1. Introduction

This document establishes guidelines for securing the software development environment, ensuring it is protected from unauthorized access, malicious attacks, and unintentional changes. A secure development environment is critical to safeguarding the integrity, confidentiality, and availability of software systems being developed.

These guidelines apply to all software development activities, including coding, testing, and deployment environments. They are relevant to all internal development teams, contractors, and third-party vendors working within the organization's software development lifecycle (SDLC).

Roles and Responsibilities

- **Developers:** Responsible for adhering to secure development environment protocols and reporting security issues.
- **System Administrators:** Responsible for maintaining secure configurations, managing user access, and monitoring the development environment.
- **Security Team:** Provides oversight and ensures compliance with secure development guidelines.
- **Third-Party Vendors:** Must comply with the organization's security policies and are responsible for maintaining secure development practices in outsourced projects.

2. Guidelines

2.1 Access Control and Authentication

- **Least Privilege Principle**: Ensure only authorized personnel can access the development environment. Access should be granted based on the minimum necessary privileges to perform their roles.
- **Multi-Factor Authentication (MFA):** Implement multi-factor authentication for all users accessing the development, testing, and production environments.
- **Role-Based Access Control (RBAC):** Use role-based access control mechanisms to limit access to sensitive resources based on job functions.
- **Regular Access Audits:** Perform periodic access rights reviews and remove access for users who no longer need it.

2.2 Secure Configuration Management

- **Baseline Configurations:** Define and enforce baseline security configurations for development environments. These configurations should include operating system security settings, application security settings, and network protections.
- **Configuration Versioning:** Track changes to configuration files and infrastructure as code (IaC) with version control systems to ensure any modifications can be audited and rolled back if needed.
- **Patch Management:** Regularly update and patch operating systems, middleware, development tools, and third-party libraries in the environment to protect against known vulnerabilities.
- **Hardening Guidelines:** Ensure development systems follow system-hardening guidelines, which may include disabling unnecessary services, closing unused ports, and enforcing strong password policies.

2.3 Network Security

- **Network Segmentation**: Separate development, testing, and production environments on distinct network segments to minimize the risk of lateral movement in case of a breach.
- **Firewalls and Network Access Control**: Configure firewalls and network access control mechanisms to restrict authorized users and systems from accessing the development environment.
- **Virtual Private Networks (VPNs)**: Secure VPNs are required to access development environments from remote locations.
- **Intrusion Detection and Prevention Systems (IDPS)**: Deploy IDPS to monitor network traffic for malicious activities and unauthorized access attempts within development environments.

2.4 Source Code Management

- **Version Control Systems (VCS):** Utilize secure, centrally managed version control systems such as Git to store and manage source code. Ensure proper access controls are enforced for all code repositories.
- **Encrypted Repositories:** Where possible, ensure that repositories containing sensitive code or proprietary algorithms are encrypted.
- **Code Review Process:** Implement code review policies to ensure that code is reviewed by at least one other developer or security expert before it is merged into the main branch.
- **Sensitive Data Protection:** Ensure that sensitive information such as credentials, API keys, and secrets are not hardcoded in source code. Use environment variables or secret management tools to store sensitive data securely.

2.5 Development Environment Security

- **Isolation of Development Environments:** Developers should work within isolated environments that are separate from production systems to reduce the risk of accidental data leakage or system disruptions.
- **Securing Development Machines:** Developer workstations should have full-disk encryption, strong passwords, and regular security updates. Anti-malware and endpoint protection tools must be deployed to prevent malware infections.
- **Containerized Development:** Use containerized environments (e.g., Docker, Kubernetes) for development and testing where appropriate, ensuring that containers are regularly scanned for vulnerabilities.
- **Access Logs:** Maintain logs of all access to development environments, ensuring that they are monitored for suspicious behavior. Logs should be retained for an appropriate duration based on organizational policies.

2.6 Third-Party Tools and Dependencies

- **Approved Tools:** Only organization-approved tools, libraries, and frameworks are used for development. Each tool should undergo a security review before being used in development.
- **Third-Party Library Monitoring:** Regularly scan third-party libraries and dependencies for known vulnerabilities using tools such as OWASP Dependency-Check, Snyk, or GitHub's security alerts.

- **License Management:** Ensure that all third-party libraries comply with licensing requirements and do not introduce legal or compliance risks to the organization.

2.7 Data Security and Privacy

- **Use of Test Data:** Avoid using production data in development and test environments. When using real data, ensure it is properly anonymized or masked to protect sensitive information.
- **Data Encryption:** Encrypt sensitive data in transit and at rest. Use secure protocols like TLS to transmit data and strong encryption standards for stored data.
- **Data Retention:** Define policies for the retention and deletion of data in development environments, ensuring that sensitive data is not retained longer than necessary.

2.8 Monitoring and Incident Response

- **Monitoring and Alerts:** Continuously monitor development environments for security threats using monitoring tools, logging systems, and SIEM (Security Information and Event Management) solutions.
- **Incident Response Plan:** Establish an incident response plan specific to development environments. This should include steps for identifying, containing, and remediating security incidents and communication protocols to notify key stakeholders.
- **Backup and Recovery:** Implement regular backups of critical development environment components and configurations. Ensure that backup data is encrypted and stored securely.

2.9 Testing and Continuous Integration/Continuous Deployment (CI/CD) Security

- **Secure CI/CD Pipelines:** Ensure that CI/CD pipelines are configured securely with limited access, strong authentication, and encryption for all data transmissions between pipeline components.
- **Automated Security Testing:** Integrate automated security testing tools, such as static analysis (SAST), dynamic analysis (DAST), and software composition analysis (SCA), into the CI/CD pipeline to identify vulnerabilities early.
- **Pre-Deployment Security Checks:** Ensure that security tests are a mandatory step in the deployment process and that any high-severity vulnerabilities are resolved before deployment to production.

3. Compliance and Auditing

- **Internal Audits:** Conduct regular audits of the development environment to ensure compliance with security policies and standards.
- **External Compliance:** Ensure compliance with applicable regulatory and industry standards such as NIST, ISO 27001, HIPAA, HITRUST, and GDPR, as they relate to secure development environments.
- **Continuous Improvement:** Regularly review and update these guidelines in response to evolving security threats, technology changes, and lessons learned from incident responses or audits.

4. Enforcement

Violations of these guidelines may result in disciplinary action, including termination of employment or contract. The organization reserves the right to investigate and address any non-compliance with secure development environment policies.

5. Review and Updates

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

6. Reference

Documents
Secure Development Policy