

Assignment 7.1: Risks and Metric in Software Systems

Joshua A. Wortz

University of San Diego

CSOL-560

Dr Varma

12/08/2025

Enterprise Risk Management Plan

Organization Profile

- **Organization Name:** AetherCode Solutions
- **Industry:** Software Development (SaaS & DevSecOps Tooling)
- **Business Model:** Provider of cloud-native CI/CD orchestration tools for enterprise clients.

Introduction & Methodology

Risk assessment in the modern software industry must evolve beyond static checklists to address the dynamic nature of Agile and DevSecOps environments. This plan adopts a **scenario-based approach** aligned with ISO 27001, but enhanced by recent academic frameworks that integrate security directly into the Continuous Integration/Continuous Deployment (CI/CD) pipeline.

We utilize the standard formula **Risk = Asset × Threat × Vulnerability**. However, recent literature emphasizes that in software supply chains, risk is "contagious"; a vulnerability in one module can cascade through the entire network of downstream dependencies.

Step 1: Asset Identification

We identify assets that are compromised, resulting in direct financial loss or a violation of the "CIA" triad (Confidentiality, Integrity, Availability). In the context of software development, assets include intangible trust mechanisms.

1. **Code Signing Infrastructure:** The cryptographic keys used to validate software authenticity. Compromise leads to a total loss of trust, as recent studies on repository security have validated.

2. **Source Code Repositories:** The core intellectual property (IP) and proprietary logic of the SaaS platform.
3. **CI/CD Build Pipeline:** The automated environment that builds and deploys software.
4. **Customer Production Data:** Tenant data managed within the SaaS environment.

Step 2: Potential Consequences

Consequences are evaluated based on financial impact, legal liability, and operational continuity.

- **Integrity Loss (Supply Chain Contagion):** If the build pipeline is compromised, the software itself becomes an attack vector for clients. Research indicates that supply chain risks propagate through "selective attacks," where adversaries target specific upstream components to maximize downstream damage.
- **Confidentiality Loss (IP Theft):** Unauthorized access to repositories can lead to cloned products or the exploitation of zero-day vulnerabilities before patches are deployed.
- **Availability Loss (Architectural Failure):** Inadequate risk assessment during the design phase can lead to systemic failures (e.g., service outages), which damage reputation and incur SLA penalties.

Step 3: Threat Identification

We focus on non-speculative threats relevant to the current landscape.

1. **Supply Chain Injection:** Adversaries compromise open-source dependencies or build tools to inject malicious code.
2. **Insider Threat:** Malicious or negligent actions by employees with elevated access, which remains a leading cause of data breaches in IT.

3. **Architectural Design Flaws:** Risks arising from ambiguity in requirements or technological constraints during the design phase, leading to unstable software behavior.
4. **Credential Theft (AiTM):** Adversary-in-the-Middle attacks bypassing Multi-Factor Authentication (MFA) to hijack administrative sessions.

Step 4: Vulnerability & Likelihood Assessment

Vulnerabilities are analyzed based on DevSecOps maturity models.

- **V1: Weak Supply Chain Practices:** Lack of signed releases or branch protection in repositories. *Likelihood: High.* Research shows a generally weak security posture in many software repositories, with rare implementation of signed releases.
- **V2: Design-Phase Ambiguity:** Failure to use fuzzy logic or formal methods to assess risks during software design. *Likelihood: Moderate.*
- **V3: Manual Security Checks:** Reliance on manual code review rather than automated security testing in the pipeline. *Likelihood: High.*
- **V4: Excessive Access Permissions:** Developers retaining standing access to production environments. *Likelihood: High.*

Step 5: Risk Assessment Matrix

Risk is calculated using the established formula.

- **Formula:** Risk = Asset Value × Threat Level × Vulnerability Level
- **Scoring:** High, Moderate, Low.

Step 6: Risk Management Plan

This table adheres to the format specified in the Assignment 7.1 Template.

Threat	Vulnerability	Asset and Consequences	Risk	Solution
Supply Chain Injection (Level: High)	Weak Supply Chain Practices (Level: High)	Code Signing Infrastructure Consequence: Total loss of trust; Clients infected with malware. (Value: Critical)	High	Automated Governance: Implement cryptographic signing verification as a "Definition of Done" in Agile sprints; Deploy Hardware Security Modules (HSM) for key storage.
Insider Threat	Excessive Access Permissions	Build Pipeline Consequence:	High	Zero Trust Pipeline: Remove standing human access to production

(Level: High)	(Level: High)	Malicious code injection into production; Supply chain contagion. (Value: Critical)		environments; Only automated, authenticated pipelines can deploy code.
Credential Theft (AiTM) (Level: High)	Lack of Phishing-Resistant MFA (Level: High)	Customer Production Data Consequence: Data exfiltration; Regulatory fines (GDPR/CCPA).	High	FIDO2/ WebAuthn: Deploy hardware-backed security keys (e.g., YubiKeys) for all administrators to prevent proxy-based phishing attacks.

		(Value: Critical)		
Architectural Design Flaws (Level: Moderate)	Design-Phase Ambiguity (Level: Moderate)	SaaS Uptime Consequence: Service outage; SLA financial penalties. (Value: Moderate)	Moderate	Fuzzy Risk Modeling: Utilize fuzzy logic methodologies to quantify and assess design ambiguity risks during the architecture phase.
IP Theft (Level: Moderate)	Manual Security Checks	Source Code Repositories Consequence: Competitor	Moderate	DevSecOps Automation: Integrate automated SAST/DAST scanning into the CI/CD pipeline to

	(Level: High)	cloning; Zero-day exploitation.		detect vulnerabilities before code commit.
		(Value: Moderate)		

Step 7: IT Infrastructure Enhancement Strategy

To mitigate these risks, we will transition to a **DevSecOps Architecture**.

1. **Automated Security Orchestration:** Integrate security tools (SAST/DAST) directly into the CI/CD pipeline. This ensures that security is a "shared responsibility" and vulnerabilities are detected in near real-time.
2. **Supply Chain Monitoring:** Deploy tools to continuously assess the "contagion" risk of third-party libraries, effectively mapping the multi-layer network of dependencies.
3. **ISO 27001-Agile Integration:** Embed ISO 27001 controls (like access control and cryptography) into the Scrum process by adding them to the product backlog.

Step 8: Mitigation Processes

Sample Mitigation Process: Supply Chain Vulnerability

- **Event:** A critical vulnerability is disclosed in an open-source library used by AetherCode (e.g., Log4j).

- **Response:** The automated pipeline halts all builds that contain the vulnerable version.
- **Analysis:** Determine if the vulnerability was "contagious", did it propagate to downstream clients?.
- **Mitigation:** Patch the dependency and implement "Branch Protection" rules to prevent future use of unverified libraries.

Step 9: Risk Assessment Policy

Policy Statement: "Risk assessment shall be continuous and integrated into the software delivery lifecycle."

Methodology:

We will move from annual assessments to Continuous Risk Monitoring. As supported by recent literature, the rapid pace of Agile development requires that security practices be mapped directly to development phases (Requirements, Design, Coding, Testing). Risk assessments will be triggered automatically by changes in the "threat landscape" or major architectural updates.

References

- Aregbesola, M., & Asghar, S. (2024). Fuzzy Logic Model for Informed Decision-Making in Risk Assessment During Software Design. *MDPI Systems*, 13(9).
- Bhardwaj, A. K., Al-Turjman, F., Kumar, M., & Rathore, S. (2024). Securing Container Images through Automated Vulnerability Detection in Shift-Left CI/CD Pipelines. *Babylonian Journal of Networking*, 2024, 162-170.
- Cadena-Martínez, R., Jiménez-Gómez, J., & Fernández-Sanz, L. (2023). Towards the Integration of Security Practices in Agile Software Development: A Systematic Mapping Review. *MDPI Applied Sciences*, 13(7).
- Cai, H., Zhang, Y., & Liu, X. (2025). Research on network security vulnerability risk contagion in software supply chain based on system dynamics. *PLoS ONE*, 20(11).
- Hegewald, R., & Beyer, R. (2025). Evaluating Software Supply Chain Security in Research Software. *arXiv preprint arXiv:2508.03856*.
- Kaur, R., Singh, P., & Sharma, A. (2025). Evolution of DevSecOps and Its Influence on Application Security: A Systematic Literature Review. *MDPI*.
- Llopez-Perez, J., Garcia-Tome, A., & Calvo-Manzano, J. A. (2025). Integrating Information Security Management System from ISO/IEC 27001 with Software Project Management Utilizing Scrum Framework. *ResearchGate*.