Software Development Plan for SSD – DefenceVenture

# 1. Security Properties in the Development Lifecycle

Confidentiality

* Access Control - Make sure users have the bare minimum of access required by enforcing stringent access rights to ShieldX components. For instance, ShieldX will impose IP limits and verify request sources in order to reduce the danger of unwanted data requests (danger ID 2).
* Data Encryption - AES-256 will be used to encrypt all sensitive data sent and stored within ShieldX in order to guard against risks such as information disclosure and spoofing (Risk IDs 1 and 2).

Integrity

* Integrity Checks - Make sure all important data and logs have integrity safeguards in place, such as digital signatures and hashing algorithms. For example, hash-based checks will verify that logs are unaltered in order to mitigate the risks of data manipulation (Risk ID 5) and repudiation (Risk ID 7).
* Authentication - To prevent unwanted access to accounts, implement multi-factor authentication (MFA) (Risk ID 4). ShieldX will use session validation tests and two-factor authentication to validate users.

Availability

* Network Monitoring - By confirming packet origins and preserving availability through traffic filtering, real-time monitoring will reduce denial-of-service (DoS) assaults (Risk ID 6).
* Redundant Systems - To ensure continuity of service in the event of an attack or failure, ShieldX will incorporate redundancy and failover techniques into the system design.

# 2. Securing the Development Environment

* Access Control in Development - Make sure that only individuals with permission can read or alter the source code by restricting access to the development environment. Only development leads will have access to the secure vault where sensitive code and passwords will be kept.
* Solation - To avoid unwanted access to ShieldX production data, develop in isolated environments. There won't be any sensitive production setups or actual data in the development and testing environments.
* Frequent Audits - To identify and resolve possible security threats early, conduct frequent security audits and vulnerability assessments of the development environment.

# 3. Use of Appropriate Tools for Secure Development

* Static and Dynamic Analysis Tools - To find security flaws in code as it is being developed, incorporate tools such as OWASP ZAP (dynamic analysis).
* Dependency Scanners - Make sure third-party libraries are safe by routinely checking dependencies for vulnerabilities using programs like Snyk or Dependabot.
* Threat Modelling Tools - To map potential vulnerabilities and control measures, use tools like Thread Dragon. Using the threat model as a guide, ShieldX's development will concentrate on mitigating threats such DoS assaults, data tampering, and spoofing.

# 4. Secure Development Practices

* Secure by Design: Make sure that every phase of the software development lifecycle, from planning to deployment, incorporates security needs. For instance, all new ShieldX features will come with audit trail tracking and access controls.
* Secure by Default: To reduce attack surfaces, default configurations will adhere to the least privilege principle and only expose necessary functionalities. Unauthorized privilege escalation will be avoided using this strategy (Risk ID 3).
* Code Reviews and Programming in Pairs: Pair programming and peer reviews will aid in the early detection of possible security vulnerabilities. This procedure will be required for all security-related critical code paths, including data management, encryption, and authentication.

# 5. Security Management

Access Control and Secrets Management

Role-Based Access Control (RBAC): Assign roles based on job responsibilities, restricting users from accessing or modifying parts of the system beyond their needs.

Secrets Management: Store sensitive information, including passwords and API keys, in a secrets management solution like HashiCorp Vault, ensuring secrets are securely encrypted and access logged.

Session Management: Implement short-lived tokens for sessions with expiration checks, preventing unauthorized long-term access (relevant for Risk IDs 1 and 4).

Preventing Vulnerabilities

Regular Patching and Updates: Ensure ShieldX dependencies and libraries are regularly patched to reduce the risk of exploitation due to outdated components.

Training and Awareness: Educate the development team on secure coding practices and new security threats, especially in areas like phishing and spear-phishing attacks (Risk ID 1).

Testing and Validation: Run regular automated tests, including security-focused tests, to ensure ShieldX remains compliant with security standards. Security testing will focus on verifying mitigations for all identified risks, including data integrity, confidentiality, and availability risks.

Responding to Vulnerabilities

Incident Response Plan: Develop and maintain an incident response plan, outlining steps to identify, contain, and resolve security incidents. Regular drills will help ensure readiness.

Logging and Monitoring: Implement comprehensive logging and monitoring tools to track user activities and flag suspicious behavior. Logs will be regularly reviewed and analyzed for signs of attacks like repudiation (Risk ID 7).

Patch Management for Vulnerabilities: Expedite patching for identified vulnerabilities, especially those in the threat model, to ensure ShieldX can address new and evolving threats promptly.

6. Reference to Threat Identifiers in the Threat Model

Threat Identifier Mitigations Applied

Risk ID 1 – Spoofing Attack MFA, IP restrictions, phishing-resistant protocols in user authentication.

Risk ID 2 – Information Disclosure Source IP monitoring, RBAC enforcement, and session validation checks.

Risk ID 3 – Privilege Elevation Least privilege role assignments, regular access audits, and restricted role assignments.

Risk ID 4 – Account Breach User authorization verification, digital signatures, and integrity checks on access data.

Risk ID 5 – Data Tampering Hash-based data validation, authorization checks, and use of digital signatures for critical operations.

Risk ID 6 – DoS Attack Network monitoring, DoS-resistant service configurations, and failover solutions.

Risk ID 7 – Repudiation Attack Enforced logging, timestamp validation, and restricted access to log modifications.

Conclusion

ShieldX’s development plan is committed to creating a secure, resilient software product that proactively addresses potential security risks. By embedding security considerations into each phase of development, adopting industry-standard tools, and adhering to best practices, we aim to provide ShieldX with robust protection against known and emerging threats.