Terraform Overview

This was based on the recording and training session Video

- Documentation
  - Infrastructure as Code (IaC)
  - Secrets Management
  - CI/CD Process
  - Kubernetes Deployments
  - Access Control and Collaboration
- 1. Deploy Repository Overview
- 2. Key Infrastructure Components
- 3. Secrets Management
- 4. CI/CD Pipeline
  - Gitlab CI/CD Pipeline
  - Build Process
  - Deployment
- 5. Helm Charts Structure
- 6. Access Control
- 7. Kubernetes Cluster Management
- Future Improvements

Documentation

Infrastructure as Code (IaC)

Raft AI uses Terraform to manage the majority of its infrastructure. The main repository for this is the "deploy" repository, which contains Terraform configurations for various components including databases, message queues, and supporting services.

Infrastructure as Code Setup and Guide

Secrets Management

Secrets are generated automatically and stored in the Terraform state file. They are then created as Kubernetes secrets during the Terraform apply process. This approach ensures that no sensitive information is stored in the codebase itself.

CI/CD Process

The CI/CD pipeline is managed through GitLab CI/CD. Each microservice repository contains files that defines the pipeline stages. Common job definitions are stored in templates in the deploy repository.Key features of the CI/CD process include:

- Use of Kaniko for Docker image building
- Deployment of review environments for feature branches
- Auto-cleanup of inactive review deployments

Kubernetes Deployments

Helm charts are used for deploying microservices to Kubernetes clusters. The charts are structured with templates, values files for different environments, and a Chart.yaml for metadata.

Access Control and Collaboration

Access to the Terraform state is restricted to the DevOps admin team. Atlantis is used to facilitate collaborative Terraform workflows, allowing team members to propose and review infrastructure changes.

1. Deploy Repository Overview

The deploy repository is the central location for most of the Terraform code that manages the infrastructure. Key points include:

- Contains Terraform files for various infrastructure components
- Houses special folders not directly related to Terraform
- Manages approximately 99% of the infrastructure deployment

2. Key Infrastructure Components

The deploy repository includes configurations for:

- Helm charts (Logging: Matamo, Prometheus, VPN (Pritunl))
- Certificate manager (using cert-manager and Let's Encrypt)
- Cloud SQL (managed in sql.tf, containing about 90% of SQL-related configurations) https://vectorai.atlassian.net/wiki/spaces/DevOps/pages/679411722
- DNS configurations
- Additional Kubernetes clusters (2 per Project)
  - Composer
  - Kafka cluster
  - Atlantis
  - Airbytes
- RabitMQ
- Redis
- Logging configuration
- VPN
- ML-related infrastructure
- MongoDB instances
- Networking (load balancers, NAT gateways)

3. Secrets Management

Any secret required by the users and services are created via terraform and stored in the K8s.

- Secrets are not stored in the code but generated automatically
- Stored in the Terraform state file, which requires special access
- Created as Kubernetes secrets during Terraform runs
- Used by applications via environment variables or mounted volumes

4. CI/CD Pipeline

Gitlab CI/CD Pipeline

- Controlled from `.gitlab-ci.yml` , is each microservice.
- Uses templates from the deploy repository
- Stages include build, check, test, and deploy

- Atlantis is used for collaborative Terraform workflows

Build Process

- Uses Kaniko for building Docker images without running Docker-in-Docker
- GitLab runners operate in the staging cluster

Deployment

- Helm charts are used for deploying microservices
- Deployments create separate environments for feature branches
- Auto-stop functionality for inactive review deployments after one week

5. Helm Charts Structure

- Templates folder contains the core logic
- Values files for different environments
- Chart.yaml for metadata
- Helm keeps state in Kubernetes secrets

6. Access Control

- Terraform state stored in GCP bucket
- Limited access to state bucket (devops/admin team)
- Atlantis used for collaborative Terraform workflows
- Service account keys required for running Terraform locally (allocated per user from the admin team)

7. Kubernetes Cluster Management

- Dynamic scaling of nodes and pods based on metrics
- Namespaces used for organization (moving towards per-deployment namespaces)
- GitLab runner namespace for CI/CD jobs

Future Improvements

- Moving towards using separate namespaces for each deployment to improve organization and cleanup processes
- Ongoing refinement of auto-scaling configurations to optimize resource usage and costs
- Considering separating certain components (e.g., DNS) into their own repositories
- Ongoing refinement of the infrastructure organization

This overview provides a high-level understanding of the deployment infrastructure and CI/CD processes at Vector/Raft AI. For more detailed information on specific components or processes, further investigation or consultation with the DevOps team may be necessary.