ISSN: 2581-3404 (Online) **IF: 5.86 (SJIF)**
IJIRTM, Volume-10, Issue-6, November-2026

A Review of Lightweight Vulnerability Assessment Tools and Their Role in Supporting Cyber security and GRC Workflows

Sapna[1], Palak[2], Vaishali Mehta[3]
Student, Department of Computer Science and Engineering (Cyber Security) [1,2]
Panipat Institute of Engineering and Technology, Samalkha, Panipat, India[1,2,3]
Affiliated to Kurukshetra University, Kurukshetra, India[1,2,3]
Professor[3]

Abstract. *Organizations frequently run vulnerability assessments first to discover network exposures and potential known weaknesses. Gathering this data has become straightforward thanks to tools like Nmap alongside public databases such as the National Vulnerability Database (NVD). Manual effort remains the tougher obstacle, involving the interpretation of scan results, validation of suspected flaws, and creation of standardized reports. This study examines lightweight vulnerability assessment tools together with their underlying standards and data sources, specifically focusing on network scanners, vulnerability databases, severity scoring systems, and reporting workflows. We also outline a basic prototype developed using Python, Streamlit, Nmap, and the NVD Common Vulnerabilities and Exposures (CVE) API, exploring its potential support for governance, risk and compliance (GRC) tasks. Four major shortcomings emerge: incomplete version detection, inaccurate or uncertain vulnerability matches, limited risk insight from severity scores alone, and a frequent lack of connection between technical findings and remediation. Ultimately, lightweight tools help reduce documentation overhead and improve reporting consistency, provided operators treat their output as provisional findings requiring careful human review.*

Keywords: Cyber security, GRC, network scanning, vulnerability assessment, vulnerability reporting.

Introduction

Any networked system presents accessible services that malicious actors can target. Unauthorized access, data loss, or service disruptions often follow if these services contain flaws. To combat this, companies conduct vulnerability assessments that inventory reachable hosts and services alongside any associated known weaknesses, allowing teams to plan patches proactively. Technical frameworks divide this process into planning, execution, and post-execution stages, with network discovery, port and service detection, and vulnerability scanning dominating the execution phase (Scarfone et al., 2008). Most of these foundational components are readily accessible today. For instance, Nmap functions as an open-source scanner capable of discovering hosts, identifying open ports, and frequently determining the underlying software running on them (Lyon, 2009; Nmap Project, n.d.-a). Public security flaws receive distinct identifiers through the Common Vulnerabilities and Exposures (CVE) Program (CVE Program, n.d.), while the National Vulnerability Database (NVD) enriches those entries with severity metrics, weakness

classifications, and product applicability details (National Institute of Standards and Technology [NIST], n.d.-b). Meanwhile, the Common Vulnerability Scoring System (CVSS) provides a standardized framework for evaluating how critical vulnerability actually is (FIRST, 2023a; Mell et al., 2006).

Gathering the raw data is rarely the difficult phase anymore. The real time sink comes afterward. Analysts must manually parse scan outputs, link identified products and versions to relevant vulnerability data, determine which matches are valid, and draft reports for peer review. Performing this entirely by hand is sluggish and frequently inconsistent across different assessments. Comprehensive suites like Tenable's Nessus and Greenbone's OpenVAS-based tools handle a large portion of this automation (Greenbone, n.d.; Tenable, n.d.), yet they still demand careful setup and human interpretation of their output. Smaller utilities that bridge the gap by combining a handful of open, transparent components occupy a middle ground between manual labor and heavy enterprise platforms, making them ideal for learners, lean teams, and initial screening.

This paper examines the underlying utilities, data repositories, and standards driving these streamlined methods, exploring their potential to aid governance, risk and compliance (GRC) efforts. Three core inquiries steer our evaluation: (a) the specific functions that scanners, vulnerability repositories, and scoring frameworks fulfill during an assessment, (b) the factors that constrain the precision and utility of workflows constructed from them, and (c) the ways structured reporting can assist GRC initiatives without being confused with actual risk or compliance verdicts. Serving as a narrative examination of official documentation, established standards, and select peer-reviewed research, this study avoids acting as a systematic literature survey and includes no empirical experiments. None of the content provided is intended to substitute for professional security audits or an enterprise's internal risk management choices.

Background information, previous studies, and a review of current tools make up the following sections of the document. We also point out existing shortcomings, connect the work to GRC, present a streamlined workflow, and wrap up with final thoughts and future directions.

Background and Related Work

Network Reconnaissance

At its core, network reconnaissance is the process of identifying reachable hosts and services across a network. Nmap stands out as a primary utility for this task, managing host discovery, port scanning, service and version detection, along with operating system identification (Lyon, 2009; Nmap Project, n.d.-a). When a port gets scanned, the tool assigns a status like open, closed or filtered based on the target response rather than indicating any security flaw (Nmap Project, n.d.-a). Speed and coverage depend directly on the volume of ports examined. Standard behavior has Nmap reviewing a preset group of frequent ports, though users can adjust settings to inspect a smaller subset for quick results or all 65,535 possibilities for thoroughness (Nmap Project, n.d.-b).

Service and version detection takes the process a stage further. Additional probes go out to active ports, and Nmap matches the returned data against a library of known signatures to determine the application name, edition, and frequently a Common Platform Enumeration (CPE) identifier (Nmap Project, n.d.-b). Success relies on how the service replies, and firewalls or proxies can easily alter, mask, or block those answers. Treat the results as clues about a service rather than absolute confirmation of its specific build. Security assessment guidance stresses (Scarfone et al., 2008) that scanning must only occur on systems where the operator holds direct authorization.

Vulnerability Databases and Identifiers

Distinct identifiers are distributed by the CVE Program for publicly known vulnerabilities, allowing tools, vendors, and analysts to discuss the exact same security flaw (CVE Program, n.d.). The NVD expands upon that catalog by incorporating CVSS scores, Common Weakness Enumeration (CWE) categories, and details regarding impacted products (NIST, n.d.-b). Fetching this information programmatically is possible via the NVD vulnerability application programming interface (API; NIST, n.d.-d), while a secondary CPE API allows users to search the product directory (NIST, n.d.-a).

CPE provides a structured naming framework for hardware and software, serving as the primary method linking vulnerability records to specific versions and products (NIST, n.d.-a). MITRE manages CWE, which pairs with CVE by detailing the underlying weakness categories driving specific security flaws. Developers and assessors rely on its annual compilation of top dangerous weaknesses to identify recurring trends (MITRE, n.d.-a, n.d.-b). These distinct identifiers all fall under the broader Security Content Automation Protocol (SCAP) initiative, an effort aimed at standardizing how security tools describe and exchange vulnerability data (NIST, n.d.-c). A single caveat matters here. Because newly published records often lack complete analysis or product mapping details, failing to find a match in the NVD does not guarantee a system is free from that vulnerability (NIST, n.d.-b).

Severity Scoring and Risk

CVSS started out as an open framework designed to evaluate the severity levels of software vulnerabilities (Mell et al., 2006). The (FIRST, 2023a) metric groups in version 4.0 span base, threat, environmental, and supplemental categories, producing scores from 0.0 up to 10.0 alongside qualitative labels ranging from none to critical. The creators emphasize that CVSS evaluates sheer severity instead of actual risk, meaning teams must pair the base score with real-world threat intelligence and context regarding the target environment (FIRST, 2023a, 2023b).

Vulnerability alone does not dictate the level of risk. NIST's risk assessment guidance treats risk as a function of how likely a threat event is and how much damage it would do, which in practice depends on the value of the asset, how exposed it is and what safeguards exist (Joint Task Force Transformation Initiative, 2012). Studies also suggest that severity scores should not be relied upon exclusively. Allodi and Massacci (2014) found that CVSS scores were weak indicators of whether vulnerabilities were exploited in the wild. The Exploit Prediction Scoring System was later proposed to estimate the probability of exploitation from observed data (Jacobs et al., 2021). Practical intelligence regarding exploitation is also available from several sources. The Cyber security and Infrastructure Security Agency [CISA] (n.d.), for example, keeps a catalog of vulnerabilities known to be exploited in the wild. NIST's patch management guidance also stresses planning maintenance around how critical assets are and what threats they face, instead of relying on severity alone (Souppaya & Scarfone, 2022).

Existing Solutions and Guidance

NIST Special Publication 800-115 details various methods for security testing and evaluation. It cautions that scan findings require verification because automated software frequently generates false positives (Scarfone et al., 2008). Scanners such as the Greenbone Community Edition, which includes OpenVAS, and Tenable's Nessus run vulnerability checks against hosts and services (Greenbone, n.d.; Tenable, n.d.).

Different approaches are necessary for web applications. The OWASP Web Security Testing Guide and the OWASP Top 10 cover testing procedures and common risks at the application layer, which a port scan and a version lookup cannot show (OWASP Foundation, n.d.-a, n.d.-b).

Review of Existing Tools and Approaches

Every available tool or standard addresses only a single phase of the assessment workflow, and no single option provides complete coverage. Nmap identifies exposed network services (Nmap Project, n.d.-a), vulnerability databases describe known weaknesses (NIST, n.d.-b) and CVSS communicates severity (FIRST, 2023a). Table 1 sums up the main purpose and a typical limitation of each.

Table 1: Comparison of Tools and Approaches Used in Vulnerability Assessment

Tool or approach	Main purpose	Limitation
Nmap (Lyon, 2009; Nmap Project, n.d.-a)	Port scanning and service detection	Does not confirm every vulnerability; version inference may be inaccurate
CVE and NVD (CVE Program, n.d.; NIST, n.d.-b)	Vulnerability identification and information	Matches require validation; records may be incomplete
CVSS (FIRST, 2023a)	Severity scoring	Does not represent complete business risk
CISA KEV catalog (CISA, n.d.)	Evidence of exploitation in the wild	Lists only known exploited vulnerabilities; absence is not evidence of safety
OpenVAS and Nessus (Greenbone, n.d.; Tenable, n.d.)	Comprehensive vulnerability scanning	Require configuration and interpretation
Lightweight integrated tool	Scanning, lookup, and reporting in one interface	Depends on data quality and matching accuracy

Note. Limitations summarize points discussed in this review and are not exhaustive. KEV = Known Exploited Vulnerabilities.

Discovery and Comprehensive Scanning

Discovery and description of services are the main purposes of Nmap. Tools like Nessus and OpenVAS take things a step further by executing vulnerability checks and generating findings (Greenbone, n.d.; Lyon, 2009; Tenable, n.d.). That extra depth comes with increased complexity. Comprehensive scanners demand configuration, frequent updates, and careful policy adjustment, leaving analysts to sort through extensive lists of outputs. In contrast, a streamlined tool built on Nmap sacrifices deep vulnerability checks to gain simplicity and clarity, since every potential finding ties directly back to a visible service banner and a public database entry.

Knowledge Sources and Prioritization

Context from databases, scoring systems, and exploitation catalogs is essential for converting a discovered service into a prioritized result. The CVE and NVD databases outline the details of known weaknesses (CVE Program, n.d.; NIST, n.d.-b), CVSS assigns a severity metric (FIRST, 2023a), and references like the CISA catalog alongside exploit prediction models indicate the probability of active exploitation (CISA, n.d.; Jacobs et al., 2021). Armed with all three perspectives, an analyst can easily distinguish between a severe flaw that exists only in theory and one that active attackers are currently exploiting.

Automation and Consistency

Consistency improves through automation, yet errors can just as easily be multiplied. When a service is incorrectly labeled, the subsequent database search may produce results completely unrelated to the target system. Because of this risk, reports ought to maintain a clear division between direct observations—like open ports and retrieved banners—and vulnerabilities derived solely from database correlations, while explicitly noting that these inferred issues require manual verification (Scarfone et al., 2008; Souppaya & Scarfone, 2022).

Scope of Web and Application Testing

Different tools operate at distinct layers of the technology stack. Network-level scanners evaluate exposed services, whereas application-level testing focuses on business logic, input validation, authentication, and session controls within web software (OWASP Foundation, n.d.-b). The OWASP Top 10 organizes prevalent application vulnerabilities into structured categories that serve as a common language for both developers and testers (OWASP Foundation, n.d.-a). Relying on port scanning and CVE matching can highlight known flaws in standard software libraries, yet it remains blind to vulnerabilities hidden inside custom application code. Documentation must clarify this limitation so that a brief set of candidate findings is never mistaken for a guarantee of application security.

Gaps Identified

Four frequent shortcomings are highlighted in the review, restricting the dependability and practical value of streamlined, automated assessment processes.

Version Detection

Precise identification relies on knowing the exact product and version, yet detecting versions frequently proves flawed or incomplete. Services often conceal or alter their banners, operate behind proxies, or receive security updates while keeping the identical reported version string. Tools must either bypass the lookup entirely or broaden their search when version data is missing or ambiguous, and both scenarios undermine reliability (NIST, n.d.-a; Nmap Project, n.d.-b).

False Matches

Just because a CVE record appears associated with a discovered product does not mean the underlying system is vulnerable. Whether a flaw actually exists or can be exploited depends entirely on the active configuration, turned-on features, patch levels, and any mitigating controls in place. Broad searches or queries with minimal filters frequently pull up entries for entirely different products or similarly named components, generating false positives (NIST, n.d.-b, n.d.-d). Assessment guidance treats validation as a separate step for this reason (Scarfone et al., 2008).

Risk Prioritization

Before a CVSS score can guide any business choice, it requires proper context. For instance, a medium-severity flaw on an exposed, mission-critical system often demands greater priority than a high-severity flaw tucked away on an isolated test machine (FIRST, 2023b; Joint Task Force Transformation Initiative, 2012). While proof of active exploitation provides additional context (Allodi & Massacci, 2014; CISA, n.d.), automated scanners generally remain blind to asset criticality unless the operator explicitly provides it.

Integration

Technical findings hold minimal value unless they are properly documented and tied directly to remediation alongside issue tracking. Recommendations for patch management emphasize structured, predictable maintenance rather than random, ad hoc patches (Souppaya & Scarfone, 2022), while guidelines for assessments explicitly treat reporting and mitigation as core phases of the assessment lifecycle (Scarfone et al., 2008). Numerous lightweight utilities halt their work right at the report generation stage, forcing teams to manually bridge the gap to ticketing platforms, asset inventories, and risk registries.

Table 2 outlines the root cause behind each deficiency along with potential remedies derived from the literature examined in this work.

Table 2: Potential Remedies and Shortcomings within Lightweight Assessment Procedures

Gap	Main cause	Possible mitigation
Version detection	Hidden or altered banners; incomplete probe responses	Use CPE output where available; flag missing versions; allow manual correction
False matches	Loose keyword matching; unknown configuration	Prefer CPE-based queries; label results as candidates; verify with vendor advisories
Risk prioritization	Severity is not the same as risk	Add asset criticality and exploitation evidence
Integration	Workflow ends at the report	Export structured findings; link to issue tracking

Note. These proposed fixes pull from recommendations provided by the National Institute of Standards and Technology (NIST, n.d.-a; Scarfone et al., 2008) and serve as ideas instead of proven fixes.

Such shortcomings highlight an operational requirement for verification procedures alongside uniform, transparent reporting that strictly separates raw facts from logical conclusions.

Relevance to GRC Workflows

GRC integrates governance, risk management and compliance functions, assisting enterprises in driving toward their objectives, managing unpredictability and satisfying regulatory duties. Well-organized vulnerability documents aid these efforts across three primary categories.

Evidence Collection

Such a document documents the boundaries of an evaluation, discovered services, suspected flaws and their corresponding severity metrics. This documentation serves as proof that vulnerability identification

actually occurred, enabling auditors to determine precisely what was included or excluded (Scarfone et al., 2008).

Risk Assessment

Technical results integrate directly into risk assessment procedures after merging with asset criticality levels and the business impact of a breach (Joint Task Force Transformation Initiative, 2012; Souppaya & Scarfone, 2022). A document capturing severity alongside discovered services and exposure levels provides risk managers a solid starting point, though it stops short of performing the complete analysis for them.

Remediation

Keeping report formats uniform is also useful for recording discoveries that require deeper analysis and for backing up remediation efforts. When every issue uses the same layout, assigning responsibility, monitoring progress, and contrasting current results with previous audits becomes much simpler.

Wider standards provide the company setting for these discoveries. The NIST Cyber security Framework 2.0 groups security results into six core categories, adding a fresh Govern category focused on governance and planning (Pascoe et al., 2024). The CIS Critical Security Controls treat ongoing vulnerability tracking as a core safeguard (Center for Internet Security [CIS], n.d.). NIST Special Publication 800-53 includes vulnerability tracking and checking within its risk evaluation group (Joint Task Force, 2020), while ISO/IEC 27001:2022 places technical vulnerability oversight among its data protection safeguards (International Organization for Standardization [ISO], 2022).

Generating reports automatically reduces tedious paperwork, though a human must still check the results. Automated scanners cannot independently confirm compliance or replace GRC professionals, who evaluate discoveries against company rules, legal mandates, and risk thresholds.

Proposed Lightweight Assessment Approach

Technology Stack

The suggested method relies on Python for the backend and Streamlit to build the web UI (Streamlit, n.d.). Nmap, driven via a Python Nmap wrapper, handles the scanning tasks (Nmap Project, n.d.-a), while the NVD CVE API pulls potential vulnerability entries (NIST, n.d.-d). Rule logic structures the findings and attaches initial remediation advice, while a reporting tool creates an exportable file. Because this logic follows strict rules rather than machine learning models, its logic remains clear and audibly verifiable. On the other hand, it stays limited strictly to what its programmed rules can detect.

Workflow

The process moves through the following stages:

1. The operator inputs a permitted destination and selects a scanning configuration.
2. Nmap discovers active ports and gathers accessible service data.
3. The application inspects the identified software and version specifics.
4. Matching vulnerability entries are fetched from the NVD.
5. Preliminary guidance and structured findings are produced through rule-based logic.
6. The generated report contains the scan results, potential matches, severity levels, and existing limitations.

Users can choose among three scan profiles, namely Quick, Standard, and Full, balancing speed against the extent of port coverage. This design reflects Nmap's option to target fewer ports, a default group, or the entire port spectrum (Nmap Project, n.d.-a). For every potential match, the tool displays the CVE

description, CVSS score, and severity rating provided by the NVD (NIST, n.d.-d), while the documentation designates them as candidates rather than verified flaws.

Expected Benefits and Limitations

Consolidating scanning, lookup functions, and reporting into a single platform can streamline the gathering and display of assessment results. Manual data entry decreases alongside documentation efforts, proving helpful for educational contexts and early evaluations. These advantages are still theoretical and require validation through subsequent testing.

Constraints demand equal attention. An open port does not guarantee a flaw exists, and candidate CVE matches require confirmation through manual review, authenticated checks, or vendor advisories (Scarfone et al., 2008). Outcomes rely heavily on version detection accuracy and NVD data completeness (NIST, n.d.-b; Nmap Project, n.d.-b). Because this methodology omits application-layer testing, it needs to be paired with techniques like those outlined by the OWASP Foundation (n.d.-b). The system serves assessment and reporting purposes and cannot replace thorough vulnerability testing.

Scan Profile Trade-offs

Selecting a scan profile requires weighing speed, coverage, and network impact. A quick scan targeting only common ports finishes rapidly and generates minimal traffic, though it might overlook services operating on atypical ports. Scanning all 65,535 ports offers far greater thoroughness, but it demands more time and generates heavy traffic that can disrupt congested or sensitive environments (Nmap Project, n.d.-a). A standard profile balances these extremes. Within academic environments involving lab targets, the full profile rarely creates issues. When targeting operational systems, the standard practice is to begin with lighter scans and escalate to a full scan solely with justification and explicit permission from the owner (Scarfone et al., 2008).

Design Considerations

Several architectural decisions influence the dependability of a lightweight utility. Request volume to the NVD is governed by public rate caps, which restrict unauthenticated clients more severely (NIST, n.d.-d). Because of this, software generating frequent requests needs to implement throttling mechanisms and cache previous lookups for identical software and release combinations. Precision matters too, meaning searches ought to leverage all available data specificity. Whenever Nmap provides a CPE identifier, querying with it yields higher accuracy than relying on generic keyword searches (NIST, n.d.-a; Nmap Project, n.d.-b). Lastly, the user interface ought to distinctly separate empirical data from derived assumptions, ensuring users can instantly differentiate measured facts from matched estimates.

The layout of a document is just as critical as the thoroughness of the scan. Official assessment advice outlines documentation that details the objectives, boundaries, techniques applied, discoveries, and suggested remediation steps (Scarfone et al., 2008). A streamlined utility mirrors this framework by incorporating the target details, scan profile, execution timestamp, identified services, potential CVE pairings along with their ratings, initial advice, and a clear disclaimer of constraints. Including these limitations in every single output prevents future readers from misinterpreting possible vulnerability matches as verified flaws.

Ethical and Legal Considerations

Security scanners carry the potential to interrupt live services and face risks of improper application. Such utilities must strictly target systems owned by the operator or explicitly authorized for testing under a defined boundary (Scarfone et al., 2008). Displaying this restriction clearly within both the software interface and the resulting document establishes proper usage boundaries. Operators must also evaluate

how aggressive scan configurations might impact delicate or production environments, opting for the gentlest profile that still fulfills the assessment goals.

Implications for Practice

Educational settings and compact teams benefit most from lightweight procedures through skill development and early evaluations. Observing how a discovered service, software edition, CVE entry, and impact score connect with each other fosters the core knowledge required for mastering enterprise platforms. For larger enterprises, this category of tool functions ideally as an initial screening mechanism that supplies a recorded baseline for further investigation. Any notable discoveries demand review by a skilled analyst, confirmation through secondary methods, and prioritization factoring in the specific asset and its operational importance (Joint Task Force Transformation Initiative, 2012; Scarfone et al., 2008).

Scan results also go stale. Services change, patches are applied and new vulnerabilities are published all the time, so a report only reflects the system at the time of the scan and the NVD at the time of the query (NIST, n.d.-b). Repeating assessments on a regular schedule and comparing the results over time turns individual reports into a record of how exposure changes.

Limitations of This Review

This review has limits of its own. It is a narrative review, so we did not run a systematic search or apply formal inclusion criteria, and the choice of sources reflects our own judgment about what is most authoritative for each topic. Many of the sources are standards and web documentation, which are revised from time to time, and details such as NVD interface behavior or CVSS versions may change after this paper is written. We also did not test any tool experimentally, so statements about the workflow's benefits describe what it is designed to do, not measured results. Readers should check current documentation before relying on any specific detail.

Conclusion and Future Directions

The utility of lightweight vulnerability assessment tools within cyber security operations and GRC frameworks was examined in this evaluation. Different phases of an assessment rely on specific tools and databases, namely Nmap, CVE, NVD, and CVSS, which handle reconnaissance, detection, and severity scoring (FIRST, 2023a; NIST, n.d.-b; Nmap Project, n.d.-a). Technical data organization and GRC documentation support can be streamlined through a straightforward workflow integrating scanning, vulnerability data retrieval, rule-driven advice, and reporting. Success ultimately hinges upon precise input, thorough validation of potential issues, and expert human review. Subsequent research could concentrate on refining product and version correlation, assessing false positive frequencies within controlled testing conditions, incorporating threat intelligence like the CISA catalog or exploit prediction scores (CISA, n.d.; Jacobs et al., 2021), enhancing layout design, and linking results directly to asset inventories and patch management systems. Real-world user testing would also help determine if these solutions genuinely lighten workloads while maintaining thorough assessment standards.

References

1. Allodi, L., & Massacci, F. (2014). Comparing vulnerability severity and exploits using case-control studies. ACM Transactions on Information and System Security, 17(1), Article 1. https://doi.org/10.1145/2630069.

2. Center for Internet Security. (n.d.). CIS critical security controls. https://www.cisecurity.org/controls
3. CVE Program. (n.d.). Common vulnerabilities and exposures. https://www.cve.org/
4. Cyber security and Infrastructure Security Agency. (n.d.). Known exploited vulnerabilities catalog. https://www.cisa.gov/known-exploited-vulnerabilities-catalog
5. FIRST. (2023a). Common vulnerability scoring system version 4.0: Specification document. https://www.first.org/cvss/v4.0/specification-document
6. FIRST. (2023b). Common vulnerability scoring system version 4.0: User guide. https://www.first.org/cvss/v4.0/user-guide
7. Greenbone. (n.d.). Greenbone community edition documentation. https://greenbone.github.io/docs/latest/
8. International Organization for Standardization. (2022). Information security, cyber security and privacy protection—Information security management systems—Requirements (ISO/IEC 27001:2022). https://www.iso.org/standard/27001
9. Jacobs, J., Romanosky, S., Edwards, B., Adjerid, I., & Roytman, M. (2021). Exploit prediction scoring system (EPSS). Digital Threats: Research and Practice, 2(3), Article 20. https://doi.org/10.1145/3436242
10. Joint Task Force. (2020). Security and privacy controls for information systems and organizations (NIST Special Publication 800-53, Rev. 5). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-53r5
11. Joint Task Force Transformation Initiative. (2012). Guide for conducting risk assessments (NIST Special Publication 800-30, Rev. 1). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-30r1 Lyon, G. F. (2009). Nmap network scanning: The official Nmap project guide to network discovery and security scanning. Insecure.Com LLC.
12. Mell, P., Scarfone, K., & Romanosky, S. (2006). Common vulnerability scoring system. IEEE Security & Privacy, 4(6), 85–89. https://doi.org/10.1109/MSP.2006.145
13. MITRE. (n.d.-a). Common weakness enumeration (CWE). https://cwe.mitre.org/
14. MITRE. (n.d.-b). CWE top 25 most dangerous software weaknesses. https://cwe.mitre.org/top25/
15. National Institute of Standards and Technology. (n.d.-a). CPE API. National Vulnerability Database. https://nvd.nist.gov/developers/products
16. National Institute of Standards and Technology. (n.d.-b). National vulnerability database. https://nvd.nist.gov/
17. National Institute of Standards and Technology. (n.d.-c). Security content automation protocol (SCAP). https://csrc.nist.gov/projects/security-content-automation-protocol
18. National Institute of Standards and Technology. (n.d.-d). Vulnerability APIs. National Vulnerability Database. https://nvd.nist.gov/developers/vulnerabilities
19. Nmap Project. (n.d.-a). Nmap reference guide. https://nmap.org/book/man.html
20. Nmap Project. (n.d.-b). Service and version detection. https://nmap.org/book/man-version-detection.html
21. OWASP Foundation. (n.d.-a). OWASP top 10. https://owasp.org/www-project-top-ten/
22. OWASP Foundation. (n.d.-b). Web security testing guide. https://owasp.org/www-project-web-security-testing-guide/

23. Pascoe, C., Quinn, S., & Scarfone, K. (2024). The NIST cyber security framework (CSF) 2.0 (NIST Cyber security White Paper 29). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.CSWP.29
24. Scarfone, K., Souppaya, M., Cody, A., & Orebaugh, A. (2008). Technical guide to information security testing and assessment (NIST Special Publication 800-115). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-115.
25. Souppaya, M., & Scarfone, K. (2022). Guide to enterprise patch management planning: Preventive maintenance for technology (NIST Special Publication 800-40, Rev. 4). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-40r4.