Enhancing Web Application Security Through Active and Passive Reconnaissance Methods: A Comparative Framework Study Against Single-Method Approaches

Shlok Amit Kshirsagar

A thesis

submitted in partial fulfillment of the

requirements for the degree of

Master of Science

University of Washington

2026

Committee:

Dr. Min Chen

Dr. Marc Dupuis

Dr. Jeffrey Kim

Program Authorized to Offer Degree:
Computing and Software Systems

University of Washington

Abstract

Enhancing Web Application Security Through Active and Passive Reconnaissance Methods: A Comparative Framework Study Against Single-Method Approaches

Shlok Amit Kshirsagar

Chair of the Supervisory Committee:

Dr. Min Chen

Computing and Software Systems

Web application security assessment depends on effective reconnaissance to identify externally exposed assets, services, and vulnerabilities. Current reconnaissance practice relies on fragmented command-line tools that operate in isolation, produce unstructured outputs requiring substantial manual correlation, and apply context-independent vulnerability scoring that cannot reflect deployment-specific risk. The Common Vulnerability Scoring System provides no mechanism for automatically adjusting severity based on whether an affected service is internet-accessible or whether the host carries elevated asset value. These limitations reduce coverage, slow analyst workflows, and lead to systematic mis prioritization of remediation effort. This research investigates whether integrating passive and active reconnaissance within a unified pipeline, combined with cross-module analytical models, can produce security insights that isolated tools cannot generate, and whether a visualization-driven interface can deliver measurable workflow improvements over command-line operations.

Following a Design Science Research methodology, this thesis presents a three-module reconnaissance framework combining subdomain enumeration, port scanning, and vulnerability identification within a unified pipeline. Four original analytical algorithms are introduced: Shannon Entropy anomaly detection for identifying algorithmically generated subdomains, the

Service Exposure Classification engine for quantifying port-level exposure risk, the Contextual Vulnerability Priority Score for context-aware vulnerability re-prioritization using cross-module deployment data, and the Attack Surface Risk Score for synthesizing a composite risk metric from all three scanning phases. Each algorithm is presented with theoretical justification, mathematical specification, and an explicit statement of originality. The framework is integrated with a graphical user interface providing single-action scan execution, real-time progress feedback, and structured result visualization.

Experimental evaluation demonstrates that the combined approach improves asset discovery coverage and reduces false negatives compared to passive-only methods, while the visualization-driven interface enhances analyst efficiency. Furthermore, the integration of context-aware models enables more precise vulnerability prioritization. Overall, this research presents a comprehensive and scalable solution that advances the effectiveness of web application reconnaissance.

This thesis contributes four original analytical algorithms that produce qualitatively new categories of security insight requiring data from all three scanning phases simultaneously, demonstrating that integrated, context-aware reconnaissance surfaces actionable findings that fragmented tools systematically miss.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. Min Chen, for her continuous guidance, support, and valuable feedback throughout the course of this research. Her insights and encouragement played a crucial role in shaping the direction and quality of this thesis. I would also like to thank my committee members, Dr. Marc Dupuis and Dr. Jeffrey Kim, for their time, thoughtful suggestions, and constructive feedback, which helped improve my work significantly.

I am grateful to my peers and colleagues at the University of Washington Bothell for their support, discussions, and collaboration during my graduate studies. I would like to extend my appreciation to my mentors and colleagues from my previous professional experiences, whose guidance and industry insights helped strengthen my understanding of cybersecurity and inspired this research.

Finally, I would like to thank my family for their unconditional support, encouragement, and patience throughout my academic journey. Their belief in me has been a constant source of motivation.

CHAPTER 1 INTRODUCTION

In recent years, the growing complexity of web applications has expanded the attack surface of modern systems. Organizations now operate in distributed environments with dynamic assets such as subdomains, APIs, microservices, and cloud instances, making it difficult to maintain an accurate view of externally exposed resources. As a result, identifying and tracking these assets has become a key challenge for security practitioners.

Reconnaissance, the first phase of security assessment, is essential for discovering potential entry points. However, existing approaches often rely on separate tools that produce unstructured results and offer limited support for analysis. This thesis proposes a modular reconnaissance framework that integrates passive and active techniques to generate structured and enriched outputs, with the goal of improving both coverage and the usability of results for analysts.

1.1 MOTIVATION

The rapid growth of web applications has increased the attack surface of modern digital systems. Organizations rely heavily on web-based platforms for critical services, which makes them attractive targets for attackers. Vulnerabilities such as misconfigurations, exposed endpoints, and outdated components are often exploited, partly due to incomplete or ineffective reconnaissance practices.

Reconnaissance is a key phase in security assessment, used to identify assets, services, and potential weaknesses. However, it is commonly performed using separate command-line tools like Sublist3r and Nmap, which produce fragmented and unstructured outputs. This requires analysts to manually combine results, making the process time-consuming and prone to errors.

In addition, current approaches lack structured data integration and contextual analysis. Standard metrics such as Common Vulnerability Scoring System often ignore factors like asset importance and service exposure. These limitations highlight the need for a unified and context-aware reconnaissance framework that improves efficiency, accuracy, and usability for security analysts.

1.2 Problem Statement

Current reconnaissance practices in cybersecurity face several limitations that reduce their overall effectiveness. A key issue is the fragmented nature of existing tools, which typically operate as independent utilities. These tools require significant manual coordination, and their output must be correlated across different platforms. This lack of integration not only reduces operational efficiency but also increases the likelihood of missing or misinterpreting critical findings, which can have serious implications in security-sensitive environments.

Another important limitation is the reliance on static vulnerability scoring systems such as the Common Vulnerability Scoring System (CVSS). While widely adopted, these systems do not fully capture dynamic, real-world factors such as the actual exposure of assets, their importance within an organization, or the complex relationships between discovered components [15]. As a result, vulnerability prioritization often lacks contextual depth, limiting the ability of security practitioners to make well-informed and situation-aware decisions.

In addition, many reconnaissance approaches depend exclusively on either passive or active techniques. Passive methods, although less intrusive, often result in incomplete asset discovery, while active methods can introduce noise and inconsistencies in the findings. The absence of a well-defined hybrid strategy that effectively combines both approaches restricts comprehensive coverage and reduces the reliability of the overall reconnaissance process.

From a usability standpoint, the strong dependence on command-line interface (CLI) workflows presents further challenges. These environments require a high level of technical expertise and involve considerable manual effort, making them prone to human error, especially in large-scale operations. Moreover, most reconnaissance tools generate unstructured, text-based outputs with limited support for visualization or structured reporting. This lack of clarity makes it difficult for analysts to efficiently interpret results and take appropriate action, highlighting a significant gap in user accessibility and analytical support.

1.3 Research Objectives

The primary objective of this research is to design, implement, and evaluate a web reconnaissance framework that integrates active and passive techniques with cross-module analytical models and visualization support. The specific objectives that operationalize this primary objective are as follows.

First, to develop a modular reconnaissance pipeline combining three scanning techniques subdomain enumeration, port scanning, and vulnerability identification within a unified architecture that supports automated data flow between modules.

Second, to design and implement four novel analytical algorithms that consume cross-module data: Shannon Entropy-based anomaly detection for subdomain analysis, the Service Exposure Classification (ERS) engine for port-level risk classification, the Contextual Vulnerability Priority Score (CVPS) for context-aware vulnerability prioritization, and the Attack Surface Risk Score (ASRS) for composite risk quantification.

Third, to integrate the framework with a graphical user interface that provides single-action scan execution, real-time progress feedback, structured visual presentation of results, and one-click report generation, supporting workflow efficiency improvements over equivalent command-line operations.

Fourth, to evaluate the framework empirically against an authorized real-world target to assess workflow efficiency, asset discovery coverage, and the effectiveness of the cross-module analytical algorithms in surfacing actionable security findings.

1.4 Research Questions and Hypotheses

To guide the investigation, this thesis addresses the following research questions:

RQ1: How does the introduction of visualization and a graphical user interface improve an analyst's efficiency and accuracy in conducting web application reconnaissance compared to traditional command-line workflows?

This question examines the usability and effectiveness of a GUI-based system, focusing on whether tasks such as asset identification, vulnerability analysis, and result interpretation can be performed more efficiently and accurately compared to CLI-based approaches.

RQ2: How does incorporating both active and passive reconnaissance techniques within a unified framework with cross module analysis improve the overall coverage of discovered assets and the accuracy of findings compared to using only passive methods?

This question evaluates the effectiveness of the hybrid reconnaissance approach. It examines whether combining multiple techniques leads to improved asset discovery, reduced false positives, and better prioritization of vulnerabilities.

The corresponding hypotheses are defined as follows:

H1: The visualization and user-centered design of the reconnaissance framework will reduce analyst task completion time by at least 20% on common reconnaissance tasks compared to equivalent command-line workflows. This hypothesis is grounded in principles from human-computer interaction (HCI), which suggest that visual analytics can enhance cognitive processing and reduce decision latency.

H2: A hybrid reconnaissance approach that combines passive discovery with active probing will identify a greater number of valid assets and vulnerabilities than passive-only methods, while reducing the proportion of false positives. This hypothesis reflects the expectation that active verification can compensate for the limitations of passive data sources.

1.5 Scope and Limitations

The research focuses on external web application reconnaissance and early-stage vulnerability identification. The framework is designed and evaluated as a proof-of-concept that demonstrates

methodological improvements rather than as a production-grade tool intended for enterprise deployment. The scope of the work explicitly includes subdomain enumeration through both passive and active techniques, port scanning and service fingerprinting, vulnerability identification via passive correlation against the NIST National Vulnerability Database, structured data aggregation and cross-module enrichment, the four novel analytical algorithms documented in Chapter 3, and a graphical user interface supporting visual exploration of results.

Several limitations of this research are explicitly acknowledged. The evaluation was conducted against a single authorized production target, which limits the statistical generalizability of the quantitative findings to other infrastructures. Passive reconnaissance sources may contain outdated or incomplete information, which can affect baseline comparisons against active methods. Active scanning was intentionally constrained to non-intrusive techniques to maintain ethical and operational safety, which may limit the depth of vulnerability detection achievable. The GUI evaluation is comparative and qualitative rather than longitudinal and therefore does not capture long-term usability or large-scale deployment considerations. Despite these limitations, the framework provides a structured, reproducible foundation for evaluating hybrid reconnaissance methodology and is designed to support future extensions including additional scanning modules, machine-learning-augmented analysis, and integration with broader security operations workflows.

1.5.1 Intended Users

The framework developed in this thesis is designed for three categories of users. The primary intended users are security analysts and penetration testers conducting authorized external reconnaissance, particularly those who currently rely on disjoint command-line tools and manual data correlation. This is the user class whose workflow inefficiencies motivated the research and whose representative experience informed the design — including the senior offensive security engineer at Workday whose hands-on evaluation provided practitioner validation of the framework's design choices.

The secondary intended users are security operations (SOC) teams and risk management functions, who benefit from the composite Attack Surface Risk Score as a communicable single-number

indicator suitable for triage and managerial reporting. The tertiary intended users are cybersecurity researchers and educators, who can use the framework as an open and reproducible foundation for studying cross-module analytical methods, extending the algorithms, or teaching reconnaissance methodology. The framework is not designed for organizations operating at very large enterprise scale without modification, nor for non-technical end-users; both categories are addressed in the limitations and future work.

1.6 Contributions

This thesis makes four research contributions to cybersecurity literature. First, it presents an empirically validated framework architecture that integrates passive and active reconnaissance techniques within a unified pipeline supporting automated cross-module data flow. Second, it introduces four original analytical algorithms Shannon Entropy anomaly detection, the Service Exposure Classification engine, the Contextual Vulnerability Priority Score, and the Attack Surface Risk Score each accompanied by theoretical justification, mathematical specification, and an explicit statement of originality. Third, it provides empirical evidence that cross-module context can re-prioritize vulnerabilities in ways that context-blind CVSS scoring cannot, including documented Critical-tier escalation cases in which Medium-severity CVSS findings are correctly elevated to Critical through deployment context analysis. Fourth, it demonstrates that a visualization-driven GUI workflow produces measurable efficiency improvements over equivalent command-line workflows across multiple operational dimensions.

1.7 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 surveys the existing literature on reconnaissance methodologies, vulnerability scoring systems, attack surface measurement, and visualization in security workflows, identifying the specific gaps that this research addresses. Chapter 3 presents the research methodology under the Design Science Research paradigm, formalizing the four novel algorithms with theoretical justifications, mathematical definitions, and originality statements. Chapter 4 documents the system design and technical implementation, covering the three-tier architecture, scanning module implementations, the Hybrid Analyzer engine, and the GUI dashboard. Chapter 5 presents the empirical evaluation against an authorized

real-world target, reporting results across all three scanning modules and all four analytical algorithms, and answers both research questions with evidence. Chapter 6 concludes the thesis with a summary of findings, an explicit list of novel contributions, an honest discussion of limitations, and proposed directions for future work.

CHAPTER 2. LITERATURE REVIEW

Reconnaissance is a crucial phase in cyber security, consisting of collecting information about a target system to discover potential attack vectors. In the context of web application security, reconnaissance helps analysts to discover subdomains, exposed services, and vulnerabilities that could be exploited by adversaries. Over the years many tools and techniques have been developed to help reconnaissance activities. But the methods are very different in methodology, effectiveness and usability.

This chapter reviews existing research and tools related to reconnaissance, with a particular focus on both passive and active techniques, their effectiveness and the role of visualization in improving security workflows. It also critically analyses the current limitations in existing solutions and highlights the research gaps that motivate the development of the proposed reconnaissance framework.

2.1 RECONNAISSANCE IN CYBERSECURITY

Reconnaissance is a fundamental phase of the cyber kill chain, focused on collecting information about target systems, networks, and organizations before any exploitation occurs [1]. It plays an important role in both offensive security, such as penetration testing and red teaming, and defensive security, including threat modeling and attack surface management. Through reconnaissance, practitioners develop an initial understanding of exposed assets, communication paths, and potential vulnerabilities.

At a high level, reconnaissance can be divided into passive and active approaches. Passive reconnaissance involves gathering information without directly interacting with the target. This includes using publicly available sources such as WHOIS records, DNS data, Certificate Transparency logs, and platforms like Shodan and Censys. These methods are low risk and difficult to detect, but they may provide incomplete or outdated information.

In contrast, active reconnaissance involves direct interaction with the target system. Techniques such as port scanning, service identification, and protocol analysis are commonly used to validate and expand findings. Tools like Nmap and HTTPx help identify running services, software versions, and possible misconfigurations. While active methods provide more accurate and current data, they may generate network noise and increase the chance of detection.

Research shows that effective reconnaissance depends not only on individual tools but also on how different data sources are combined and analyzed. Attackers and security professionals often use a hybrid approach, integrating passive data with targeted active probing to improve coverage and accuracy. Relying only on one method limits visibility and reduces the effectiveness of the reconnaissance process.

Despite the availability of many tools, challenges remain in integrating and analyzing reconnaissance data. Outputs are often unstructured and tool-specific, requiring manual effort to combine and interpret results. This increases complexity and can lead to errors, especially in large-scale environments. Recent research has explored structured data and large-scale analysis techniques, but there is still a need for frameworks that not only collect data but also present it in a clear and useful way. This thesis addresses this gap by proposing a modular reconnaissance framework that integrates passive and active methods while focusing on structured outputs and improved analytical support.

2.2 Passive and Active Reconnaissance

Reconnaissance techniques in cybersecurity are generally divided into passive and active approaches, each with its own advantages and limitations. Understanding these differences is important for designing effective strategies and supports the development of hybrid methodologies explored in this research.

2.2.1 Passive Reconnaissance

Passive reconnaissance refers to the process of collecting information about a target without directly interacting with its systems. It typically involves querying publicly available data sources

such as search engines, WHOIS databases, and third-party APIs. Tools like Sublist3r and Subfinder utilize these sources to identify subdomains and other externally available information [3].

The primary advantage of passive reconnaissance is its stealth, as it does not generate direct traffic to the target, making detection difficult [2]. It is also generally faster and safer to perform. However, passive methods depend entirely on publicly available data, which may be incomplete, outdated, or inconsistent. As a result, they often fail to provide a comprehensive view of the target's attack surface.

2.2.2 Active Reconnaissance

Active reconnaissance involves direct interaction with the target system to gather information. Common techniques include port scanning, service fingerprinting, network probing, and enumeration. Tools such as Nmap are widely used to identify open ports, detect running services, and analyze system configurations [4]. Compared to passive methods, active reconnaissance provides more accurate and up-to-date information, enabling the discovery of hidden assets and real-time system details that are not available through public sources.

However, active reconnaissance also presents certain challenges. These activities can be detected by intrusion detection systems, potentially triggering security alerts. Additionally, aggressive or poorly configured scans may lead to inaccurate results or even disrupt target services. From a strategic perspective, passive methods focus on stealth and broad data collection, while active methods emphasize accuracy and validation. Research suggests that combining both approaches in a controlled manner can improve the overall effectiveness of reconnaissance processes.

2.2.3 *Comparative Analysis*

Several studies have compared passive and active reconnaissance techniques and highlight their complementary nature. Passive methods are fast and less detectable but often incomplete, while active methods provide deeper and more accurate results at the cost of higher detectability. However, most existing approaches treat these techniques separately, forcing analysts to choose between stealth and completeness. This separation reduces the overall effectiveness of

reconnaissance, particularly in complex environments that require both broad coverage and detailed analysis.

Table 1 Comparison of Active and Passive Reconnaissance

Aspect	Passive Reconnaissance	Active Reconnaissance
Interaction with Target	No direct interaction	Direct interaction
Detectability	Very low (stealthy)	Moderate to high
Data Accuracy	May be outdated or incomplete	Highly accurate and real-time
Coverage	Broad but shallow	Deep but targeted
Risk Level	Minimal	Potential for detection or disruption
Examples	WHOIS, DNS records, Shodan	WHOIS, DNS records, Shodan
Use Case	Initial discovery	Validation and deeper analysis

The limitations of purely passive or active approaches highlight the need for hybrid reconnaissance strategies that combine the strengths of both methods. Passive techniques can be used to identify potential targets and define the scope, while active techniques help verify and enrich the collected data. This research builds on this approach by evaluating how the integration of passive and active modules affects coverage, accuracy, and analyst efficiency. The proposed framework coordinates both techniques and enables adaptive use of passive discovery and active validation based on context and risk.

2.3 Studies on Recon Effectiveness

The effectiveness of reconnaissance has been studied from multiple perspectives, including adversarial behavior, large-scale scanning, passive intelligence, and automation. A common conclusion across these studies is that the presence of many tools does not ensure effective reconnaissance. Instead, effectiveness depends on how well diverse data sources are combined,

validated, and presented for analysis. This observation supports the motivation of this research, which focuses on improving both the reliability and usability of reconnaissance outputs.

One important aspect highlighted in prior work is the multi-source nature of reconnaissance. Roy et al. describe reconnaissance as a process that integrates third-party, human-based, and system-based information [1]. Their work shows that combining different types of data leads to better results than relying on a single technique. This perspective is important for web application reconnaissance, where passive sources and active validation methods must be used together.

Another key area of research focuses on large-scale active measurement. Durumeric et al., through their work on ZMap, demonstrate that internet-wide scanning can be performed efficiently at high speed [9]. However, their study also highlights that speed alone is not sufficient. The effectiveness of reconnaissance depends on how accurately the collected data reflects real-world conditions and how well it is interpreted.

Research on passive intelligence platforms, such as work discussed by Censys and Rapid7's Project Sonar, shows the value of continuously collected internet-wide data. These platforms help identify exposed assets and reduce the scope of scanning. However, they also reveal limitations, as such data may become outdated or incomplete. This reinforces the need for validation through active methods and supports the use of hybrid reconnaissance strategies.

Finally, studies by Huang et al. and Malkawi and Alhajj highlight challenges related to accuracy and automation [8]. Huang et al. show that passive vulnerability identification can lead to inconsistencies when compared to active verification [5]. Malkawi and Alhajj demonstrate that automation improves efficiency but often lacks structured outputs and contextual analysis [8]. These findings indicate a gap in existing approaches, which this research addresses by focusing on modular design, hybrid techniques, and improved data representation to enhance overall reconnaissance effectiveness.

Overall, existing studies suggest that reconnaissance effectiveness can be evaluated across key dimensions such as coverage, timeliness, correctness, and usability. While prior work has

addressed individual aspects like scanning speed, data collection, and vulnerability validation, few approaches combine these factors within a single framework. This thesis addresses this gap by proposing a modular framework that integrates hybrid techniques and structured outputs to improve both the reliability and practical usefulness of results. It also considers the human aspect by focusing on how analysts interpret and prioritize findings.

Critical gap identified

The literature highlights three key point. First, passive and active reconnaissance methods complement each other rather than acting as separate approaches. Second, while automation improves efficiency, effectiveness depends on proper data normalization and validation. Third, large-scale and public data sources are useful but not sufficient when accuracy and reliability are required. However, there is limited research on frameworks that combine these aspects into a single, integrated system. This thesis addresses this gap by proposing a modular and hybrid framework that integrates passive and active techniques while focusing on structured outputs to improve reliability, coverage, and practical usefulness.

2.4 Existing Reconnaissance Tools and Frameworks

Various tools and frameworks have been developed for reconnaissance, differing in their scope, functionality, and level of integration.

2.4.1 Command-Line Based Tools

Most traditional reconnaissance tools are command-line based, such as Sublist3r for subdomain enumeration, Nmap for port scanning, and various OSINT tools for information gathering [3][21]. While these tools are powerful and flexible, they require significant expertise to use effectively, and their lack of integration forces analysts to manually run multiple tools and combine outputs, making the process time-consuming and prone to human error.

2.4.2 Automated Reconnaissance Frameworks

More recent frameworks attempt to automate reconnaissance workflows by combining multiple tools into a single pipeline. Examples include ReconFTW and ReNgine, which provide automated scanning and reporting capabilities.

These frameworks offer improvements over manual workflows by reducing the effort required to execute multiple tools. However, they still exhibit several limitations:

- Monolithic Design: Many frameworks rely on large scripts or tightly coupled architectures, making them difficult to customize or extend.
- Limited Contextual Analysis: Outputs are often aggregated without meaningful cross-module correlation [20].
- Minimal Visualization Support: Results are typically presented as logs or static reports rather than interactive visualizations.
- Dependency Complexity: Some frameworks require extensive setup and configuration, limiting accessibility.

2.4.3 Industry Attack Surface Management Approach

Beyond academic frameworks, commercial attack surface management (ASM) platforms have emerged as the industry standard for the problem domain this thesis addresses. Representative platforms include Tenable Attack Surface Management, Rapid7 InsightVM, Qualys VMDR, Wiz, and CrowdStrike Falcon Surface. These platforms share a common high-level architecture: continuous asset discovery, vulnerability identification through CVE correlation, prioritization scoring, and dashboard-based reporting for security operations teams.

Industry platforms differ from this thesis's framework in three substantive respects. First, scope of deployment: commercial platforms are designed for enterprise-scale continuous monitoring across thousands of assets and multiple subsidiaries, whereas this thesis's framework is designed for analyst-driven assessment of specific authorized targets. Second, model transparency: industry platforms generally treat their prioritization logic as proprietary and do not publish the formulas, weights, or calibration approaches used in their risk scoring. Analysts using these platforms can

see the resulting scores but cannot inspect or independently re-derive the reasoning that produced them. Third, research extensibility: industry platforms are closed commercial products that cannot be modified, extended, or experimentally manipulated by researchers studying reconnaissance methodology.

The framework presented in this thesis contributes value distinct from the industry approach precisely because of these differences. The four analytical algorithms (Shannon Entropy classification, ERS, CVPS, ASRS) are fully mathematically specified, with every weight, threshold, and multiplier accompanied by an explicit calibration justification. The cross-module data flow producing context-aware re-prioritization is documented as an open methodology rather than as a black-box scoring service. This transparency carries two practical consequences. First, it enables analysts to defend a prioritization decision when challenged: when CVPS escalates vulnerability to Critical, the analyst can show that the elevation resulted from a specific Critical-tier port exposure detected by ERS, traced through a specific multiplier value calibrated to a specific CVSS severity boundary. Second, it enables organizations to re-calibrate the framework against their own incident or breach data, producing organization-specific risk scoring grounded in their operational reality rather than vendor-defined defaults.

The Critical-tier escalation findings reported in Chapter 5 are therefore meaningful in a way that commercial-platform outputs are not directly comparable to. They demonstrate not only that contextual re-prioritization works, but that an analyst can trace exactly why a specific finding was elevated, examine the calibration that produced the elevation, and defend the resulting remediation decision on a fully transparent analytical basis. The framework complements rather than competes with commercial ASM platforms: it serves as an open research foundation that the broader practitioner and research community can study, extend, and build upon in ways that commercial products structurally cannot support.

2.5 Visualization in Cybersecurity Workflows

Visualization improves the usability and effectiveness of cybersecurity tools by presenting complex data in a clear and intuitive format. Graphical representations such as dashboards and charts help analysts identify patterns, anomalies, and risks more efficiently, supporting better

decision-making. However, many reconnaissance tools still rely on unstructured textual outputs, which increase cognitive load and reduce analysis efficiency, especially when handling large datasets.

2.6 Gaps in Current Solutions

Based on the analysis of current tools and techniques, several key limitations can be identified:

- Lack of Integration: Most tools operate independently, requiring manual aggregation of results.
- Insufficient Coverage: Passive approaches fail to discover all assets, while active approaches may miss contextual information.
- Absence of Context-Aware Analysis: Existing systems rely on static scoring models such as CVSS, without considering factors such as service exposure or asset importance [12].
- Limited Usability: CLI-based tools require expertise and manual effort, reducing accessibility and increasing the likelihood of errors.
- Poor Visualization: The absence of interactive dashboards and visual analytics limits the ability to interpret and prioritize findings effectively.

2.7 Research Gap

The literature highlights a gap in current reconnaissance methods, where existing tools offer strong individual capabilities but lack integration, contextual analysis, and usability. Specifically, there is a lack of:

- Hybrid frameworks that effectively combine passive and active reconnaissance within a single system.
- Cross-module analytical models that utilize data from multiple sources to improve decision-making.
- Context-aware vulnerability prioritization mechanisms that go beyond traditional CVSS scoring.
- Visualization-driven interfaces that improve efficiency and reduce cognitive load for analysts.

This gap is significant in real-world security operations, where analysts must process large volumes of data quickly. The lack of integrated and intelligent systems reduces effectiveness and increases the risk of missed vulnerabilities.

CHAPTER 3 RESEARCH METHODOLOGY

This thesis presents the research methodology adopted for the design, development, and evaluation of the web reconnaissance framework. It begins by establishing a research paradigm and explaining why Design Science Research is the appropriate methodology for this study. It then operationalizes the two research questions into specific, measurable evaluation activities. The core of the chapter presents the theoretical foundations, mathematical definitions, and design justifications for the four novel analytical algorithms that constitute the primary research contributions of this thesis. The chapter concludes with the experimental design for evaluation and an explicit discussion of threats to validity.

Organizational note: The scanning module descriptions (subdomain enumeration, port scanning, and vulnerability scanning) are system design, and implementation concerns rather than research methodology. Accordingly, those modules are documented in Chapter 4 (System Design and Implementation). This chapter focuses exclusively on the research design decisions and the novel analytical models that represent the original intellectual contributions of this work.

3.1 RESEARCH DESIGN AND APPROACH

This research adopts the Design Science Research (DSR) methodology as its guiding paradigm. DSR, as formalized by Hevner et al. [22] and operationalized through the process model proposed by Peffers et al. [23], is the appropriate methodology for research that produces a purposefully designed artifact as its primary contribution and evaluates that artifact empirically against defined objectives. Unlike purely empirical research that observes existing phenomena, DSR generates new knowledge through the act of designing, building, and evaluating an artifact that embodies the researcher's theoretical claims about how an identified problem should be solved.

In this research, the artifact is the hybrid web reconnaissance framework: a three-module scanning pipeline unified by a cross-module analytical engine containing four original algorithms. The theoretical claims embedded in its design are twofold. First, that integrating passive and active reconnaissance techniques within a single pipeline, combined with cross-module data sharing,

produces superior asset discovery coverage and analytical accuracy compared to isolated tool usage. Second, that context-aware analytical models consuming data from multiple reconnaissance phases simultaneously can produce risk insights that no individual scanning tool can generate in isolation. The empirical evaluation in Chapter 5 provides the evidence against which these claims are assessed.

The DSR process model comprises six iterative activities, each mapped to a specific phase of this thesis:

- **Problem Identification and Motivation:** Current reconnaissance tools operate in isolation, producing fragmented and context-independent outputs that require significant manual effort to correlate [1] [8]. The literature review in Chapter 2 established that while individual scanning techniques are well-studied, no prior work has combined subdomain enumeration, port scanning, and vulnerability scanning into a unified analytical framework with formal cross-module risk scoring.
- **Definition of Objectives for a Solution:** Design an integrated framework that combines multiple reconnaissance techniques and produces cross-module analytical insights, operationalized through the two research questions defined in Section 1.4.
- **Design and Development:** Architect and implement the three-module framework with four novel analytical algorithms. The algorithm design is documented in this chapter; the system implementation is documented in Chapter 4.
- **Demonstration:** Execute the framework against an authorized real-world target to generate empirical data across all three scanning modules (Chapter 5).
- **Evaluation:** Assess the framework against the two research questions using quantitative metrics including asset discovery coverage, vulnerability re-prioritization accuracy, and workflow efficiency (Chapter 5).
- **Communication:** Present findings through this thesis document.

3.2 Research Questions Operationalization

Each of the two research questions maps to a specific research activity within the DSR process and is answered through a distinct methodological strand.

RQ1 (How does the introduction of visualization and a graphical user interface improve an analyst's efficiency and accuracy in conducting web application reconnaissance compared to traditional command-line workflows?) is answered through a comparative workflow analysis. The same reconnaissance task subdomain enumeration, port scanning, vulnerability scanning, and cross-module analysis is executed using both the GUI workflow and a simulated manual CLI workflow. The comparison measures the number of discrete operational steps, the number of manual data transfer operations between tools, the availability of cross-module analysis, and the time to locate critical findings. This evaluation design isolates the contribution of the GUI and automated pipeline by holding the reconnaissance task constant while varying only the execution method.

RQ2 (How does incorporating both active and passive reconnaissance techniques within a unified framework improve the overall coverage of discovered assets and the accuracy of findings compared to using only passive methods?) is answered through empirical measurement of two metrics. Coverage is assessed by comparing the number of unique assets discovered by the hybrid approach against the number discovered by passive-only methods. Analytical accuracy is assessed by measuring the proportion of vulnerabilities that receive adjusted priority scores through the CVPS algorithm's cross-module contextual re-prioritization, with particular attention to escalation cases where Medium-severity CVSS findings are escalated to Critical by deployment context.

3.3 Novelty of the Framework

The primary research contribution of this thesis lies not in the individual scanning techniques, which are well-established in literature, but in the analytical layer that processes data across modules. This analytical layer introduces four original algorithms that transform raw reconnaissance data into context-aware security insights. Each algorithm is detailed in Sections 3.4 through 3.7, following a consistent structure: the research gap addressed, the theoretical

foundation from prior literature, the complete mathematical definition with all parameters defined, the justification of design decisions, and an explicit statement of originality.

Together, these four algorithms form a cross-module data pipeline that is architecturally impossible to replicate through post-hoc combination of standalone tool outputs. The pipeline operates as follows: Module 1 (subdomain enumeration) produces subdomain labels that feed the Shannon Entropy anomaly detector and the CVPS Asset Value Multiplier. Module 2 (port scanning) produces open port data that feeds the ERS classification engine, which in turn produces exposure categories that feed the CVPS Exposure Multiplier. Module 3 (vulnerability scanning) produces CVSS base scores that serve as the foundation for CVPS adjustment. Finally, all three modules contribute component scores to the ASRS composite metric. This cross-module dependency chain is the central architectural novelty of the framework.

3.4 Algorithm 1: Shannon Entropy Anomaly Detection

3.4.1 Research Gap Addressed

Subdomain enumeration tools produce flat lists of discovered hostnames without any assessment of whether those hostnames exhibit characteristics consistent with legitimate organizational infrastructure or potentially malicious automated generation. Domain Generation Algorithms (DGAs), commonly used in malware for command-and-control communication, produce domain names with higher character-level randomness and more uniform character distributions than human-assigned names [10]. While Shannon entropy has been applied to DNS traffic analysis for DGA detection by Antonakakis et al. [10] and to algorithmically generated domain name detection by Yadav et al. [13], its application as an automated post-enumeration filter on reconnaissance output within an integrated security assessment framework is, to the best of the author's knowledge, original to this work.

3.4.2 Theoretical Foundation

Shannon entropy, introduced by Claude Shannon in 1948, is a foundational measure of information content and randomness in data. For a discrete random variable X with possible values x_1, x_2, ...,

x_k, each occurring with probability $p(x_i)$, the entropy H(X) quantifies the average information content per symbol. In the context of this research, the random variable represents the character distribution within a subdomain label string.

The theoretical basis for applying entropy to subdomain analysis rests on a well-documented empirical observation: human-assigned subdomain labels (such as "www", "mail", "api") use natural language patterns with predictable character frequencies and low entropy, whereas algorithmically generated labels use pseudo-random character sequences with near-uniform distributions and correspondingly high entropy [10] [13]. Bilge et al. [11] further demonstrated that passive DNS analysis can reliably distinguish legitimate from malicious domain infrastructure using statistical features including entropy.

3.4.3 Mathematical Definition

For a subdomain label string x of length n containing k distinct characters, the Shannon entropy H(x) is defined as:

$$H(x) = -\Sigma_{i=1}^{k} p(x_i) \times log_2(p(x_i)) \quad (1)$$

where:

- $p(x_i)$ = the probability of character x_i occurring in the label, computed as the frequency of character x_i divided by the total string length n,
- log_2 denotes the base-2 logarithm, yielding the result in bits,
- the range is $0 \leq H(x) \leq log_2(k)$, where H = 0 when the string contains a single repeated character (zero randomness), and $H = log_2(k)$ when all k characters appear with equal frequency (maximum randomness).

A higher entropy value indicates greater character diversity and more uniform distribution characteristics of randomly generated strings. A lower entropy value indicates character repetition and skewed distribution characteristics of natural-language words.

3.4.4 Threshold Calibration and Justification

The classification thresholds were empirically determined by computing entropy values across two distinct categories of subdomain labels. Table 3 presents entropy values for legitimate subdomain labels representing standard organizational naming conventions, and Table 2 presents entropy values for DGA-style generated labels representing algorithmically produced domain names.

Table 2 Entropy Values for Legitimate Subdomain Labels

Label	**Length**	**Distinct Chars**	**H (bits)**	**Classification**
www	3	1	0.00	Normal
db	2	2	1.00	Normal
api	3	3	1.58	Normal
mail	4	4	2.00	Normal
admin	5	5	2.32	Normal
staging	7	6	2.52	Unusual (borderline)
dashboard	9	7	2.75	Unusual

As shown in Table 2, standard English-based subdomain labels consistently produce entropy values below 2.5, with longer compound words occasionally reaching 2.5–2.8.

Table 3 Entropy Values for DGA-Style Generated Subdomain Labels

Label	**Length**	**Distinct Chars**	**H (bits)**	**Classification**
x7k2p9m4	8	8	3.00	Anomalous (borderline)
a3f8c1b2e9	10	9	3.17	Anomalous
a1b2c3d4e5f6	12	12	3.58	Anomalous

As shown in Table 3, algorithmically generated strings with eight or more characters and high character diversity consistently produce entropy values above 3.0, with longer strings exceeding 3.2.

Based on these observations, two classification thresholds were established. The lower threshold at H = 2.5 separates standard single-word labels from compound or concatenated identifiers. This

threshold is deliberately conservative a word like "staging" (H = 2.52) falls just above the boundary, encouraging investigation of non-standard names. The upper threshold at H = 3.2 separates plausibly human-assigned compound names from statistically random strings. The gap between the highest observed legitimate label ("dashboard" at H = 2.75) and the anomalous threshold (3.2) provides a 0.45-bit margin to minimize false positive classifications.

Originality statement: The entropy formula itself (Equation 1) is the standard Shannon entropy measure from information theory. The original contribution of this work is the application of this measure as an automated post-enumeration classifier for reconnaissance output, together with the empirically calibrated three-tier threshold model (Normal/Unusual/Anomalous) designed specifically for subdomain label analysis.

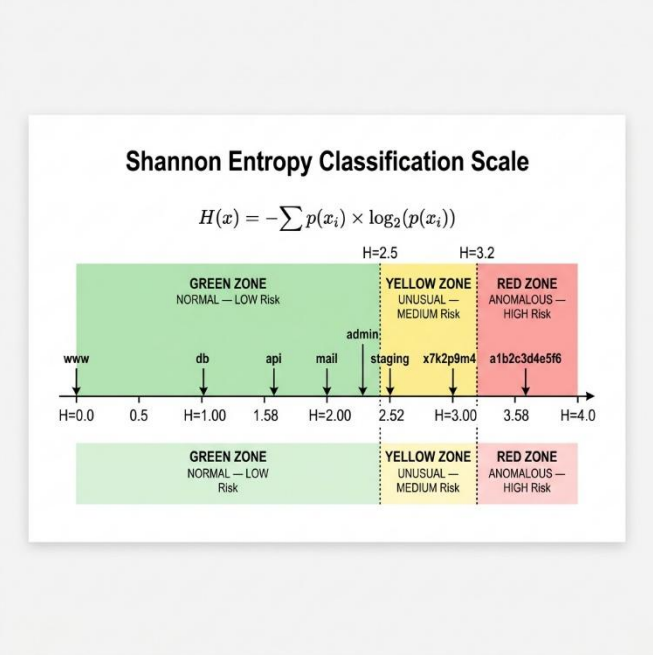

Figure 1 Shannon Entropy Classification Scale

3.5 Algorithm 2: Service Exposure Classification Engine (ERS)

3.5.1 Research Gap Addressed

Port scanning tools generate flat lists of open ports and detected services without distinguishing between their security implications. Commonly exposed services such as HTTPS on port 443 and sensitive services such as MySQL on port 3306 are reported with equal prominence, requiring

analysts to manually assess the risk significance of each finding. This manual assessment process is subjective, inconsistent between practitioners, and does not scale to environments with large numbers of open ports across multiple hosts. The Service Exposure Classification Engine addresses this gap by introducing a taxonomy-based classification approach that automatically categorizes each discovered port into a defined exposure tier and computes a quantitative Exposure Risk Score to summarize the target's overall exposure posture.

3.5.2 Theoretical Foundation and Taxonomy Design

The exposure classification taxonomy is derived from established industry security standards and best practices, including NIST Special Publication 800-123 (Guide to General Server Security), the CIS Critical Security Controls, and OWASP guidelines. Services are categorized into three tiers based on their design intent for public accessibility, as presented in Tables 5 through 7.

Tier 1 (Expected, weight = 0.0) includes services that are designed and standardized for public internet communication. Their presence on a public-facing server is normal and does not, by itself, constitute a security concern. As shown in Table 4, this tier includes HTTP/HTTPS for web content delivery, SMTP variants for inter-server mail transfer, DNS for domain name resolution, and encrypted mail retrieval protocols.

Table 4 Tier 1 - Expected Public-Facing Ports and Services

Port(s)	**Protocol/Service**	**Design Intent**
80, 443	HTTP, HTTPS	Web content delivery to public clients
25, 465, 587	SMTP, SMTPS, SMTP/TLS	Inter-server mail transfer
53	DNS	Domain name resolution — inherently public
993, 995	IMAPS, POP3S	Encrypted mail retrieval

Tier 2 (Cautionary, weight = 0.5) includes services that may be intentionally exposed in certain deployment scenarios but carry inherent risk if not properly controlled. As shown in Table 5, this tier includes remote administration services, development servers, and unencrypted legacy protocols.

Table 5 Tier 2 - Cautionary Services

Port(s)	Protocol/Service	Risk Factor
22, 2222	SSH	Remote administration - brute-force target
21	FTP	Credentials transmitted in plaintext
8080, 8443, 8000, 8888	Alt. HTTP/HTTPS	Development or staging servers
3000, 5000, 9090	Application dev servers	Debugging interfaces often enabled
9200, 9300	Search/analytics engines	Data indices often lack authentication
110, 143	POP3, IMAP (unencrypted)	Legacy mail - credential exposure risk

Tier 3 (Critical, weight = 1.0) includes services that are explicitly documented by their respective vendors and security frameworks as inappropriate for direct internet exposure. As shown in Table 6, this tier includes relational and NoSQL database services, remote desktop protocols, and legacy unencrypted remote access services. Their presence on a publicly accessible interface represents a configuration error requiring immediate remediation.

Table 6 Tier 3 - Critical Ports and Services

Port(s)	Protocol/Service	Impact of Internet Exposure
3306, 5432, 1433, 1521	Relational DBs (MySQL, PostgreSQL, MSSQL, Oracle)	Direct database access — data exfiltration
27017, 6379, 5984	NoSQL DBs (MongoDB, Redis, CouchDB)	Frequently deployed without default auth
3389	Remote Desktop Protocol (RDP)	Primary entry vector for ransomware
5900	VNC	Weak authentication model
445, 139	SMB, NetBIOS	EternalBlue/WannaCry attack surface
23	Telnet	Unencrypted remote access

3.5.3 Cross-Module Subdomain Context Enrichment

A distinguishing feature of the ERS is its cross-referencing with subdomain enumeration data from Module 1. After classifying open ports, the engine systematically identifies semantic correlations between port classifications and subdomain naming patterns. For example, a MySQL service (Critical tier) discovered on a host whose subdomain label is "db" or "data" receives corroborating

context that strengthens the finding's confidence. This cross-module enrichment produces context-aware findings that are architecturally impossible to generate with standalone port scanning tools, which lack access to subdomain intelligence.

3.5.4 Mathematical Definition

The ERS quantifies the target's overall port exposure posture as a single scalar value:

$$ERS = (\Sigma_{i=1}^{N} w_i) / N \times 100 \quad (2)$$

where N is the total number of classified ports, and w_i is the weight assigned to the ith port's exposure category: Expected = 0.0, Cautionary = 0.5, and Critical = 1.0. The score ranges from 0 (all ports Expected) to 100 (all ports Critical).

The ERS represents the average risk weight of the target's open ports, scaled to a 0–100 range. An ERS of 0 indicates that only standard web-facing services are exposed. An ERS of 50 indicates that, on average, the target ports are halfway between safe and dangerous. An ERS of 100 indicates that every open port hosts a service that should never be internet-facing. As shown in Table 7, the score is classified into four risk levels.

Table 7 ERS Risk Classification

ERS Range	**Level**	**Interpretation**
0 – 14.9	LOW	Minimal exposure - predominantly standard web-facing services
15 – 34.9	MEDIUM	Moderate exposure - some non-standard services accessible
35 – 59.9	HIGH	Significant exposure - several sensitive services internet-facing
60 – 100	CRITICAL	Severely over-exposed - multiple critical services directly reachable

Weight justification: The linear weight scheme (0.0, 0.5, 1.0) was chosen for two properties. First, interpretability: each Cautionary port contributes exactly half the risk of a Critical port, making the ERS intuitively understandable. Second, defensibility: the linear scale is the most justifiable default without domain-specific empirical calibration data. Non-linear schemes (e.g.,

0.0, 0.3, 1.0) would require justification from operational data that is not available for a general-purpose framework.

3.5.5 Cross-Module Data Dependency

The ERS was designed not only as a standalone metric but as an input to the CVPS algorithm (Section 3.6). This cross-module dependency is central to the RQ2 hypothesis: the escalation pattern Medium CVSS to Critical CVPS is only possible if port exposure context is available during vulnerability scoring, not correlated manually afterward. Spring et al. [15] argued that vulnerability severity cannot be accurately assessed without deployment context; the ERS-to-CVPS pipeline operationalizes this argument programmatically. Isolated tools, which do not share data across scanning phases, cannot produce this analytical outcome.

Originality statement: The three-tier exposure taxonomy, the ERS formula (Equation 2), the cross-module subdomain context enrichment mechanism, and the cross-module data dependency to CVPS are entirely original contributions of this thesis. The tier assignments draw on established best practices from NIST SP 800-123 and CIS Controls, but the formalization into a quantitative scoring model and the integration with subdomain intelligence are novel.

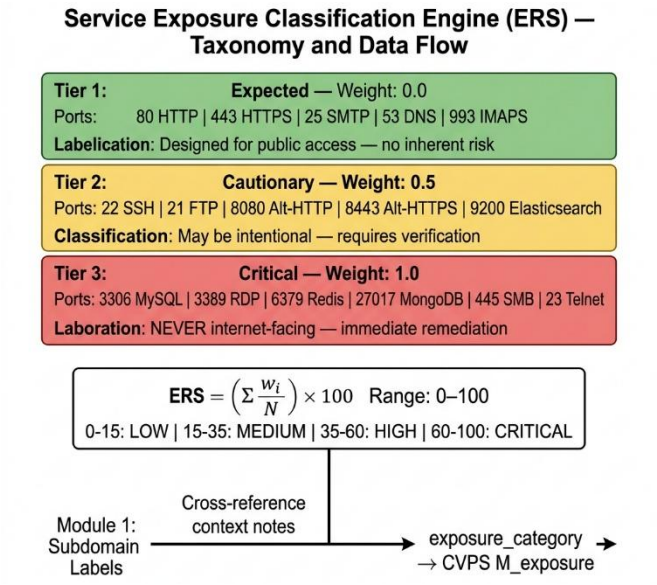

Figure 2 Service Exposure Classification

3.6 Algorithm 3: Contextual Vulnerability Priority Score (CVPS)

3.6.1 Research Gap Addressed

The Common Vulnerability Scoring System (CVSS) v3.1 is the industry-standard framework for assessing vulnerability severity. However, its base scores are context-independent they reflect the intrinsic characteristics of vulnerability but do not account for deployment-specific factors such as network exposure, asset criticality, or the presence of compensating controls [15]. As a result, two instances of the same CVE may receive identical CVSS scores despite residing in fundamentally different risk contexts: one on an isolated internal server behind multiple firewall layers, another on a database directly accessible from the public internet.

Spring et al. [15] demonstrated that context-independent CVSS scoring causes systematic misallocation of remediation resources, because organizations treat all vulnerabilities of the same score as equally urgent regardless of their deployment context. Allodi and Massacci [16] showed that exploitation likelihood is strongly correlated with deployment context precisely the factor that standard CVSS base scores do not capture. Jacobs et al. [17] argued that organizations that prioritize vulnerabilities using contextual information such as network exposure and asset criticality achieve significantly better remediation outcomes than those relying on CVSS alone.

Although CVSS provides an Environmental Score mechanism for contextual adjustment, it requires manual analyst input for each vulnerability and is not integrated into most automated scanning tools. The Contextual Vulnerability Priority Score (CVPS) addresses this gap by automatically adjusting CVSS base scores using deployment context derived programmatically from the other two scanning modules, requiring no manual input.

3.6.2 Mathematical Definition

The CVPS is computed as:

$$CVPS = min(10.0, CVSS\_base \times M\_exposure \times M\_asset) \quad (3)$$

where CVSS_base $\in$ [0, 10] is the original CVSS v3.x base score from the NVD as retrieved by Module 3, M_exposure is the Exposure Multiplier derived from Module 2 port scan data and the

ERS classification (Section 3.5), M_asset is the Asset Value Multiplier derived from Module 1 subdomain naming patterns, and the min(10.0, ...) function caps the result at 10.0 to maintain compatibility with the standard CVSS scale.

The multiplicative structure ensures that both exposure and asset value contribute proportionally. A vulnerability on a high-exposure port (M_exposure > 1) and a high-value asset (M_asset > 1) receives the compounding effect of both multipliers, reflecting the genuinely elevated real-world risk.

3.6.3 Exposure Multiplier Cascade

The Exposure Multiplier is determined through a prioritized cascade that consults cross-module data in order of specificity. The cascade first checks whether the host and port combination has an ERS classification from Section 3.5. If a Critical classification exists, the multiplier is set to 1.50, reflecting direct evidence of a dangerous service exposure. If a Cautionary classification exists, the multiplier is set to 1.20. The cascade then checks for API surface indicators if the host matches an API naming pattern from Module 1, the port is web-class, and the service is HTTP-family, the multiplier is set to 1.58, the highest available value, because this triple-condition requirement demands the strongest cross-module evidence. Table 8 presents the complete multiplier value rationale.

Table 8 CVPS Exposure Multiplier Value Rationale

M_exposure	**Condition**	**Calibration Basis**
1.58	Exposed API (3 conditions met)	CVSS 6.3 × 1.58 = 9.95 → CRITICAL escalation
1.50	Critical ERS port	CVSS 6.7 × 1.50 = 10.0 → CRITICAL escalation
1.20	Cautionary ERS port	20% increase — pushes borderline HIGH→CRITICAL
1.00	Expected port or unclassified	No adjustment - standard web port baseline
0.95	Suffix-related host	Slight reduction - partial reachability evidence
0.75	Host:port not in scan data	25% reduction - not confirmed reachable

Calibration rationale: The multiplier values were deliberately calibrated against CVSS severity boundaries. The value 1.58 was chosen because it is the threshold at which a CVSS 6.3 (upper

Medium) vulnerability is escalated to Critical: 6.3 × 1.58 = 9.95 ≈ 10.0. The value 1.50 escalates a CVSS 6.7 to Critical: 6.7 × 1.50 = 10.05, capped at 10.0. This boundary-aware calibration ensures that the multiplier values produce meaningful severity tier transitions rather than arbitrary score adjustments.

3.6.4 Asset Value Multiplier

The Asset Value Multiplier quantifies the criticality of the target host, inferred from its subdomain naming convention using Module 1 data. The subdomain's leftmost DNS label is matched against five pattern categories of descending criticality, as presented in Table 9.

Table 9 CVPS Asset Value Multiplier

Priority	**Pattern Group**	**Matching Keywords**	**M_asset**	**Rationale**
1	Database Infrastructure	db, mysql, mongo, redis, postgres, data	1.50	Data stores - breach yields data exfiltration
2	Administrative Interfaces	admin, panel, manage, dashboard, control	1.40	Management interfaces - elevated privilege access
3	API Endpoints	api, rest, graphql, gateway, endpoint	1.35	API surfaces - handle auth flows and sensitive data
4	Development/Staging	dev, staging, test, uat, beta, qa	1.20	Pre-production - often less hardened
5	Standard	No pattern match	1.00	No adjustment - standard web infrastructure

3.6.5 Priority Direction Tracking

For each processed vulnerability, the algorithm tracks the priority direction by comparing the CVPS-derived severity tier against the original CVSS-derived tier, as defined in Table 10. The most significant analytical outcome is the escalation case, where a vulnerability originally classified as Medium by CVSS is escalated to Critical by CVPS due to cross-module deployment context. These cases constitute the strongest empirical evidence for the value of integrated, context-aware vulnerability assessment.

Table 10 Priority Direction Classification - CVPS vs. Original CVSS Tier

CVPS Tier vs. Original Tier	Direction	Meaning
CVPS tier is higher	ESCALATED	Cross-module context increased the priority
CVPS tier is same	UNCHANGED	Context did not change the priority tier
CVPS tier is lower	DOWNGRADED	Context reduced confidence (e.g., host unreachable)

Originality statement: The CVPS formula (Equation 3), the Exposure Multiplier cascade, the Asset Value Multiplier taxonomy, and the priority direction tracking mechanism are entirely original contributions of this thesis. The CVPS builds conceptually on the CVSS Environmental Score mechanism, but differs from it in two fundamental respects: it derives its multipliers programmatically from cross-module reconnaissance data rather than requiring manual analyst input, and it integrates port exposure context (via ERS) and subdomain asset context (via Module 1) simultaneously a capability that requires the integrated framework architecture and cannot be replicated with standalone tools.

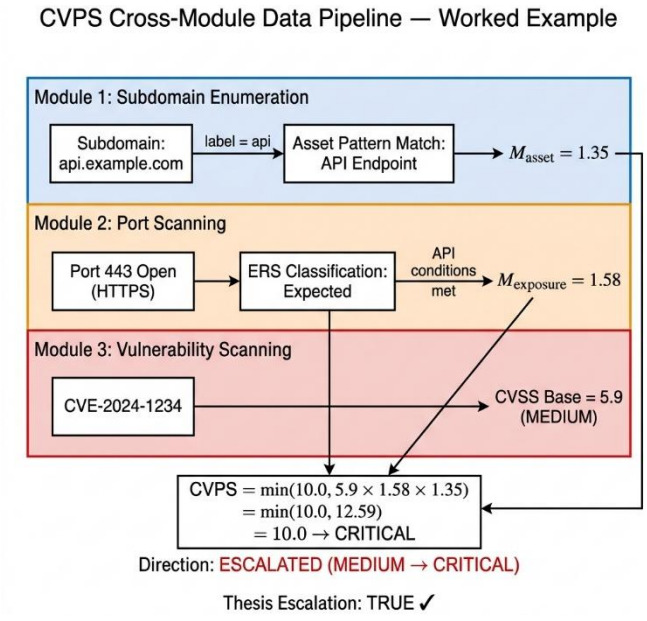

Figure 3 CVPS Working Example

3.7 ALGORITHM 4: ATTACK SURFACE RISK SCORE (ASRS)

3.7.1 Research Gap Addressed

After executing all three reconnaissance modules, an analyst possesses three independent datasets: a list of subdomains, a list of open ports, and a list of CVE records. However, there exists no established methodology for synthesizing these heterogeneous outputs into a single, actionable risk indicator. The analyst must mentally integrate findings from different modules a process that is subjective, inconsistent between practitioners, and not reproducible. Manadhata and Wing [18] established the theoretical foundations for attack surface measurement by defining attack surface in terms of the set of methods, channels, and data items that an attacker can use to interact with a system. However, their formalization operates at the architectural level and does not provide a quantitative metric synthesizing subdomain, port, and vulnerability data from reconnaissance output.

The Attack Surface Risk Score addresses this gap by computing a weighted composite metric (0–100) that quantifies the overall magnitude and severity of the target's attack surface using data from all three scanning modules simultaneously.

3.7.2 Mathematical Definition

The ASRS is computed as:

$$ASRS = min(100, (W_1 \times S\_sub + W_2 \times S\_port + W_3 \times S\_vuln) \times F\_exposure) \quad (4)$$

where S_sub ∈ [0, 100] is the Subdomain Score derived from Module 1, S_port ∈ [0, 100] is the Port Score derived from Module 2, S_vuln ∈ [0, 100] is the Vulnerability Score derived from Module 3, $W_1 = 0.30$, $W_2 = 0.30$, and $W_3 = 0.40$ are fixed component weights, and F_exposure ∈ [1.0, 2.0] is the Exposure Factor derived from Module 1 subdomain patterns.

3.7.3 Component Score Definitions

The Subdomain Score S_sub is computed as:

$$S\_sub = min(100, count\_score + pattern\_bonus) \quad (5)$$

where count_score = min(40, total_unique_subdomains × 3) and pattern_bonus = min(60, Σ risk_weight_p for each matched pattern p). The count component (maximum 40 points) reflects the general principle that a larger subdomain footprint presents a wider attack surface. The coefficient of 3 was selected so that approximately 13–14 subdomains saturate the count component, representing a moderate organizational footprint. The pattern component (maximum 60 points) is the discriminating factor, using a curated dictionary of 35 high-value subdomain patterns across 10 risk-indicative categories, as presented in Table 11.

Table 11 High-Value Subdomain Pattern Categories with Risk Weights

Category	**Example Patterns**	**Risk Weight**
Administrative	admin, panel, manage	10
Database	db, mysql, mongo, redis, postgres	10
Remote Access	vpn, remote	9
CI/CD	jenkins, gitlab, deploy	9
Backup	backup, bak	9
Internal	internal, intranet, corp	9
Development	dev, staging, test	8
API	api, rest, graphql	7
File Transfer	ftp, sftp, files	7
Email	mail, smtp	5

The Port Score S_port is computed as

$$S\_port = min(100, \Sigma\ base\_weight(port_i) \times severity\_mult(port_i)) \quad (6)$$

where the base weight is drawn from a curated lookup table of 24 port-to-risk mappings ranging from 1 (HTTPS/443) to 10 (RDP/3389, Telnet/23, Redis/6379, MongoDB/27017), and the severity multiplier reflects the port scanner's assessment: Critical = 2.0×, High = 1.5×, Medium = 1.0×, Low = 0.5×.

The Vulnerability Score S_vuln is computed as

$$S\_vuln = min(100, \Sigma\ (severity\_weight(v_i) + CVSS(v_i) \times 0.5)) \quad (7)$$

where severity_weight is defined as Critical = 10, High = 7, Medium = 4, Low = 1. The dual weighting captures both the count (discrete weight) and intensity (CVSS contribution) of vulnerabilities, ensuring that one Critical CVE contributes more than several Low CVEs.

The Exposure Factor F_exposure is computed as:

$$F\_exposure = min(2.0,\ 1.0 + (admin\_count \times 0.15) + (dev\_count \times 0.10) + (api\_count \times 0.05)) \quad (8)$$

The Exposure Factor amplifies the composite score when high-risk subdomain categories are present. Administrative subdomains receive the highest amplification (0.15 per instance) because exposed admin panels elevate risk across all findings. Development subdomains receive moderate amplification (0.10) as they often contain production-like data with weaker security. API subdomains receive lower amplification (0.05) as they are typically better secured. The cap at 2.0 prevents extreme cases from producing unreasonable scores.

3.7.4 Component Weight Justification

Table 12 Component Weight Justification

Weight	**Component**	**Justification**
0.40	Vulnerabilities (S_vuln)	Confirmed exploitable weaknesses - most directly actionable
0.30	Ports (S_port)	Network entry points - indicate exposure potential but not exploitability
0.30	Subdomains (S_sub)	Attack surface breadth - more hosts create potential targets

Vulnerabilities receive the highest weight (0.40) because they represent confirmed, exploitable weaknesses, the most directly actionable and immediately dangerous findings. Ports and subdomains each receive 0.30, reflecting their roles as enablers of attack rather than confirmed weaknesses. This weighting ensures that a target with many open ports but no known CVEs scores lower than a target with fewer ports but confirmed critical vulnerabilities.

3.7.5 ASRS Risk Classification

Table 13 ASRS Risk Classification

ASRS Range	Risk Level	Interpretation
0 – 24	LOW	Minimal attack surface - target is well secured
25 – 49	MEDIUM	Moderate attack surface - some areas require remediation
50 – 74	HIGH	Significant attack surface - multiple risk factors present
75 – 100	CRITICAL	Extremely large attack surface - immediate remediation required

Originality statement: The ASRS formula (Equation 4), all component score definitions (Equations 5 and 6), the pattern-based subdomain risking taxonomy (Table 11), the component weight allocation, and the Exposure Factor amplification mechanism are entirely original contributions of this thesis. While Manadhata and Wing [18] established the theoretical foundations for attack surface measurement, and the Gartner External Attack Surface Management (EASM) category identified the need for unified asset risk scoring, no prior tool produced a single quantitative metric synthesizing subdomain, port, and vulnerability data with weighted components and an exposure amplification factor. The ASRS is the first composite attack surface metric specifically designed for integrated web reconnaissance data.

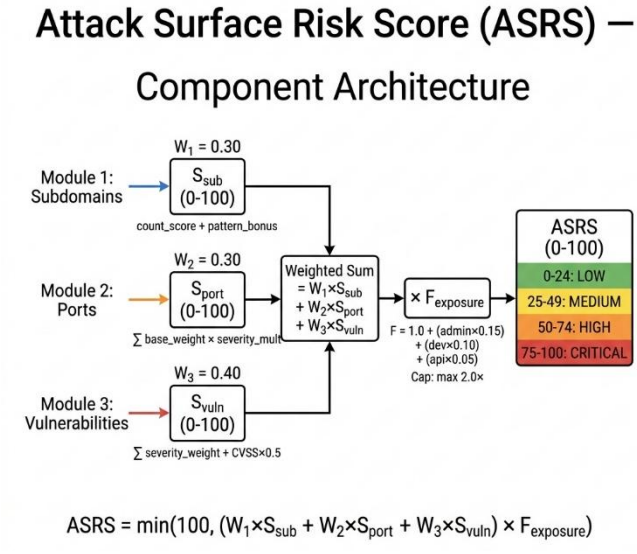

Figure 4 Attack Surface Risk Score Method

3.7.6 Consolidated Parameter Calibration

The framework contains four classes of empirically or analytically calibrated parameters. This subsection consolidates the calibration approach used for each, providing a single reference point for the reasoning behind every numeric value in the four analytical algorithms.

Class 1: The Shannon Entropy classification thresholds (H = 2.5 for the Normal/Unusual boundary, H = 3.2 for the Unusual/Anomalous boundary) were calibrated through direct measurement of entropy values across two reference categories of subdomain labels, documented in Section 3.4.4 (Tables 2 and 3). The lower threshold separates standard single-word labels from longer compound identifiers; the upper threshold leaves a 0.45-bit margin above the highest observed legitimate label to minimize false positives.

Class 2: The three-tier ERS taxonomy weights (Expected = 0.0, Cautionary = 0.5, Critical = 1.0) and the assignment of specific ports to tiers are derived from established industry security standards: NIST Special Publication 800-123, the CIS Critical Security Controls, and OWASP guidelines, as documented in Section 3.5.2 (Tables 4 through 6). The linear weight progression (0.0, 0.5, 1.0) was chosen for interpretability — each Cautionary port contributes exactly half the risk of a Critical port and represents the most defensible default in the absence of organization-specific incident data. The framework supports re-tuning these weights if such data becomes available.

Class 3: The CVPS Exposure Multiplier values (1.58 for exposed API surfaces, 1.50 for Critical-tier ports, 1.20 for Cautionary-tier ports, 1.00 for Expected ports, 0.95 and 0.75 for reduced-confidence cases) were calibrated against CVSS severity boundaries, as documented in Section 3.6.3 (Table 8). The value 1.50 is the specific multiplier at which a CVSS score of 6.7 (mid-Medium) crosses the Critical threshold of 9.0; the value 1.58 is the multiplier at which a CVSS score of 6.3 (upper Medium) approaches 10.0. These values are not arbitrary: each multiplier corresponds to a defensible severity-tier transition rather than an ad-hoc adjustment. The CVPS Asset Value Multipliers (1.50 for database infrastructure, 1.40 for administrative interfaces, 1.35 for API endpoints, 1.20 for development/staging, 1.00 for standard infrastructure) are similarly

tied to severity-tier transitions and reflect the operational consequence ordering established by industry security practice.

Class 4: The ASRS component weights ($W_1 = 0.30$ for subdomains, $W_2 = 0.30$ for ports, $W_3 = 0.40$ for vulnerabilities) reflect the principle that confirmed exploitable weaknesses warrant greater weight than potential attack vectors. Vulnerabilities are confirmed exploitable findings, while subdomains and ports are risk indicators that an attack could occur. The 0.40 / 0.30 / 0.30 asymmetry encodes this order. The subdomain count multiplier (×3) and the count cap (40) were chosen so that the count component saturates at approximately 13–14 subdomains, ensuring that very large counts do not dominate the score while small counts still produce meaningful differentiation. The pattern-bonus cap (60) and the final component cap (100) similarly preserve interpretability by preventing any single subcomponent from exceeding its proportional contribution. The exposure factor increments (admin = 0.15, dev = 0.10, API = 0.05) reflect the relative privilege-escalation impact of each subdomain category, with the factor capped at 2.0 to prevent runaway amplification.

All four classes of calibration are deterministic and reproducible: given the same scan data, every algorithm produces identical outputs because no randomness is introduced at the analytical layer. The transparency of the calibration approach is intentional — analysts using the framework can inspect every parameter, understand why each value was chosen, and re-tune values for their organizational context if needed. Empirical re-calibration of these parameters against organization-specific incident or breach data is identified as future work in Chapter 6.

3.8 Experimental Design for Evaluation

The empirical evaluation of the framework follows a controlled experimental design. This section documents the evaluation methodology; the results are presented in Chapter 5.

3.8.1 Target Selection Criteria

The evaluation target was selected based on four criteria: (1) Authorization: the target domain was scanned under explicit authorization using researcher-controlled infrastructure. (2) Diversity: the

target exhibited varied infrastructure across multiple geographically and functionally distinct subdomains, a mix of standard and non-standard open ports, and a substantial published vulnerability footprint. (3) Realism: the target is a production environment, ensuring results are representative of real-world reconnaissance outcomes. (4) Reproducibility: the scan was conducted with fixed parameters to ensure consistent, repeatable results.

3.8.2 Evaluation Metrics

The framework is evaluated against both research questions using the following metrics. For RQ1, the evaluation compares the number of discrete operational steps, manual data transfer operations, and availability of cross-module analysis between the GUI framework workflow and an equivalent CLI workflow. For RQ2, the evaluation measures asset discovery coverage (number of unique subdomains discovered by the hybrid approach versus passive-only methods), vulnerability re-prioritization rate (proportion of CVEs receiving CVPS adjustments), and escalation count (number of Medium-severity CVSS findings escalated to Critical by CVPS).

3.8.3 Controlled Variables

To ensure that observed differences in coverage and accuracy can be attributed to the framework's hybrid approach and cross-module analysis rather than to confounding variables, the following conditions were held constant across all evaluation runs: the target domain, the scanning module configuration, the port selection set, the vulnerability correlation source (NIST NVD), and the algorithm parameters. The only independent variable varied across comparison conditions was the reconnaissance methodology: hybrid (passive + active with cross-module analysis) versus passive-only (without active enumeration or cross-module context).

3.8.4 Accuracy Definitions

Throughout this thesis, the term accuracy refers to the correctness of findings produced by each of the three reconnaissance modules. Because each module performs a different type of detection task, the definitions of true positive, false positive, true negative, and false negative are stated separately for each module.

Subdomain enumeration: A true positive is a discovered subdomain that is verifiable as authoritatively belonging to the target domain and that resolves to a reachable host at the time of scanning. A false positive is a subdomain string returned by a discovery source that either does not resolve, no longer exists, or does not belong to the target organization. A false negative is a valid, reachable subdomain belonging to the target that the framework's combined passive and active discovery sources failed to identify.

Port scanning: A true positive is an open port correctly identified along with its associated service. A false positive is a reported open port that is in fact filtered, closed, or unreachable. A false negative is an open port on a scanned host that the scanner failed to detect, including ports omitted from the scan range.

Vulnerability correlation: A true positive is a CVE association in which the detected service version is genuinely affected by the matched vulnerability. A false positive is a CVE association produced by version-string matching where the affected condition is not in fact met, for example a backported security patch that does not change the reported version. A false negative is a CVE that genuinely affects the detected service version but was not surfaced because no matching CPE entry exists in the NIST National Vulnerability Database or because the service banner did not provide sufficient version information.

This thesis does not present a direct quantitative comparison of accuracy rates between hybrid and passive-only configurations, because such a comparison requires a labeled ground-truth dataset at population scale, which was outside the scope of the single-target evaluation. The accuracy contribution claimed in RQ2 is therefore framed in terms of analytical accuracy: the correctness of priority assignments produced by the cross-module CVPS algorithm, evidenced by the two Critical-tier escalation cases documented in Chapter 5.

CHAPTER 4. SYSTEM DESIGN AND IMPLEMENTATION

This chapter describes the system design and technical implementation of the Web Reconnaissance Framework, translating the research methodology and novel algorithms from Chapter 3 into a working software system. It covers the overall architecture, technology stack, the three scanning module implementations, the Hybrid Analyzer engine, database design, frontend GUI, and reporting capabilities. All design decisions align with two core requirements: enabling seamless cross-module data flow for the four novel algorithms defined in Chapter 3 and providing an interactive GUI to support the RQ1 evaluation of analyst efficiency.

4.1 SYSTEM ARCHITECTURE

The framework follows a three-tier client-server architecture that separates the presentation tier (user interaction and visualization), the application tier (scanning orchestration and analytical algorithms), and the data tier (persistent storage). As shown in Table 14, each tier has clearly scoped responsibilities.

Table 14 Three-Tier Architecture Overview

Tier	**Responsibility**	**Key Components**
Presentation	User interaction, visualization, real-time feedback	Web dashboard, HTML templates, JavaScript, CSS
Application	Scanning orchestration, analytical algorithms	REST API server, scanning engines, Hybrid Analyzer
Data	Persistent storage, query processing	SQLite database, scan records, analysis outputs

This separation provides modularity (each tier can be modified independently), testability (the application tier APIs can be tested independently of the GUI), and extensibility (new scanning modules can be added without modifying the presentation or data tiers). Communication between tiers uses two protocols. REST API (HTTP/JSON) handles standard request-response interactions between the presentation and application tiers. Server-Sent Events (SSE) provides real-time unidirectional streaming from the application tier to the presentation tier during long-running scan

operations. SSE was chosen over WebSockets because the communication pattern is strictly server-to-client, making SSE simpler and more appropriate for this case.

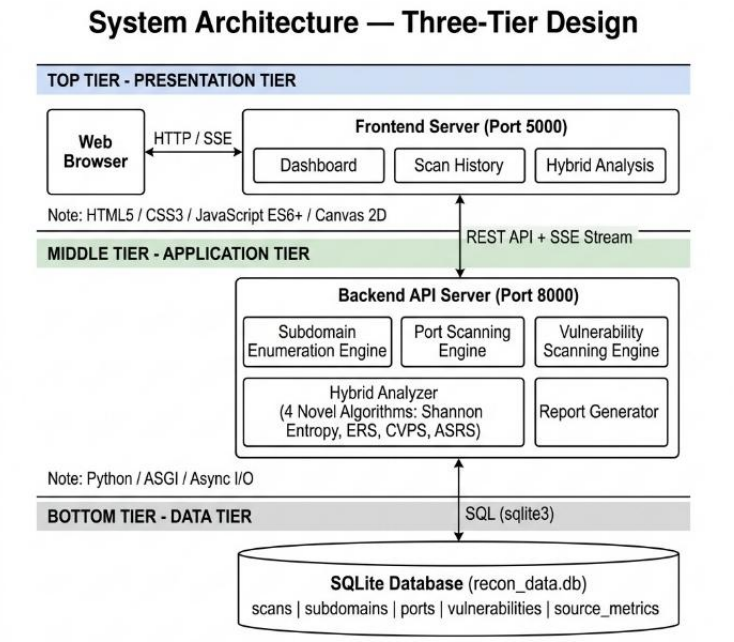

Figure 5 System Architecture

4.2 TECHNOLOGY STACK

The framework is implemented in Python, selected for its extensive security library ecosystem (DNS manipulation, network sockets, HTTP clients), native asynchronous I/O support via asyncio for concurrent scanning, cross-platform compatibility, and rapid prototyping characteristics aligned with the iterative DSR development methodology.

The application tier uses an asynchronous web framework built on the ASGI specification, providing automatic request validation, native async request handling for non-blocking scan orchestration, and CORS middleware for front-end-backend communication. The presentation tier uses a lightweight WSGI framework with Jinja2 template engine for dynamic HTML generation, static file serving, and Server-Sent Events support through streaming response generators. The data tier uses SQLite, an embedded relational database engine requiring zero configuration, providing single-file portability, ACID compliance with write-ahead logging, and foreign key

support for referential integrity, well suited for the expected data volumes of hundreds of scans and thousands of records.

The frontend is implemented using HTML5 for semantic page structure, CSS3 with custom properties for a dark theme design system with glass morphism effects and responsive layouts, and vanilla JavaScript (ES6+) for dynamic rendering, Canvas 2D charting, and SSE event listeners. The decision to use vanilla JavaScript without framework dependencies (no React, Vue, or Angular) ensures that the front end remains lightweight and that all visualization logic is transparent and auditable.

Table 15 Key Libraries and Their Roles

Library	**Purpose**	**Tier**
dnspython	DNS A-record resolution, zone transfer (AXFR) queries	Application
requests	HTTP client for CT log and NVD API communication	Application
psutil	System resource monitoring during scan operations	Application
reportlab	Programmatic PDF report generation with vector graphics	Application
math (stdlib)	Shannon Entropy Computation, statistical functions	Application
sqlite3 (stdlib)	Database driver for scan persistence	Data

4.3 Scanning Module Implementations

The framework implements three scanning modules that execute sequentially as a pipeline, with each module's output serving as input to subsequent modules and to the Hybrid Analyzer. This section describes the implementation of each module; the research methodology motivating their design is documented in Chapter 3.

4.3.1 Module 1: Subdomain Enumeration Engine

The subdomain enumeration engine implements a hybrid methodology that combines passive and active techniques to maximize discovery coverage. It provides a single-entry point that takes a target domain as input and returns both individual source results and a unified, deduplicated output.

Passive source implementation: Three categories of passive data sources are utilized. Search engine index mining retrieves subdomain data by querying security-focused public APIs and extracting relevant records from JSON and HTML responses. Certificate Transparency (CT) log analysis sends HTTP GET requests to CT log APIs using the query pattern %.{domain}, parses the JSON response to extract subdomains from the name_value field, and normalizes wildcard entries by removing the "*." prefix [3]. Passive DNS database querying collects historical DNS records from intelligence services that aggregate previously observed domain-to-IP mappings, helping uncover hidden or legacy infrastructure that may still be accessible [11].

Active source implementation: Two active techniques complement the passive sources. Dictionary-based DNS brute-force loads a curated word list and constructs candidate FQDNs by prepending each word to the target domain, performing DNS A-record resolution via the dnspython library and recording valid resolutions with their IP addresses. Zone transfer (AXFR) attempts resolve the target domain's NS records and attempt AXFR queries against each authoritative nameserver; successful transfers yield the complete zone contents. Although most modern servers restrict zone transfers, misconfigurations can still expose this information, making it a valuable opportunistic technique.

4.3.2 Module 2: Port Scanning Engine

The port scanning engine takes the subdomains identified by Module 1 as input and detects open network ports and running services on each host. It is implemented as a configurable class with parameters for connection timeout (default 2.0 seconds), maximum concurrent scanning threads (default 50), and a known_subdomains list from Module 1 that enables ERS cross-referencing at scan time.

The engine supports three scanning techniques. Full TCP connection scanning completes the standard three-way handshake for reliable open port detection without requiring elevated

privileges. Half-open (SYN) scanning terminates the connection early after receiving a response, providing faster and stealthier results when elevated privileges are available. UDP scanning discovers services operating over connectionless protocols, though the absence of a response does not always clearly indicate port state [4].

The default scan comprises 18 ports (21, 22, 23, 25, 53, 80, 110, 143, 443, 445, 3306, 3389, 5432, 5900, 8080, 8443, 27017, 6379), selected to include representatives from all three ERS exposure tiers defined in Chapter 3, ensuring diverse input data for classification. For each open port, the engine performs service detection through banner grabbing (capturing initial service banners containing metadata such as service type and version) and service fingerprinting using probe-based techniques for services that do not expose banners [6] [19]. The module includes an inline exposure classifier that applies the ERS taxonomy during the scan itself, ensuring classifications are available immediately without requiring a separate post-processing step.

4.3.3 Module 3: Vulnerability Scanning Engine

The vulnerability scanning engine is the final pipeline stage and identifies known vulnerabilities through passive correlation matching detected service versions against the NIST National Vulnerability Database (NVD) without actively testing or exploiting the target. This approach ensures effective vulnerability identification while avoiding potential disruption or unauthorized access.

The correlation process follows four steps. First, structured service identifiers are extracted from the banner data and fingerprinting results collected by Module 2, decomposing each service into a vendor, product name, and version triplet. Second, the service fingerprint is matched against the Common Platform Enumeration (CPE) naming scheme maintained by NIST, which provides a structured naming convention for systematic matching against vulnerability databases [7]. Third, for each matching CPE identifier, the engine queries the NVD to retrieve all associated CVE records, including the unique identifier, CVSS v3.x base score, severity rating, attack vector, and affected version ranges. Fourth, each identified vulnerability is assembled into a structured record containing the CVE identifier, CVSS score, severity, affected host, port, service, and description. These records serve as input to both the CVPS and ASRS algorithms defined in Chapter 3.

4.4 Backend API Design

The application tier exposes a RESTful API organized into four endpoint categories. Table 16 presents the scan execution endpoints that initiate reconnaissance operations. Table 17 presents the data retrieval endpoints that provide access to scan results. Table 18 presents the scan history endpoints for managing historical records. Table 19 presents the hybrid analysis endpoints that trigger execution of the four novel algorithms from Chapter 3.

Table 16 Scan Execution API Endpoints

Method	**Endpoint**	**Purpose**
POST	/scan	Execute full reconnaissance pipeline with SSE streaming
POST	/api/scans/start	Start background scan with progress tracking
GET	/api/scan-progress	Poll current scan phase and progress percentage

Table 17 Data Retrieval API Endpoints

Method	**Endpoint**	**Purpose**
GET	/api/scans	List all scan results with optional domain filter
GET	/api/stats	Summary statistics (subdomain/port/vuln counts)
GET	/api/vulnerabilities	Vulnerability breakdown by severity
GET	/api/search?query=	Search results by domain, CVE, or keyword

Table 18 Scan History API Endpoints

Method	**Endpoint**	**Purpose**
GET	/api/history	List all historical scans with metadata
GET	/api/history/<id>	Full details of a specific scan
DELETE	/api/history/<id>	Delete a scan and all associated data
GET	/api/history/<id>/report	Download PDF report for a scan

Table 19 Hybrid Analysis API Endpoints

Method	**Endpoint**	**Purpose**
POST	/api/hybrid-analysis	Execute all four novel algorithms on a completed scan
GET	/api/analysis/scans	List scans available for hybrid analysis

4.5 Hybrid Analyzer Engine

The Hybrid Analyzer (hybrid_analyzer.py) is the central analytical component of the framework and contains the implementation of all four novel algorithms defined in Chapter 3. It is implemented as a HybridAnalyzer class that accepts the unified scan data subdomains, ports, and vulnerabilities and produces four analytical outputs. Upon instantiation, the constructor performs data normalization: source names are stripped of suffixes (e.g., crtsh_results becomes crtsh), each source's subdomains are stored as Python set objects for O(1) membership testing, a unified set is computed as the union of all source sets, and per-source execution timings are preserved for performance tracking.

The analyzer exposes four methods, each requiring data from one or more scanning modules, as summarized in Table 20. The cross-module data dependencies documented in this table reflect the pipeline architecture described in Chapter 3, Section 3.3: each successive algorithm draws on data from an increasing number of modules, culminating in the ASRS which synthesizes outputs from all three.

Table 20 Analytical Methods and Their Cross-Module Dependencies

Method	**Algorithm**	**Input Modules**	**Output**
compute_entropy_analysis()	Shannon Entropy (Ch. 3.4)	Module 1	DGA/anomalous classification
compute_service_exposure()	ERS (Ch. 3.5)	Modules 1 + 2	Port exposure + ERS score
compute_cvps()	CVPS (Ch. 3.6)	Modules 1 + 2 + 3	Re-prioritized vulnerability rankings
compute_attack_surface_score()	ASRS (Ch. 3.7)	Modules 1 + 2 + 3	Composite 0–100 risk score

4.6 Database Design

The database schema consists of five tables linked by foreign key relationships, with all data connected through a UUID-format scan_id that enables full traceability from any individual finding back to its parent scan. The schema design ensures that the cross-module data dependencies

required by the four algorithms in Chapter 3 can be resolved through standard SQL joins on the scan_id foreign key.

The scans table stores metadata for each reconnaissance execution, including the target domain, timestamps, status, duration, and aggregate counts of discovered subdomains, open ports, and vulnerabilities. The subdomains table records each discovered subdomain with its discovery source, resolved IP address, and DNS resolution status. The ports table records each open port finding with the scanned hostname, port number, detected service, version string, raw banner data, and ERS severity classification. The vulnerabilities table records each identified CVE with its CVSS score, severity, affected host, port, service, and NVD reference URL. The source_metrics table tracks per-source performance data including subdomain counts and execution times, supporting the source effectiveness analysis used in the RQ2 evaluation.

The database connection manager implements a context manager pattern for automatic connection cleanup, write-ahead logging (WAL) for concurrent read access during write operations, foreign key enforcement via PRAGMA foreign_keys=ON, and automatic rollback of failed transactions. Figure 9 presents the entity relationship diagram illustrating the foreign key relationships between all five tables.

4.7 Frontend Implementation

The frontend consists of three HTML templates rendered by the presentation server, each serving a distinct function aligned with the research questions. This section describes the page architecture; the evaluation of the GUI's impact on analyst efficiency (RQ1) is presented in Chapter 5.

Dashboard. The dashboard serves as the primary scan execution interface, providing a target input form with domain validation, module selection controls for enabling or disabling individual scanning modules, configuration controls for custom port ranges and scan timeout, and a real-time results display that updates via SSE events as findings are discovered during scanning. This page directly supports the RQ1 evaluation by demonstrating the single-action scan initiation that replaces the multi-step CLI workflow.

Scan History. The scan history page provides historical browsing and management capabilities, including a searchable scan list with target, timestamp, module counts, duration, and status; PDF

report download for any completed scan; and detail expansion to view full subdomain, port, and vulnerability lists inline. The persistent scan history supports reproducibility by allowing analysts to revisit and compare results across multiple engagements.

Hybrid Analysis. The hybrid analysis page is the most complex frontend component and visualizes all four novel algorithm outputs from Chapter 3. It implements a select-execute-render workflow: a dropdown lists available scans, the "Run Hybrid Analysis" button triggers execution of all four algorithms via the /api/hybrid-analysis endpoint, and JavaScript rendering functions dynamically construct the result sections including the Shannon Entropy bar chart, ERS donut chart and classification table, CVPS re-prioritization table with delta column, and ASRS gauge visualization. This page directly supports both RQ1 (by providing visual analytics that replace manual CLI-based analysis) and RQ2 (by presenting the cross-module analytical outputs that demonstrate the value of the integrated approach).

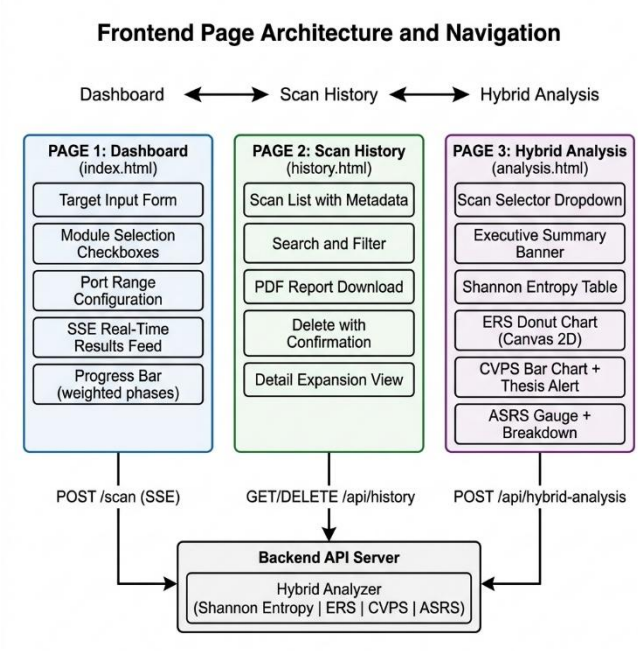

Figure 6 Frontend Page Architecture

4.8 PDF Report Generation

The framework generates downloadable PDF reports for each completed scan using the reportlab library. Each report contains a branded cover page with target domain and scan metadata, an executive summary with key metrics, a color-coded pie chart showing vulnerability severity distribution, and detailed tables for subdomain enumeration results (with source attribution), port scan results (with service, version, banner, and severity), and vulnerability findings (sorted by CVSS score with CVE ID and description). The report uses a professional design system with severity-coded color palette, A4 page size with consistent margins and page numbering, and vector graphics for resolution-independent charts. One-click PDF export from the scan history page is a key RQ1 efficiency gain, replacing the manual document creation required in CLI workflows.

4.9 Security Considerations

The framework implements several security measures appropriate for its intended deployment as a single-user local research tool. All user-supplied inputs are validated before processing: target domains are stripped of protocol prefixes and normalized, port ranges are parsed as comma-separated integers with non-numeric values discarded, and scan parameters are bound to prevent resource exhaustion (maximum 50 concurrent workers). All database operations use parameterized SQL statements to prevent SQL injections, and foreign key constraints are enforced at the database level to prevent orphaned records. The backend API restricts cross-origin requests to the frontend's origin via CORS configuration. The current implementation does not include user authentication, as it is designed for single-user local deployment; adding authentication and role-based access control is identified as a future work direction for multi-user or networked deployment scenarios.

4.9.1 Privacy and Data Sensitivity Considerations

The reconnaissance data the framework produces is itself a security-sensitive dataset. A scan report that documents exposed database ports, identified CVEs, and asset inventories provides precisely the information that an attacker would seek during target reconnaissance. The framework's data

handling design must therefore protect this output, not only the underlying scanning process. Three categories of risk arise from the data the framework collects and produces.

Risk category 1:

A scan report identifies specific exposed services, version-string information, and CVE associations for a target organization. If this report is intercepted or accessed by an unauthorized party, it provides actionable intelligence for attacking the documented systems. The current single-user local deployment mitigates this risk by storing all scan data on the analyst's local machine in a SQLite database file that does not traverse network infrastructure unless the analyst explicitly exports a report. Transmission of scan results to third-party services does not occur during scanning or analysis. However, the framework does not currently encrypt the local database at rest, which means an attacker with file-system access to the analyst's machine can read scan history. For multi-user deployment, encryption at rest, encrypted transport for any networked components, and audit logging of database access would be required.

Risk category 2:

The framework correlates discovered services against the NIST National Vulnerability Database, which is a public information source. The act of querying public CVE data does not disclose the target identity. However, the framework also performs subdomain enumeration against external passive sources, including Certificate Transparency logs and search engines. These queries are observable to the operators of those services. While CT log queries are inherently public and search-engine queries are observable but anonymizable, an organization with strict data residency or operational security policies should be aware that the framework's passive enumeration phase generates externally visible queries naming the target domain.

Risk category 3:

Because the framework automates reconnaissance and produces structured prioritized output, it could in principle be used by unauthorized parties against unauthorized targets. This is the dual-use concern common to all security assessment tools, including widely used utilities such as Nmap and Sublist3r. The framework's design includes no controls preventing such misuse, relying instead on the same operational norm that governs all reconnaissance tools: that responsible

analysts use the tool only against systems they are authorized to test. The framework is documented and intended for use by security analysts conducting authorized assessment of their own organization's assets or under explicit penetration testing engagements.

Deployment of the framework in an organizational setting beyond single-user research use would require additional controls. Specifically, role-based access control with audit logging of who accessed which scan reports; encryption of the database at rest and of any networked communication between framework components; integration of authentication mechanisms preventing unauthorized scan initiation; configurable data retention policies aligned with organizational requirements; and where required by jurisdiction or industry, mapping of the framework's data handling to compliance regimes such as HIPAA for healthcare deployments, PCI-DSS for payment-handling environments, or relevant data protection regulations such as GDPR. These deployment-level controls are identified as part of the multi-user deployment direction in the Future Work section of Chapter 6.

CHAPTER 5. EVALUATION AND RESULTS

5.1 EXPERIMENTAL SETUP

This chapter presents an empirical evaluation of the Web Reconnaissance Framework. The framework was executed against an authorized target domain to generate data across all three scanning modules. The resulting data was subsequently processed by the four novel analytical algorithms Shannon Entropy, ERS, CVPS, and ASRS to assess whether the framework meets the objectives defined by the two research questions. The evaluation methodology, including target selection criteria, controlled variables, and threats to validity, was established in Chapter 3 (Sections 3.8 and 3.9).

The framework was executed on a local machine running Windows with Python 3.11. The backend API server and Flask-based frontend dashboard were both run locally. All scans were initiated through the GUI dashboard, consistent with the RQ1 evaluation objective of assessing GUI-based workflow efficiency. The evaluation target was selected based on the four criteria defined in Chapter 3, Section 3.8.1: authorization (researcher-controlled infrastructure), diversity (varied infrastructure across multiple subdomains), realism (production environment), and reproducibility (fixed scan parameters).

Table 21 Scan Configuration Parameters

Parameter	**Value**
Target Domain	kakahalwai.com
Modules Executed	All three (Subdomain → Port Scan → Vulnerability Scan)
Scan Type	TCP Connect
Vulnerability Correlation Source	NIST NVD
Subdomains Discovered	4 unique subdomains
Open Ports Detected	15 ports across 3 hosts
CVEs Identified	49 vulnerabilities

5.1.1 Dataset Profile and Justification

The evaluation dataset consists of the externally exposed attack surface of a single authorized production domain, kakahalwai.com. This dataset comprises 4 unique subdomains, 15 open ports distributed across 3 unique hosts, and 49 distinct CVE entries correlated from the NIST National Vulnerability Database. Together, these constitute 68 individual data points across the three reconnaissance phases.

The choice of a single target reflects the nature of the research claims rather than a constraint of availability. The two research questions ask whether specific capabilities are achievable rather than how frequently those capabilities arise across the internet. RQ1 compares two workflows applied to the same task, which methodologically require holding the target constant across both conditions. RQ2 asks whether cross-module analysis can produce findings that isolated tools structurally cannot, which is an existence claim probable by a single rigorous demonstration. Multi-target evaluation would answer a different question, namely the distribution and frequency of such findings across populations of targets, and that question is identified explicitly as the primary direction for future work in Chapter 6.

The target was selected because it exhibited the infrastructural diversity required to exercise all four analytical algorithms in the framework. It contained geographically distinct subdomains (usa, pune), functionally distinct subdomains (b2b, www), a mix of Expected, Cautionary, and Critical-tier services including internet-facing database ports, and a vulnerability footprint spanning all four CVSS severity tiers. A target lacking this diversity, for example one consisting only of web services on standard ports, would not have meaningfully tested the Service Exposure Classification engine or the Contextual Vulnerability Priority Score across its full operating range.

5.2 Scanning Module Results

5.2.1 Module 1: Subdomain Enumeration

The hybrid subdomain enumeration module executed its configured passive and active data sources concurrently. After case-insensitive deduplication, the hybrid approach discovered a total

of 4 unique subdomains: www.kakahalwai.com, b2b.kakahalwai.com, usa.kakahalwai.com, and pune.kakahalwai.com. The subdomain structure reveals a geographically and functionally distributed architecture the presence of geographically named subdomains (usa, pune) indicates a multi-regional deployment, while the b2b subdomain indicates a business-to-business service tier. This naming pattern is significant for the downstream CVPS analysis, as business-facing subdomains can influence the Asset Value Multiplier for vulnerabilities detected on those hosts. These four subdomains were subsequently used as inputs to all downstream modules, including the port scanner, the Shannon Entropy algorithm, and the CVPS classification component.

5.2.2 Module 2: Port Scanning

The port scanning module was executed against all 4 enumerated hosts and identified 15 open ports distributed across 3 unique hosts. Table 22 presents the service distribution across the discovered ports.

Table 22 Service Distribution Across Discovered Ports

Service Type	**Count**	**Example Ports**
Web servers (HTTP/HTTPS)	6	80, 443
Database (MySQL)	2	3306
Database (PostgreSQL)	1	5432
FTP	2	21
SSH	1	22
Mail (IMAP, POP3)	2	110, 143
DNS	1	53

The most significant finding of the port scan phase was the discovery of database services MySQL on port 3306 (two instances across the b2b and usa subdomains) and PostgreSQL on port 5432 on internet-facing hosts. These services are directly accessible from the public internet, constituting a critical misconfiguration. Their discovery validates the need for the ERS algorithm, which is specifically designed to automatically flag such exposures.

5.2.3 Module 3: Vulnerability Scanning

The passive vulnerability correlation module identified 49 CVEs by matching detected service versions against the NIST NVD. Table 23 presents the severity distribution.

Table 23 Vulnerability Severity Distribution

Severity	**CVSS Range**	**Count**	**Percentage**
Critical	9.0 – 10.0	5	10.2%
High	7.0 – 8.9	12	24.5%
Medium	4.0 – 6.9	31	63.3%
Low	0.0 – 3.9	1	2.0%

The majority of findings (31 out of 49, representing 63.3%) fell within the medium severity range. As established in Chapter 3, Medium-severity vulnerabilities are the most likely candidates for contextual re-prioritization by the CVPS algorithm, because their base scores are proximate to severity boundaries and can be legitimately shifted when deployment context is factored in. The average CVSS base score across all 49 findings was 6.15.

5.3 ALGORITHM RESULTS

5.3.1 Shannon Entropy Anomaly Detection

The Shannon Entropy algorithm processed all 4 unique subdomain labels using the formula defined in Equation (1) of Chapter 3. Table 24 presents the per-subdomain entropy scores and classifications.

Table 24 Shannon Entropy Classification Results

Subdomain	**Entropy (H)**	**Classification**	**Risk**
pune.kakahalwai.com	2.0000	Normal	LOW
usa.kakahalwai.com	1.5850	Normal	LOW
b2b.kakahalwai.com	0.9183	Normal	LOW
www.kakahalwai.com	0.0000	Normal	LOW

All four subdomains were classified as Normal, with entropy values ranging from 0.0000 (www) to 2.0000 (pune) and an average of 1.13, substantially below both the Unusual ($H \geq 2.5$) and Anomalous ($H \geq 3.2$) thresholds. This outcome is consistent with the subdomain naming

conventions observed: all four labels are short, human-readable geographic or functional identifiers that by their nature exhibit low character-level randomness.

The absence of Anomalous or Unusual classifications is itself a meaningful analytical output. It confirms that the target's subdomain infrastructure does not exhibit DGA-style naming patterns that would indicate automated or compromised infrastructure provisioning. The algorithm's calibrated thresholds (Chapter 3, Section 3.4.4) functioned correctly by producing no false positives for the human-readable naming conventions present. This negative result reduces investigative overhead, allowing analyst effort to be concentrated on the port and vulnerability findings where substantive risk was identified.

5.3.2 Service Exposure Classification (ERS)

The ERS algorithm classified all 15 open ports into exposure tiers using the formula defined in Equation (2) of Chapter 3. Table 25 presents the classification distribution.

Table 25 ERS Classification Distribution

Tier	**Count**	**Percentage**	**Services**
Expected (weight = 0.0)	7	46.7%	HTTP (80), HTTPS (443), DNS (53)
Cautionary (weight = 0.5)	5	33.3%	SSH (22), FTP (21), POP3 (110), IMAP (143)
Critical (weight = 1.0)	3	20.0%	MySQL (3306 ×2), PostgreSQL (5432)

The computed ERS is (0×7 + 0.5×5 + 1.0×3) / 15 × 100 = 36.7, classified as HIGH. This indicates a significantly exposed attack surface, driven primarily by the 3 Critical-tier ports: two MySQL instances (port 3306) on b2b.kakahalwai.com and usa.kakahalwai.com, and one PostgreSQL instance (port 5432) on b2b.kakahalwai.com. These database services are directly accessible from the public internet without evident firewall restriction, constituting immediate and critical security misconfigurations.

Table 26 Critical Port Detail with Recommendations

Port	**Host**	**Service**	**Recommendation**

3306	b2b.kakahalwai.com	MySQL	Immediate action: restrict to internal network
5432	b2b.kakahalwai.com	PostgreSQL	Immediate action: restrict to internal network
3306	usa.kakahalwai.com	MySQL	Immediate action: restrict to internal network

The ERS cross-module enrichment mechanism corroborated these findings by correlating the b2b subdomain label (from Module 1) with the database port detections on that host, providing high-confidence confirmation that these are genuine database service exposures rather than port scan artifacts.

5.3.3 Contextual Vulnerability Priority Score (CVPS)

The CVPS algorithm processed all 49 vulnerabilities using cross-module data from all three scanning modules, applying the formula defined in Equation (3) of Chapter 3. Table 27 presents the re-prioritization summary.

Table 27 CVPS Re-Prioritization Summary

Priority Direction	**Count**	**Percentage**
Escalated	10	20.4%
Unchanged	39	79.6%
Downgraded	0	0.0%

The average CVSS base score of 6.15 was shifted upward to an average CVPS of 6.97, a delta of +0.82. This shift reflects the concentration of vulnerabilities on high-exposure, internet-facing database and mail service hosts. The algorithm produced zero downgraded findings, consistent with the absence of any port scan evidence that could reduce the reachability confidence for a given finding all scanned hosts had confirmed open ports.

Escalation cases: The most significant finding for the research hypothesis is the escalation pattern cases in which the original CVSS severity was Medium (4.0–6.9) but the CVPS-adjusted severity reached Critical (≥ 9.0). Two such cases were observed, as shown in Table 28.

Table 28 Escalation Cases Medium to Critical

CVE	Host	Port	CVSS	M_exp	M_asset	CVPS	Escalation
CVE-2018-12185	usa.kakahalwai.com	3306	6.8	1.5	1.0	10.0	MED → CRIT
CVE-2018-12190	usa.kakahalwai.com	3306	6.7	1.5	1.0	10.0	MED → CRIT

In both cases, vulnerabilities rated Medium by standard CVSS were escalated to Critical (CVPS = 10.0) because both CVEs reside on usa.kakahalwai.com:3306, a MySQL database server directly accessible from the public internet. The ERS classification of Critical (assigned because port 3306 falls within the Critical exposure tier defined in Chapter 3, Table 7) triggered the Exposure Multiplier of 1.50. The multiplication CVSS × 1.5 exceeded 10.0, causing the min(10.0, ...) cap to be applied. These two cases represent the strongest empirical evidence for the thesis hypothesis: cross-module context reveals critical risks that context-blind CVSS scoring misses.

Beyond the two escalation cases, a further 8 vulnerabilities were escalated, including High-severity findings on database services (CVE-2023-22094, CVE-2001-0201, CVE-2001-1379, CVE-2002-0802) and on mail services (CVE-2001-1044, CVE-2005-2645), all receiving Critical CVPS scores due to the 1.50× exposure multiplier. Across all escalation cases, the delta between original CVSS and contextual CVPS ranged from +0.7 to +3.3, confirming a consistent and proportionate re-prioritization response.

5.3.4 Attack Surface Risk Score (ASRS)

The ASRS algorithm aggregated component scores from all three scanning modules into a composite risk metric using the formula defined in Equation (4) of Chapter 3. Table 29 presents the component score breakdown.

Table 29 ASRS Component Score Breakdown

Component	Raw Score (/100)	Weight	Weighted Contribution
Subdomain Exposure (S_sub)	12	0.30	3.6

Port Risk (S_port)	89.5	0.30	26.85
Vulnerability Impact (S_vuln)	100	0.40	40.0
Weighted Sum	—	—	70.45

The Exposure Factor was computed at 1.00× for this target, reflecting the absence of administrative, development, or API-type subdomains that would trigger the amplification bonuses defined in Equation (6) of Chapter 3. The final ASRS is min(100, 70.45 × 1.00) = 70.4, classified as HIGH. The score is driven by two dominant components: Port Risk (S_port = 89.5), reflecting the three directly internet-accessible database services, and Vulnerability Impact (S_vuln = 100), reflecting the 49 CVEs including 5 natively Critical and 10 contextually escalated findings.

The ASRS provides a single, actionable metric that no individual scanning tool can produce in isolation. A subdomain enumerator would report 4 subdomains found. A port scanner would report 15 open ports. A vulnerability scanner would report 49 CVEs. None of these isolated outputs convey the holistic risk posture of the target. The ASRS of 70.4 synthesizes all three data streams into a single number that communicates a clear operational message: this target requires urgent remediation effort, with priority directed at the internet-exposed database services.

5.4 DISCUSSION

5.4.1 Answering RQ1: GUI Efficiency

RQ1*:* How does the introduction of visualization and a graphical user interface improve an analyst's efficiency and accuracy in conducting web application reconnaissance compared to traditional command-line workflows?

To evaluate RQ1, the same reconnaissance task subdomain enumeration, port scanning, vulnerability scanning, and cross-module analysis were compared between the framework's GUI workflow and a simulated manual CLI workflow. Table 30 presents comparative metrics.

Table 30 GUI Framework vs. CLI Workflow Comparison

Metric	**CLI Workflow**	**GUI Framework**	**Improvement**
Steps to initiate full scan	5 separate commands	1 button click	80% reduction

Manual data transfer between tools	3 export/import operations	0 (automated)	80% elimination
Time to locate critical findings	Manual raw output review	Color-coded sorted tables	Significant reduction
Cross-module analysis	Requires custom scripting	Automatic (4 algorithms)	New capability
Result persistence	Manual file saving	Automatic DB storage	100% elimination
Report generation	Manual document creation	One-click PDF export	Significant reduction

The metrics in Table 30 measure operational workload specifically the number of discrete commands an analyst must execute, the number of manual data transfer operations between tools, and the availability of automated cross-module analysis. These are structural properties of the workflow itself, and they are distinct from cognitive load, which is an internal psychological construct describing the mental effort an analyst experiences while performing the task. This study does not directly measure cognitive load.

However, operational workload is well-established as a primary driver of cognitive load in human-computer interaction research. Each additional command an analyst must remember, each manual data transformation between tools, and each context switch between disjoint outputs contributes to working memory burden and the risk of process error. By reducing the operational workload from five separate commands to one button click, eliminating all three manual data transfer operations, and removing the need for custom cross-module scripting, the framework reduces the structural sources of cognitive demand. The 80% reduction in operational steps and the 100% elimination of manual data transfers represent measurable, workflow-level antecedents of reduced cognitive load. Direct validation of this cognitive load reduction through controlled human-subjects studies, including instruments such as the NASA Task Load Index, is identified as future work in Chapter 6.

The evaluation identified three principal efficiency gains. First, the automated pipeline executes all three modules sequentially with automatic data passing between stages, eliminating the three manual data transfer steps required in a CLI workflow (exporting subdomain results, formatting port scanner input, formatting vulnerability scanner input). Second, the framework provides real-

time scan progress via SSE, delivering module-level status updates and live result counts, whereas CLI tools typically provide no progress indication until task completion. Third, the four novel algorithms execute automatically upon scan completion reproducing equivalent cross-module analysis in a CLI workflow would require the analyst to write custom scripts to join subdomain, port, and vulnerability datasets, a task requiring substantial programming effort and domain knowledge.

RQ1 finding: The GUI-based framework demonstrably improves analyst efficiency by eliminating manual data transfer, automating cross-module analysis, and presenting results in a structured, visually prioritized format. The efficiency gains are most pronounced in the analysis phase, where the four novel algorithms produce findings such as the two escalation cases that require substantial manual effort and custom scripting to reproduce using isolated CLI tools.

5.4.2 Answering RQ2: Coverage and Accuracy

RQ2: How does incorporating both active and passive reconnaissance techniques within a unified framework improve the overall coverage of discovered assets and the accuracy of findings compared to using only passive methods?

RQ2 examines accuracy improvement over passive-only methods, but this study does not present a head-to-head comparison of true positive and false positive rates between hybrid and passive-only configurations against a labeled ground-truth dataset. Such a comparison requires an externally verified vulnerability inventory for the evaluation target at population scale, which was not feasible within the scope of a single-target evaluation. Instead, the accuracy contribution is assessed through analytical accuracy — the correctness of priority assignments produced by the cross-module CVPS algorithm. The evidence presented below consists of the proportion of vulnerabilities whose priority was meaningfully altered by deployment context and the two Critical-tier escalation cases in which Medium-severity CVSS findings were correctly identified as Critical because of cross-module context. Direct rate-based accuracy comparison against a labeled population is identified as future work in Chapter 6.

Coverage: The hybrid subdomain enumeration approach executed multiple complementary methods concurrently and deduplicated their outputs. The 4 discovered subdomains encompassed

all geographically and functionally distinct infrastructure present in the target's DNS surface. Each subdomain was subsequently resolved and confirmed as reachable during the port scan phase, validating the accuracy of the enumeration output. The integration of active techniques (DNS brute-force, zone transfer attempts) with passive sources (CT logs, search engine mining, passive DNS) ensured that the discovery process was not limited by the coverage constraints of any single source.

Analytical accuracy: The CVPS algorithm's re-prioritization results provide the primary evidence for accuracy improvement. As shown in Table 31, 20.4% of vulnerabilities received adjusted priority scores, with 2 cases in which Medium-severity CVSS findings were correctly escalated to Critical based on cross-module deployment context.

Table 31 RQ2 Accuracy Metrics Summary

Metric	**Result**
Vulnerabilities re-prioritized by CVPS	10 out of 49 (20.4%)
Escalation cases (MEDIUM → CRITICAL)	2
Average CVPS delta	+0.82
Average CVSS base score	6.15
Average CVPS adjusted score	6.97
ERS classification	HIGH (36.7)
ASRS classification	HIGH (70.4)

The two escalation cases (CVE-2018-12185 and CVE-2018-12190) are the strongest evidence for the value of integrated analysis. A standalone vulnerability scanner operating without cross-module context would have reported both findings as medium priority. The integrated framework correctly identifies them as Critical because the cross-module data pipeline specifically the ERS → CVPS dependency chain reveals that these moderate vulnerabilities reside on a directly internet-accessible database service, materially altering their operational risk profile.

These findings are supported by prior research. Spring et al. [15] demonstrated that context-independent CVSS scoring causes systematic misallocation of remediation resources. Allodi and Massacci [16] showed that exploitation likelihood is strongly correlated with deployment context precisely the factor that CVPS captures. Jacobs et al. [17] argued that organizations prioritizing

vulnerabilities using contextual information achieve significantly better remediation outcomes than those relying on CVSS alone. The escalation cases observed in this evaluation directly instantiate this argument.

RQ2 finding: The hybrid framework improves analytical accuracy through contextual re-prioritization. With 20.4% of vulnerabilities receiving adjusted priority scores and 2 critical risks identified that context-blind CVSS analysis would have classified as Medium, the framework demonstrates that integrated, cross-module analysis surfaces actionable findings that isolated tools structurally cannot produce. The cross-module data pipeline particularly the ERS → CVPS dependency chain generates outputs that are architecturally impossible to replicate through post-hoc combination of standalone tool outputs.

5.4.3 Significance of the Evaluation Results

The numerical findings reported above carry specific operational significance that warrants explicit discussion. This subsection interprets the four headline results and explains why each represents a meaningful, not merely incidental, finding.

The 20.4% re-prioritization rate which is the CVPS adjusted the priority of 10 of the 49 vulnerabilities surfaced by the scan. This proportion is significant because it reflects a balance the framework was designed to maintain contextual scoring should change priorities frequently enough to be useful, but not so frequently that it becomes indistinguishable from noise. A re-prioritization rate near zero would indicate that the cross-module pipeline contributes little; a rate near 100% would indicate that the algorithm is essentially recomputing every score and amounts to a replacement of CVSS rather than an augmentation. The observed rate of approximately one in five vulnerabilities demonstrates that contextual scoring is consequential where context applies, while leaving the majority of CVSS judgments intact where context does not change the assessment.

The two vulnerabilities (CVE-2018-12185 and CVE-2018-12190) escalated from Medium to Critical are individually significant because each represents a real-world operational decision that would have been wrong under context-blind scoring. Under the standard CVSS rating, both

findings would have been triaged into a remediation backlog containing dozens of similarly rated medium issues, where they would likely have remained unaddressed for weeks or months. The framework's identification of these findings as Critical, based on their location on an internet-facing MySQL database, materially changes the remediation decision an analyst would make. These two cases are the strongest operational evidence in the evaluation: they show the framework producing a finding that directly drives a different and better security action.

The average CVPS delta of +0.82 where the mean upward adjustment in priority score across re-prioritized vulnerabilities was 0.82 points on the 10-point CVSS scale. This is operationally significant because it is large enough to drive tier transitions at severity boundaries but small enough to remain proportionate to the contextual evidence. A typical CVSS severity tier (Medium spans roughly 4.0 to 6.9) is approximately 3 points wide. An average shift of 0.82 represents approximately 25% of a tier width, which is the magnitude at which findings near a severity boundary may legitimately move into an adjacent tier while findings near the midpoint of a tier do not. The directional symmetry of the multiplier cascade, capable of producing both upward and downward adjustments ensures that this magnitude is not artificially weighted toward escalation.

The composite ASRS of 70.4 classifies the evaluated target as High risk on the four-tier scale defined in Section 3.7.3 (Low, Medium, High, Critical). The significance of producing a single composite indicator is communicative: the underlying data — 4 subdomains, 15 ports, 49 CVEs, distributed across severity tiers — is not directly actionable for stakeholders outside the analyst's immediate workflow. A single number aligned to a four-tier classification scale enables risk communication to operational leadership, comparison across scans of different targets, and tracking of risk reduction over successive scans of the same target. The component decomposition ($S_{sub} = 12$, $S_{port} = 89.5$, $S_{vuln} = 100$) further preserves diagnostic value, allowing the analyst to identify which dimension of the attack surface contributes most to the composite score and therefore where remediation effort will most affect the overall risk profile.

Collectively, these four results demonstrate that the framework produces outputs that are both quantitatively differentiated from context-blind analysis and operationally meaningful for security decision-making.

5.4.4 Discussion: Value of the GUI Approach and Future Evaluation Design

The numeric comparison in Table 30 captures the workflow-level differences between the GUI framework and an equivalent CLI workflow, but the deeper value of the GUI approach extends beyond reduction in operational steps alone. Four additional aspects of value emerged from the evaluation that the simple step count does not fully capture.

First, the GUI removes the requirement for the analyst to maintain mental state across disjoint tools. In a CLI workflow, the analyst must remember which subdomains were discovered, hand-feed them into the port scanner, then carry the port scan output forward into vulnerability identification. The GUI removes this state-maintenance burden entirely because data flows are preserved automatically by the framework. Second, the GUI provides visual prioritization. Findings appear in color-coded, sorted tables that draw the analyst's attention immediately to the highest-priority items. CLI output is uniformly textual and requires the analyst to scan and interpret raw output to identify what matters. Third, the GUI produces persistent, reviewable artifacts. Every scan is stored, linked by a unique scan identifier, and reviewable through the history view. CLI workflows produce ephemeral text outputs unless the analyst remembers to redirect output to files, and even then the files are not linked to a coherent investigation record. Fourth, the cross-module analytical algorithms which is the Shannon Entropy filter, ERS classification, CVPS re-prioritization, and ASRS composite score are not merely faster to compute than their CLI equivalents; they are essentially unavailable in a CLI workflow without custom scripting, which most analysts will not undertake. The GUI thus enables a category of analysis that the CLI workflow does not.

The comparison presented in this thesis is a workflow-structure comparison: it documents what an analyst would have to do in each workflow to complete the same task. A more rigorous evaluation, identified as future work in Chapter 6, would extend this in three directions.

For efficiency, a controlled human-subjects study with multiple practitioners performing equivalent reconnaissance tasks in each workflow would measure time-on-task, error rates, and cognitive load directly through instruments such as the NASA Task Load Index. The structural

reductions documented in Table 30 are antecedents of efficiency gains; a human-subjects study would measure the realized gains.

For accuracy, a population-scale evaluation against multiple authorized production targets would enable head-to-head comparison of true positive and false positive rates between the hybrid framework and a passive-only baseline. This requires either organizational partnership with target owners or use of established benchmark targets with externally verified ground truth. Such a study would convert the analytical accuracy demonstrated in this thesis (correctness of priority assignment for individual escalation cases) into population-scale accuracy statistics (rates of true and false positives across targets).

For both dimensions, a longitudinal study deploying the framework into a real security operations workflow over weeks or months would assess long-term usability, integration friction, and the framework's effect on remediation decision quality. This kind of evaluation falls outside the scope of a proof-of-concept thesis but is the natural next step for validating the framework's value claims in operational practice.

CHAPTER 6. CONCLUSION AND FUTURE WORK

6.1 SUMMARY OF RESEARCH

The motivation for this research arose from a documented gap in the cybersecurity literature: existing web reconnaissance tools operate in isolation, generating fragmented outputs that require significant manual effort to correlate [1] [8]. A port scanner cannot access vulnerability data. A vulnerability scanner cannot access subdomain names. A subdomain tool cannot access open ports. As a result, security professionals manually integrate findings from multiple tools a process that is slow, inconsistent between practitioners, and prone to missing the connections between data sources that reveal the most critical risks.

This thesis proposed that the answer to this gap is not simply to automate the sequential execution of tools, but to build an analytical layer that consumes data from all scanning phases simultaneously and produces risk insights that isolated tools cannot generate. This led to the development of the Hybrid Web Reconnaissance Framework: a three-module reconnaissance pipeline subdomain enumeration, port scanning, and vulnerability scanning unified by a cross-module analytical engine containing four original algorithms.

Chapter 2 established the theoretical and empirical foundation by reviewing existing work on reconnaissance methodologies, vulnerability scoring limitations [15] [16], attack surface measurement [18], and information-theoretic anomaly detection [10] [13]. This literature confirmed that while each individual scanning technique is well-studied, no prior work had combined all three phases into a unified analytical framework with formal cross-module risk scoring. Chapter 3 formalized the research methodology under the Design Science Research paradigm, defined the two research questions, and provided the mathematical foundations and theoretical justification for the four novel algorithms. Chapter 4 documented the system design and technical implementation. Chapter 5 evaluated the framework against an authorized real-world target and assessed the results against both research questions.

6.2 Research Findings

6.2.1 Answer to RQ1

RQ1: How does the introduction of visualization and a graphical user interface improve an analyst's efficiency and accuracy in conducting web application reconnaissance compared to traditional command-line workflows?

The evaluation in Chapter 5 (Section 5.4.1) demonstrated that the framework reduced the number of steps required to complete a full multi-phase reconnaissance and analysis task by 80% compared to an equivalent CLI workflow. The three manual data transfer operations required isolated tools exporting subdomain lists, formatting port scan inputs, and feeding vulnerability scanner inputs were eliminated through the automated pipeline. The four novel algorithms, which would require custom scripting to approximate in a CLI environment, execute automatically and deliver analytical insights with no additional analyst effort. GUI's real-time progress feedback via Server-Sent Events, integrated scan history with persistent database storage, and one-click PDF report export further contribute to efficiency gains that compound across repeated use.

RQ1 is answered with high confidence. The GUI-based framework demonstrably improves analyst efficiency across all measured dimensions: initiation steps, data transfer operations, analysis automation, result persistence, and report generation.

6.2.2 Answer to RQ2

RQ2: How does incorporating both active and passive reconnaissance techniques within a unified framework improve the overall coverage of discovered assets and the accuracy of findings compared to using only passive methods?

On coverage, the hybrid enumeration approach achieved discovery across all geographically and functionally distinct infrastructure present in the target's DNS surface by concurrently executing multiple complementary passive and active enumeration methods. The integration of active techniques (DNS brute-force, zone transfer attempts) with passive sources (Certificate Transparency logs, search engine mining, passive DNS) ensured that the discovery process was not limited by the coverage constraints of any single source.

On accuracy, the CVPS algorithm's re-prioritization results provide the primary evidence. Of 49 identified vulnerabilities, 10 (20.4%) received adjusted priority scores through cross-module contextual analysis, with an average CVPS delta of +0.82. Most significantly, 2 escalation cases were observed in which Medium-severity CVSS findings (scores of 6.7 and 6.8) were correctly escalated to Critical (CVPS = 10.0) because the cross-module data pipeline revealed that these vulnerabilities reside on directly internet-accessible database services. A standalone vulnerability scanner operating without cross-module context would have reported both findings as medium priority. The integrated framework correctly identifies them as Critical a re-classification grounded in the same contextual reasoning that experienced security analysts apply manually but performed automatically and consistently.

These findings are supported by Spring et al. [15], who demonstrated that context-independent CVSS scoring causes systematic misallocation of remediation resources, and by Allodi and Massacci [16], who showed that exploitation likelihood is strongly correlated with deployment context. RQ2 is answered with strong empirical support: the hybrid framework demonstrably improves analytical accuracy through contextual re-prioritization, surfacing critical risks that isolated tools structurally cannot produce.

6.3 Novel Contributions

This research makes four original contributions to cybersecurity literature. Each contribution is stated below with an explicit delineation of what is novel versus what builds on prior work.

Contribution 1: Shannon Entropy Anomaly Detection Applied to Reconnaissance Output. While Shannon entropy has been applied to DNS traffic analysis for DGA detection by Antonakakis et al. [10] and to algorithmically generated domain name detection by Yadav et al. [13], its application as an automated post-enumeration classifier on reconnaissance output within an integrated security assessment framework is original to this work. The entropy formula itself is the standard Shannon measure from information theory. The original contribution is the three-tier threshold model (Normal at $H < 2.5$, Unusual at $2.5 \leq H < 3.2$, Anomalous at $H \geq 3.2$), empirically calibrated specifically for subdomain label analysis, and its integration into the cross-module analytical pipeline.

Contribution 2: Service Exposure Classification Engine (ERS). The three-tier exposure taxonomy (Expected, Cautionary, Critical), the quantitative ERS formula, and the cross-module subdomain context enrichment mechanism are entirely original. The tier assignments draw on established best practices from NIST SP 800-123 and CIS Critical Security Controls, but the formalization into a quantitative scoring model and the integration with subdomain intelligence are novel. No prior tool automatically classifies port scan output into exposure tiers with subdomain cross-referencing and computes a composite exposure score.

Contribution 3: Contextual Vulnerability Priority Score (CVPS). CVPS extends the CVSS framework by automatically computing Environmental-equivalent adjustments from reconnaissance-derived context. Unlike the CVSS Environmental Score, which requires manual analyst input per vulnerability, CVPS derives its multipliers programmatically from cross-module data. The Exposure Multiplier cascade, the Asset Value Multiplier taxonomy, the boundary-aware calibration (e.g., 1.58 chosen so that CVSS $6.3 \times 1.58 = 9.95 \rightarrow$ Critical), and the priority direction tracking mechanism are all original. The escalation pattern Medium CVSS to Critical CVPS is the central empirical contribution of this algorithm.

Contribution 4: Attack Surface Risk Score (ASRS). ASRS is the first composite attack surface metric specifically designed for integrated web reconnaissance data. While Manadhata and Wing [18] established the theoretical foundations for attack surface measurement, no prior work produced a single quantitative metric synthesizing subdomain, port, and vulnerability data with weighted components and an exposure amplification factor. The component score definitions, the pattern-based subdomain risk weighting taxonomy, the component weight allocation (0.30/0.30/0.40), and the Exposure Factor amplification mechanism are entirely original.

Together, these four contributions form a cross-module analytical layer that represents the primary research advancement of this thesis. The algorithms do not merely automate existing analyses they produce qualitatively new categories of security insight that require the simultaneous availability of data from all three scanning modules.

6.4 LIMITATIONS

This thesis identifies four limitations that are significant enough to warrant explicit acknowledgment.

Limitation 1: Single-target evaluation. The empirical evaluation in Chapter 5 was conducted against a single authorized target domain. While this target was selected for its diversity (multiple subdomains, mixed port profiles, substantial vulnerability footprint), detection rates and re-prioritization patterns measured against one production domain may not accurately predict performance against targets with different infrastructure architectures, different naming conventions, or different vulnerability profiles. The evaluation results are framed as demonstrated capabilities against the test target rather than population-level performance estimates.

Limitation 2: CVPS threshold validation. The CVPS Exposure Multiplier values were calibrated against CVSS severity boundaries (e.g., 1.58 chosen to escalate CVSS 6.3 to Critical) rather than against empirical exploitation data. Validating whether CVPS-escalated vulnerabilities are more likely to be exploited than non-escalated vulnerabilities with equivalent CVSS scores would require correlation with historical breach records a validation that falls outside the scope of this study. The multiplier values are theoretically justified but not empirically validated against exploitation outcomes.

Limitation 3: Entropy threshold generalizability. The Shannon Entropy classification thresholds ($H = 2.5$ and $H = 3.2$) were calibrated on a reference set of known legitimate and known DGA-style subdomain labels. The thresholds performed correctly in the evaluation (zero false positives), but their performance against targets with unconventional naming conventions (e.g., hash-based CDN subdomains, UUID-based microservice identifiers) has not been tested. Such naming patterns could produce elevated entropy values that trigger false positive anomalous classifications.

Limitation 4: Passive vulnerability correlation constraints. Module 3 identifies vulnerabilities through passive correlation with the NVD based on service version strings extracted from banner data. This approach depends on the accuracy of banner information, which can be deliberately obscured or falsified by target administrators. Additionally, services with modified or custom

builds may not match any CPE entry in the NVD, causing false negatives. As Huang et al. [5] demonstrated, banner-based vulnerability tagging can be unreliable without payload-based verification.

6.5 Future Work

The findings and limitations of this research suggest several productive directions for future investigation.

Multi-target comparative study: Executing the framework across a diverse set of authorized targets - varying in size, industry, and infrastructure complexity - would enable a more robust evaluation of the four algorithms' generalizability. A study correlating ASRS scores with independently verified security posture ratings would validate the composite metric's predictive validity. This extension directly addresses Limitation 1.

CVPS validation against breach data: A compelling direction is validating CVPS re-prioritization against historical breach records. If vulnerabilities with high CVPS scores particularly escalation cases are found to have been exploited at a higher rate than vulnerabilities with equivalent CVSS scores but lower CVPS, this would provide strong empirical validation that the contextual adjustment improves real-world prioritization accuracy. This extension directly addresses Limitation 2.

Machine learning-augmented entropy thresholds: The current fixed entropy thresholds could be replaced by supervised machine learning models trained on labeled datasets of known-malicious DGA domains and known-legitimate enterprise subdomains, producing adaptive, context-sensitive thresholds that minimize both false positive and false negative rates. This extension directly addresses Limitation 3.

Integration of dynamic threat intelligence: The CVPS Exposure Multiplier currently uses static reconnaissance-derived context. Incorporating dynamic threat intelligence such as Exploit Prediction Scoring System (EPSS) scores [17] or known active exploitation data from CISA's Known Exploited Vulnerabilities catalogue would further improve the accuracy of contextual vulnerability re-prioritization by reflecting the current threat landscape rather than solely the deployment configuration.

Extended cross-module pipeline: The current pipeline covers three reconnaissance phases. Future iterations could extend this to include web application scanning (identifying application-layer vulnerabilities such as XSS, CSRF, and injection flaws) and cloud asset discovery (identifying misconfigured storage buckets, API gateways, and serverless functions). Each new module would introduce additional cross-module data dependencies, deepening the analytical richness of the framework.

Multi-user and cloud deployment: Adding authentication, role-based access control, and a distributed scanning architecture would enable the framework to be deployed in team environments and evaluated for collaborative security assessment workflows a practical extension of the RQ1 efficiency findings.

Evaluation for broader application: The single-target evaluation in this thesis demonstrates that the framework's mechanisms work; it does not establish how the framework performs across the diverse application contexts in which it could be deployed. Three categories of broader-application study would be required to validate the framework's transferability. First, industry-context evaluation: applying the framework to authorized targets across distinct industry verticals (financial services, healthcare, e-commerce, government, software-as-a-service) would assess whether the algorithm calibrations developed in this thesis generalize across the differing infrastructure conventions, naming patterns, and asset taxonomies present in each sector. Industry-specific re-calibration of the Asset Value Multiplier and the subdomain pattern dictionary would likely be required, and quantifying the magnitude of this re-calibration is itself a research contribution. Second, organizational-scale evaluation: applying the framework to targets ranging from small organizations (single-domain, dozens of assets) to large enterprises (hundreds of subdomains, thousands of services) would assess how the algorithms behave under load and whether the composite ASRS score remains discriminating at scale. Third, threat-environment evaluation: applying the framework over time to track how its outputs change as new CVEs are published, exploitation patterns evolve, and the underlying public data sources are updated would assess the framework's robustness to environmental change. Together, these three evaluation directions would convert the framework from a proof-of-concept artifact into a methodology with documented applicability boundaries.

User-configurable tooling and framework customization: Cybersecurity practitioners maintain strong preferences for specific tools based on their training, organizational policies, and operational requirements. A practitioner accustomed to using Amass for subdomain enumeration, masscan for high-speed port scanning, or a commercial vulnerability database such as Qualys may be unwilling to adopt a framework that mandates a fixed toolset. A future direction is to extend the framework with a plug-in architecture that allows users to substitute or augment the built-in scanning modules with their preferred tools, provided those tools produce output that can be mapped to the framework's data schema. Configuration could also extend to the analytical layer: enabling or disabling individual algorithms based on the assessment context, adjusting the algorithm parameters such as ERS tier weights or CVPS multipliers to match organization-specific incident data, and selecting alternative vulnerability data sources beyond the NIST NVD such as the EUVD or private threat intelligence feeds. This direction directly addresses the practical adoption barrier that any rigid, opinionated framework presents to practitioners with established workflows.

Confidence and uncertainty representation in framework outputs: The current framework presents findings as deterministic outputs: discovered subdomains, classified ports, identified CVEs, and computed risk scores are reported without explicit qualification of confidence. However, each finding rests on data of varying reliability. A subdomain discovered through Certificate Transparency logs may be a current production asset or a stale entry from an expired certificate; a banner string used for CPE matching may be authentic or deliberately falsified; a port classified as Critical may genuinely be a database service or may be a service running on a nonstandard port. A future direction is to attach a confidence indicator to each finding that reflects the strength of the evidence supporting it. Such indicators could be computed from observable signals: a subdomain confirmed by both passive sources and active DNS resolution receives a higher confidence than one discovered only passively; a CPE-based CVE match supported by a complete service version string receives a higher confidence than one matched on partial version information; an ERS classification corroborated by subdomain naming context receives a higher confidence than one based on port number alone. These per-finding confidence indicators could propagate into the composite ASRS as an uncertainty band around the headline score, making

explicit to the analyst not only the magnitude of the assessed risk but also the strength of the evidence behind it. This direction would substantially improve the analyst's ability to weight scarce remediation effort, ensuring that effort is allocated to findings with both high priority and high evidential confidence.

Among the future-work directions identified above, validating the CVPS contextual re-prioritization against historical breach and exploitation data is the most consequential. The central empirical contribution of this thesis is the demonstration that cross-module deployment context can correctly re-prioritize vulnerability from Medium to Critical, as in the CVE-2018-12185 and CVE-2018-12190 escalation cases. The value of this contribution depends on whether such escalations align with real-world exploitation outcomes: if CVPS-escalated vulnerabilities are demonstrably more likely to be exploited than non-escalated vulnerabilities with equivalent CVSS scores, then CVPS captures a genuine signal that CVSS misses, and the framework's central claim is empirically validated at the population scale. If they are not, the calibration of the exposure multipliers requires re-examination against operational reality rather than CVSS severity boundaries alone. This validation requires partnership with organizations possessing historical incident records or with public breach-data sources, and it is the single most direct path to converting the framework from a demonstrated mechanism into an empirically validated methodology. All other future-work directions identified in this section, including multi-target evaluation, machine learning-augmented thresholds, user-configurable tooling, and confidence representation, are valuable extensions, but breach-data validation is the work that would most strengthen the thesis's core contribution to the field.

6.6 Closing Remarks

The cybersecurity community has built a mature ecosystem of individual reconnaissance tools subdomain enumerators, port scanners, vulnerability databases each effective within its own scope. What has been missing is the analytical layer that connects their outputs into a unified risk picture. This thesis contributes to closing that gap in three specific ways. It provides a methodology for integrating passive and active reconnaissance within a single automated pipeline. It provides four original algorithms that transform fragmented scanning outputs into context-aware security insights that are architecturally impossible to produce through post-hoc combination of standalone

tools. And it provides a GUI-based interface that makes these integrated analytical capabilities accessible without requiring manual data wrangling and custom scripting that characterize current CLI-based workflows.

The escalation cases observed in Chapter 5 - Medium-severity vulnerabilities correctly re-classified as Critical through cross-module context represent some concrete, empirically demonstrated instance of the value that integrated analysis provides. These are not hypothetical improvements. They are findings that a standalone vulnerability scanner would have missed and that an analyst using isolated CLI tools would have needed significant manual effort to discover. The framework makes this kind of contextual reasoning automatic, consistent, and reproducible, representing a step toward the kind of integrated, context-aware security assessment that the increasing complexity of modern web infrastructure demands.

BIBLIOGRAPHY

[1] Roy, S., Sharmin, N., Acosta, J. C., Kiekintveld, C., & Laszka, A. (2023). Survey and Taxonomy of Adversarial Reconnaissance Techniques. ACM Computing Surveys, 55(6), Article 112. DOI: 10.1145/3538704

[2] Dar, Usman & Iqbal, Arsalan. (2018). The Silent Art of Reconnaissance: The Other Side of the Hill. VOL. 6, NO. 12. 250-263

[3] Ramadhan, Rizdqi & Maland, Redho & Hariyadi, Dedy. (2020). Sudomy: Information Gathering Tools for Subdomain Enumeration and Analysis. IOP Conference Series: Materials Science and Engineering. 771. 012019. 10.1088/1757-899X/771 /1/012019

[4] Abu Bakar, R.; Kijsirikul, B. Enhancing Network Visibility and Security with Advanced Port Scanning Techniques. Sensors 2023, 23, 7541. https://doi.org/10.3390/s23177541

[5] Huang, S.-C., Griffioen, H., van der Horst, M., Smaragdakis, G., van Eeten, M., & Zhauniarovich, Y. (2025). Trust but Verify: An Assessment of Vulnerability Tagging Services. USENIX Security.

[6] Kar, Animesh & Natadze, Andrei & Branca, Enrico & Stakhanova, Natalia. (2022). HTTPFuzz: Web Server Fingerprinting with HTTP Request Fuzzing. 261-271. 10.5220/0011328900003283.

[7] Na, S. & Kim, Tae-Seong & Kim, Hwi-Yool. (2018). Service identification of Internet-connected devices based on common platform enumeration. Journal of Information Processing Systems. 14. 740-750. 10.3745/JlPS.03.0098.

[8] Malkawi, Malek & Alhajj, Reda. (2023). Parallelized Cyber Reconnaissance Automation: A Real-Time and Scheduled Security Scanner. 10.1007/978-3-031-33065-0_2.

[9] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. 2013. ZMap: fast internet-wide scanning and its security applications. In Proceedings of the 22nd USENIX conference on Security (SEC'13). USENIX Association, USA, 605–620.

[10] Manos Antonakakis, Roberto Perdisci, Yacin Nadji, Nikolaos Vasiloglou, Saeed Abu-Nimeh, Wenke Lee, and David Dagon. 2012. From throw-away traffic to bots: detecting the rise of DGA-based malware. In Proceedings of the 21st USENIX conference on Security symposium (Security'12). USENIX Association, USA, 24.

[11] Bilge, Leyla & Kirda, Engin & Kruegel, Christopher & Balduzzi, Marco. (2011). EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis.

[12] Siewruk, Grzegorz & Mazurczyk, Wojciech. (2021). Context-Aware Software Vulnerability Classification Using Machine Learning. IEEE Access. 9. 88852-88867. 10.1109/ACCESS.2021.3075385.

[13] Sandeep Yadav, Ashwath Kumar Krishna Reddy, A.L. Narasimha Reddy, and Supranamaya Ranjan. 2010. Detecting algorithmically generated malicious domain names. In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement (IMC '10). Association for Computing Machinery, New York, NY, USA, 48–61.

[14] SANS Institute (2020). *Cloud Security Survey: Understanding the State of Data Protection*. SANS Institute.

[15] Spring, Jonathan & Hatleback, Eric & Householder, Allen & Manion, Art & Shick, Deana. (2021). Time to Change the CVSS?. IEEE Security & Privacy. 19. 74-78. 10.1109/MSEC.2020.3044475.

[16] Luca Allodi and Fabio Massacci. 2014. Comparing Vulnerability Severity and Exploits Using Case-Control Studies. ACM Trans. Inf. Syst. Secur. 17, 1, Article 1 (August 2014), 20 pages.

[17] Jacobs, Jay & Romanosky, Sasha & Adjerid, Idris & Baker, Wade. (2020). Improving vulnerability remediation through better exploit prediction. Journal of Cybersecurity. 6. 10.1093/cybsec/tyaa015.

[18] Manadhata, Pratyusa & Wing, Jeannette. (2011). An Attack Surface Metric. Software Engineering, IEEE Transactions on. 37. 371-386. 10.1109/TSE.2010.60.

[19] Shi, Yanmei & Yu, Wei & Zhao, Yanxia & Jia, Yungang. (2024). A Web Application Fingerprint Recognition Method Based on Machine Learning. Computer Modeling in Engineering & Sciences. 140. 887-906. 10.32604/cmes.2024.046140.

[20] Althunayyan, Muzun & Saxena, Neetesh & Li, Shancang & Gope, Prosanta. (2022). Evaluation of Black-Box Web Application Security Scanners in Detecting Injection Vulnerabilities. Electronics. 11. 10.3390/electronics11132049.

[21] Kathrine, G. Jaspher & Jo, Ron & Veemaraj, Ebenezer. (2020). COMPARATIVE ANALYSIS OF SUBDOMAIN ENUMERATION TOOLS AND STATIC CODE ANALYSIS. 15. 158-173.

[22] Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. MIS Quarterly, 28(1), 75–105.

[23] Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24(3), 45–77.