Project Overview

WebMapper is an automated web discovery tool designed to dramatically speed up penetration testing and web reconnaissance by enumerating hosts, services, web application endpoints, fingerprinting services, and mapping relationships into an interactive graph model. WebMapper captures metadata and scan history, supports import/export, and provides a GUI graph visualization with search and drill-down capabilities so both beginners and advanced pentesters can quickly understand web application structure and potential attack paths.

Chapter 1: Introduction

1.1. Introduction

In the modern digital landscape, web applications have become increasingly complex, often comprising multiple subdomains, virtual hosts, services, and interlinked endpoints. For security professionals, gaining a comprehensive understanding of this landscape is crucial to identify vulnerabilities, assess risks, and prioritize remediation. Traditional manual reconnaissance and enumeration methods are not only time-consuming but also prone to oversight, particularly in large-scale environments.

WebMapper addresses these challenges by providing an automated platform that streamlines web discovery, enumeration, and mapping into a single, cohesive workflow. By integrating host and service discovery, endpoint enumeration, and fingerprinting with a graph-based visualization, WebMapper allows security practitioners to quickly visualize relationships, detect potential attack paths, and make informed decisions. The tool maintains detailed metadata and scan histories to ensure repeatability and traceability of reconnaissance efforts.

Designed for usability, WebMapper caters to both novice and experienced penetration testers. Beginners benefit from the intuitive GUI and automated workflows, which reduce the learning curve and eliminate the need for extensive manual correlation of results. Advanced users gain access to granular controls, custom scanning profiles, and API/CLI integration, enabling tailored scans and seamless incorporation into existing security operations. Ultimately, WebMapper enhances efficiency, accuracy, and insight during penetration testing engagements, transforming complex web reconnaissance data into actionable intelligence.

1.2. Project Motivation

Manual web reconnaissance and enumeration during penetration testing can be time-consuming and prone to errors, particularly for complex applications with multiple subdomains, virtual hosts, and services. WebMapper aims to automate this process, providing a centralized, interactive visualization that enables faster decision-making, deeper insights, and more efficient identification of potential attack paths.

1.3. Problem Statement

Security professionals face challenges in managing large amounts of reconnaissance data, correlating subdomains, virtual hosts, services, and endpoints, and identifying high-risk areas efficiently. Existing tools often lack integrated visualization and historical tracking, requiring extensive manual effort to piece together findings. There is a need for a unified system that combines enumeration, fingerprinting, and graph-based mapping in a user-friendly interface.

1.4. Project Scope and Objectives

- Automate enumeration of web hosts, subdomains, virtual hosts, services, and endpoints.
- Provide accurate fingerprinting for services, web frameworks, and endpoints.
- Build an interactive graph model to represent relationships between assets.
- Maintain scan history and metadata for auditing and comparative analysis.
- Enable import/export and integration with existing pentesting tools.
- Offer both GUI and CLI/API interfaces to accommodate different user preferences and workflows.

1.5. Time Schedule

Phase	Duration	Activities
Requirement Analysis	1 Weeks	Gather requirements, define scope
Design	2 Weeks	System architecture, database design, UI mockups
Implementation	8 Weeks	Develop scanning engine, fingerprinting, graph visualization
Testing & QA	2-3 Weeks	Unit/integration/load tests, user feedback
Deployment	3-6 Days	Docker setup (if needed), sample deployment
Evaluation & Optimization	2 Weeks	Performance tuning, UX adjustments

1.6. Document Structure

Chapter 1: Introduction, Motivation, and Problem Statement
Chapter 2: Literature Survey on existing fraud detection methods
Chapter 3: Methodology data processing, feature engineering, and modeling
Chapter 4: Implementation system design, training, and evaluation
Chapter 5: Results, discussion, and visualization
Chapter 6: Future work, challenges, and trends
Chapter 7: Conclusion and acknowledgments

Chapter 2: Litrature Review

2.1. Background

Web reconnaissance and mapping are essential components of penetration testing and cybersecurity assessment. Understanding the architecture, services, endpoints, and interconnections within web applications allows security professionals to identify potential attack surfaces and vulnerabilities. Traditional approaches rely on multiple standalone tools for subdomain enumeration, port scanning, service fingerprinting, and endpoint discovery. While effective, these methods require extensive manual correlation and can be inefficient, especially when dealing with complex multi-host environments. The evolution of automated web discovery tools aims to consolidate these functions into integrated frameworks, providing centralized visibility and historical context for continuous security evaluation.

2.2. Related Works

Several tools and frameworks have addressed aspects of web discovery and mapping, each with specific focus areas:

- Nmap: Widely used for network scanning and service enumeration, providing detailed information on hosts and open ports.
- Amass: Focused on subdomain enumeration and passive discovery techniques, leveraging public datasets and brute-forcing methods.
- Sublist3r: Tool for subdomain enumeration using multiple search engines, often integrated with DNS and certificate transparency records.
- DirBuster / Gobuster / ffuf: Tools aimed at directory and endpoint discovery through wordlist-based fuzzing.
- Burp Suite / OWASP ZAP: Comprehensive web application security testing platforms that provide scanning and mapping capabilities, including some level of service fingerprinting.

2.3. Evaluation of Current States

An important capability of WebMapper is precise timestamping of each discovery and scanning action. Every pipeline step from passive collection and subdomain discovery to port probes, vhost checks, and endpoint enumeration records an exact timestamp and contextual metadata. This temporal data provides two key benefits:

1. **Incident Correlation**: If an actual attack or unexpected activity occurs on a target during an engagement, timestamps allow analysts to correlate WebMapper's automated actions with external logs (e.g., IDS/IPS alerts, web server logs, or SIEM events). This helps determine whether a detected incident was the result of the tool's scan traffic or an independent malicious event, reducing false positives and supporting forensic timelines.

2. **Professional Reporting & Compliance**: For professional engagements and audits, including detailed timestamps in reports improves transparency and traceability. Reports can include per-step timestamps, duration metrics, and historical comparisons (e.g., first-seen vs. last-seen), which are valuable for clients, legal records, and quality assurance.

WebMapper stores timestamps alongside raw responses and scan artifacts; these entries are immutable within the scan history log to preserve the integrity of forensic timelines. Access controls and export options enable teams to include or redact timestamped data according to client policies and privacy considerations.

Screenshots & Visual Evidence: In addition to timestamped artifacts, WebMapper allows users to capture screenshots of the graph canvas, node detail panels, and rendered endpoint responses directly from the GUI. These visual snapshots can be embedded in reports to provide immediate visual context for example, showing how a vhost was detected, the exact response used for fingerprinting, or the graph view that highlights an attack path. Screenshots are stored with metadata (timestamp, author, related node/scan ID) so they can be traced back to the originating scan action. Export options support embedding screenshots into HTML/PDF reports or exporting them as image files for separate documentation.

These combined capabilities precise timestamps, immutable scan history, and integrated visual evidence make WebMapper suitable for professional pentesting engagements, forensic timelines, and compliance-driven reporting.

Chapter 3: Methodology

This chapter outlines the step-by-step methodology adopted in the development of the Fraud Guard system. It includes the procedures for data collection, preprocessing, feature engineering, model selection, training, and evaluation.

3.1. Data Processing

Data processing in WebMapper focuses on reliably ingesting, normalizing, and storing diverse reconnaissance artifacts produced by passive and active discovery modules. Inputs include DNS records, certificate transparency entries, HTTP responses, service banners, directory listing results, JavaScript artifacts, and external feeds. The processing pipeline performs the following key tasks:

- **Normalization**: Convert heterogeneous inputs into a canonical schema (hosts, services, endpoints, fingerprints, metadata). Standardize timestamps to ISO 8601 and normalize encoding/character sets of raw responses.
- **Deduplication & Merging**: Identify and merge duplicate discoveries across sources using fingerprinting and unique keys (IP+port, host header+path). Maintain provenance metadata listing discovery sources and confidence scores.
- **Sanitization**: Strip or redact sensitive fields when configured before storing or exporting. Optionally hash or tokenise fields based on retention policies.
- **Storage**: Persist processed artifacts into the graph database and blob storage for raw responses and screenshots, linking them via stable IDs.

3.2. Feature Engineering

Feature engineering prepares raw discovery data into attributes suitable for fingerprinting models, heuristics, and scoring engines. Effective feature design improves detection of technologies, service behaviors, API endpoints, and anomalies.

- **API Endpoint Features**: Automated detection of API-specific endpoints through analysis of path patterns (e.g., /api/, /v1/, /graphql), OpenAPI/Swagger discovery, and heuristic checks in JavaScript files or HTTP responses. Features include method diversity (GET, POST, PUT, DELETE), parameter naming conventions, response structure, and presence of authentication headers. These signals enhance fingerprinting accuracy and enable endpoint classification for REST, GraphQL, or custom APIs.
- **Structural Features**: Presence of particular headers, HTTP status codes, path depth, URL parameter counts, and content-length distributions.

- **Temporal Features**: Time-based patterns such as scan frequency, response-time variability, and first/last-seen windows.
- **Network Features**: Port patterns, TTL variations, reverse DNS patterns, and TLS cipher suites.

3.3. Modeling

In this context, modeling refers to the logical organization of relationships and entities within the system rather than AI or ML prediction. WebMapper models the discovered environment using a graph-based data structure that reflects the real-world connections between web assets.

- **Nodes**: Represent hosts, subdomains, virtual hosts, services, and endpoints.
- **Edges**: Represent relationships such as resolves-to, runs-service, hosts-vhost, or links-to.
- **Attributes**: Each node and edge includes metadata like discovery source, timestamp, confidence score, and associated fingerprints.

This structured modeling allows users to visualize dependencies, pivot between related entities, and understand the hierarchy of web assets with clarity.

3.4. Model Evaluation

Model evaluation in WebMapper focuses on verifying data integrity, accuracy, and correlation reliability not on machine learning metrics. Validation routines ensure that:

- Relationships in the graph model are consistent and bidirectional where applicable.
- Timestamp ordering and metadata linkage remain intact.
- Duplicates and conflicting entries are automatically reconciled.
- Endpoints, vhosts, and subdomains correctly inherit parent relationships.

Periodic internal audits confirm the accuracy and completeness of the data model, ensuring that visualization and reporting are based on verified, traceable information.

3.5. System Integration

System integration ensures all modules in WebMapper communicate seamlessly and function as a unified platform.

- **Backend Integration**: Combines enumeration, fingerprinting, and storage modules under a central controller.
- **Database Integration**: Uses a database for efficient storage and querying of relationships.
- **Frontend Integration**: Connects the backend data model to an interactive web-based GUI for graph visualization and analysis.
- **Timestamp and Reporting Integration**: All collected data and screenshots include timestamps, allowing analysts to generate accurate, time-aware reports. This feature is particularly useful for forensic investigations and differentiating legitimate scan actions from real attacks.
- **Export & Reporting**: Users can export scans, metadata, and screenshots directly into reports for client delivery or documentation.

Together, these components create a cohesive system that simplifies complex reconnaissance data into actionable insights, supporting both efficiency and accountability in professional pentesting operations.