SECURITY CONTROLS IN SHARED SOURCE CODE REPOSITORIES

BEST PRACTICES FOR IMPLEMENTING SECURITY CONTROLS

CLAY LANKFORD

7/17/2024

INTRODUCTION TO SHARED SOURCE CODE REPOSITORIES

- Shared source code repositories are platforms that allow multiple developers to collaborate on codebases.
- Popular uses include:
  - Version Control
  - Collaborative Coding
  - Project Management
- Example include: GitHub, GitLab, and Bitbucket.

WHY SECURITY IN SOURCE CODE REPOSITORIES IS CRUCIAL

- Insecure repositories can lead to unauthorized access, data breaches, intellectual property theft, injection of malicious code, and compliance issues.
- The impact of such breaches includes financial loss, reputational damage, and operational disruptions, making security a top priority.

ACCESS CONTROL MANAGEMENT

- Principle of Least Privilege
  - Grant access only to users who need it and regularly review permissions. Implement role-based access control (RBAC) to assign roles based on job functions, using predefined roles in platforms.
  - Enhance security with multi-factor authentication (MFA) for user logins.

CODE REVIEW AND APPROVAL PROCESSES

- Mandatory Code Reviews
  - Conduct peer reviews to catch vulnerabilities early and use automated tools for static code analysis.
  - Enforce branch protection rules to require pull request approvals and status checks before merging.
  - Use GPG keys to sign commits and verify code authenticity.

MONITORING AND AUDITING

- Audit Logs
  - Maintain detailed logs of repository activities and regularly review them for suspicious actions.
  - Integrate automated security scanning tools like Dependabot and Snyk for continuous monitoring.
  - Set up alerts for critical changes or access attempts.

SECURE DEVELOPMENT PRACTICES

- Secure Coding Guidelines
  - Follow industry standards such as OWASP for secure coding. Provide regular training for developers.
  - Avoid hardcoding secrets in the codebase by using secrets management tools like HashiCorp Vault or AWS Secrets Manager.
  - Keep dependencies updated with management tools.

DATA ENCRYPTION AND BACKUP

- Encryption
  - Encrypt data both at rest and in transit using strong algorithms.
  - Regularly perform backups of repositories and store them securely.
  - Test recovery procedures to ensure data integrity and availability.

COMPLIANCE AND LEGAL CONSIDERATIONS

- Regulatory Compliance
  - Ensure repository practices align with regulations like GDPR and HIPAA.
  - Manage intellectual property properly, respecting licenses and third-party code usage.
  - Maintain compliance with legal requirements to avoid penalties.

RESOURCES

- https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/
- https://docs.github.com/en/code-security/supply-chain-security/end-to-end-supply-chain/securing-accounts
- https://docs.gitlab.com/ee/user/application_security/