

Secure SDLC Practices at Progress

At Progress Software, we are committed to integrating security at every stage of our Software Development Lifecycle (SDLC). Our approach to Secure SDLC (SSDLC) is to treat security not as an afterthought but as a fundamental component of our development process. This commitment is reflected in our adoption of the OWASP Software Assurance Maturity Model (SAMM), which provides a structured framework for assessing, formulating and implementing a robust software security lifecycle, threat modeling and Secure-by-Design principles.

Key Principles of Our Secure SDLC:

- **Proactive Security Integration:** We embed security practices from the initial design phase through to deployment and maintenance so potential vulnerabilities are addressed early and continuously.
- **Continuous Improvement:** By leveraging OWASP SAMM, we regularly evaluate and enhance our security practices to align them with emerging threats and industry standards.
- **Comprehensive Training:** We mandate comprehensive security training for all personnel involved in the software lifecycle to equip everyone with the knowledge to contribute to secure development.
- **Risk Management:** Our SSDLC framework includes rigorous risk assessment and mitigation strategies so that security risks can be identified, prioritized and addressed effectively.
- **Transparency and Accountability:** We maintain clear documentation and reporting mechanisms to facilitate transparency in our security practices and accountability at all levels of the organization.

Adoption of OWASP SAMM:

OWASP SAMM is integral to our SSDLC framework. It provides a measurable and effective way to analyze and improve our software security posture. The model supports our entire software lifecycle and is adaptable to various development methodologies, whether agile or waterfall.

- **Assessment and Strategy:** We use SAMM to assess our current security practices and define a strategic roadmap for improvement.

- **Implementation and Measurement:** SAMM guides us in implementing security activities at different maturity levels, allowing us to measure progress and demonstrate concrete improvements.
- **Tailored Approach:** The flexibility of SAMM enables us to tailor our security practices to the specific risks and needs of our organization, supporting a balanced and effective security assurance program.

SSDLC Tooling and Functions:

We apply a comprehensive security approach to our software that integrates best-in-class Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST) and Software Composition Analysis (SCA) tools. Our SAST tools scrutinize our source code, uncovering vulnerabilities early in the development cycle. Meanwhile, our DAST tool probes our running applications, identifying potential security flaws from an external perspective. Finally, our SCA tools examine third-party components and open-source libraries for known vulnerabilities, helping to maintain the integrity of our software supply chain. Together, these methods bolster our overall security posture, enhancing both the reliability and safety of our applications.

Secure by Design Principles:

We adhere to Secure by Design principles so security is integrated into every phase of development. This proactive approach helps us mitigate risks and protect our software from potential threats. The key principles we follow include:

- **Security-Centric Culture:** We foster a culture where security is a priority for everyone involved in the development process. This includes regular training, awareness programs and ad-hoc workshops to keep our team updated on the latest security practices.
- **Incorporating Security into Requirements:** Security requirements are defined and integrated from the very beginning of the project. This allows for security considerations to be a fundamental part of the design rather than an afterthought.
- **Secure Coding Practices:** Our developers follow secure coding practices to identify and fix common vulnerabilities such as SQL injection, cross-site scripting (XSS) and buffer overflows. We use static and dynamic analysis tools to identify and fix security issues during development.
- **Layered Defense:** We implement multiple layers of defense to protect our software from various types of attacks. This includes firewalls, intrusion detection systems and encryption to help safeguard data at rest and in transit.

- **Secure and Fail-Safe Defaults:** Our software is designed with secure and fail-safe defaults to minimize the risk of security breaches. This means that the default configurations are secure, and the system fails securely if an error occurs.
- **Automated Security Testing:** We automate security testing to support continuous and consistent security checks throughout the development lifecycle. This includes automated vulnerability scanning, penetration testing and code reviews.
- **Robust Vulnerability Management:** We have a robust vulnerability management program in place to quickly identify, assess and remediate vulnerabilities. This enables us to promptly address and mitigate security issues.
- **Threat Modeling:** During the design phase, the team takes a risk-based approach. Before the team builds new applications and features, they will complete risk assessments that model aspects of both the attack and defense sides of the application, system or environment. The methodology centers on data flows, knowledge of the software being analyzed and a cross-functional team approach so all threats can be discovered, evaluated and addressed. Use of industry-standard resources from MITRE, OWASP, CERT and others are incorporated into the threat modeling process.