Requirements

Create a general architecture, considering the following requirements:

- Requirement 001: Use Gitlab as a CI/CD Platform for cloud infrastructure and application deployment on AWS.
- Requirement 002: Create two separate environments, Dev & Prod.
- Requirement 003: Use a separate AWS account to secure access to Dev & Prod Accounts without creating long-term access keys in deployment environments.
- Requirement 004: Have the ability to add a new environment at any point in time with a repeatable approach using the same pattern.
- Requirement 005: The Gitlab pipeline should interact and authenticate only with a DevOps account to perform deployments into the Dev & Prod account.

The task also considers the use of CDK, CLI, and npm (node.js)

The requirements describe the natural and best-practice-based model for the deployment of infrastructure/products. Different accounts respect the concept of multi-account isolation allowing a control of the area of impact, the service limits, and workload visibility. Also, dividing the accounts in the DevOps and developer/production takes into consideration the Control Plane/Data Plane paradigm, where data accounts (the latest) have a trust relationship with the control plane accounts (the former).

Tools considerations

The requirements are consistent with the tools mentioned: CDK allows programming language design of infrastructure integrating seamlessly with GitLab pipelines, npm is required for CDK/GitLab integration, and calling of CDK commands is done using CLI.

Authorization consideration (bottom area of the cicd_sourav_20122023.operational_auth.pdf diagram)

Considering the requirements, and always establishing a trust relationship between any of the Development and Production accounts to the DevOps account, we have two main options:

a) Creation of programming access keys in the DevOps account and use of those keys in the definition of the tasks in GitLab.
b) Setting of GitLab as an Identity Provider and the use of temporary tokens between AWS (using STS) to GitLab.

The second alternative is more secure because, in that way, we will never have any kind of credentials for GitLab in the DevOps account that can be lost or stolen.

Operational mode and construction instructions (top area of the cicd_sourav_20122023.operational_auth.pdf diagram)

Three accounts need to be created: developer, production, and DevOps, but more of the developer's or production's environment can be added after if needed with minimum changes in the configuration. The developer and production accounts need to be bootstrapped with CDK and configured with a trust relationship with the DevOps account (

A CDK definition file will include the infrastructure that will be deployed.

A GitLab/pipeline definition file (.gitlab-ci.yml) needs to be created with all the authorization, stages, tasks, and script definitions to:

- Authorize using temporary tokens received from the DevOps account
- Build components for CDK and any building that will be needed for software or application components.
- Synth CDK definition in Cloud Formation stack
- Deploy the Cloud Formation Stack in the Developer account
- Ask for authorization from the Product Owner and, having it, proceed to deploy to Production accounts

Adding more accounts

New accounts can be added anytime, respecting the following:

a) Bootstrapping for CDK and authorization for DevOps account.
b) Changes in the GitLab pipeline definition file, adding more tasks related to new accounts and including those tasks in the corresponding stage.

Future additions

Some improvements could be considered in the future, for example, the validation of CloudFormation and CDK files during the commit and unit testing steps in the CI/CD flow.

References

a) “Building Cross-Account Deployment in GitLab Pipelines Using AWS CDK” in https://aws.amazon.com/es/blogs/apn/building-cross-account-deployment-in-gitlab-pipelines-using-aws-cdk/.
b) “Configure OpenID Connect in AWS to retrieve temporary credentials” in https://docs.gitlab.com/ee/ci/cloud_services/aws/
c) “Deploy to AWS from GitLab CI/CD” in https://docs.gitlab.com/ee/ci/cloud_deployment/
d) “Configure OpenID Connect for GitLab and AWS” in https://www.steynhuizinga.nl/2022/03/configure-openid-connect-for-gitlab-and-aws/
e) “AWS re:Invent 2021 - Automating cross-account CI/CD pipelines” in https://www.youtube.com/watch?v=AF-pSRSGNks&t=60s