Security Controls in Shared Source code Repositories

James Cortes, CSD-380

Why Security in Shared Repos Matters

- Shared Code repositories are a critical part of the software supply chain like GitHub, GitLab, Bitbucket etc. A breach, leak, or malicious commit in a repo can compromise entire applications, leak secrets, or introduce backdoors.

Key Principles of Secure Repository Design

- Least Privilege: Grant only the minimal permissions needed
- Strong Authentication: Require multi factor authentication for all users accessing repositories.
- Separation of duties: Require that code merging or branch changes be reviewed/approved by a separate party.

Preventing Secrets, Vulnerabilities, and Supply Chain risks

- Secret scanning: Use automated tools like GitGuardian, Gitleaks, etc. To detect hardcoded credentials or secrets.
- Dependency Scanning: Integrate security scanning in CI/CD to detect vulnerabilities in code and dependencies.
- Pinning: avoid "floating" dependency versions that may pull in malicious or vulnerable updates, prefer version locking.

Commit Integrity, Logging, and Monitoring

- Signed Commits: Require developers to sign commits to verify origin and integrity.
- Audit Logging: enable detailed logs for repo access, merges, pushes, pull requests, permission changes. Review logs regularly for anomalies.
- Malicious Commit Detection: Tools and research suggest using commit metadata, Change patterns, file sensitivity to flag suspicious commits.

Access Governance and Lifecycle Controls

- Periodic Access Reviews: Regularly review who has access and adjust as roles change.
- Time bound access: For contractors or external contributors, assign temporary access that expires
- Strong credential policies: For SSH keys, personal access tokens, rotate or expire keys regularly.

Challenges and Tradeoffs

- Overhead vs agility: Too strict policies may slow development.
- False positives in automated scanning can annoy developers.
- Managing legacy repositories with poor hygiene
- Balancing public vs private repo needs (open-source vs proprietary)

References

- OpenSSF. (2023, September 14). OpenSSF releases Source Code Management Best Practices guide. Open Source Security Foundation. https://openssf.org/blog/2023/09/14/openssf-releases-source-code-management-best-practices-guide/
- Orca Security. (2023, January 17). Source code security best practices. https://orca.security/resources/blog/source-code-security-best-practices
- GitHub Resources. Software development strategy essentials: A security-first approach. https://resources.github.com/security/software-development-strategy-essentials