Best Practices for Implementing Security Controls in Shared Source Code Repositories

Tazmin Somerville

CSD-380 DevOps – Module 11

May 05 – 11, 2025

Why Secure Source Code Repositories?

- Central hub for intellectual property and collaboration
- Target for attackers and insider threats
- Regulatory and compliance requirements

Centralize Security Tools & Libraries

- Store pre-approved security libraries (auth, encryption, logging) in the repository
- Make security tools easy to find and reuse
- 'Putting our information security toolchain and approved libraries there makes it much easier to influence the daily work of Dev and Ops, because anything we create is available, searchable, and reusable.' — Kim, Gene, et al.

Enforce Access Controls

- Use role-based access controls (RBAC)
- Limit access to sensitive code and resources
- Require two-factor authentication (2FA) for all users
- 'Limiting user access to your source code is crucial to prevent unauthorized modifications and leaks.' — Assembla

Automate Security Scanning

- Integrate static code analysis and vulnerability scanning in CI/CD
- Tools: SonarQube, Checkmarx, etc.
- Detect and fix issues early in the development lifecycle

Secure Secrets Management

- Never store passwords, API keys, or secrets in code
- Use secret management tools (Vault, Keywhiz, etc.)
- Encrypt sensitive data at rest and in transit

Standardize Security Configurations

- Provide secure base images, cookbooks, and OS/database configs
- Include recommended settings for logging, authentication, and encryption
- Use container registries with signed and hashed images

Monitor & Audit Changes

- Enable logging and alerting for repository changes
- Regularly review access logs and code modifications
- Collaborate across teams to respond to incidents

Train & Support Teams

- Offer security training for Dev and Ops
- Provide documentation and support for using security tools
- Review and guide teams, especially with new tools and libraries

Summary & Key Takeaways

- Centralize and standardize security controls
- Enforce strong access and secrets management
- Automate, monitor, and continuously improve
- Foster a security-aware culture across all teams

Sources:

- Kim, Gene, et al. *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations*, 2nd ed., IT Revolution Press, 2021.
- "Source Code Security Best Practices." Assembla, https://get.assembla.com/blog/source-code-security
- "Secure Code Repositories: Best Practices." Harness DevOps Academy, https://www.harness.io/harness-devops-academy/secure-code-repositories-best-practices
- "Quickstart for Securing Your Repository." GitHub Docs, https://docs.github.com/en/code-security/getting-started/quickstart-for-securing-your-repository
- "Source Code Security Best Practices." Stop Source Code Theft, https://www.stop-source-code-theft.com/source-code-security-best-practices/