Project 5: Building a Secure CI/CD Pipeline

Detailed Report

Project Goal: Develop a secure Continuous Integration and Continuous Deployment (CI/CD) pipeline for an AWS-based application.

Guide

1. **Set Up CI/CD Pipeline:**
   - Configure AWS CodePipeline and AWS CodeBuild to create the pipeline.
2. **Integrate Security Checks:**
   - Add static code analysis and dependency scanning to the pipeline.
3. **Configure IAM Roles and Policies:**
   - Set up least-privilege permissions for IAM roles and policies.
4. **Manage Credentials Securely:**
   - Use AWS Secrets Manager to securely handle credentials within the pipeline.
5. **Implement Secure Deployment:**
   - Utilize AWS CodeDeploy for secure application deployment.

Key Activities and Implementation:

1. **Using AWS CodePipeline and AWS CodeBuild:**
   - Configured a CI/CD pipeline with AWS CodePipeline and AWS CodeBuild.

```
json
{
  "pipeline": {
    "name": "MyPipeline",
    "roleArn": "arn:aws:iam::123456789012:role/AWS-CodePipeline-
Service",
    "artifactStore": {
      "type": "S3",
      "location": "my-codepipeline-artifact-bucket"
    },
    "stages": [
      {
        "name": "Source",
        "actions": [
          {
            "name": "Source",
            "actionTypeId": {
              "category": "Source",
              "owner": "AWS",
              "provider": "S3",
              "version": "1"
            },
            "outputArtifacts": [
              {
                "name": "SourceArtifact"
              }
            ],
            "configuration": {
              "S3Bucket": "my-source-bucket",
              "S3ObjectKey": "source.zip"
            }
          }
```

```
        ]
      },
      {
        "name": "Build",
        "actions": [
          {
            "name": "Build",
            "actionTypeId": {
              "category": "Build",
              "owner": "AWS",
              "provider": "CodeBuild",
              "version": "1"
            },
            "inputArtifacts": [
              {
                "name": "SourceArtifact"
              }
            ],
            "configuration": {
              "ProjectName": "MyBuildProject"
            }
          }
        ]
      },
      {
        "name": "Deploy",
        "actions": [
          {
            "name": "Deploy",
            "actionTypeId": {
              "category": "Deploy",
              "owner": "AWS",
              "provider": "CodeDeploy",
              "version": "1"
            },
            "inputArtifacts": [
              {
                "name": "BuildArtifact"
              }
            ],
            "configuration": {
              "ApplicationName": "MyCodeDeployApplication",
              "DeploymentGroupName": "MyDeploymentGroup"
            }
          }
        ]
      }
    ]
  }
}
```

2. **Integrating Security Checks into the Pipeline:**
    - Incorporated static code analysis and dependency scanning into the pipeline.

```
bash
# Example CodeBuild buildspec file
version: 0.2

phases:
  install:
    runtime-versions:
      python: 3.8
```

```
    commands:
      - pip install -r requirements.txt
      - pip install bandit safety
  build:
    commands:
      - bandit -r .
      - safety check
```

3. **Configuring IAM Roles and Policies:**
   - Set up least-privilege permissions for IAM roles and policies to enhance security.

```
json
Code kopieren
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codepipeline:*",
        "codebuild:*",
        "codedeploy:*"
      ],
      "Resource": "*"
    }
  ]
}
```

4. **Using AWS Secrets Manager:**
   - Utilized AWS Secrets Manager for secure management of credentials within the pipeline.

```
bash
aws secretsmanager create-secret --name MyDatabaseSecret --secret-
string '{"username":"admin","password":"password"}'
```

5. **Secure Deployment with AWS CodeDeploy:**
   - Used AWS CodeDeploy for secure deployment of the application to EC2 instances or Amazon ECS.

```
json
{
  "applicationName": "MyCodeDeployApplication",
  "deploymentGroupName": "MyDeploymentGroup",
  "deploymentConfigName": "CodeDeployDefault.OneAtATime",
  "ec2TagFilters": [
    {
      "Key": "Name",
      "Value": "MyAppServer",
      "Type": "KEY_AND_VALUE"
    }
  ],
  "serviceRoleArn":
"arn:aws:iam::123456789012:role/CodeDeployServiceRole"
}
```

Outcome: A secure and efficient CI/CD pipeline that ensures continuous and secure deployment of the application.