BEMS Reports, 2025; 11(2):41-43.
https://www.bemsreports.org

Brief Report

Scaling Product Security with AI and Automation

Chaitra Bhat*

Paranoids, Technical Security Engineer, Yahoo Inc., Dublin, IRELAND.

ABSTRACT

Ensuring strong product security is a fundamental part of building trustworthy and resilient software. However, conducting thorough security reviews especially in fast-moving, agile engineering teams can be challenging to scale effectively. Traditional review processes are often time-consuming, resource-intensive, and difficult to keep consistent across teams and projects. This article introduces a practical, staged framework designed to improve the product security review lifecycle by leveraging automation and generative AI. The focus is on two key areas: threat modelling and secure design validation both of which are critical in identifying potential vulnerabilities early in the development cycle. By automating repetitive tasks and enhancing human decision-making with AI-powered tools, teams can accelerate reviews without sacrificing depth or quality. Targeted at product security engineers and cybersecurity professionals, the framework addresses real-world pain points such as lack of scalability, inconsistent processes, and limited security resources. It offers actionable insights into how modern tools can be integrated into existing workflows to support faster, smarter, and more scalable security practices. Ultimately, this approach aims to bridge the gap between strong security standards and the need for development speed, helping organizations build more secure products without slowing down innovation.

Keywords: Product Security, Threat modelling, Secure design, Automation, Generative AI.

Correspondence:
Mrs. Chaitra Bhat
Paranoids, Principal Technical Security Engineer, Yahoo Inc., Dublin, IRELAND.
Email: chaitra.bhat@gmail.com

Received: 01-06-2025;
Revised: 24-06-2025;
Accepted: 07-07-2025.

INTRODUCTION

Modern engineering teams are releasing software at an unprecedented pace, driven by agile methodologies and continuous delivery practices. However, product security often struggles to keep up. Security reviews particularly those focused on architecture and design are still largely manual, heavily reliant on context, and difficult to scale across fast-moving development teams.[1,2] This mismatch creates a significant bottleneck in the software development lifecycle, especially when security teams are understaffed or stretched thin. As a result, critical security checks may be delayed or skipped altogether, increasing the risk of vulnerabilities making it into production. Conducting product security reviews early in the development process during the design and planning phases is essential for identifying and mitigating risks before they become costly to address.[1,3] Fixing security issues at this stage is significantly more efficient than remediating them post-deployment. If neglected, these gaps can lead to serious consequences, including data breaches, regulatory non-compliance, and expensive rework late in the lifecycle. To keep pace with rapid development cycles, organizations need scalable approaches to embed security into the software design process. This includes exploring automation, integrating security into developer workflows, and leveraging emerging technologies like AI to reduce manual overhead and improve coverage.

Many organizations have adopted Application Security Posture Management (ASPM) tools to monitor systems after deployment.[4] However, few have successfully automated the earlier and arguably more critical phases of product security, such as threat modelling and secure design validation.[1,5] These stages still depend heavily on business context and manual effort. A typical product security review involves evaluating application code, infrastructure, deployment pipelines, and overall system design to identify and mitigate potential risks.[2,3] It also includes performing threat modelling to anticipate possible attack vectors, reviewing code changes, and ensuring the secure implementation of critical components like authentication, access control, and payment logic. These activities often require close collaboration with developers and a deep understanding of the product's architecture and business use cases. Recent advancements in generative AI offer a promising opportunity to address these challenges. If automation can reliably handle even 70-80% of common security tasks, security teams could be freed up to focus on higher-risk, higher-impact areas. This shift has the potential to make product security practices more scalable, efficient, and effective across fast-paced development environments.

The following definitions apply to the two key terms used in this report:

DOI: 10.5530/bems.11.2.9

- **Critical Review/Change:** Involves systems with high business impact (e.g., authentication infrastructure, data stores handling PII or payment data, central access control systems).
- **Standard Review:** Applies to low-risk, well-documented happy-path changes using known secure patterns and approved frameworks.

The Scalability Challenge in Product Security

Drawing from hands-on experience across cloud migrations, greenfield projects, and feature upgrades, here are some common pain points to scale product security reviews:

- **Lack of effective automation:** There are just no good, automated solutions today for threat modelling and secure design review.
- **No good asset or resource classification:** Poor asset classification leads to uniform prioritisation, wasting time on low-risk reviews while neglecting high-risk ones.
- **Lack of good or no change management system:** Weak change management allows critical changes to bypass review, increasing risk.
- **Lack of visibility and measurable impact:** Preventative work is hard to measure, leading to underinvestment. One idea I am exploring is tracking all code changes made via Pull-Requests (PRs) that stem from product security reviews - a metric that could finally show real, tangible influence.
- **Operational bottlenecks and burnout:** Too much manual review and busy work means less time for innovation and critical reviews. This is a recipe for burnout.

These challenges demand a new approach - one that leverages automation and reimagines the product security review process.

My Vision: A Staged, Automated Approach

Here is the high-level blueprint I believe can work. It breaks security reviews into three SDLC checkpoints, each with automation and clear roles:

Stage 1: Design & Development

- Who: Developers, Architects.
- What: Product Requirement Documents (PRDs), design docs, Git repos containing application and Infrastructure-as-Code (IaC), Static Application Security Testing (SAST) tools findings for application and IaC and Software Composition Analysis (SCA) outputs.

How it works

- A self-serving GenAI-powered Threat Modelling Assistant reviews PRDs, design docs, and code changes, spotting security gaps (e.g., missing auth, insecure dependencies).
- Developers get real-time feedback-risks, remediation links, secure code templates - so they can fix issues early, without waiting for review from security engineers.
- For low-risk, standardised changes, trained security champions can approve deployment.
- Security champions approved deployments can form the first baseline for the repo/app, useful for future drift detection via ASPM.

Note: Commits and PRs trigger integrated scans by security tools integrated into the CI/CD pipeline, feeding data to ASPM tools.

Stage 2: Point-in-Time Security Review (Pre-Production - dev/staging)

- Who: Product Security Engineers.
- What: All of the above artefacts, plus runtime/cloud/container/network scanner results.

How it works

- Reviews are triggered for critical/non-standard changes - either manually by dev/security champions or automatically via change management systems.
- Engineers focus on business logic-based threat modelling and secure code review; they can also use the Threat Modelling Assistant by adding custom rules for org-specific checks if possible.
- Leverage ASPM or security tools dashboard to view all the reported issues from the scanners for the concerned application or repo.
- Dynamic testing is done as needed.
- Findings from this review update the app/service baseline. Security tickets are created for issues for tracking purposes.

Stage 3: Continuous Security Posture Monitoring

- **Who:** Data Security Teams, DevSecOps.
- **Scope:** Drift detection in baselines for all artefacts mentioned in stages 1 and 2.

How it works (not the focus of this discussion):

- ASPM tools aggregate findings from various sources.
- Drift from the last secure baseline only triggers alerts if risk exceeds a set threshold - reducing noise and focusing attention where it matters.

Enhancements to Make This Efficient

- **Streamline discovery of data for reviews:** Standardise required inputs (cloud accounts, Git links, design docs, diagrams) and build a pre-review tool to validate completeness and gather tool findings.
- **Automate review/ticketing:** Use a data lake to correlate, deduplicate, and risk-rank tool outputs - auto-creating tickets for high-confidence, high-risk issues.

What is Needed to Make This Work

1. Automated secure design & threat modelling: Generative AI/expert systems that analyse PRDs, design docs, IaC and code, providing real-time, contextual feedback. (There is a lot of startup activity here - so I am hopeful we'll see real solutions soon!)

2. Asset/resource classification: Tag systems by criticality, sensitivity and exposure to prioritise reviews.

3. Automated change management: Detect and notify stakeholders of critical changes (e.g., login/auth/payment logic). Can be part of ASPM solutions to detect drifts in artefacts.

4. Integrated security tools: SAST, SCA, IaC, container/cloud scanning, network scanners, EDR - all feeding into a data lake where the central intelligence layer can be applied.

5. Curated security knowledge base: Baseline requirements, templates, checklists, secure coding patterns and libraries.

6. Documentation hygiene: PRDs and design docs tracked in version control or tagged via metadata. Over time, we may see these documents evolve into the authoritative source for AI-generated code, effectively bridging the gap between design and implementation - this is a separate topic!

Intelligence Tier: Risk-Based Triage

To make product security reviews scalable and less disruptive, it's essential to introduce an intelligence layer that can triage security findings based on asset or resource classifications. This intelligence layer should assess the criticality of each system, service, or data resource involved leveraging metadata such as asset tags, business impact scores, or data sensitivity levels. By applying this contextual understanding, the system can prioritize security findings appropriately. Only high or critical issues those that affect sensitive assets or present a substantial risk to the business should interrupt development workflows or trigger immediate action. These are the findings that must be addressed before code can be merged or deployed. On the other hand, lower-priority findings, especially those tied to less critical assets or minor design flaws, can be safely deferred for review during scheduled security assessments or backlog grooming sessions. This tiered approach minimizes unnecessary friction for engineering teams while ensuring that meaningful risks receive prompt attention. It also allows security teams to focus their limited resources where they matter most, reducing alert fatigue and enabling better strategic oversight. Ultimately, integrating this intelligence-driven triage mechanism helps balance development speed with robust security, ensuring smarter and more efficient product security operations.

CONCLUSION: A SMARTER PATH FORWARD

This blueprint is not about eliminating human effort - it's about optimising it, so skilled security engineers can focus where they are most needed. By shifting security left, layering automation, intelligence and structuring responsibilities, we can scale product security. There is still lots of detail to work out, and every item under "What is Needed to Make This Work" is a challenge on its own. If your team has found creative solutions for these challenges, I would love to connect and swap notes. Let's make scalable product security a reality together!

ACKNOWLEDGEMENT

None.

CONFLICT OF INTEREST

The author declares that there is no conflict of interest.

ABBREVIATIONS

AI: Artificial Intelligence; **ASPM:** Application Security Posture Management; **CI/CD:** Continuous Integration and Continuous Deployment; **DevSecOps:** Development, Security, and Operations; **IaC:** Infrastructure-as-Code; **PII:** Personally Identifiable Information; **PRDs:** Product Requirement Documents; **PRs:** Pull-Requests; **SAST:** Static Application Security Testing; **SCA:** Software Composition Analysis; **SDLC:** Software Development Life Cycle.

REFERENCES

1. Murat D, Berkan U, Ali I. An Overview of Secure by Design: Enhancing Systems Security through Systems Security Engineering and Threat Modeling. Paper/Poster presented at: 2024, 17th International Conference on Information Security and Cryptology (ISCTürkiye); 2024.
2. Taherdoost H. Understanding cybersecurity frameworks and information security standards-A review and comprehensive overview. *Electronics*. 2022; 11: 2181.
3. Karie NM, Sahri NM, Yang W, Valli C, Kebande VR. A review of security standards and frameworks for IoT-based smart environments. *IEEe Access*. 2021; 9: 121975-95.
4. Jim MMI. Cloud Security Posture Management Automating Risk Identification and Response in Cloud Infrastructures. *Academic Journal on Science, Technology, Engineering & Mathematics Education*. 2024; 4: 10.69593.
5. Crothers EN, Japkowicz N, Viktor HL. Machine-generated text: A comprehensive survey of threat models and detection methods. *IEEE Access*. 2023; 11: 70977-1002.

Cite this article: Bhat C. Scaling Product Security with AI and Automation. BEMS Reports. 2025;11(2):41-3.