Bedrock

Release 3.0

Papina

Apr 21, 2022

CONTENTS

Bedrock is an AWS landing zone generator utilising either CloudFormation or Terraform

Check out the *Usage* section to get your landing zone code

Note: This project is under active development.

CHAPTER

ONE

CONTENTS

1.1 Usage

1.1.1 Quick Start Landing Zone Code

To use Bedrock Landing Zone, first fork/clone the Repo with your preferred CI/CD framework:

GitHub Actions

```
git clone https://github.com/trustypangolin/bedrock-foundation-template
```

BitBucket Pipelines

```
git clone https://bitbucket.org/trustypangolin/bedrock-foundation-template
```

GitLab Pipelines

```
git clone https://gitlab.com/trustypangolin/bedrock-foundation-template
```

1.1.2 Management AWS Account

The landing zone will need the initial AWS account to be created. All other sub accounts are created as part of the CI/CD process

1.1.3 Add The Variables in your Git Secrets

Table 1: Git Secrets/Variables to set

Repository Secrets, Variables	Example	Description
AWS_ROOT_ACCOUNT	111111111111	Your 12 digit AWS Management Account ID
BEDROCK_TF_STATE (Optional)	dGVycmFmb3JtIHsKICBiYWNrZW5kICJzMyIge wogICAgcmVnaW9uICAgICAgICAgPSAiYXAtc2 91dGhlYXN0LTIiCiAgICBkeW5hbW9kYl90YWJ sZSA9ICJiZWRyb2NrLXRmc3RhdGUiCiAgfQp9	Single line Base64 version of the remote_state.tf Note that Key and Bucket are not included.
BEDROCK_TF_VARS (Optional)	dW5pcXVlX3ByZWZpeCA9ICJpbmRpZ29jYXB5Ym FyYSIgIApiYXNlX3JlZ2lvbiA9ICJhcC1zb3V0 aGVhc3QtMiIKcm9vdF9lbWFpbHMgPSB7CiAgIl NlY3VyaXR5IiAgID0gImF3cytiZWRyb2NrLnNl Y0Bkb21haW4iCiAgIlNoYXJlZCIgICAgID0gIm F3cytiZWRyb2NrLnNoYXJlZEBkb21haW4iCiAg IlByb2R1Y3Rpb24iID0gImF3cytiZWRyb2NrLn Byb2RAZG9tYWluIgp9Cm5vdGlmaWNhdGlvbnMg PSB7CiAgYmlsbGluZyAgICA9ICJhd3MrYmVkcm 9jay5iaWxsaW5nQGRvbWFpbiIKICBvcGVyYXRp b25zID0gImF3cytiZWRyb2NrLm9wZXJhdGlvbn NAZG9tYWluIgogIHNlY3VyaXR5ICAgPSAiYXdz K2JlZHJvY2suc2VjdXJpdHlAZG9tYWluIgp9	Single line Base64 version of the terraform.tfvars
ENCKEY	Password123!	Artifacts such as STS credentials are encoded between jobs with OpenSSL, so that non-admins can't access temporary credentials from an artifcats file

1.1.4 Bootstrap your AWS Managment Account

You will need the AWS CLI tools and a local copy of Terraform installed

1. Activate SSO in your preferred region
2. Configure SSO with the preferred IdP (eg AWS/Azure/Google/OKTA).
3. Create and Assign yourself AdministratorAccess permissions via the PermissionsSets to the Management Account
4. Log into the AWS account landing page (http://d-someid.awsapps.com/start)
5. Either grab the temporary keys from the AWS Landing Page and input them into the `~/.aws/credentials` file, or
6. configure your `~/.aws/config` file for SSO and use `aws sso login --profile <your profile>`
7. Ensure the credentials/profile is set as default by setting `export AWS_PROFILE=<your profile>`

Typical `~/.aws/config` file setup

```
[profile bedrock]
sso_start_url = https://d-1234567890.awsapps.com/start
sso_region = ap-southeast-2
sso_account_id = 111111111111
sso_role_name = AdministratorAccess
region = ap-southeast-2
output = json
```

You should now have admin access to the account via SSO, confirmed by running a simple cli command such as `aws organizations list-roots` should return organizations values for the Management account Id

You now need a way for GitHub/GitLab/BitBucket to have access to your new AWS account, there is some terraform files in the /tf folder that will allow you bootstrap the various OIDC and roles required

1. copy the terraform.tfvars.template file to terraform.tfvars and
2. change the values to suit your repo and naming for the OIDC
3. terraform init
4. terraform apply

Your CI/CD process should now be able to assume the basic roles setup if you set the repo values up corectly

1.2 Terraform

1.2.1 OIDC Terraform files

AWS will require the correct OIDC settings depending on your Git provider

The following OIDC tf files have been included, along with associated pipeline setups

1. Gitlab (CI/CD) `gitlab-oidc.tf` and `.gitlab-ci.yml` CI/CD
2. Github (Github Actions) `github-oidc.tf` and `.github` folder with Github Actions
3. Bitbucket (Pipelines) `bitbucket-oidc.tf` and `bitbucket-pipelines.yml` Pipelines

There is some initial bootstrapping involved with Terraform before the pipeline code can takeover the hardwork

1.2.2 Customising the Terraform environment variables

First, open a cli and move into the tf folder

`cd /tf`

copy the `terraform.tfvars.template` to `terraform.tfvars`

`cp terraform.tfvars.template terraform.tfvars`

this file should stay untracked in your repo via `.gitignore`, as it will generally have secret or semi-secret information

1.2.3 Intialise Terraform

Ensure terraform has been installed

rename the git repository tf files that are not utilised to -oidc.tf.disabled, however leaving them as-is will not give additonal access without proper variables

1.3 CloudFormation

1.3.1 Begin OIDC

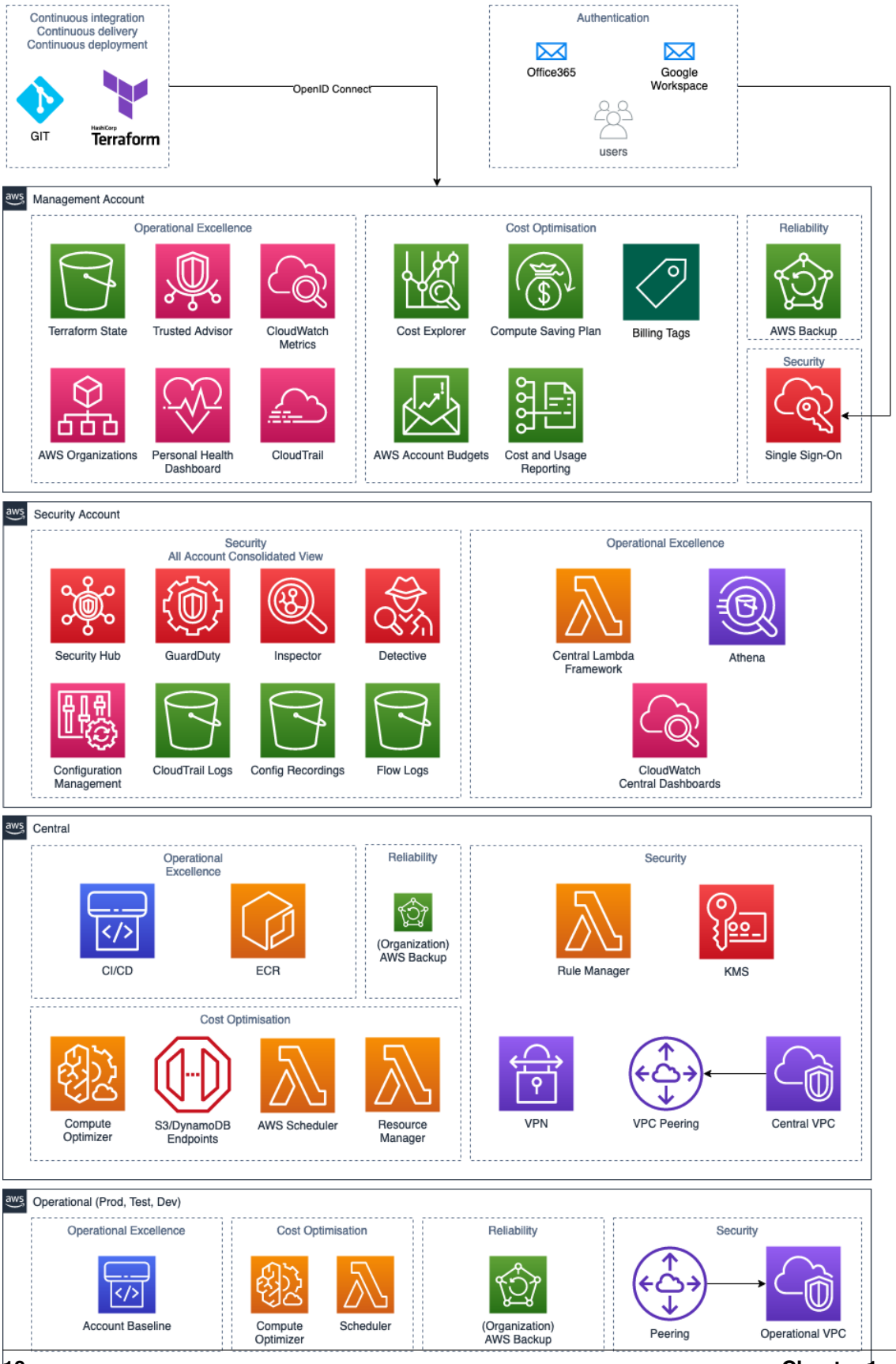
AWS will require the correct OIDC settings depending on your Git provider

The following OIDC setups have been included, along with associated pipeline setups

1. Gitlab (CI/CD)
2. Github (Github Actions)
3. Bitbucket (Pipelines)

1.4 High Level Design

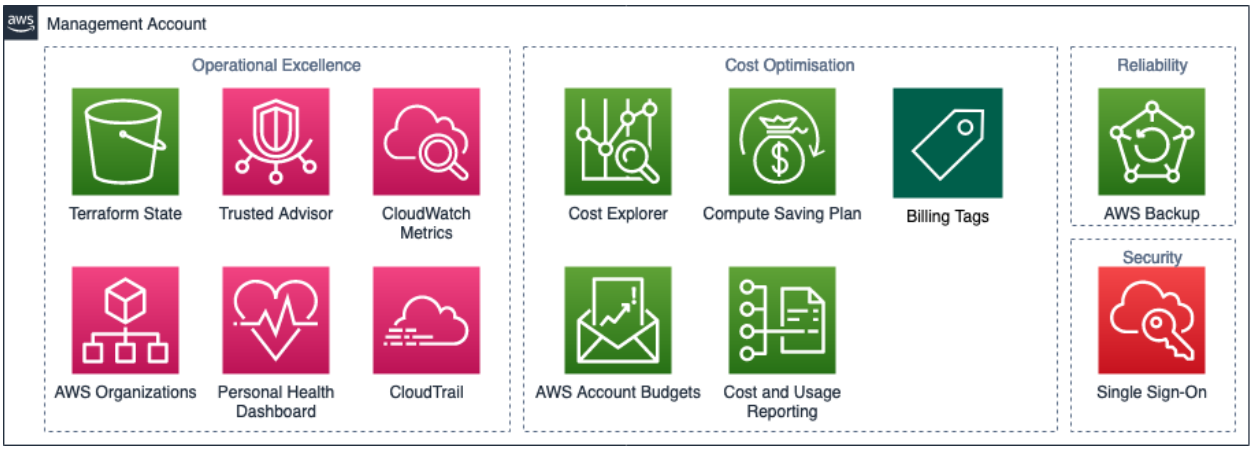
1.4.1 Overall Account Design



Management

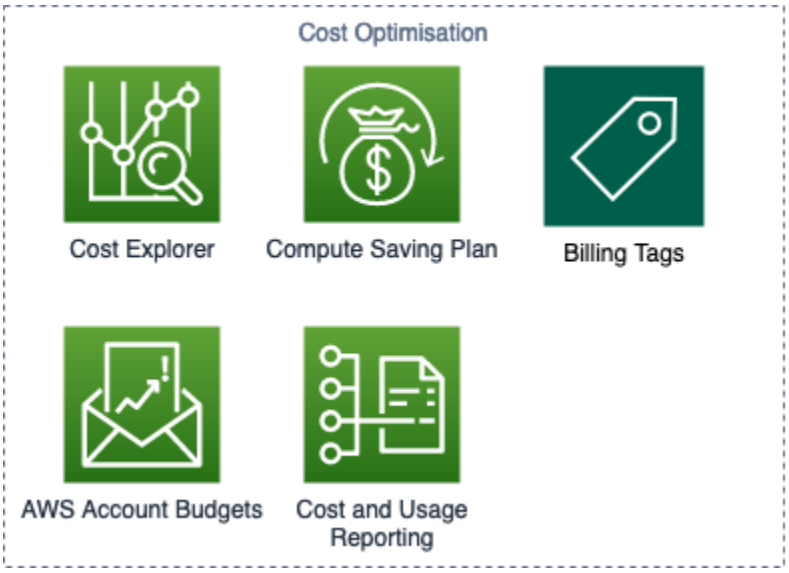
Overall Managment Account Architecture

Management Account Design



Operational Excellence Design

Cost Optimisation Design

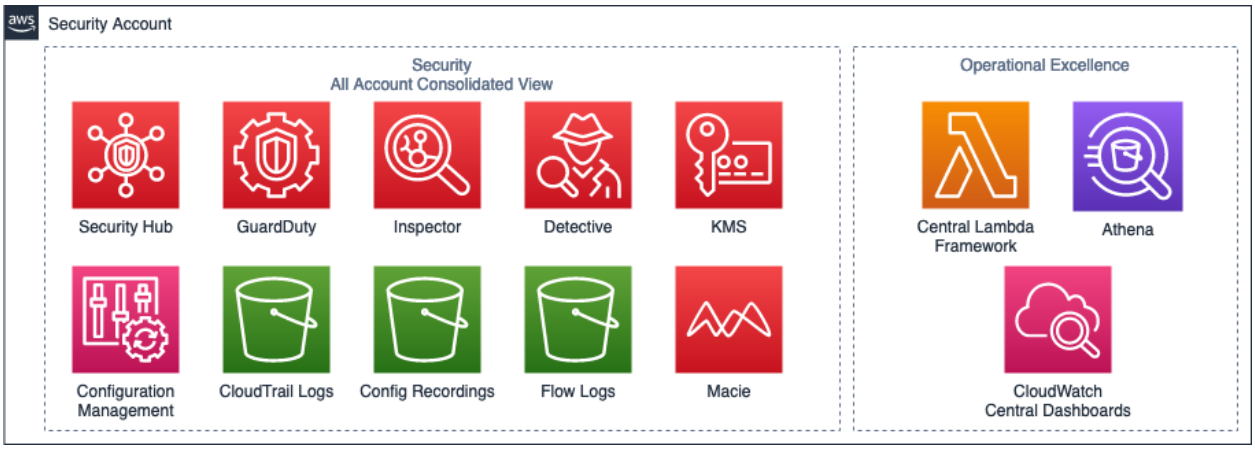


Reliability Design

Security Design

Security

Account Design



Operational Excellence Design

Security Design

Central

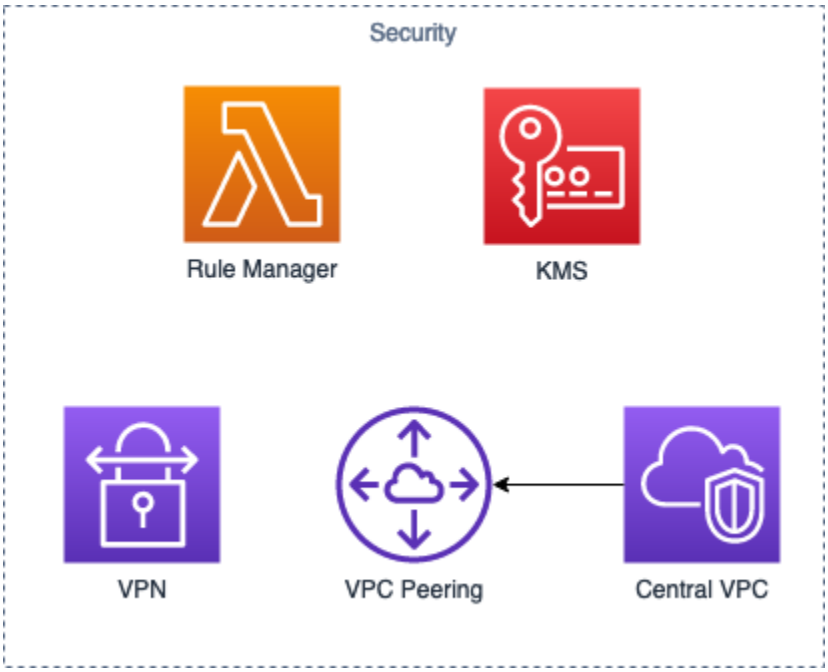
Account Design

Operational Excellence Design

Cost Optimisation Design

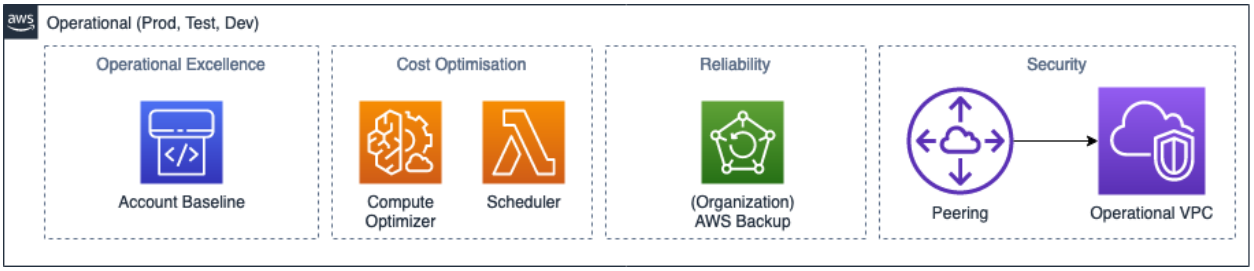
Reliability Design

Security Design

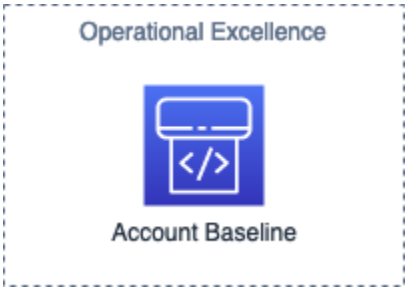


Operational

Account Design



Operational Excellence Design



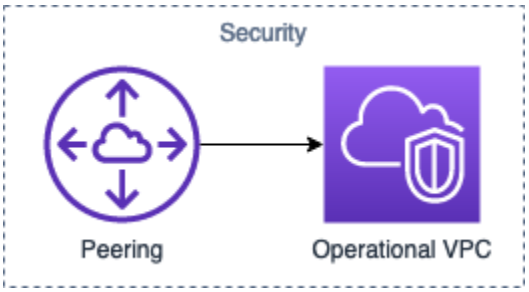
Cost Optimisation Design

Scheduler has a role in these accounts, actual Lambda and CloudWatch schedule is in the Central account

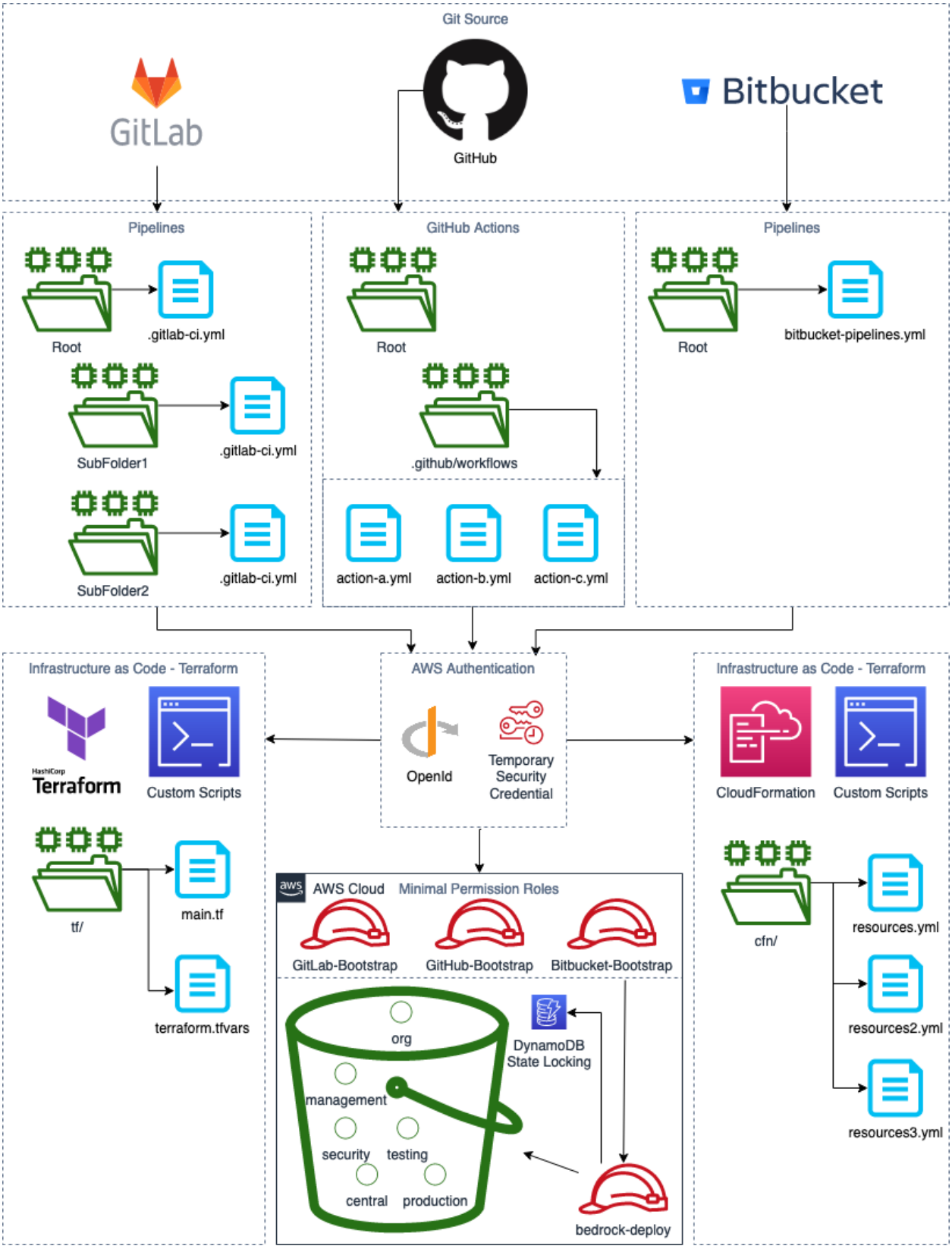
Reliability Design

AWS Backup is set from the Organisation Level, refer to Management-Reliability

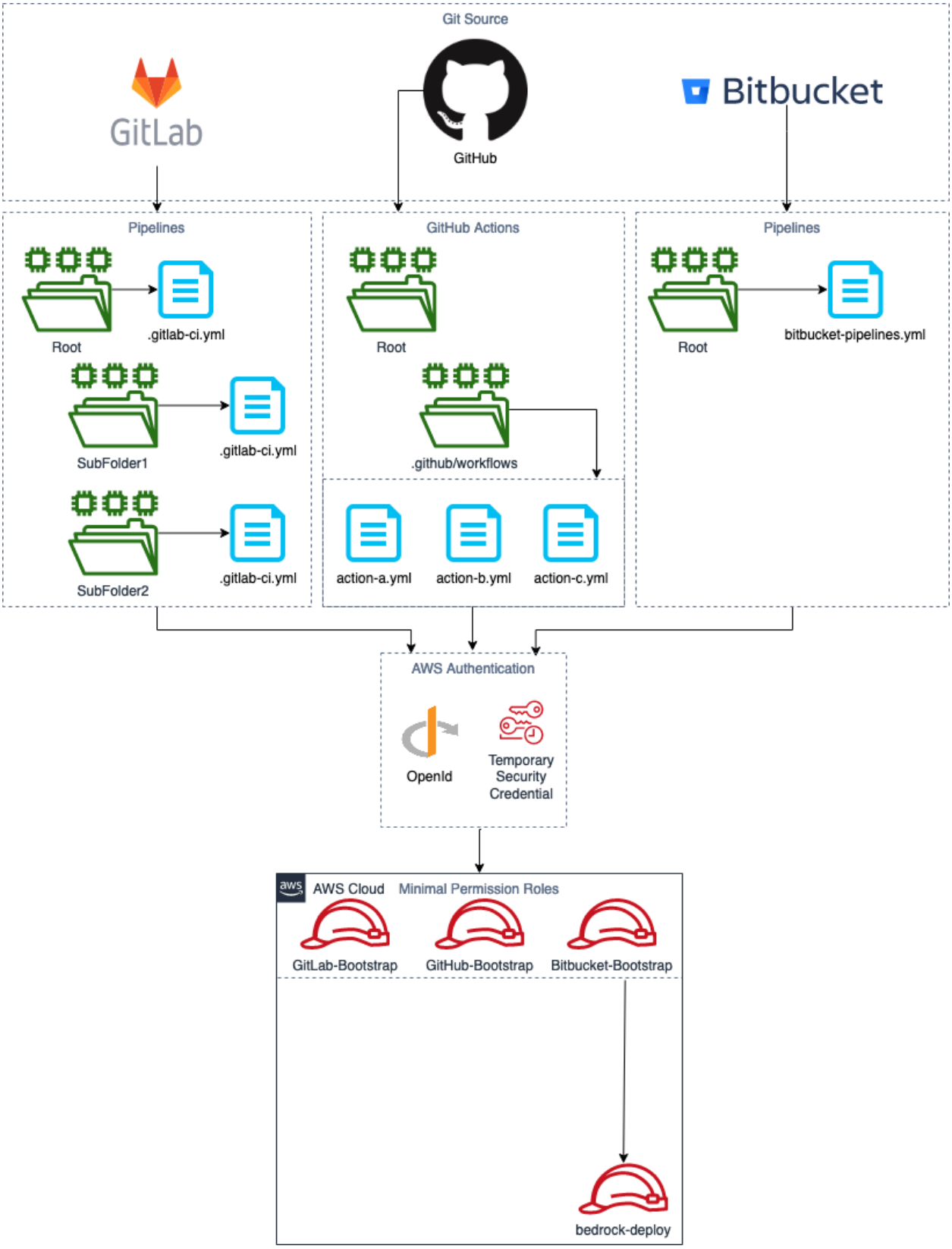
Security Design



1.4.2 Overall CI CD Design

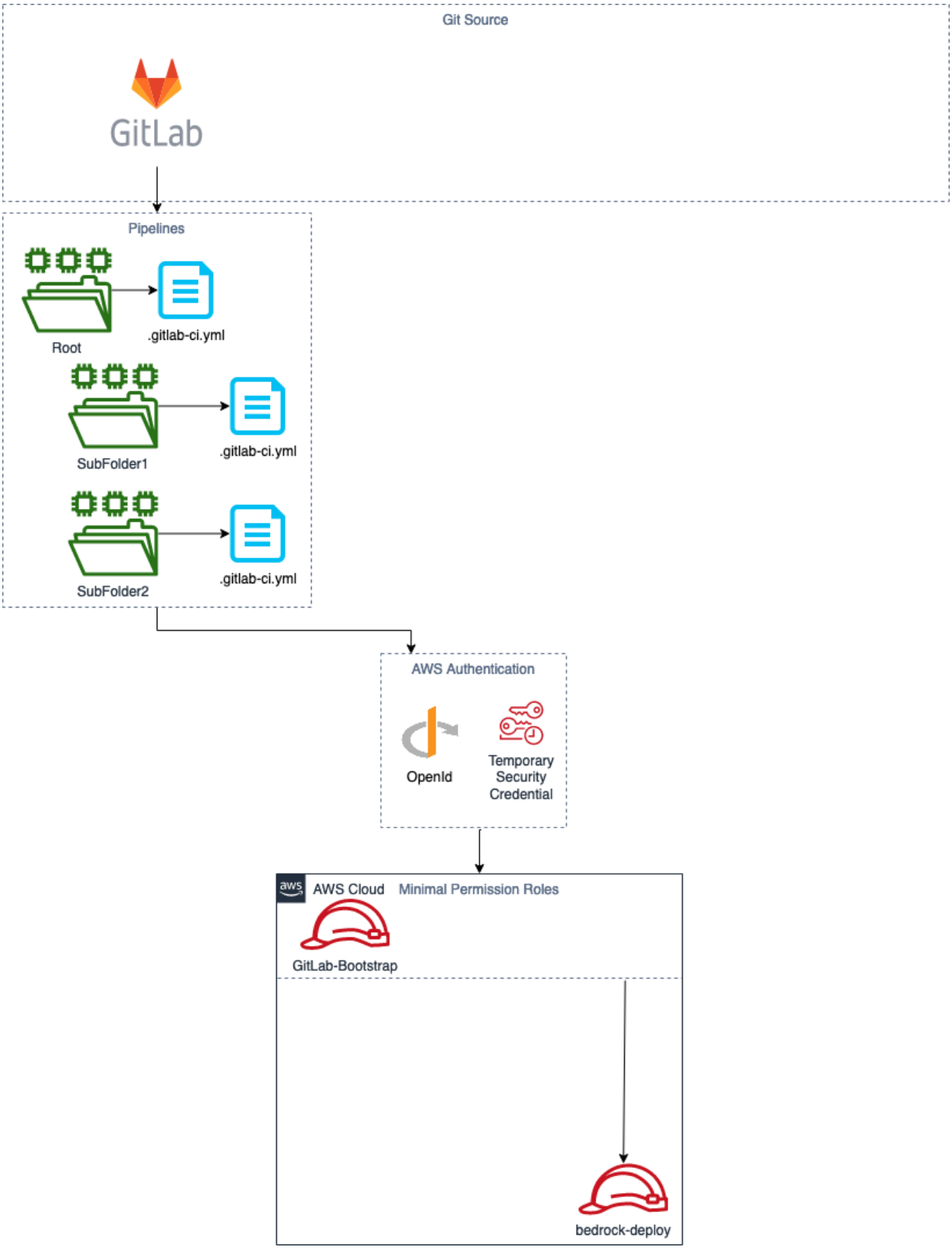


Git Sources

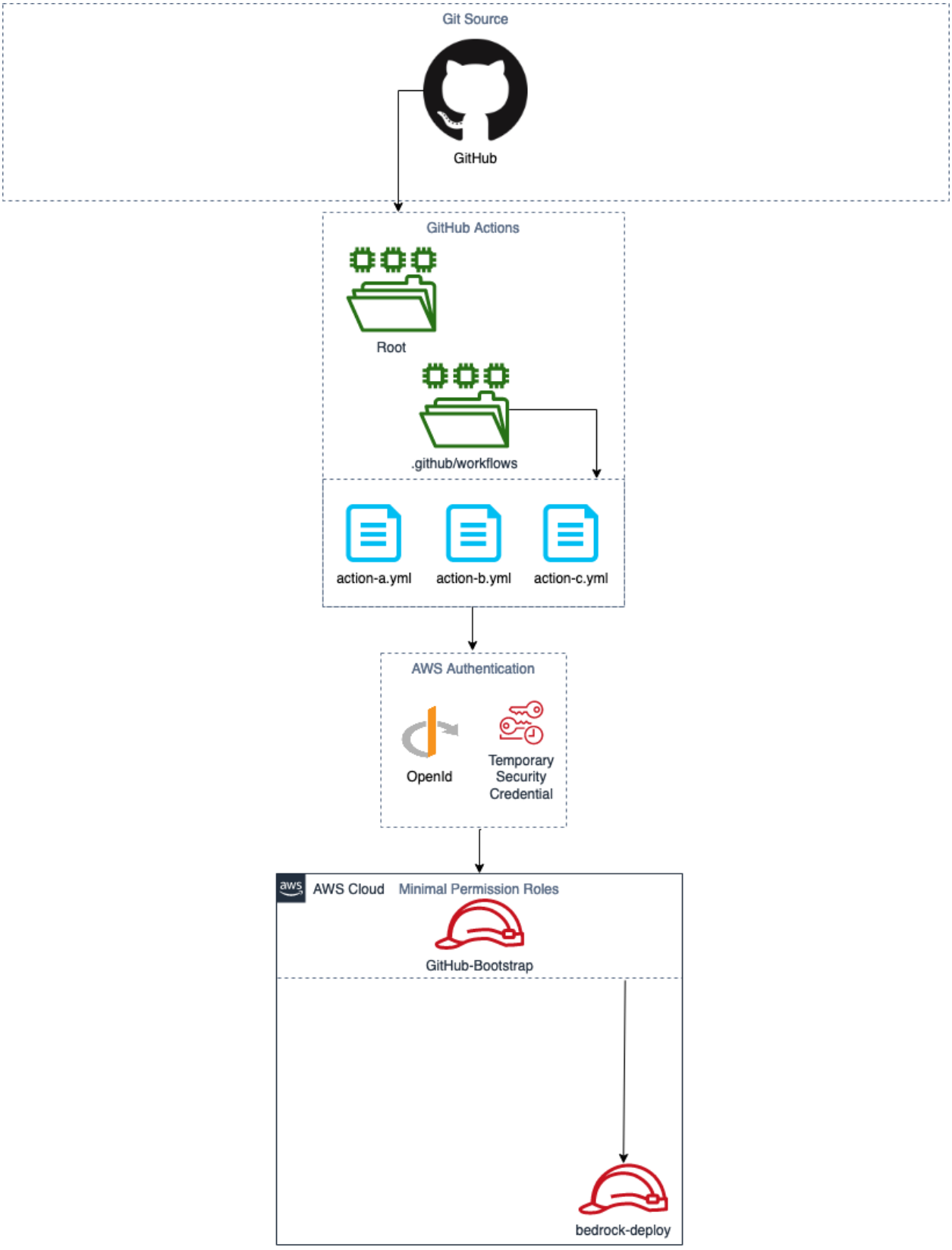


Overall Git Source Architecture

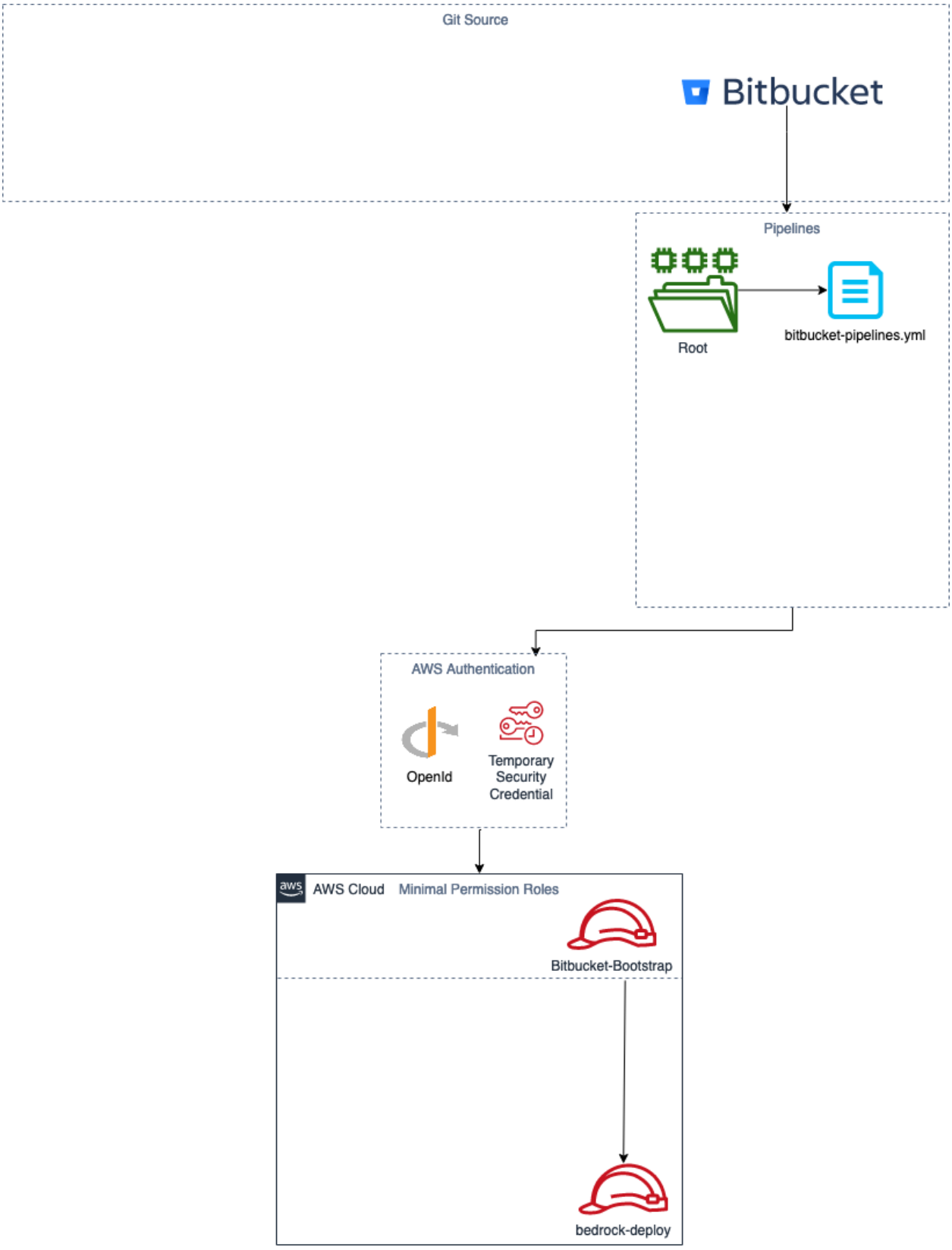
GitLab Design



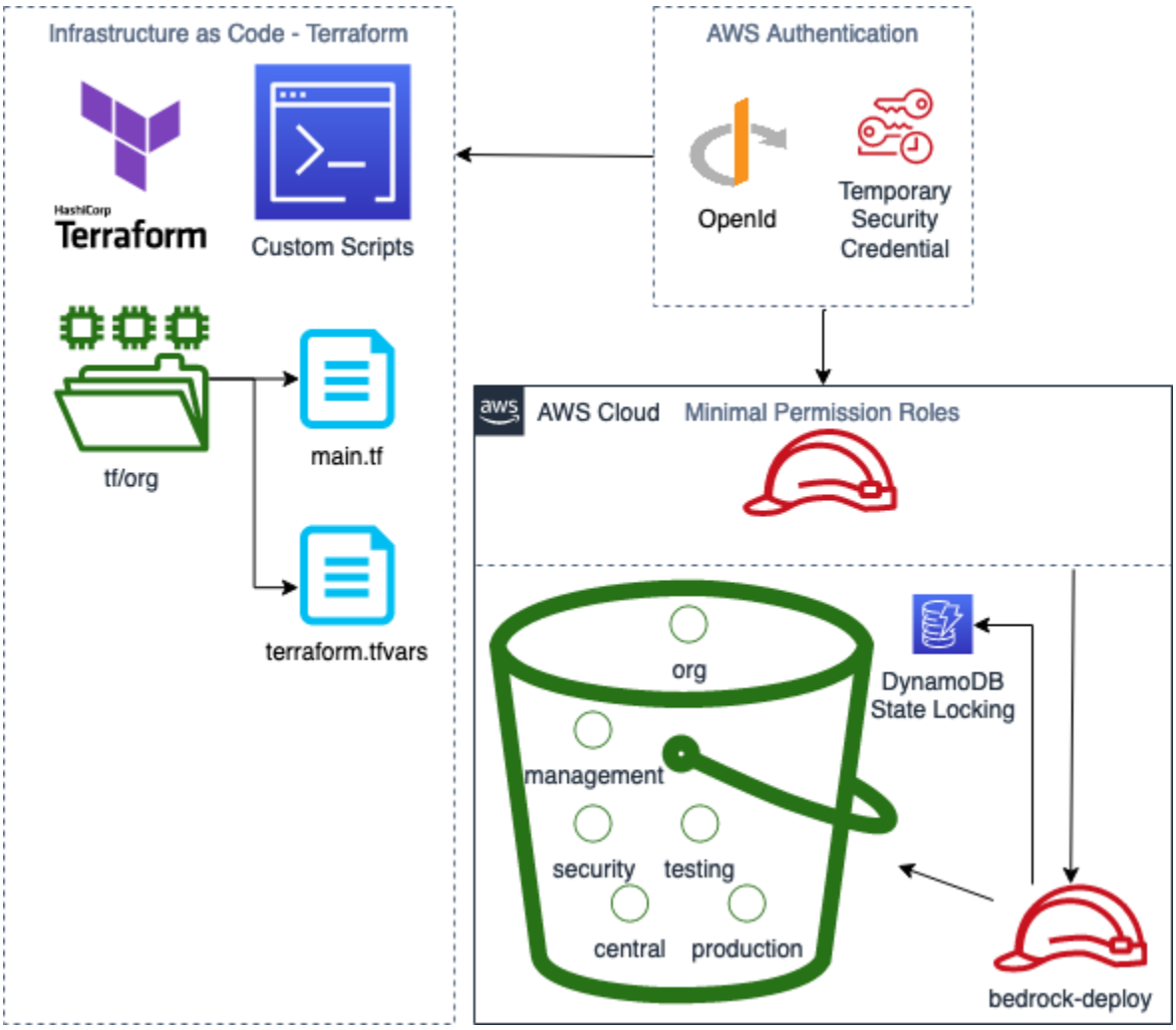
GitHub Design



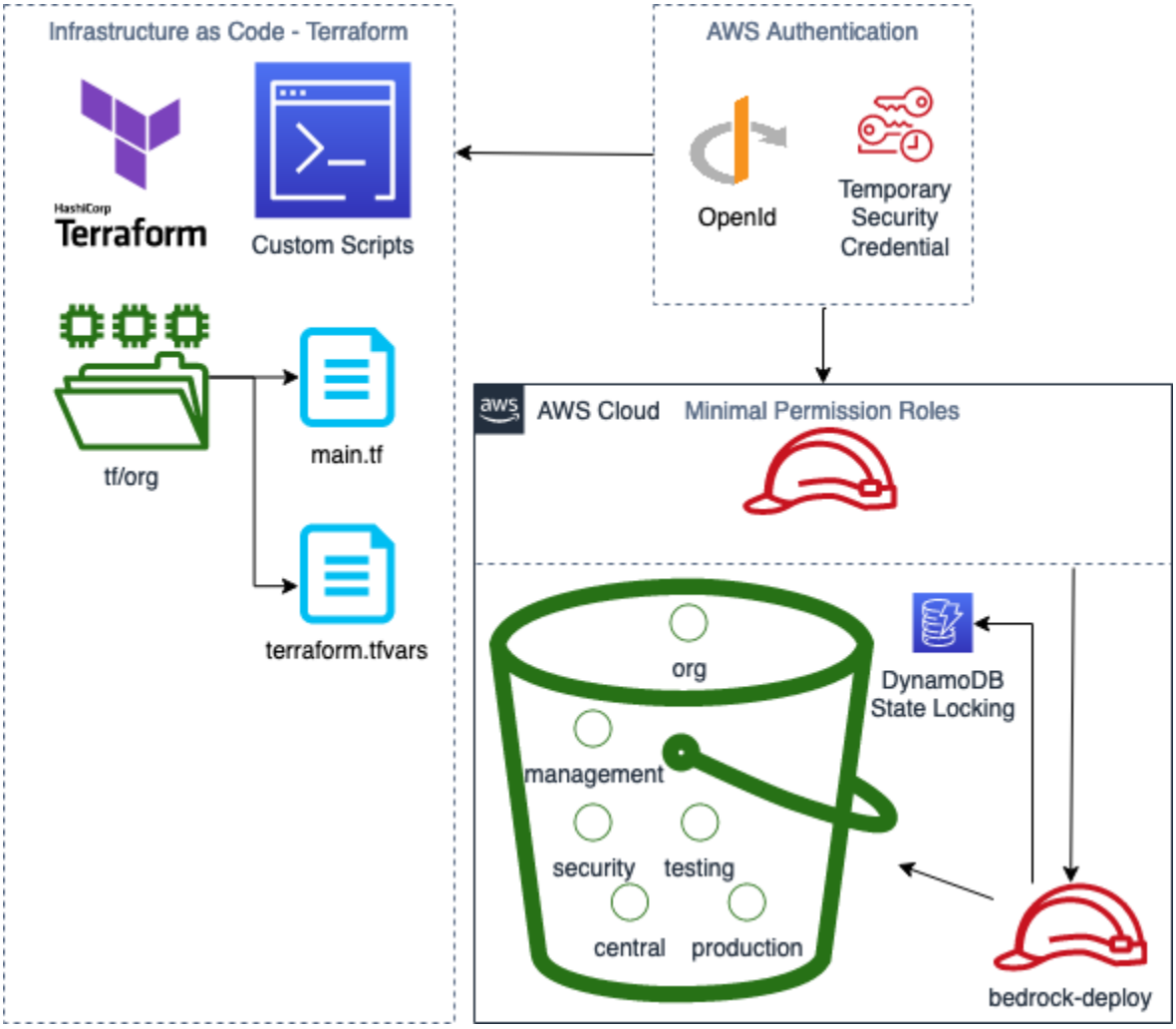
Bitbucket Design



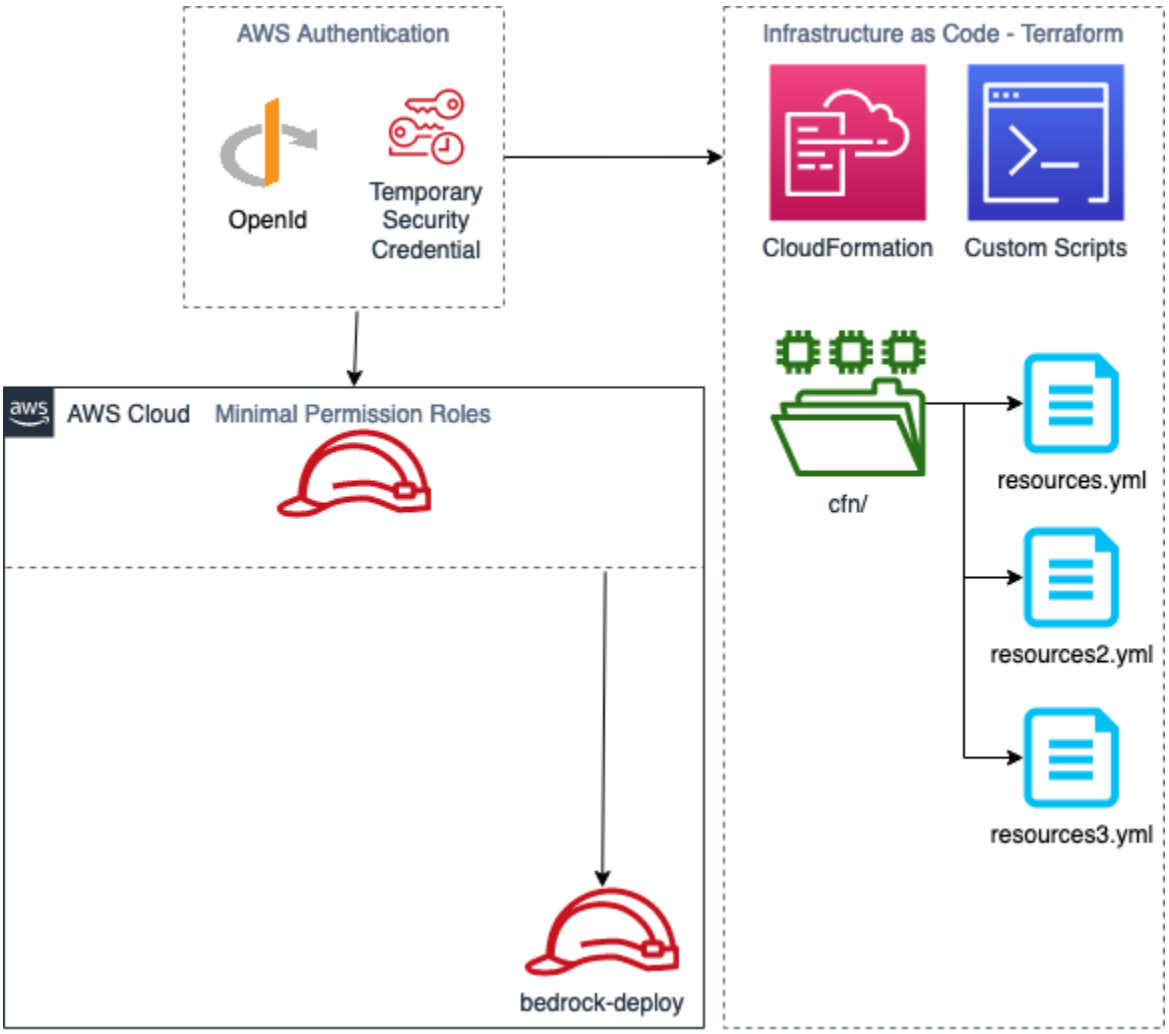
1.4.3 Overall Terraform Design



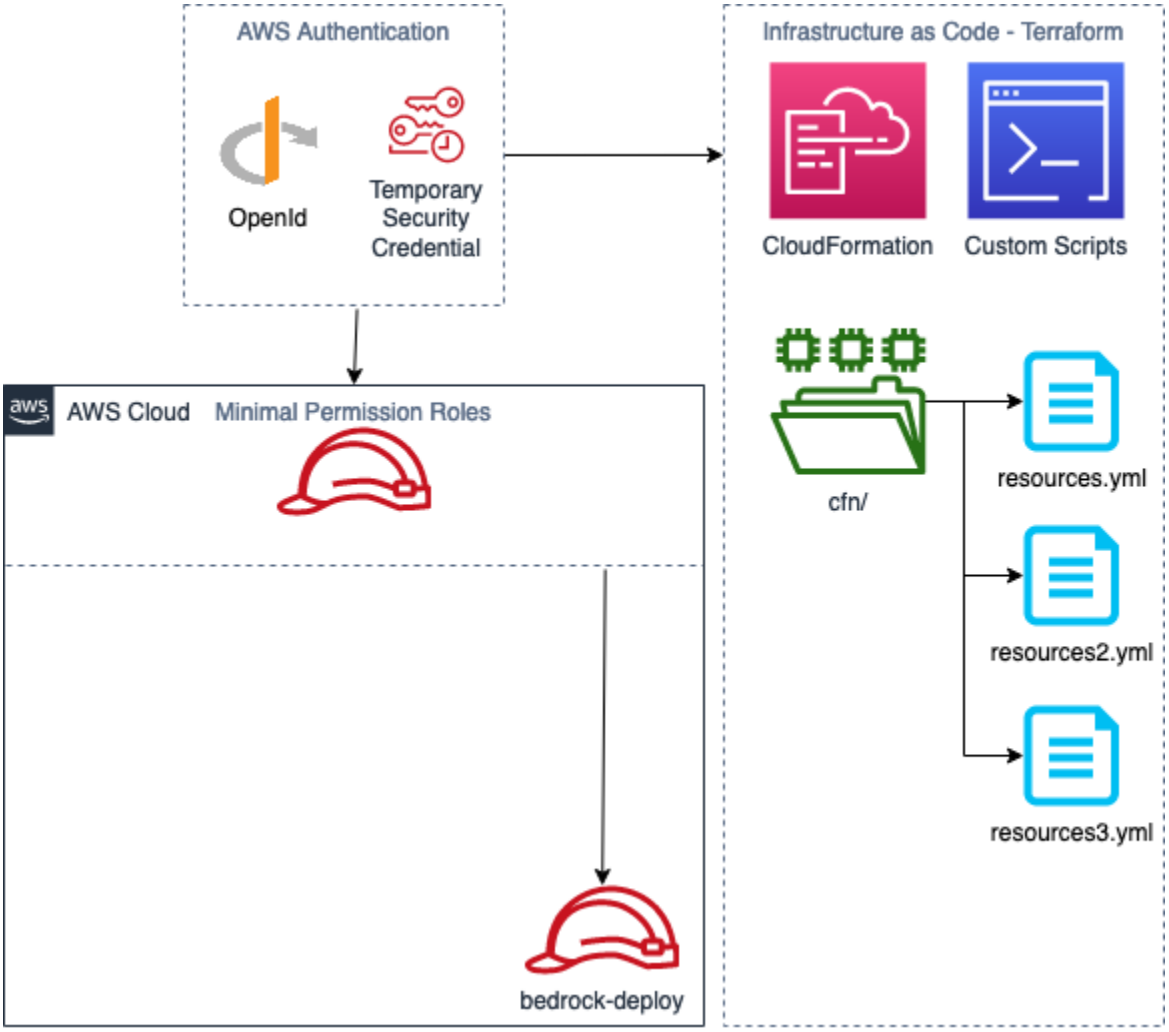
Terraform


Infrastructure as Code - Terraform
HashiCorp
Terraform
Custom Scripts
tf/org
main.tf
terraform.tfvars
AWS Authentication
OpenId
Temporary
Security
Credential
aws
AWS Cloud
Minimal Permission Roles
org
DynamoDB
State Locking
management
security
testing
central
production
bedrock-deploy

1.4.4 Overall CloudFormation Design



CloudFormation



1.5 Frequently Asked Questions

1.5.1 Performance Pillar?

There are no real workloads aside from the VPN and some basic Lambda functions

1.5.2 Sustainability Pillar?

Aside from region selection, No workloads are available to really account for this

Bedrock has its documentation hosted on Read the Docs.