Secure Development Practices
Last updated: October 2024

1. Introduction

This document outlines the secure development practices to ensure that security is integrated into every software development lifecycle (SDLC) stage.

2. Security Practices

2.1 Version Control

We use a version control system where all changes are managed. Each commit to a branch includes a detailed description, making changes transparent and understandable to all team members. We follow a trunk-based development approach, allowing for frequent releases of properly tested and secure features into production. Our CI/CD pipelines are robust, with security checks integrated throughout the process. Before a feature moves to production, it must pass all stages in the pipeline.

- **SAST (Static Application Security Testing)**: Automated checks to identify security vulnerabilities in the source code.
- **DAST (Dynamic Application Security Testing)**: Testing the running application for security flaws.
- **Dependency and License Scanning**: The pipeline includes checks for vulnerable dependencies and license compliance before deployment.
- **Secret Management**: All sensitive information, such as database secrets, API keys, and passwords, is securely stored in environment variables using AWS SSM Parameter Store.

2.2 Multi-Environment and Access Control

- **Multi-Environment Setup**: We use separate AWS accounts for development, testing, and production environments, ensuring complete isolation between environments. Developers have access only to the development environment and are restricted from accessing production resources.
- **Least Privilege**: Access controls follow the principle of least privilege, granting team members only the permissions they need to perform their jobs.
- **Multi-Factor Authentication (MFA)**: MFA is enforced for all IAM users accessing AWS accounts to provide an additional layer of security.

2.3 Secrets and Configuration Management

- **AWS SSM for Secret Management**: All secrets and sensitive information are stored securely in AWS SSM. Environment-specific variables are managed through separate library groups within the pipeline.
- **Access Control on Pipelines**: Developers only have access to the development environment. Environment variables for production are restricted to ensure confidentiality and security.
- **Infrastructure as Code**: Secure deployment is achieved using infrastructure as code (IaC) tools, ensuring that all infrastructure changes are repeatable, documented, and auditable.

2.4 Security and Monitoring

- **AWS Monitoring**: Monitoring is enabled across all AWS environments to track security events and anomalies in real time.
- **AWS Best Practices**: Our architecture follows AWS security best practices, including encryption, logging, and IAM policies.

2.5 Testing and Release Management

- **Frequent Releases**: New features are tested thoroughly before being released. No code goes to production without passing all stages in the CI/CD pipeline, which includes security, functional, and regression tests.
- **Testable Features**: Every feature is fully testable before reaching production, with a dedicated test environment that mirrors the production setup.

3. Review and Updates

These practices must be reviewed and updated annually or when significant changes occur to the development environment, security landscape, or organizational structure.

4. Reference

Documents
Secure Development Policy