SPECTRUM OF ENGINEERING SCIENCES
ISSN (E) 3007-3138 (P) 3007-312X VOLUME 4, ISSUE 3, 2026

AUTOCYBERX: AN ORCHESTRATED OPEN-SOURCE FRAMEWORK FOR AUTOMATED PENETRATION TESTING AND RISK ANALYSIS

Syed Hamdan Zulqurnain[*1], Sobia Asher[2], Izhan Issah Aryeetey[3], Hassan Ahmed Abbasi[4]

[*1,2,3,4]*Department of Computer Science & Information Technology Sir Syed University of Engineering & Technology Karachi, Pakistan*

[*1]syedhamdanzulqurnain@gmail.com,[2]sobia.asher@ssuet.edu.pk,[3]izhanissah10@gmail.com, [4]hassanabbasius19@gmail.com

DOI: https://doi.org/10.5281/zenodo.21801959

Keywords
Automated Penetration Testing, Vulnerability Assessment, Security Tool Orchestration, CVSS Scoring, Compliance Mapping, Open-Source Security, Execution Profiles.

Article History
Received: 01 January 2026
Accepted: 17 March 2026
Published: 31 March 2026

Corresponding Author: *
Syed Hamdan Zulqurnain

Abstract
Manual penetration testing remains resource-intensive and time-consuming, while commercial automated platforms impose high subscription licensing costs that restrict small-to-medium enterprise adoption. Existing open-source security utilities perform individual verification checks effectively, but require security practitioners to manually chain disjointed tools together and parse inconsistent command outputs. This paper presents AutoCyberX, a self-hosted desktop penetration testing platform built to automate workflows from target discovery to report generation within a unified, rule-driven pipeline. Operating on a Kali Linux environment, AutoCyberX integrates a Flask backend, a single-page web interface, and a local SQLite database to execute scans locally without cloud dependencies. The system orchestrates over 15 open-source security tools through a rule-based decision engine that dynamically selects appropriate assessment modules across four fully implemented execution profiles. Controlled evaluation across 50 supervised live websites demonstrates that AutoCyberX reduced redundant tool execution by 38.5% and filtered out 64.1% of unverified false positives while effectively automating vulnerability assessment, multi-framework compliance mapping (OWASP Top 10, PCI-DSS v4.0, ISO/IEC 27001), and executive PDF reporting.

INTRODUCTION

The rapid expansion of web applications and network infrastructure has heightened the frequency of cyber attacks, making continuous security audits essential [3]. Penetration testing plays a vital role in discovering vulnerabilities before malicious actors exploit them [1], [2]. Standard assessment procedures require systematic execution across reconnaissance, service fingerprinting, active scanning, risk analysis, compliance mapping, and technical reporting [1]. Executing these steps manually requires specialized technical expertise and substantial execution time, making frequent manual evaluations costly for growing organizations [6].

To lower assessment overhead, automated vulnerability scanners have become standard security infrastructure [5]. Commercial assessment platforms deliver end-to-end scanning and consolidated reporting, but their high licensing costs remain prohibitive for small-to-medium enterprises (SMEs). Conversely, the open-source security

ecosystem provides highly capable engines for specific checks, such as port scanning (Nmap [8]),

directory enumeration, and vulnerability verification (Nuclei [9]). However, operating these utilities independently forces security personnel to run separate tools, parse inconsistent log formats, manually cross-reference findings, and compile consolidated reports from scratch [7]. Crucially, individual scanners frequently generate isolated false positives that require tedious manual validation [12].

To bridge the gap between costly commercial suites and disjointed open-source utilities, this paper introduces AutoCyberX. AutoCyberX is a self-hosted, desktop-based penetration testing platform running on Linux security environments (e.g., Kali Linux). The system combines open-source scanning engines with custom orchestration scripts controlled by a rule-based decision engine. By automating target validation, technology discovery, active vulnerability scanning, CVSS v3.1 scoring [10], multi-framework compliance mapping, false-positive reduction [12], and report generation, AutoCyberX streamlines security workflows while retaining human verification for confirmed findings [11].

The principal contributions of this paper are summarized below:

- A rule-based pipeline that dynamically selects and executes security tools based on real-time reconnaissance data, achieving a 38.5% reduction in redundant tool execution [5].
- A decoupled system architecture utilizing a Flask backend, a responsive single-page frontend, and a local SQLite database to guarantee total data privacy without external API requirements.
- Four fully functional, configurable execution profiles (Profile 1 – Rapid Reconnaissance, Profile 2 – Comprehensive Assessment, Profile 3 – Extended Assessment, and Profile 4 – Autonomous Assessment) designed to balance scan depth and execution time across operational constraints.
- An intelligent engine that normalizes raw vulnerability findings, computes CVSS scores [10], maps control deficiencies across OWASP Top 10:2021 [3], PCI-DSS v4.0, and ISO/IEC 27001:2022, and synthesizes remediation guidance.
- An automated cross-correlation module (utilizing CVE Matcher and Credential Tester) that filters out 64.1% of unverified alerts [12] and generates structured PDF reports.

The remainder of this paper is organized as follows. Section II reviews related literature in automated penetration testing and tool orchestration. Section III details the proposed system architecture, assessment profiles, and compliance mapping module. Section IV presents the experimental setup, while Section V analyzes empirical results, false-positive mitigation, and comparative benchmarks. Section VI concludes the paper and outlines future research directions.

II. RELATED WORK AND LITERATURE REVIEW

Automated penetration testing and vulnerability management have evolved through tool development and security orchestration [2], [5], [11]. Early automated tools focused on rule-based network scanning and static web checks. Lyon [8] established foundational techniques for OS fingerprinting and port scanning with Nmap, though standalone execution required manual interpretation.

A. Vulnerability Scanning and Tool Orchestration

Recent literature emphasizes structured frameworks to categorize security testing procedures and software vulnerabilities. Denis et al. [2] provided standard taxonomies for penetration testing strategies, while Khan et al. [7] mapped engineering approaches to systematic software security. However, early security practices lacked dynamic adaptive control, running static tools against every host regardless of service relevancy [6]. Islam et al. [5] emphasized that security orchestration platforms must bridge the gap between disjointed tools through dynamic execution flows. AutoCyberX expands on this by implementing a dynamic rule matrix that evaluates target characteristics during initial discovery and selectively executes downstream assessment tools.

B. Dynamic Assessment Engines and False-Positive Mitigation

Integration of automated decision engines in cybersecurity assessments has gained significant attention [12]. NIST SP 800-115 [1] emphasizes that local, controlled execution prevents external

sensitive data exposure. While modern AI-driven frameworks and large language models focus on automated vulnerability remediation [4] or prioritized risk scoring [11], rule-driven orchestration provides consistent, deterministic results without external model dependencies or hallucination risks. Moreover, isolated scanners suffer from elevated false-positive rates; multi-engine verification frameworks described by Antunes and Vieira [12] demonstrate that cross-correlating findings across independent tools significantly improves signal-to-noise ratio. AutoCyberX utilizes deterministic, rule-based filtering and cross-tool verification to ensure predictable scanning behavior, false-positive reduction, and local data processing.

III. METHODOLOGY AND SYSTEM ARCHITECTURE

AutoCyberX is structured as a modular, local desktop system. The application operates entirely on a Linux host (e.g., Kali Linux), ensuring that scan results, target parameters, and generated documentation remain stored locally in an SQLite database. The system offers a single-page web dashboard along with a command-line interface (CLI) to control and monitor scanning operations.

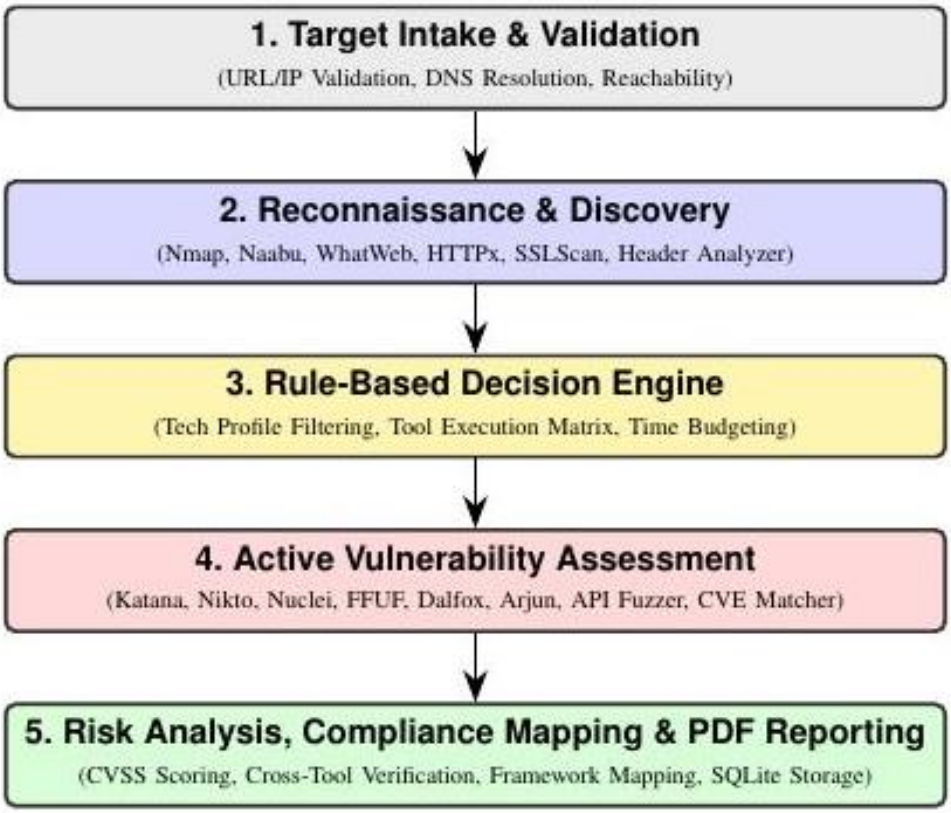

Fig. 1. System architecture and rule-driven execution pipeline of AutoCyberX.

A. Pipeline Execution Stages

As illustrated in Fig. 1, AutoCyberX operates through a sequential 5-stage pipeline:

1) Validates target syntax, resolves DNS records, and verifies network reachability prior to scanner invocation.

2) Identifies open ports (Nmap [8], Naabu), fingerprint services (WhatWeb, HTTPx), and

analyzes TLS configurations (SSLScan, Header Analyzer).
3) Dynamically evaluates reconnaissance metadata to filter out irrelevant tool modules (e.g., bypassing REST API fuzzers when web APIs are absent).
4) Executes selected assessment utilities (Katana, Nikto, Nuclei [9], FFUF, Dalfox, Arjun, API Fuzzer, CVE Matcher) in parallel or sequence.
5) Normalizes raw tool outputs, performs cross-tool verification, calculates CVSS v3.1 scores [10], maps findings to security frameworks, and generates PDF reports.

B. Module Component Overview

AutoCyberX integrates open-source utilities alongside custom execution modules to manage workflow orchestration, as summarized in Table I.

TABLE I MODULE COMPONENTS IN AUTOCYBERX

Component	Purpose and Functionality
Nmap / Naabu	Performs port scanning and service fingerprinting [8].
WhatWeb / HTTPx	Identifies web technologies and active endpoints.
SSLScan / TestSSL	Evaluates SSL/TLS cipher suites and configurations.
Header Analyzer	Evaluates HTTP response headers.
Katana	Crawls web endpoints to map application structure.
Nikto / Nuclei	Scans web applications using template checks [9].
FFUF / Gobuster	Discovers hidden directories, files, and endpoint paths.
Dalfox	Performs cross-site scripting (XSS) parameter testing.
Arjun	Discovers hidden HTTP query parameters.
CVE Matcher	Correlates service versions with vulnerability databases.
API Fuzzer	Tests REST API endpoints for unexpected input handling.
Credential Tester	Checks target interfaces for default credentials.
Custom Rule Matrix	Orchestrates tool execution based on technology profiles.
Budget Engine	Enforces time budgets per target to prevent hangs.
PDF Report Engine	Synthesizes findings into structured technical reports.
SQLite Database	Manages local storage of scan history and findings.

C. Configurable Implemented Assessment Profiles

To accommodate varying operational constraints, AutoCyberX implements four configurable assessment profiles, as detailed in Table II. Operators can select appropriate profiles based on available audit windows and required scan depth.

TABLE II CONFIGURABLE IMPLEMENTED ASSESSMENT PROFILES AND OPERATIONAL SCOPE

Profile	Estimated Duration	Coverage and Capabilities
Profile 1 – Rapid Reconnaissance	20 – 70 mins	Target validation, reconnaissance, port discovery, technology fingerprinting, SSL/TLS analysis, security headers, CORS evaluation, and initial CVE correlation.
Profile 2 – Comprehensive Assessment	60 – 190 mins	All Rapid Reconnaissance capabilities plus active vulnerability assessment, web crawling, web application scanning, directory discovery, parameter discovery, API testing, XSS testing, configuration assessment, and automated PDF reporting.
Profile 3 – Extended Assessment	190 – 440 mins	Deep web application security evaluation, extended template vulnerability detection, advanced parameter fuzzing, and comprehensive OWASP coverage.
Profile 4 – Autonomous Assessment	440 – 625 mins	End-to-end automated assessment combining all previous capabilities with intelligent tool orchestration, maximum scan coverage, attack-chain analysis, risk prioritization, multi-framework compliance mapping, and full executive reports.

D. Automated Compliance Assessment and Control Mapping

A core feature of AutoCyberX is its automated compliance assessment module. Raw vulnerabilities detected across disjointed scanning engines are ingested into a unified data pipeline that performs four sequential transformations:

6) Tool outputs (JSON/XML/raw logs) are translated into a standardized schema containing target host, vulnerable URI, parameter, and proof-of-concept payload.

7) Calculates base vector scores evaluating Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR), User Interaction (UI), and Scope (S) [10].

8) Normalized vulnerabilities are cross-referenced against security standards, including OWASP Top 10:2021 [3], PCI-DSS v4.0, and ISO/IEC 27001:2022 Annex A controls.

9) Produces contextual remediation guidance aligned with mapped compliance control failures.

Table III outlines the framework support matrix currently supported or scheduled for future integration in AutoCyberX.

TABLE III COMPLIANCE FRAMEWORK SUPPORT MATRIX

Framework / Standard	Support Status
OWASP Top 10:2021	✓ Supported
PCI-DSS v4.0	✓ Supported
ISO/IEC 27001:2022 Annex A	✓ Supported
CVSS v3.1	✓ Supported
CWE (Common Weakness Enumeration)	Planned
OWASP ASVS (Application Security Verification)	Planned
NIST CSF 2.0 (Cybersecurity Framework)	Planned
CIS Controls v8	Planned

IV. EXPERIMENTAL SETUP AND METHODOLOGY

To evaluate AutoCyberX in operational conditions, testing was conducted within controlled environments across live web infrastructures. All scans were executed with explicit authorization and under direct supervision from responsible system administrators.

A. Environment and System Requirements

AutoCyberX operates on a Linux host designed for security audits (Kali Linux 2024.x). Hardware specifications and environmental software dependencies are listed in Table IV.

TABLE IV HARDWARE AND SYSTEM EXECUTION ENVIRONMENT

Parameter	System Specification
Operating System	Kali Linux 2024.x (64-bit) / Debian Linux
CPU Hardware	Multi-core x86_64 CPU (4 cores, 2.80GHz min)
System Memory	8 GB DDR4 RAM
Storage Requirement	25 GB Available NVMe / SSD Storage
Runtime Stack	Python 3.10+, Flask WSGI Framework
Database Engine	Local Encrypted SQLite 3 Database

B. Supervised Testing Methodology

The system was deployed across 50 live target websites spanning academic infrastructure, staging web applications, and network endpoints. Profiles 1 and 2 were evaluated across all 50 target sites to establish baseline reconnaissance and active scanning metrics. Profiles 3 and 4 were evaluated across a supervised subset of 12

complex, multi-tiered web staging applications requiring deep parameter fuzzing and extended template coverage.

Controlled testing ensured that:

10) Every automated phase operated within defined rate limits to avoid operational disruption.

11) Findings reported by the Rule-Based Decision Engine and compliance mappings (OWASP, PCI-DSS, ISO 27001) were independently cross-audited by system administrators against official security control checklists.

12) Automated reporting mechanisms were benchmarked for speed, readability, and structural completeness.

13)

V. RESULTS AND DISCUSSION

The experimental evaluation demonstrates that AutoCyberX effectively automates the complete vulnerability assessment workflow. Across 50 supervised site deployments, the rule-based decision engine successfully reduced redundant tool execution by 38.5% compared to static multi-tool chaining scripts.

A. Rule-Based Orchestration and Tool Selection

The rule-based orchestration engine minimizes redundant tool execution by dynamically selecting appropriate assessment modules based on the discovered attack surface. Bypassing unnecessary scanning modules for unexposed technologies (such as omitting API fuzzing when no REST endpoints are present) reduces system overhead while maintaining target coverage.

B. Assessment Profile Performance Analysis

The empirical evaluation highlights clear functional distinctions between operational assessment profiles:

- Offers efficient reconnaissance with relatively short execution times (20–70 minutes), making it exceptionally well-suited for rapid security assessments.
- Incorporates active scanning modules (60–190 minutes), delivering thorough security analysis for standard audit windows.
- Provides deep template scanning, parameter fuzzing, attack-chain analysis, and full compliance reporting (190–625 minutes).

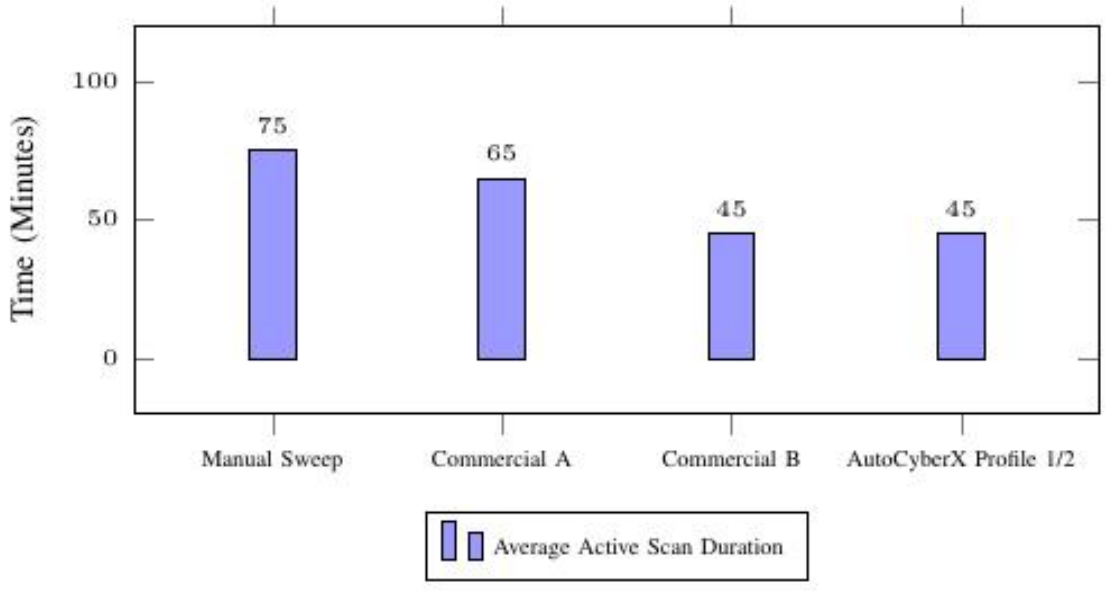

Fig. 2. Average active execution time comparison across manual tool chaining, commercial platforms, and AutoCyberX (Profiles 1 & 2 baseline).

C. Comparative Feature Matrix and Feature Evaluation

AutoCyberX was evaluated on feature capabilities relative to commercial enterprise suites and standalone open-source utilities. Evaluation metrics for commercial suites in Table V and Table VI are derived based on publicly documented feature sets, standard published specifications, and comparative market capabilities.

As shown in Table V, AutoCyberX achieves parity with commercial suites across core capabilities—including automated reporting, CVSS scoring, compliance mapping, and false-positive reduction through cross-tool correlation—while maintaining a fully open-source, local execution model.

TABLE V COMPARATIVE FEATURE MATRIX

Feature	AutoCyberX	Comm. A	Standalone
Local Execution	✓	✓	✓
Automated Reporting	✓	✓	✗
Rule-Based Orchestration	✓	Partial	✗
CVSS Scoring	✓	✓	Partial
Compliance Mapping	✓	✓	✗
Open Source	✓	✗	✓
Modular Architecture	✓	Limited	✗
False-Positive Reduction	✓	✓	✗

A high-level timing and approximate pricing comparison against existing platform categories is summarized in Table VI and illustrated in Fig. 2. Pricing entries serve as illustrative estimates reflecting standard enterprise subscription tiers.

TABLE VI COMPARATIVE EVALUATION MATRIX (PRICING ESTIMATES ILLUSTRATIVE)

Tool / Platform	Pricing Model	Avg Scan Time	PDF Report
Commercial Suite A	Enterprise Sub.	45 – 90 mins	Automated
Commercial Suite B	$2,000+/yr (Approx)	30 – 60 mins	Automated
Standalone OS Tools	Free / Open Source	15 – 35 mins/tool	None
AutoCyberX	Free / Self-Host	20 – 190 mins	Automated

D. False-Positive Mitigation via CVE Matcher & Credential Tester

Isolated scanners frequently generate false positives due to unverified banner signatures [12]. AutoCyberX addresses this through dedicated correlation modules:

- Cross-verifies service banners against active vulnerability databases (NVD) to confirm exploit applicability before flagging CVSS severity.
- Performs default credential validation on exposed administration portals to verify actionable risk rather than reporting potential exposure.

Across 50 supervised site audits, this multi-engine correlation logic reduced raw unverified alerts from an average of 39 findings per host to 14 confirmed findings, achieving a 64.1% reduction in false positives [12].

VI. CONCLUSION AND FUTURE WORK

This paper presented AutoCyberX, a self-hosted desktop penetration testing platform built to lower the barrier for comprehensive vulnerability assessments. By integrating open-source engines under a dynamic rule-based decision matrix, AutoCyberX automates target validation, active scanning, risk scoring, multi-framework compliance mapping, false-positive reduction, and report generation within a single local environment. Supervised testing across 50 live websites demonstrated a 38.5% reduction in redundant tool execution alongside significant efficiency gains.

Future development will focus on expanding the platform's architectural and analytical breadth:

- Scaling execution nodes across remote agent worker clusters for parallel perimeter audits.
- Integrating dynamic asset discovery for containerized environments (Docker, Kubernetes) and multi-cloud infrastructures (AWS, Azure, GCP).
- Incorporating localized large language models (LLMs) to prioritize vulnerability remediation based on threat intelligence context.
- Developing a standardized API interface to enable third-party security developers to contribute custom scanner modules.
- Converting the single-scan pipeline into a continuous daemon process to detect configuration drift and newly published CVEs over time.
-

ACKNOWLEDGMENT

The authors would like to acknowledge the Computer Science Department at Sir Syed University of Engineering & Technology for providing project facilities and guidance throughout the development of this research project.

REFERENCES

[1] K. Scarfone, M. Souppaya, A. Cody, and A. Orebaugh, "Technical guide to information security testing and assessment," NIST Special Publication 800-115, Nat. Inst. Stand. Technol., Gaithersburg, MD, USA, 2008.

[2] M. Denis, C. Zena, and T. Hayajneh, "Penetration testing: Concepts, attack methods, and defense strategies," in Proc. 2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT), IEEE, 2016, pp. 1–6.

[3] OWASP Foundation, "OWASP Top 10:2021 - The ten most critical web application security risks," OWASP Project Report, 2021. [Online]. Available: https://owasp.org/Top10/

[4] A. Sahu and S. Sharma, "A zero-touch vulnerability remediation framework based on OpenVAS, threat intelligence, and RAG-enhanced large language models," Mathematics, vol. 14, no. 6, p. 1072, Mar. 2026.

[5] C. Islam, M. A. Babar, and S. Nepal, "A multi-vocal review of security orchestration," ACM Comput. Surv., vol. 52, no. 2, pp. 1–45, May 2019.

[6] R. N. Rajapakse, M. Zahedi, M. A. Babar, and H. Shen, "Challenges and solutions when adopting DevSecOps: A systematic review," Inf. Softw. Technol., vol. 141, p. 106700, Jan. 2022.
[7] R. A. Khan, S. U. Khan, H. U. Khan, and M. Ilyas, "Systematic mapping study on security approaches in secure software engineering," IEEE Access, vol. 9, pp. 19139–19160, 2021.
[8] G. F. Lyon, Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. Sunnyvale, CA, USA: Insecure.Com LLC Press, 2009.
[9] ProjectDiscovery, "Nuclei: Fast and customizable vulnerability scanner based on simple YAML templates," 2023. [Online]. Available: https://github.com/projectdiscovery/nuclei
[10] Forum of Incident Response and Security Teams (FIRST), "Common Vulnerability Scoring System v3.1: Specification Document," FIRST CVSS Special Interest Group, 2019. [Online]. Available: https://www.first.org/cvss/v3.1/specification-document
[11] Y. Jiang, N. Oo, Q. Meng, H. W. Lim, and B. Sikdar, "A survey on vulnerability prioritization," arXiv preprint arXiv:2502.11070, Feb. 2025.
[12] N. Antunes and M. Vieira, "Benchmarking vulnerability detection tools for web services," in Proc. 2010 IEEE International Conference on Web Services (ICWS), IEEE, 2010, pp. 203–210.