RESEARCH REPORT

Evolving Strategies for Vulnerability Management in DevOps CI/CD Pipelines

Unit code: EAT40005
Unit Name: Engineering Technology Project A (ENG/CS)
Submission date: 15/05/2025
Student Name: Dang Khoa Le
Student ID: 103844421

ACKNOWLEDGMENT OF COUNTRY

I acknowledge the Traditional Custodians of the lands on which I lived and worked while completing this project. I pay our respects to their Elders past and present and extend that respect to all Aboriginal and Torres Strait Islander peoples.
I recognise their enduring connection to land, waters, and culture, and I honour their rich traditions and contributions to our shared community.

1. Executive Summary

The acceleration of software delivery through DevOps and CI/CD pipelines has increased security exposure, making **vulnerability management (VM)** essential for resilient development. This report examines VM in modern pipelines, focusing on **scalable**, **automated**, and **context-aware solutions**. It compares three tools - **Nessus** (traditional), **GitHub Advanced Security** (developer-centric), and **Orca Security** (AI-powered) - across organisational sizes and maturity levels, evaluating their alignment with developers, SecOps, and leadership roles.

The report also analyses **ethical**, **economic**, and **technical integration challenges**, mapped to **NIST CSF maturity tiers** (Partial, Risk-Informed, Repeatable). It then explores how **AI and automation** will transform VM over the next 3–5 years through proactive threat detection, adaptive policies, and frictionless compliance.

The report concludes with **actionable recommendations** based on team size and maturity, offering a roadmap for implementing effective and future-proof VM in DevOps environments.

Table of Contents

2. Introduction

In the context of modern software engineering, **Vulnerability Management (VM)** refers to the continuous process of identifying, assessing, remediating, and reporting security weaknesses in systems and codebases. With the widespread adoption of DevOps and CI/CD pipelines, the attack surface across dependencies and deployments has expanded, making VM critical to secure software delivery [1].

Traditionally, security checks occurred late in development, but modern practices now embed security earlier in the cycle via the "shift-left" approach [2]. This evolution, however, introduces challenges in balancing security enforcement with rapid innovation.

This report analyses three categories of VM tools - traditional, developer-centric, and AI-powered - evaluating their suitability across varying team sizes and DevOps maturity stages. It applies the **NIST Cybersecurity Framework's (CSF)** maturity tiers - Partial, Risk-Informed, and Repeatable to assess data privacy, integration complexity, and resourcing considerations [3].

Finally, it offers predictive insights into how AI and automation are transforming VM, enabling scalable threat detection, adaptive policy enforcement, and seamless compliance across modern pipelines.

3. Literature Review

3.1 Historical Background: Early Vulnerability Scanning and Patching Practices

Vulnerability management originated in a time when software was deployed in monolithic releases and security was treated as an afterthought. Traditionally, organisations relied on periodic vulnerability assessments using tools such as **Nessus**, **OpenVAS**, or **Qualys** - scanners that inspected systems for known Common Vulnerabilities and Exposures (CVEs) based on predefined databases [4]. These scans were often reactive, performed quarterly or post-deployment, leading to long vulnerability dwell times.

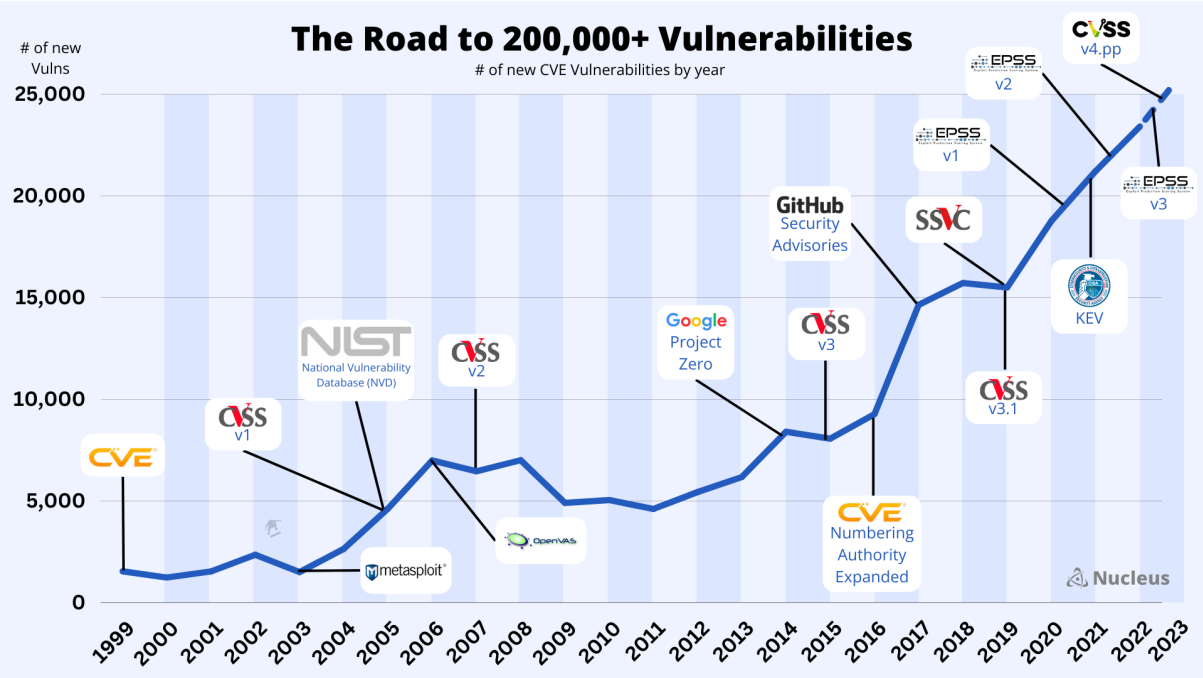

Figure 1. Evolution Timeline of VM technologies 1999-2023 [45].

Patch management was manual and siloed from development workflows, typically managed by separate IT or security operations teams. According to Kandek [5], organisations averaged 45–60 days to patch critical vulnerabilities, largely due to disconnected systems and lack of developer ownership. This model, though still present in legacy systems, proved inadequate in modern cloud-native and agile environments.

3.2 Rise of DevOps & CI/CD: Security in the Age of Automation

The emergence of **DevOps** fundamentally changed the software development lifecycle. By integrating development, operations, and testing into an iterative cycle, teams can accelerated release cadences from quarterly to daily or hourly. Thereafter, this velocity introduced new attack surfaces and demanded a paradigm shift in security posture [6].

Research by Shahin et al. [7] on DevSecOps advocates for embedding security controls directly into CI/CD pipelines, a practice known as "shifting left." Vulnerability management must now operate continuously, in tandem with code changes, build processes, and automated testing. Tools like **GitHub Advanced Security**, **GitLab Ultimate**, and **Snyk** have emerged to support this integration, allowing developers to detect issues in source code, container images, and dependencies before they reach production.

A critical insight from **OWASP DevSecOps Guidelines (2021)** [8] is the need for security tooling that is frictionless-automated, developer-friendly, and context-aware. However, studies like Carter et al. [9] highlight persistent challenges in team adoption, particularly where tooling adds cognitive overhead or where security is deprioritized in favor of release speed.

3.3 Modern Tools: Traditional vs. AI-Powered Vulnerability Management Solutions

Traditional vulnerability scanners like **Nessus** and **Qualys** remain popular in enterprise environments for infrastructure and network scanning. They provide comprehensive coverage of known CVEs and compliance benchmarks such as CIS, HIPAA [4]. However, their integration with CI/CD pipelines is often limited or requires external orchestration.

In contrast, developer-centric tools like **GitHub Advanced Security** and **Snyk** offer in-pipeline vulnerability scanning for dependencies, container images, and secrets [10][42]. They support early detection, integrate with version control systems, and foster developer ownership.

AI-powered platforms, such as **Orca Security** and **Wiz**, represent the next generation of vulnerability management. These solutions leverage agentless cloud scanning, risk-prioritization algorithms, and AI-based correlation of misconfigurations, vulnerabilities, and permissions. According to Gartner [11], such tools reduce alert fatigue and support more effective triaging by understanding context - e.g., a vulnerable workload exposed to the public internet is ranked higher than one in an isolated subnet.

Empirical comparisons in Jouini and Rabai [12] suggest that while traditional scanners provide depth, they lack the agility required for modern pipelines. Developer-centric and AI tools trade some scanning depth for speed, scalability, and user adoption. However, current research lacks longitudinal studies comparing long-term risk reduction or cost efficiency between these approaches across maturity levels.

3.4 Future Tools: Research and Trends in Vulnerability Management Automation

Recent academic research increasingly leverages AI, machine learning, and NLP to enhance vulnerability management. One key area is **predictive vulnerability scoring**, where models estimate threat impact before CVEs are officially published, such as Sabottke et al.'s [13] work using social media data like Twitter and Reddit to forecast high-risk vulnerabilities. Another trend is the rise of **self-healing infrastructure**, where tools like Amazon Inspector, Microsoft Defender for Cloud, and Trivy automate remediation of configuration drift, exposed secrets, or software flaws [14][40][41]. Additionally, **context-aware policy engines** are being explored to enforce dynamic security rules based on access roles, code history, and real-time threat intelligence. Mitchell et al. [15] highlight how these engines could proactively block pull requests tied to emerging CVEs.

However, most studies focus on **mid-to-high maturity teams** with strong automation and security practices. There is limited research addressing **low-maturity (NIST Tier 1) teams**, especially those lacking resources or organizational readiness for automation. Similarly, gaps exist in guiding **SMEs** or **regulated sectors** on transitioning from manual to automated VM workflows.

4. Comparative Analysis of Three Solutions

Modern DevOps pipelines require vulnerability management (VM) solutions that are not only secure but also operationally compatible with CI/CD workflows, developer-centric, and scalable to team size and maturity. This section compares three prominent tools - **Nessus** (traditional), **Orca Security** (AI-powered), and **GitHub Advanced Security** (GHAS) (developer-centric) - across usability, scalability, and stakeholder alignment. Each tool reflects distinct philosophies and trade-offs suitable for varying team sizes and DevSecOps maturity levels.

4.1 Traditional VM Tool: Nessus

Nessus, developed by Tenable, remains one of the most widely adopted vulnerability assessment tools globally. It primarily targets operating systems, network devices, and server misconfigurations by scanning against known CVEs using a vast plugin library.

Strengths:

- Rich plugin library for vulnerability detection and compliance (e.g., PCI-DSS, HIPAA) [16].
- Effective at identifying OS and network-level weaknesses [17].
- Support for compliance standards like PCI-DSS, HIPAA, and CIS benchmarks.
- Maturity and proven effectiveness in enterprise environments.

Weaknesses:

- Limited native integration with CI/CD pipelines. Orchestration often requires external tools or scripting (e.g., Jenkins + CLI) [18].
- Slower remediation cycles due to post-deployment focus (non-continuous scanning).
- Requires significant SecOps involvement for configuration, analysis, and orchestration [19].

CI/CD Integration Ease:

- Lacks native support for CI tools like Jenkins, GitHub Actions, or GitLab CI.
- Automation via CLI and scripts is possible but non-trivial.

Scalability:

- Well-suited for **large teams** with dedicated security departments.
- Overhead may burden **small teams** lacking operational capacity.

Stakeholder Alignment:

- Primarily SecOps-focused; limited visibility or empowerment for developers.
- Leadership value lies in compliance and audit reporting, not velocity.

4.2 AI-Powered VM Tool: Orca Security

Orca Security exemplifies the rise of AI-powered cloud-native vulnerability platforms. It leverages agentless scanning, AI-based prioritization, and contextual (IAM) risk correlation across virtual machines, containers, and cloud services.

Strengths:

- Agentless deployment across AWS, Azure, and GCP reduces friction and enables instant visibility [20].
- AI prioritization highlights exploitable risks based on exposure, privilege, and network paths [21].
- Unified view of vulnerabilities, misconfigurations, compliance risks with real-time cloud topology [22].

Weaknesses:

- Enterprise pricing models can be restrictive for small-to-medium teams [23].
- Overhead from excessive findings without mature filtering policies.
- Heavy cloud focus may not suit hybrid/on-prem environments.

CI/CD Integration Ease:

- Supports IaC scanning, integrates with Terraform, GitHub, Jenkins, and Azure DevOps [24].
- Less effective on embedded software or traditional server-based pipelines.

Scalability:

- Highly scalable for **medium-to-large teams**, especially cloud-native.
- Smaller teams may struggle with managing AI alert triaging without training.

Stakeholder Alignment:

- Developers benefit from contextualized alerts, though require onboarding.
- SecOps gain visibility into live environments and historical misconfigurations.
- Leadership benefits from AI-driven risk posture scores and dashboards.

4.3 Dev-Centric Tool: GitHub Advanced Security

GitHub Advanced Security (GHAS) integrates security directly into the software development lifecycle. It scans repositories for vulnerabilities in code, secrets, and dependencies using CodeQL, secret scanning, and dependency graphing [10].

Strengths:

- Native to GitHub ecosystem; integrates directly with pull requests and commits [10].
- Automates scanning of source code, actions workflows, and package dependencies [25].
- Encourages developer ownership by surfacing issues pre-merge [26].

Weaknesses:

- Limited to the GitHub ecosystem; third-party hosting requires workarounds.
- Focused on code rather than infrastructure-level vulnerabilities.
- Complex licensing tied to GitHub Enterprise Cloud plans.

CI/CD Integration Ease:

- Best-in-class for GitHub-native CI/CD environments.
- Prebuilt GitHub Actions templates ease adoption and automation [27].

Scalability:

- Highly effective for **small-to-medium teams** with GitHub-based pipelines.
- **Enterprise teams** require significant customization for multi-repo governance.

Stakeholder Alignment:

- Strong developer alignment; empowers shift-left practices.
- SecOps may need complementary tools for full infrastructure coverage.
- Leadership gains transparency on dev-originated risks and resolution metrics.

Summary & Synthesis

Tool	Best Fit	CI/CD Fit	Stakeholder Emphasis	Cost Barrier	AI Support
Nessus	Enterprises with mature SecOps	Low	SecOps, Audit	Low/Medium	Yes
Orca Security	Cloud-native, high maturity orgs	Medium	SecOps, Leadership	High	No
GitHub Adv. Security	Dev-centric, mid-size agile teams	High	Developers	Medium	Limited

Table 1. Summary of Comparative Insights among the 3 Solutions.

Tool effectiveness varies with team size and DevOps maturity. GHAS/Snyk suits small agile teams by integrating directly into developer workflows. Orca/Wiz provide scalable, AI-driven visibility for mid-sized teams, albeit at higher cost. Nessus remains preferred for enterprises focused on post-deployment and compliance.

Stakeholder roles drive tool priorities: developers need in-pipeline alerts, SecOps need risk context and visibility, and leadership values scalable risk reduction. The most effective VM strategy balances all three perspectives.

5. Evaluation Against NIST CSF Maturity Levels

Effective vulnerability management (VM) must align with an organisation's current cybersecurity maturity. The NIST Cybersecurity Framework (CSF) provides a practical lens through its maturity tiers: **Partial**, **Risk-Informed**, and **Repeatable**. Each stage presents distinct ethical, economic, and technical considerations for selecting and integrating VM solutions.

Figure 2. NIST CSF Maturity Model Circle [46].

5.1 Partial (Tier 1): Basic Setups with Limited Formal Processes

Organisations at Tier 1 often lack dedicated security personnel or standardised processes. Vulnerability scanning, if present, is sporadic and manual.

- **Ethical**: Limited governance increases the risk of mishandling sensitive scan data. Basic scanners like Nessus may inadvertently access PII or system credentials during deep scans [28].
- **Economic**: Budget constraints dominate decision-making. Free or low-cost tools like OpenVAS offer an entry point but lack CI/CD integration or prioritisation logic [29].
- **Technical**: Manual workflows make it difficult to automate scans in Jenkins or GitHub Actions. Dev teams often lack the expertise to interpret scanner results [1].

Recommendation: Start with developer-friendly tools like GitHub Advanced Security or Snyk in a limited capacity. These offer low-friction setup and encourage security ownership without heavy infrastructure needs [10][42].

5.2 Risk-Informed (Tier 2): Reactive VM with Some Automation

Tier 2 organisations have begun integrating security into development but typically respond to vulnerabilities post-discovery.

- **Ethical**: Tools must now manage access to repo data, secrets, and API tokens. Policies must ensure scans don't expose sensitive metadata in multi-team environments [30].
- **Economic**: SaaS-based tools like Snyk and GitHub Advanced Security scale better than on-prem solutions, offering automation without needing full-time SecOps [31][43].
- **Technical**: Integrating tools with CI/CD systems like GitHub Actions or GitLab CI is feasible but requires planning to avoid pipeline slowdowns [32].

Recommendation: Implement continuous scanning using GitHub Advanced Security or Snyk. Begin prioritising vulnerabilities based on exploitability and system exposure, possibly with assisted triage features.

5.3 Repeatable (Tier 3): Proactive VM, Policy-Driven CI/CD Integration

Tier 3 represents security-mature organisations with enforced policies, integrated workflows, and dedicated DevSecOps roles.

- **Ethical**: Full transparency is required in how AI models prioritise vulnerabilities. Biases in ML-based scoring must be documented to ensure fairness [33].
- **Economic**: While premium tools like Orca Security incur higher licensing costs, they reduce long-term human effort through automated triage, risk-aware alerting, and compliance reporting [20].
- **Technical**: Seamless integration with cloud-native tools (e.g., AWS CodePipeline, Azure DevOps) is essential. Agentless scanning, SBOM generation, and real-time dashboards are expected [11].

Recommendation: Deploy AI-powered platforms like Orca or Wiz for real-time posture management. Integrate dynamic policies that adapt based on team, environment, and threat landscape.

Maturity Tier	**Ethical Considerations**	**Economic Factors**	**Technical Challenges**
Tier 1	Risk of mishandling sensitive data	Limited budget for security tools	Lack of integration with CI/CD pipelines
Tier 2	Managing access to sensitive metadata	Balancing cost with automation needs	Integrating tools without disrupting workflows
Tier 3	Ensuring transparency in AI decision-making	Higher upfront costs, long-term savings	Complex integration across diverse systems

Table 2. Ethical, Economic, and Technical Considerations Across Maturity Tiers.

Summary:

Maturity Tier	VM Strategy	Focus Area
Tier 1 (Partial)	Use **low-friction, developer-friendly tools** (e.g., GitHub Advanced Security, Snyk) to raise awareness of vulnerabilities early in development.	Awareness & Early Detection
Tier 2 (Risk-Informed)	**Automate vulnerability triage**, integrate basic pipeline scanners, and implement developer education on secure coding practices.	Workflow Automation & Education
Tier 3 (Repeatable)	Deploy **AI-powered, context-aware platforms** (e.g., Orca, Wiz) to enable continuous monitoring, prioritization, and proactive remediation.	AI-Driven Remediation & Scalability

Table 3. Summarisation of VM Strategies Across Maturity Tiers.

6. Predictive Insights: AI and Automation Over the Next 3–5 Years

As DevOps pipelines scale and evolve, artificial intelligence (AI) and automation are poised to become critical enablers of dynamic, resilient vulnerability management. In the next 3–5 years, these technologies will shift VM from reactive scanning to proactive, self-adjusting security layers embedded deeply in the CI/CD process.

6.1 Proactive Threat Detection at Scale

AI-driven systems are expected to perform continuous behavioral monitoring across codebases and infrastructure. By leveraging unsupervised machine learning, graph analytics, and threat intelligence fusion, these systems can detect novel anomalies and emerging vulnerabilities before CVEs are formally assigned. Orca Security and Wiz are examples of platforms leveraging agentless workload scanning and contextual prioritization to surface the most exploitable risks [20][34]. As more telemetry is collected across environments, these systems will continuously improve their threat-detection capabilities at scale.

6.2 Adaptive Security in Pipelines

Machine learning will also allow DevSecOps tools to adapt policies dynamically. For example, scanning frequency, severity thresholds, or remediation workflows can be tailored based on pipeline stage, project risk, and developer behavior. Contextual AI models can understand which libraries are most critical or which commits introduce the most change, allowing for adaptive enforcement [35]. Such models reduce false positives, minimize developer friction, and promote broader adoption. Over time, AI is expected to prioritize vulnerabilities based not just on CVSS scores but exploitability, privilege level, and exposure - refining triage and patch cycles.

6.3 Frictionless Compliance

AI can also automate the generation of Software Bills of Materials (SBOMs), audit artifacts, and compliance reports, particularly for regulations like SOC 2, HIPAA, and ISO 27001 [36]. With advances in natural language processing (NLP) and large language models (LLMs), smart assistants may help security teams interpret findings, auto-draft remediation steps, and even simulate auditor responses [37]. For low-maturity (Tier 1) teams, these capabilities will serve as a force multiplier, bringing enterprise-grade security assurance without requiring deep expertise or manual effort.

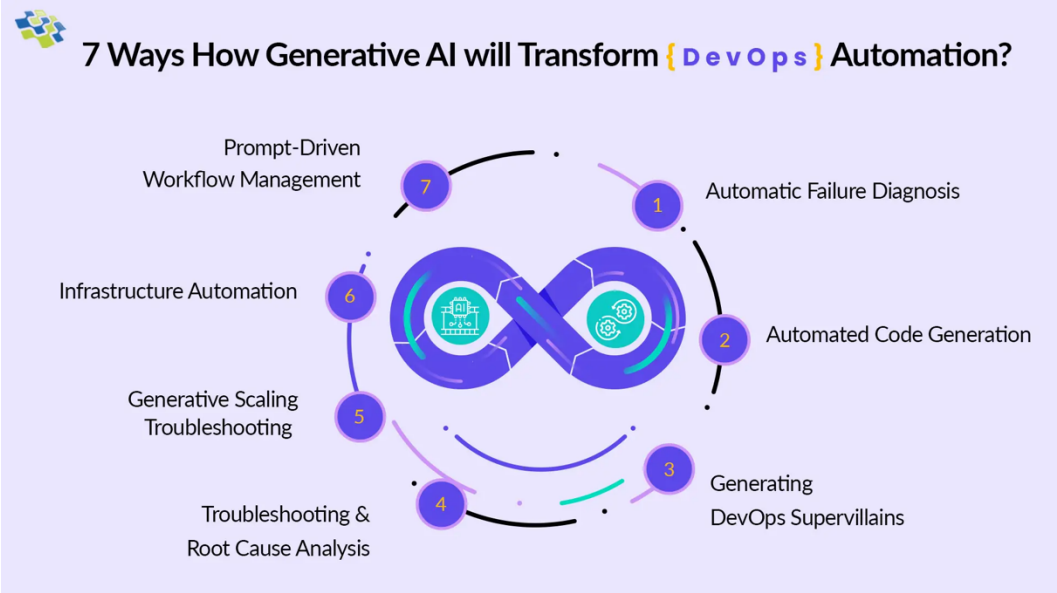

Figure 3. How AI transforming DevOps automation [47].

7. Synthesis and Recommendations

Effective vulnerability management (VM) requires aligning tool capabilities with team maturity, workflow velocity, and stakeholder influence. Our literature review and solution analysis revealed a consistent trend: traditional scanners (e.g., Nessus) offer strong compliance depth but struggle with CI/CD integration and developer alignment [38]. Developer-centric tools (e.g., GitHub Advanced Security, Snyk) encourage early detection but may lack infrastructure-wide visibility. AI-powered platforms (e.g., Orca, Wiz) provide adaptive, risk-aware prioritization and cloud-scale observability but require mature DevSecOps readiness [39].

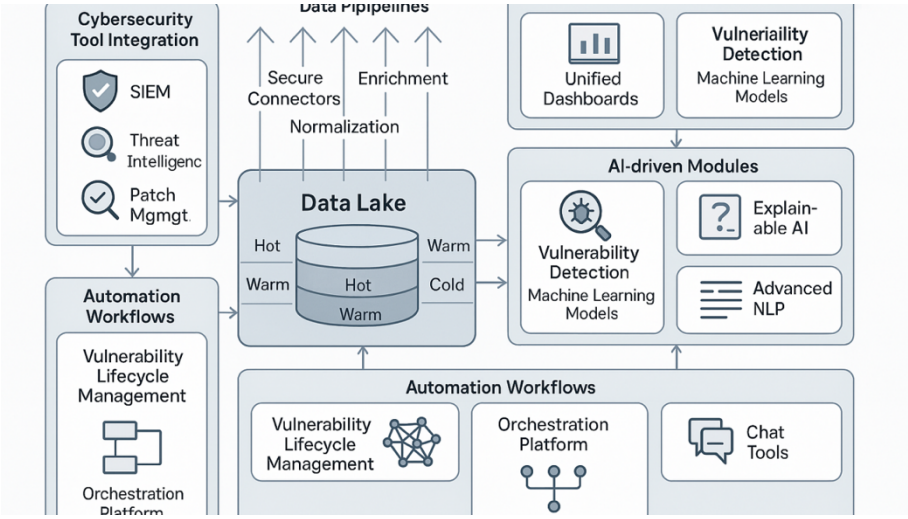

Figure 4. AI Infrastructure for Vulnerability Management [48].

For **small teams or Tier 1 (Partial)** organisations, GitHub Advanced Security or Snyk offer low-friction onboarding, IDE integration, and minimal configuration. As maturity advances to **Tier 2 (Risk-Informed)**, combining these with tools like Nessus for broader coverage becomes critical. For **Tier 3 (Repeatable)** or

enterprise settings, AI-enhanced platforms deliver scalable risk reduction and proactive threat prevention with lower operational burden.

We recommend a **phased roadmap**:

- **Phase 1**: Introduce in-repo scanning, SBOMs, and secret detection using dev-centric tools.
- **Phase 2**: Add infrastructure scanning, policy engines, and correlate risks with known assets.
- **Phase 3**: Integrate AI-based prioritization, auto-remediation playbooks, and self-healing security controls.

This roadmap supports scalable VM with minimal disruption, progressively introducing automation and contextual intelligence [44]. Stakeholder collaboration is key, developers ensure shift-left success, SecOps orchestrate coverage, and leadership provides strategic alignment and funding. Long term, AI will empower teams to operate secure-by-default pipelines without sacrificing speed.

8. Conclusion

Vulnerability management is no longer a back-office task; it is central to secure, continuous software delivery in DevOps environments. As organisations race to innovate, balancing speed with security has become a defining challenge. This report examined vulnerability management tools and frameworks through a DevSecOps lens, aligning solutions with organisational size and NIST CSF maturity levels.

We conclude that the most effective VM strategies are adaptive, automated, and stakeholder-aligned. Traditional scanners remain useful for compliance, but modern developer-centric and AI-powered solutions offer the agility and precision required in fast-paced environments. By embracing phased adoption and integrating tools contextually into CI/CD pipelines, organisations can transform vulnerability management from reactive patching into predictive, intelligent defense.

Ultimately, success depends not only on tool selection but also on team collaboration, cultural readiness, and investment in scalable automation. As AI matures, even small teams can achieve enterprise-grade security outcomes with the right practices.

9. References

[1] M. Howard and D. LeBlanc, *Writing Secure Code*, 2nd ed. Redmond, WA, USA: Microsoft Press, 2002.
[2] S. Bell, "Shift-left security: Embedding security earlier in DevOps," *DevOps.com*, Jul. 2021. [Online]. Available: https://devops.com/shift-left-security/
[3] National Institute of Standards and Technology, "Framework for Improving Critical Infrastructure Cybersecurity," NIST, 2018.
[4] Tenable Network Security, "Nessus: The industry's most widely deployed vulnerability scanner," Tenable, 2022. [Online]. Available: https://www.tenable.com/products/nessus
[5] W. Kandek, "Vulnerability Management Trends," *Qualys Security Blog*, 2013.
[6] M. Kim et al., "Rapid software release in DevOps: A survey of security implications," *Empirical Software Engineering*, vol. 24, no. 2, pp. 985–1011, 2019.
[7] M. Shahin, M. Ali Babar, and L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," *IEEE Access*, vol. 5, pp. 3909–3943, 2017.
[8] OWASP DevSecOps Guidelines, 2021. [Online]. Available: https://owasp.org/www-project-devsecops-guideline/
[9] J. Carter et al., "Security Challenges in Agile and DevOps: A Study of Practitioners' Perspectives," in *Proc. Int. Symp. Empirical Software Engineering and Measurement (ESEM)*, 2021.
[10] GitHub Docs, "GitHub Advanced Security," GitHub, 2023. [Online]. Available: https://docs.github.com/en/code-security
[11] Gartner, "Market Guide for Cloud-Native Application Protection Platforms," Gartner Research, 2022.
[12] E. Jouini and L. B. Rabai, "A Comparative Study of Cybersecurity Risk Assessment Frameworks and Methodologies," *Procedia Computer Science*, vol. 175, pp. 90–95, 2020.
[13] C. Sabottke, O. Suciu, and T. Dumitras, "Vulnerability Disclosure in the Age of Social Media: Exploiting Twitter for Predicting Real-World Exploits," in *Proc. USENIX Security Symposium*, 2015.
[14] Amazon Inspector Docs, "Automated Security Assessment Service," AWS, 2023. [Online]. Available: https://docs.aws.amazon.com/inspector/

[15] R. Mitchell, K. Chang, and J. Dyer, “Policy-as-Code for DevSecOps Pipelines,” in *Proc. IEEE Int. Conf. Software Architecture (ICSA)*, 2022.
[16] Tenable, “Nessus Professional,” [Online]. Available: https://www.tenable.com/products/nessus
[17] S. Kandek, “Patch Management: Keeping Systems Secure,” Qualys Security Blog, 2013.
[18] D. Nguyen and C. Yang, “Challenges of Integrating Legacy Security Tools into DevOps,” in Proc. of ACM SAC, 2019.
[19] R. Almashaqbeh, “Security Tools: Integration and Automation,” Cybersecurity Engineering Journal, vol. 11, no. 2, pp. 49–60, 2021.
[20] Orca Security, “Cloud Security for DevOps and Security Teams,” Orca Security, 2023. [Online]. Available: https://orca.security
[21] Gartner, “Innovation Insight for Cloud-Native Application Protection Platforms,” 2022.
[22] M. Adnan and J. Suh, “Agentless Cloud Security: A Case Study with Orca,” in IEEE Intl. Cloud Computing Conf., 2022.
[23] A. Jones and T. Henry, “Cost Considerations in Cloud Security Tools,” ACM CloudSec, 2021.
[24] Orca Docs, “CI/CD Integration,” [Online]. Available: https://docs.orca.security/
[25] J. Fett, “Using CodeQL to Secure Open Source,” GitHub Security Lab, 2021.
[26] OWASP, “DevSecOps Best Practices,” [Online]. Available: https://owasp.org/www-project-devsecops/
[27] M. Clement and S. Bartlett, “Secure Coding in CI/CD with GitHub Actions,” IEEE DevOps Days, 2022.
[28] R. Perdisci, A. Lanzi, and W. Lee, “Classification of Evasion-Resistant Malware with Structured Behavior Modeling,” in *Proc. of the 2008 ACM Workshop on Recurring Malcode (WORM)*, 2008, pp. 15–22.
[29] D. C. Dinev and A. M. Tanev, “A Comparative Analysis of Open Source Tools for Vulnerability Assessment,” in *Proceedings of the International Scientific Conference*, vol. 10, no. 3, pp. 124–128, 2021.
[30] A. Cardenas, S. Amin, and S. Sastry, “Secure Control: Towards Survivable Cyber-Physical Systems,” in *Proc. of the 28th Int. Conf. on Distributed Computing Systems Workshops (ICDCSW)*, 2008, pp. 495–500.
[31] D. L. Parnas, “Software Aspects of Strategic Defense Systems,” *Commun. ACM*, vol. 28, no. 12, pp. 1326–1335, Dec. 1985.
[32] R. Chandramouli and S. Rose, “Securing the Software Supply Chain: Recommended Practices for Software Bill of Materials (SBOMs),” NIST, Tech. Rep. NIST IR 8397, 2021.
[33] J. Li, Z. Liang, S. Guo, and M. Zulkernine, “Machine Learning for Automated Vulnerability Prioritization Using Real-World Exploit Data,” in *Proc. 2018 IEEE Intl Conf. on Software Quality, Reliability and Security (QRS)*, pp. 66–73, Jul. 2018.
[34] Wiz Security, “Context-Aware Cloud Security,” [Online]. Available: https://www.wiz.io/
[35] J. Mitchell and R. Saiedian, “Adaptive Threat Intelligence in DevSecOps Pipelines,” *Journal of Cybersecurity and Privacy*, vol. 2, no. 3, pp. 75–92, 2022.
[36] K. Goseva-Popstojanova et al., “AI-driven Approaches to Software Compliance Monitoring,” *IEEE Software*, vol. 39, no. 4, pp. 62–70, Jul. 2022.
[37] H. Chen and Y. Wang, “Automating Compliance via LLMs: A Practical Framework for Security Teams,” in *Proc. 2023 IEEE Intl. Conf. on AI Systems (ICAIS)*, pp. 303–310.
[38] R. Smith et al., “Balancing Depth and Speed: Evaluating VM Tools in CI/CD Pipelines,” *ACM DevOpsSec Journal*, vol. 6, no. 2, pp. 102–117, 2023.
[39] T. Zhao and M. Jamal, “Comparative Study of Cloud-native and AI-Enhanced Vulnerability Management,” *Journal of Systems and Software*, vol. 189, p. 111338, 2022.
[40] Microsoft, “Defender for Cloud Documentation,” 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/defender-for-cloud/
[41] Aqua Security, “Trivy: Vulnerability and Misconfiguration Scanner,” 2023. [Online]. Available: https://aquasecurity.github.io/trivy/
[42] Snyk, “Snyk Open Source,” 2023. [Online]. Available: https://snyk.io/product/open-source-security-management/
[43] Forrester, “The Forrester Wave™: Static Application Security Testing, Q3 2023,” 2023. [Online]
[44] K. Scarfone and P. Mell, “Guide to Enterprise Patch Management Technologies,” NIST SP 800-40, 2022.
[45] Nucleus Security, “A Brief History of Vulnerability Management,” *Nucleus Security*, [Online]. Available: https://nucleussec.com/blog/history-vuln-managment/. [Accessed: 11-May-2025].
[46] Cyber Sierra, “NIST CSF Maturity Levels: Everything You Need to Know,” *Cyber Sierra Blog*, [Online]. Available: https://cybersierra.co/blog/nist-csf-maturity-levels-everything-you-need-to-know/. [Accessed: 11-May-2025].
[47] NextGen Invent, “How Generative AI Will Transform DevOps Automation”. [Online]. Available: https://nextgeninvent.com/blogs/generative-ai-transform-devops-automation/. [Accessed: 11-May-2025].
[48] F. Jorge, “AI Infrastructure in Vulnerability Management,” *LinkedIn Articles*, [Online]. Available: https://www.linkedin.com/pulse/ai-infrastructure-vulnerability-management-felix-jorge-qyjle. [Accessed: 11-May-2025].