secret-mgt-proposal

Secret Management in Kubernetes on AWS

Introduction

This document suggests using AWS Secrets Manager and AWS Systems Manager Parameter Store to enhance secret management in the Opsfleet Kubernetes environment on AWS. It describes the advantages of each service and explains how they can be combined with the existing EKS and Terraform infrastructure.

Current Problems

- **Security Risks of Storing Secrets in Configuration Files**: Storing sensitive data, such as database passwords, in configuration files alongside code in GitHub can pose significant security risks.
- **Complexity and Exposure with Environment Variable Management**: Managing secrets through environment variables adds complexity and increases the potential for exposure.
- **Challenges of Self-Hosted Solutions**: The team's limited capacity makes it impractical to maintain and secure self-hosted secret management systems.

Proposed Solutions

1: AWS Secrets Manager

Benefits:

- **Automatic Secret Rotation**: Automatically rotates secrets to reduce the risk of using outdated or compromised credentials.
- **Detailed Access Control**: Provides detailed policies to define who can access or manage specific secrets, improving security.
- **Secure Storage**: Encrypts secrets both at rest and in transit to protect sensitive data.
- **Direct Integration with AWS Services**: Seamlessly integrates with other AWS services and EKS, simplifying the architecture.
- **Audit and Compliance**: Integrates with AWS CloudTrail to audit access and changes to secrets, helping with compliance monitoring.

Integration with Kubernetes:

- **Kubernetes External Secrets**: Use this tool to automatically copy secrets from AWS Secrets Manager to Kubernetes secrets.
- **IAM Roles for Service Accounts (IRSA)**: Give specific AWS permissions to pods in EKS without managing credentials directly.
- **Deployment Update**: Modify Kubernetes deployments to use Kubernetes secrets instead of fixed environment variables.

2: AWS Systems Manager Parameter Store

Benefits:

- **Cost-Effectiveness**: A cost-friendly choice for startups, particularly when there are few secrets and requests.
- **Hierarchical Storage**: Arrange configuration data and secrets in a hierarchy to simplify management at scale.
- **Secure and Simple**: Offers encryption for stored data and a user-friendly interface for managing fewer or simpler secrets.
- **AWS Service Integration**: Seamlessly integrates with EKS and other AWS services, ensuring a unified cloud environment.
- **Access Control and Auditing**: Supports permissions and monitoring via AWS IAM and CloudTrail.

Integration with Kubernetes:

- **Kubernetes External Secrets**: Syncs Parameter Store parameters into Kubernetes secrets, similar to Secrets Manager.
- **IRSA**: Enables secure and granular access to the Parameter Store from Kubernetes pods.
- **Deployment Modification**: Updates application deployments to consume secrets as Kubernetes secrets.

3: HashiCorp Vault Cloud Service

Benefits:

- **Managed Service**: Provides a fully managed Vault environment, reducing operational overhead.
- **Dynamic Secrets**: Generates on-demand, time-limited credentials.
- **Encryption as a Service**: Offers encryption capabilities for sensitive data.
- **Advanced Access Control**: Allows detailed policy creation and enforcement.
- **Audit Logs**: Maintains comprehensive logs for access and usage of secrets.

Integration with Kubernetes:

- **Setup Vault Agent Injector**: Automatically injects secrets into Kubernetes pods using the Vault Agent sidecar.
- **Kubernetes Authentication**: Configures Vault to authenticate pods using Kubernetes service account tokens.
- **Secure Configuration**: Ensures secure connectivity between EKS and Vault, often over TLS.

Execution Strategy

1. **Evaluation**: Conduct a thorough assessment of current and future secret management needs.
2. **Prototype**: Implement a small-scale prototype with both AWS Secrets Manager and Parameter Store.
3. **Policy Definition**: Define IAM policies and roles for secure access to secrets.
4. **Integration**: Update Kubernetes deployments and CI/CD pipelines to use the new secret management system.
5. **Monitoring and Auditing**: Establish procedures for regular monitoring and auditing of secret access and usage.

Summary

Transitioning to AWS Secrets Manager or AWS Systems Manager Parameter Store for secret management will significantly enhance the security and efficiency of Opsfleets Kubernetes environment. Both options offer robust features and integrate seamlessly with AWS services, providing a scalable, secure, and manageable approach to handling sensitive data.