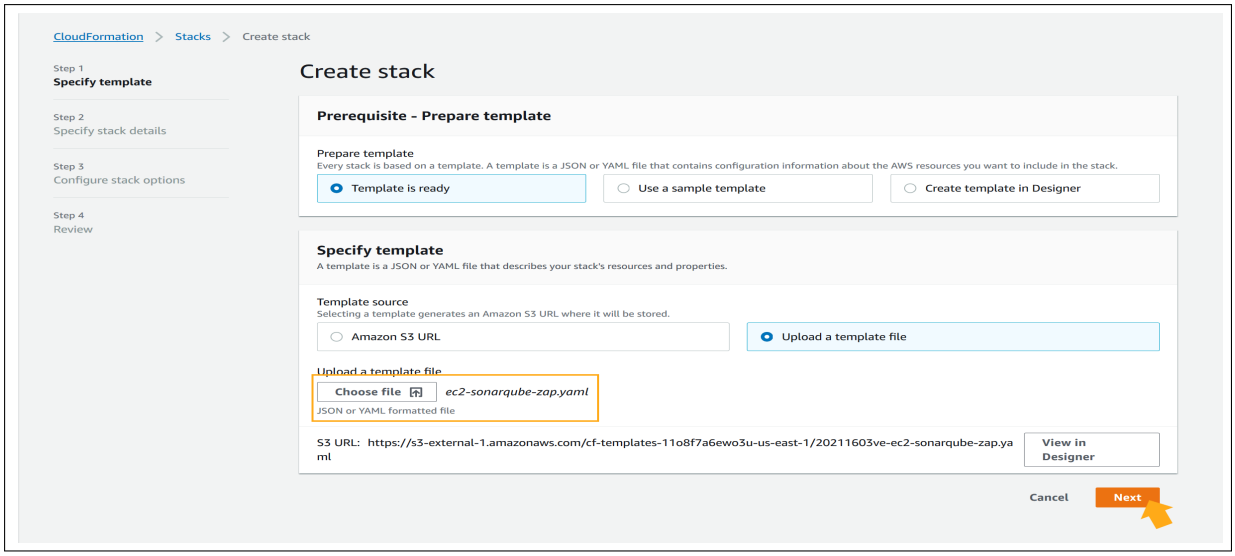
The creation will take time and keep checking the stack by refreshing the page.



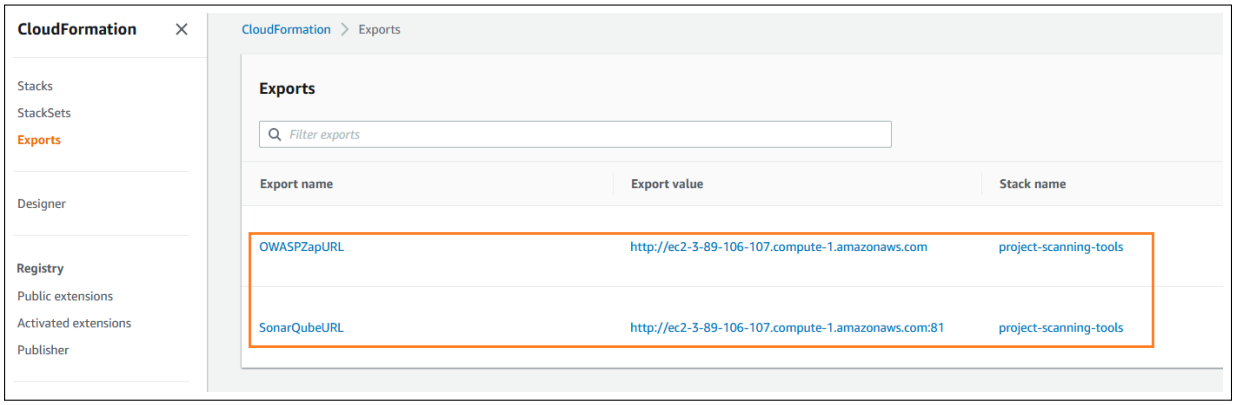
In the figure 4 you can see the environment is created.

5 SAST and DAST Tools setup and deployment

Use the templates of Sonarqube for the static scanning and OWASP ZAP for dynamic scanning provided in the Github repo. The link for Github repo is `https://github.com/ganeshpatil97/Thesis_project` Search for Cloudformation and select create stack the following screen will appear. Choose the options as shown in the picture.

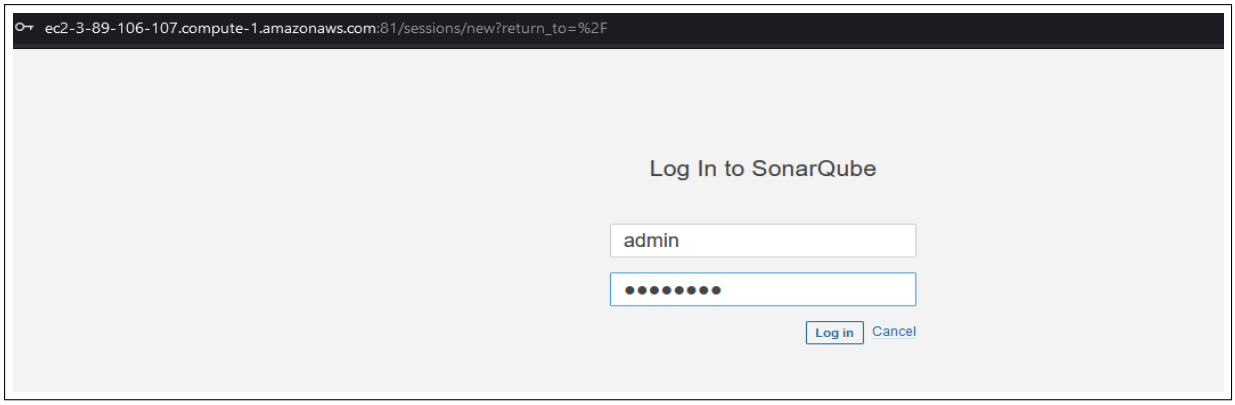


Provide the name of stack and select instance type as t2.micro and click next. Finally click on create stack and the stack will be created in sometime. It will look something like the image shown below screenshot 5

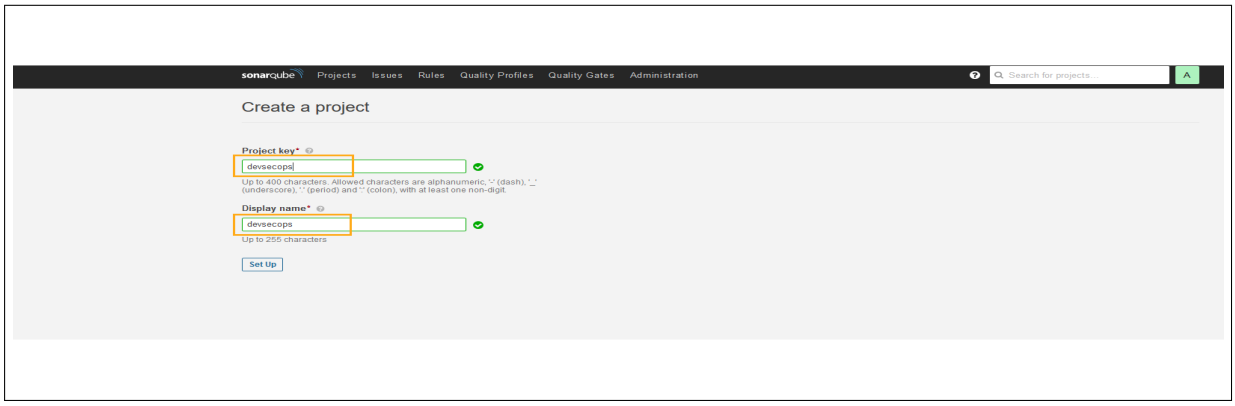


5.1 SonarQube Configuration

Using the url in Image5 access the SonarQube and enter the following as shown in the picture below. Enter password as admin123



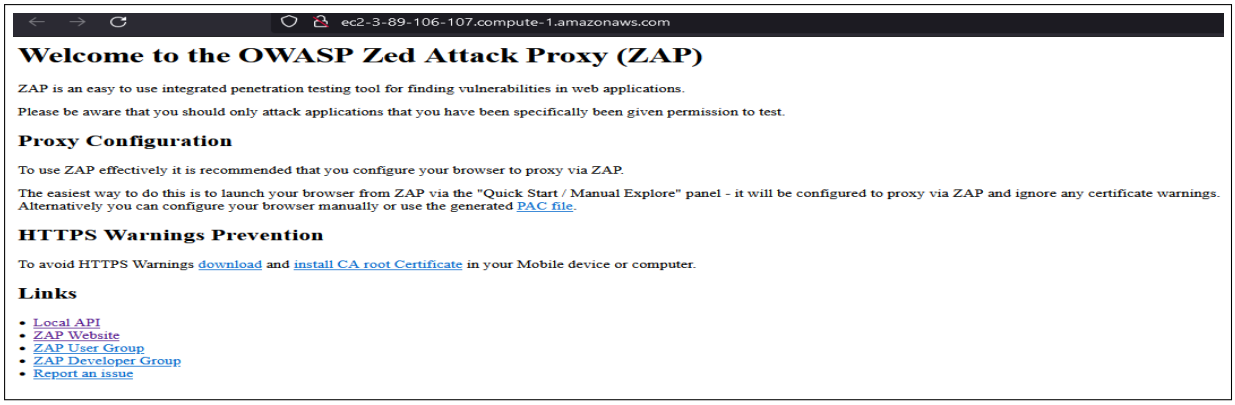
Then click on create new project and select option manual.Enter the project as shown in the picture below and click on setup.



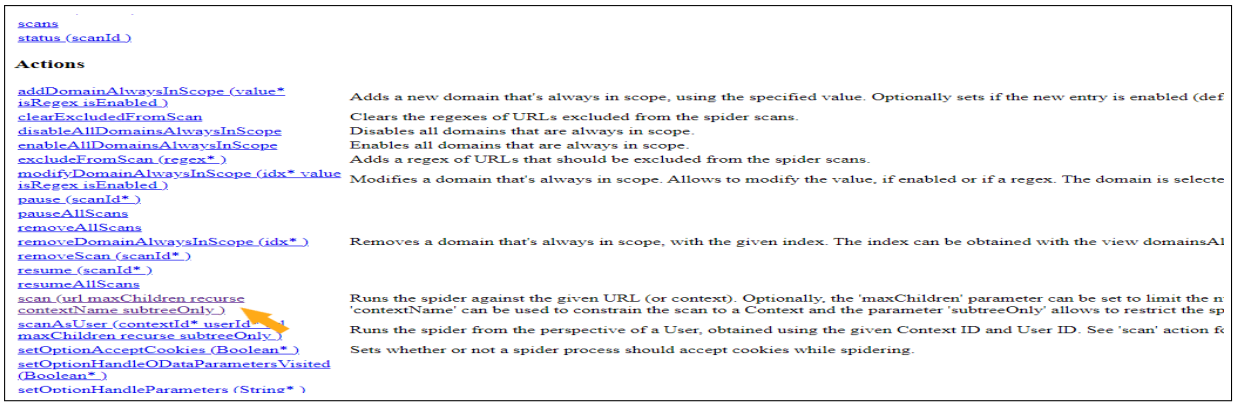
Now generate the token and save it locally. Click on continue and select Maven as the technology to for building the project. Token for this project is SONARQUBE API TOKEN=”6b6f205b3905690f1f03b63b7424d05df4e28c48”

5.2 OWASP ZAP Configuration

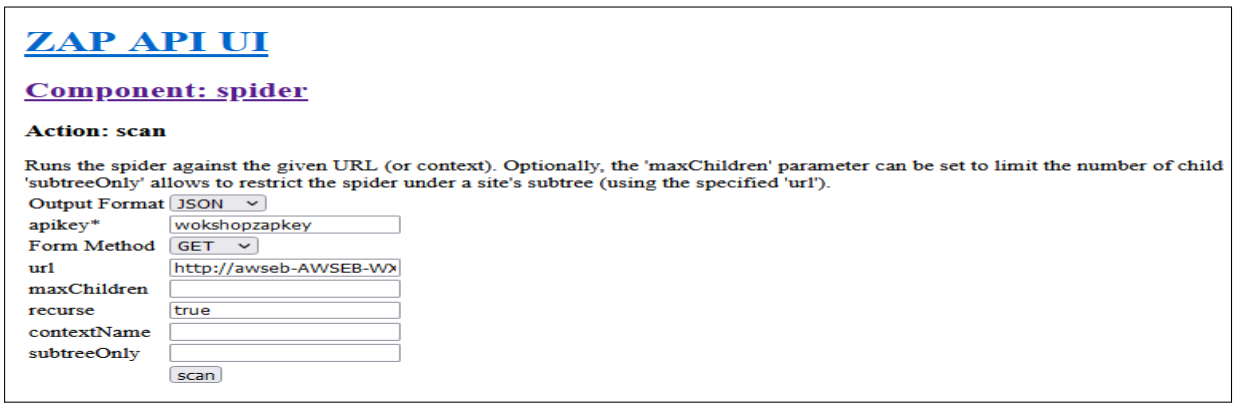
Click on the link a page will open like below where you need to navigate to Local API.



Before configuring the OWASP ZAP server one thing needs to be done. OWASP ZAp has a tool named spider which is used for mapping of the whole application. Hence after clicking local *API => spider => scan.Asindicatedinthepicture.*



Now a screen will appear as shown below hwere you need to enter the details as shown in the image. Enter the respective details such as apikey, url, recurse and click on get.



clicking on scan should provide the output as scan : "0"

Now come back to the homepage and select the option of */acan => scan => entersameapikeyasabove => url => recurseastrue => clickscan.*

These steps are important so that our application gets scanned based on the endpoint url provided.

6 Configuring the Security Hub

Download the template from the Github repo and go to the CloudFormation and repaet the same steps as done previously and shown in image 5. click next and provide name for the stack and after that click next. Acknowledge the permission and create stack.

7 Deployment of Pipeline and Configuring the SNS

To create the pipeline we must first retrieve the details of the Elastic Beanstalk environment, the application and URL. These values can be further on passed while configuring the pipeline.

```
EB_ENVIRONMENT=$(aws cloudformation describe-stack-resources --stack-name elasticbeanstalk-project | jq '.StackResources[] |
select(.ResourceType=="AWS::ElasticBeanstalk::Environment").PhysicalResourceId' | tr -d '[\"\n]')
```

```
EB_APPLICATION=$(aws cloudformation describe-stack-resources --stack-name elasticbeanstalk-project | jq '.StackResources[] |
select(.ResourceType=="AWS::ElasticBeanstalk::Application").PhysicalResourceId' | tr -d '[\"\n]')
```

```
EB_URL=$(aws cloudformation describe-stacks --stack-name elasticbeanstalk-project | jq '.Stacks[].Outputs[] |
select(.OutputKey=="EBEndPointURL").OutputValue' | tr -d '[\"\n]')
```

Gather the information such as apikeys and other details using the commands listed below.

```
SONARQUBE_URL=$(aws cloudformation list-exports | jq '.Exports[] | select(.Name=="SonarQubeURL").Value' | tr -d '[\"\n]')

ZAP_URL=$(aws cloudformation list-exports | jq '.Exports[] | select(.Name=="OWASPZapURL").Value' | tr -d '[\"\n]')

export ZAP_API_KEY="workshopzapkey"
```

Export all the details using the commands provided below.

```
export S3_BUCKET="dsop-bucket-${ACCOUNT_ID}"
export  REPO_NAME="docker-repo-${ACCOUNT_ID}"

export SONARQUBE_API_TOKEN="6b6f205b3905690f1f03b63b7424d05df4e28c48"

export PIPELINE_APPROVER_EMAIL="ganeshpatil.1997@gmail.com"
export PIPELINE_NOTIFICATIONS_EMAIL="ganeshpatil.1997@gmail.com
```

This will be the final step towards the deployment of pipeline. Use the below command to create the pipeline.

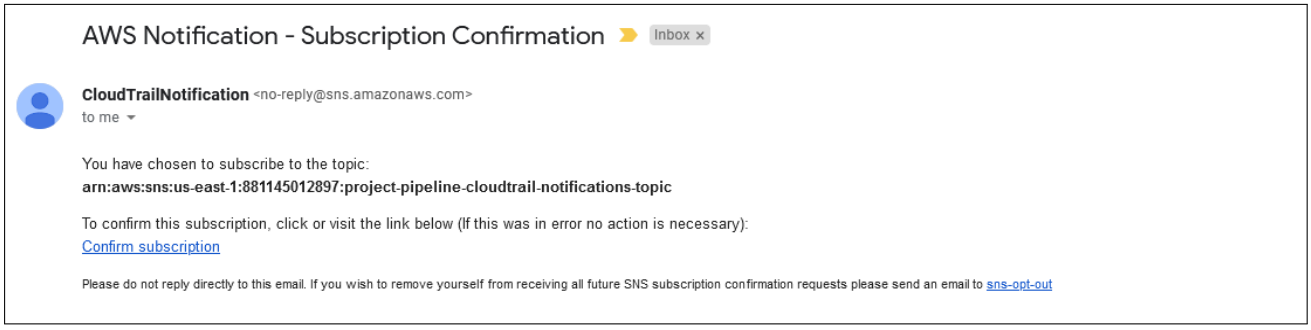
```
aws cloudformation create-stack --stack-name project-pipeline --template-body file://$HOME/environment/devsecops-cicd/application/templates/devsecops-codepipeline.yaml --parameters
ParameterKey=RepositoryName,ParameterValue=${REPO_NAME} ParameterKey=ApplicationName,ParameterValue=${EB_APPLICATION}
ParameterKey=ElasticBeanstalkEnvironment,ParameterValue=${EB_ENVIRONMENT} ParameterKey=PRDApplicationName,ParameterValue=${EB_APPLICATION}
ParameterKey=PRDElasticBeanstalkEnvironment,ParameterValue=${EB_ENVIRONMENT} ParameterKey=SonarQubeScanToken,ParameterValue=${SONARQUBE_API_TOKEN}
ParameterKey=SonarQubeURLName,ParameterValue=${SONARQUBE_URL} ParameterKey=LambdaPackageLoc,ParameterValue=${S3_BUCKET} ParameterKey=OwaspZapURLName,ParameterValue=${ZAP_URL}
ParameterKey=OwaspZapApiKey,ParameterValue=${ZAP_API_KEY} ParameterKey=ApplicationURLForDASTScan,ParameterValue=http://${EB_URL}
ParameterKey=PipelineApproverEmail,ParameterValue=${PIPELINE_APPROVER_EMAIL} ParameterKey=PipelineNotificationsEmail,ParameterValue=${PIPELINE_NOTIFICATIONS_EMAIL} --capabilities
CAPABILITY_NAMED_IAM
```

The above command can be found in the documentation and has ben adapted using the help of the two url links provided below.

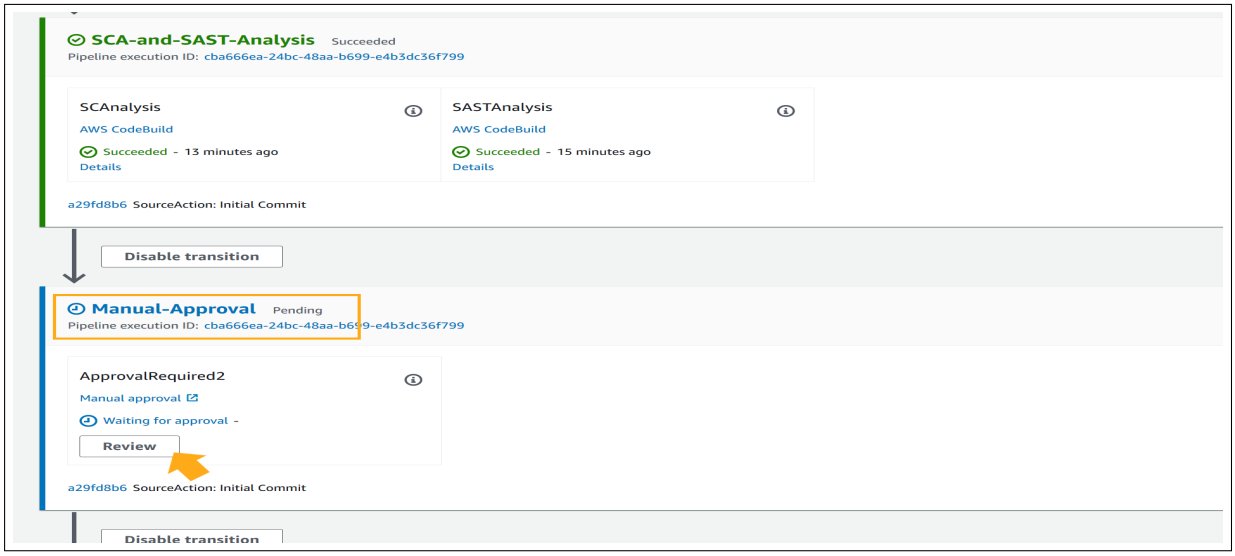
- `https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-cli-creating-stack.html`
- `https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiIsYDCoMT5AhWNS8AKHQwZBDgQFnoECAsQAQ&url=https%3A%2F%2Fdocs.aws.amazon.com%2Fcli%2Flatest%2Freference%2Fcloudformation%2Fcreate-stack.html&usg=AOvVaw2I1BTusWeGCAOSTlDKde4k`

A pipeline will be deployed after running the following command mentioned above.

Once the pipeline is deployed, the email provided will receive an subscription notification as shown below.

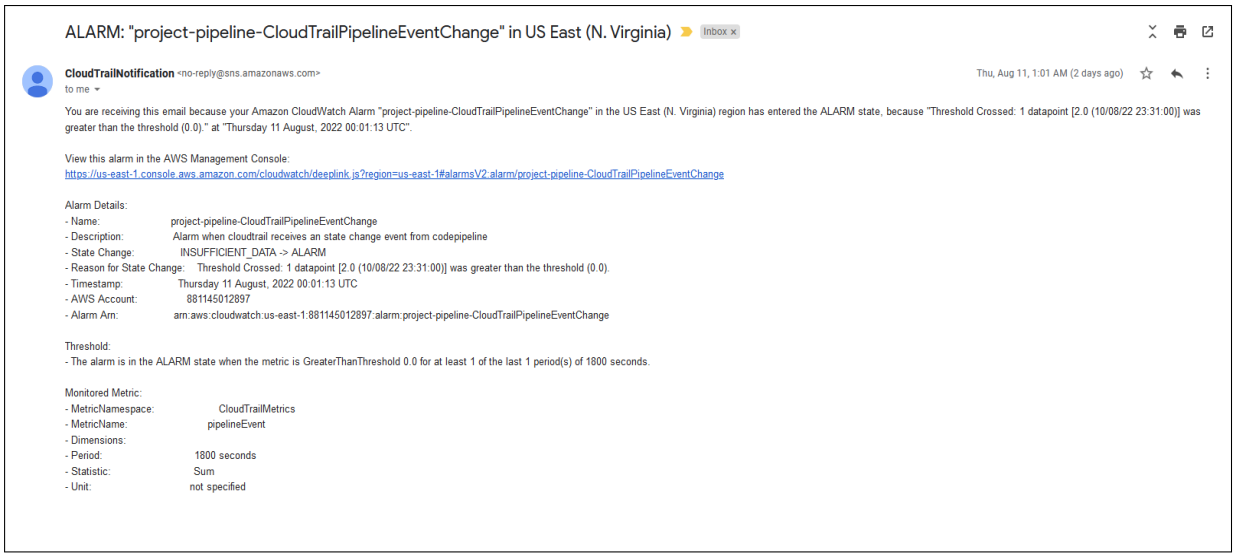


To initiated the pipeline just make changes in the file and commit the changes. The pipeline needs manual approval



Also an email is triggered once the pipeline reaches the approval stage, which will have alink for approval which will direct directly to aws codepipeline.

If the pipeline has failed or encountered a problem an alarm will be triggered which will look something like this



In this way the DCVS can be configured to automate the work of static and dynamic testing.