Application Deployment using Terraform

Arun Harimurthy

Topics

High level architecture

Assumptions

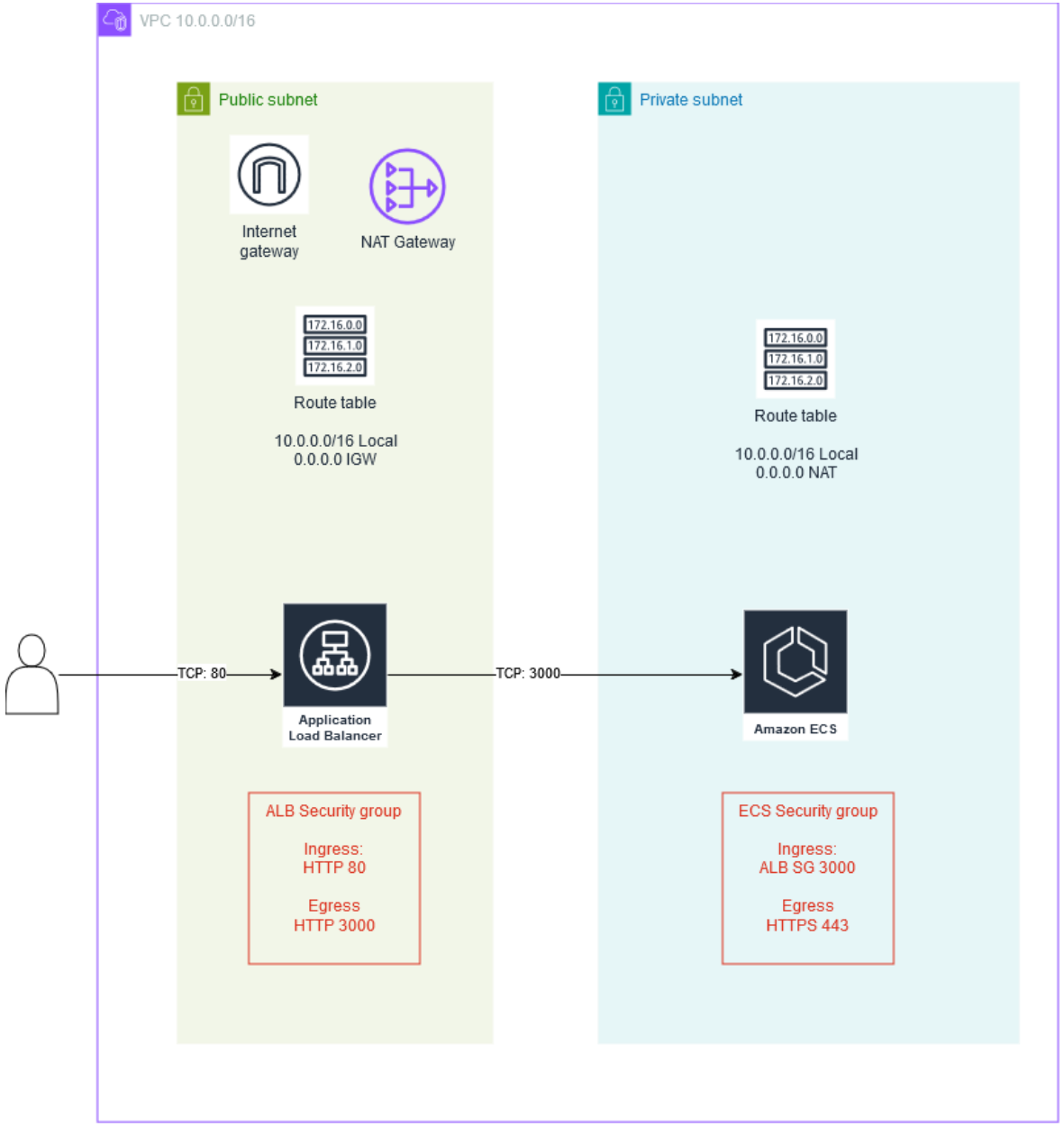
Build

Deploy

Adapt

Future version

High level Architecture



Assumptions

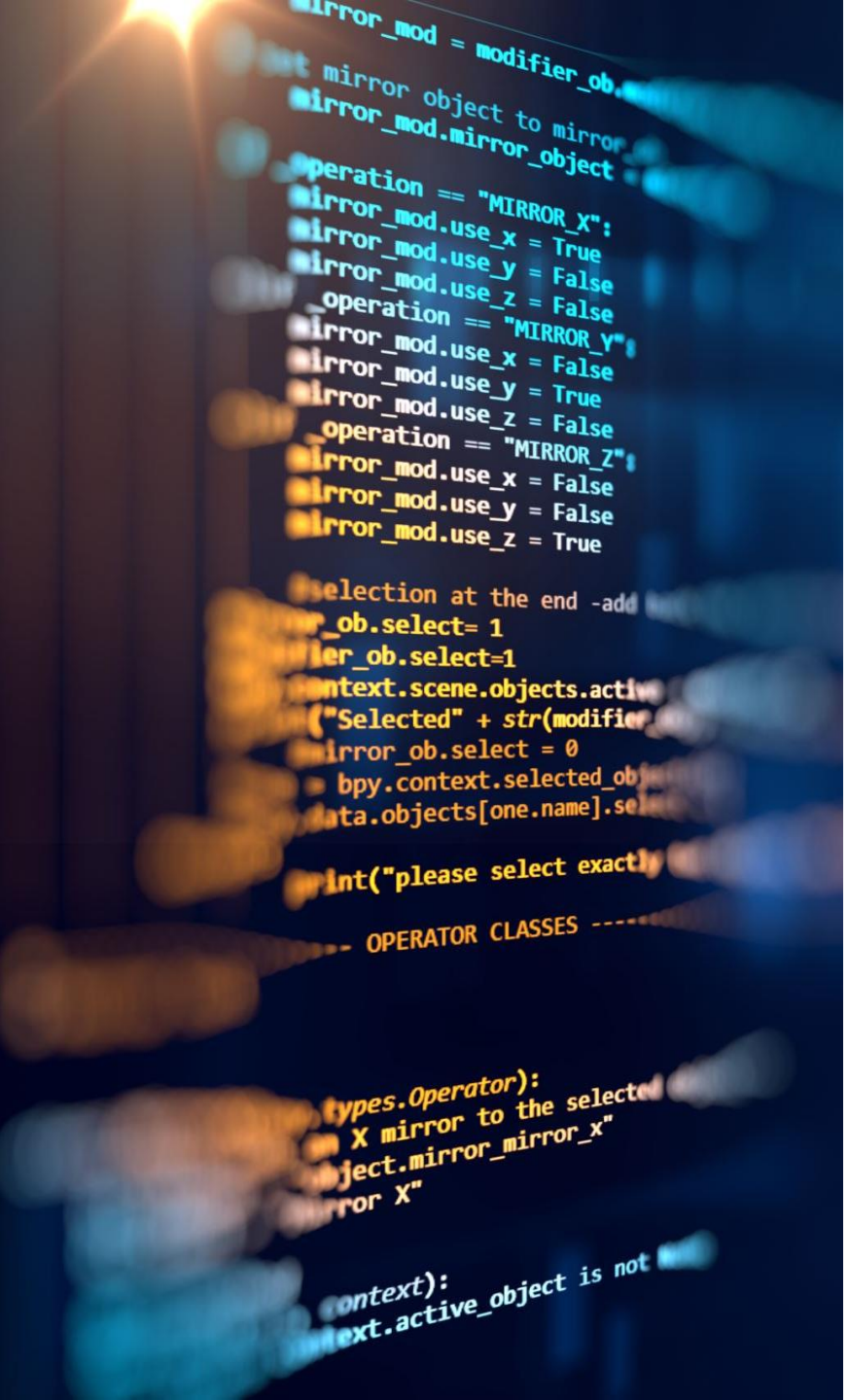
- No SSL provider and DNS resolver
  - Focused data at transit at minimum level
  - Security is enabled at transport layer instead of application layer
- Create AWS setup from scratch
  - Project focused on network and other pre-requisite required for application deployment
- Sample NodeJs application
  - Sample NodeJs application from third party used to demonstrate the infra deployment
  - Sample application is compatible with node 16, end-of-life exempted for application to run.
  - The code is used as-is, no static scanning performed

Build

- Application code is sourced from https://github.com/andrewagain/calculator and node 16 required for application to start
- After installing dependencies npm install and npm start command will start the application
- AWS public node 16 docker image used to build the app image.
- The image can be used to test the application locally

Deploy

- Application image stored in AWS ECR registry with data at rest encryption. Image scanning performed in ECR.
- Terraform script used to deploy the infrastructure
- Script deploys network components for the application
- ECS task definition created, and task executed as service
- The components protected by security groups. Minimum privileges provided to ECS, to run task and get image from ECR
- Application can be accessed via LB URL, user IP needs to be whitelisted



Adapt

- New application feature in same source code
  - Updated code can be deployed with same process by creating new image with latest tags.
  - The code will perform rolling deployment by creating new version of task.

- New feature as separate microservice
  - Separate ECS task can be created with respective target groups and re-write url in loadbalancer

- WAF in Loadbalancer
  - Token authentication can be included in WAF for each requests
  - Sensitive URLs can be whitelisted to specific Origin

Future version

- Necessary :
  - Data at transit encryption with help of SSL certificates and DNS
  - Enable logging in load balancer, vpc flow logs and store the logs in cloudwatch logs
  - Federated sign in Application load balancer

- Different approach:
  - Application in EC2 instead of ECS
  - Token authentication in WAF

Questions