

“Red-Team Attack Simulation Dashboard”

Name of the Candidate

1. Rucha Bhalerao
2. Akshada Gundal

Enrollment No

MITU23BTCSD053
MITU23BTCSD004

Under the guidance of

Prof. Arun Ghandat(Guide)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MIT School OF COMPUTING

MIT Art, Design and Technology University

Rajbaug Campus, Loni-Kalbhori, Pune 412201

2025-26

ABSTRACT

Modern cyber-attacks are becoming highly sophisticated, multi-staged, and automated, making it difficult for instructors and learners to clearly understand how attacks evolve across different phases of the cyber kill chain. Traditional attack tools such as Nmap, SQLMap, MySQLMap, Metasploit etc. produce fragmented, command-line based outputs which require deep technical expertise to interpret. Because of this, students and analysts struggle to visualize how reconnaissance leads to exploitation, and how payload delivery eventually results in data exfiltration. This gap highlights the urgent need for a unified simulation environment that can visually demonstrate each step of the attack process in a structured and simplified manner.

To address this challenge, this research proposes a **Red Team Simulation Dashboard**, a centralised platform that graphically represents the entire attack flow from initial scanning to final data extraction. The dashboard integrates outputs from commonly used offensive tools and maps them into a visual kill-chain model. This allows users to easily understand which exploit succeeded, which stage failed, and how the attack lifecycle progresses. The system also supports real-time visualization, live log monitoring, and

accurate stage tracking, enabling both students and security practitioners to see the immediate outcomes of every step.

By combining multiple offensive tool results into one unified interface, the Red Team Simulation Dashboard improves practical learning, reduces complexity, and supports faster interpretation of attack data. It enhances understanding for college-level teaching, cyber-labs, and training scenarios, while also serving as a practical framework for demonstrating real-world red team methodologies. This makes the proposed dashboard a powerful educational and operational tool for modern cybersecurity environments.

Keywords: *Red Team Simulation, Cyber Kill Chain, Attack Visualization, Offensive Security Tools, Real-Time Monitoring*

Chapter 1

INTRODUCTION

1.1 Introduction

The rapid increase in cyber-attacks and the advancement of attacker techniques have made it necessary for organizations to adopt intelligent and proactive offensive security simulations. Modern cybercrime groups use automated scripts, AI-driven exploitation, and multi-stage attack chains to compromise systems, making manual penetration testing and traditional security audits insufficient. As systems grow more connected through cloud platforms, APIs, and shared networks, it becomes increasingly difficult for security teams to clearly visualize how an attacker progresses from reconnaissance to data exfiltration. Due to this complexity, Red Team Simulation Dashboards are now becoming essential. This project presents the design and development of a Red Team Simulation Dashboard that visually demonstrates the full cyber kill chain—including Reconnaissance, Exploitation, Payload Execution, and Data Exfiltration—in a centralized platform.

Unlike individual tools that give raw technical output (like Nmap, SQLMap, Metasploit, etc.), this dashboard aggregates and visualizes all the execution stages in one interface. Each stage of the simulated attack is displayed in real-time using graphs, kill-chain diagrams, timelines, and logs. The goal is to make offensive security more understandable, not only for red team professionals, but also for students and analysts who need to learn how real-world cyberattacks flow. When a stage succeeds or fails, the dashboard immediately updates the status indicators and visualizations, making it easier to analyze the attacker's path and the system's weaknesses. Additionally, the dashboard automatically generates detailed reports with attack results, affected assets, exploited vulnerabilities, and final outcomes. This makes documentation easier and helps both learning and professional reporting.

At the core of this simulation system is a unified engine that triggers execution of selected offensive tools and collects their results in structured form. The system parses tool outputs, maps them to kill-chain stages, and visualizes them in a graphical timeline for better analysis. Critical events such as successful payload delivery or privilege escalation are clearly highlighted. The modular architecture allows scaling for cloud targets, internal networks, or specific service targets. In the future, advanced features such as AI-based attack path prediction, automated exploit selection, and integration with MITRE ATT&CK datasets can be added to make the simulation more intelligent and realistic.

The dashboard's design focuses heavily on usability and accessibility. The UI is designed so that even a non-expert learner can understand how a cyberattack evolves step-by-step. Color-coding, risk indicators, stage completion status, and real-time logs provide clarity for decision-making and analysis. Exportable reports (in formats like JSON, PDF, CSV) help users share results or maintain documentation. Access control and security logging ensure that only authorized users can run simulations, view results, or modify configurations.

In conclusion, the Red Team Simulation Dashboard provides a powerful and modern approach to conducting offensive security simulations. Instead of reading raw terminal outputs or separate tool logs, security professionals and students can learn, analyze, and visualize cyberattacks in a structured and easy-to-understand manner. This system bridges the gap between theoretical red team knowledge and practical attack simulation, making proactive cybersecurity assessment more effective, visual, and actionable.

1.2 Existing Work

In the last several years, various Red Teaming platforms and offensive security frameworks have been developed to help cybersecurity teams simulate real-world attacks. Tools such as **Metasploit**, **SQLMap**, **Nmap**, **Hydra**, and **Feroxbuster** are widely used to perform reconnaissance, exploitation, privilege escalation, brute force attempts, and post-exploitation tasks. However, these tools mostly work individually, and their outputs are often in raw technical log format, making it difficult for new learners, small organizations, or academic users to clearly visualize the complete attack flow.

Frameworks like **MITRE CALDERA** and **Atomic Red Team** introduced automation in adversary simulation, enabling security teams to emulate ATT&CK Tactics, Techniques & Procedures (TTPs) with less manual scripting. These systems allow execution of pre-built attack chains and measure detection capability of defense tools. However, most of these platforms are hard to configure, require advanced knowledge, and are mainly used in enterprise SOC or government-level cyber labs. They also require strong infrastructure and trained professionals for proper setup.

Some commercial platforms such as **SafeBreach**, **Picus Security** and **AttackIQ** provide automated Breach & Attack Simulation (BAS) focused on enterprise environments. They offer dashboards, reports, and analytics to visualize attack impacts. But their licensing cost is extremely expensive, and therefore unrealistic for academic students, small cyber labs, colleges, and startup organizations. Additionally, they mostly focus on defensive validation and do not give detailed view of each stage of attack in a simplified visual manner that beginners can easily understand.

Therefore, there is a gap between raw offensive tools and enterprise-grade BAS systems. Existing solutions lack a **simple, unified, and visual dashboard** that shows the entire kill-chain step-by-step with clear graphical indicators. Our proposed **Red Team Simulation Dashboard** fills this gap by combining multiple offensive tools into one platform, converting their outputs into visual elements, and presenting the full attack path in a structured kill chain. This makes attack simulations easier to understand, use, teach, and document—especially for students, trainers, and entry-level security researchers

1.2 Objectives

The main objective of this project is to develop a Red-Team Attack Simulation Dashboard that allows users to clearly visualize every stage of a cyber attack in a structured and simplified manner. Instead of receiving raw and complex tool outputs like Nmap, SQLMap, or Metasploit logs, users will be able to view clean, organized, and readable results through a single graphical interface. The dashboard will present attack stages such as reconnaissance, exploitation, payload delivery, and data exfiltration step-by-step using the cyber kill chain model. This helps students, researchers, and security teams understand how attacks evolve from start to finish.

Another objective is to provide a safe and controlled platform for offensive testing and learning. The dashboard will allow simulated attacks to run in a restricted environment, preventing any real damage while still demonstrating realistic attacker behavior. It also aims to generate automated reports that summarize the results in clear and shareable formats for academic use, documentation, or further analysis. Ultimately, the project seeks to make red team simulations more accessible, more visual, and easier to understand, especially for beginners who struggle to interpret raw command-line outputs.

By simplifying offensive security workflows, integrating multiple tools into a single platform, and enabling real-time visualization, this Red-Team Simulation Dashboard intends to bridge the gap between traditional complex attack tools and modern educational requirements. The objective is not just to run attacks, but to make learning, observing, and reporting those attacks easier, faster, and more effective for everyone from cybersecurity students to professionals.

1.3 Scope:

➤ AI-Driven Attack Orchestration and Evaluation

The next generation of red-team simulators will use AI to autonomously generate, select, and adapt attack sequences based on observed system behavior and historical exercises.

- *Smart Scenario Generation*: ML models will craft realistic multi-stage attack paths that reflect current threat actor TTPs.
- *Automated Outcome Assessment*: AI will score effectiveness, highlight detection gaps, and recommend follow-up actions for defenders.

➤ **Integration with DevSecOps and CI/CD**

Red team simulations will be embedded into software delivery pipelines to validate security controls continuously and early.

- *Pre-deployment Attacks*: Simulated adversary runs triggered by CI jobs to test new builds.
- *Shift-left Emulation*: Rapid feedback to developers showing how new code might be abused in real attack chains.

➤ **Cloud-Native & Multi-Cloud Emulation**

Support for cloud-specific kill chains, ephemeral hosts, and API-driven services across hybrid and multi-cloud infrastructures.

- *Container & Serverless Scenarios*: Simulate lateral movement and privilege abuse in Kubernetes and serverless stacks.
- *Cloud Policy-Aware Execution*: Safe orchestration that respects cloud IAM, quotas, and policies.

➤ **Real-Time Simulation & Continuous Validation**

Move from episodic exercises to persistent, lightweight adversary emulation for continuous security validation.

- *Persistent Attack Agents*: Non-invasive agents that run benign emulations and telemetry in test/approved environments.
- *Live Integration with SIEM/SOC*: Immediate ingestion of simulation telemetry to measure detection and response performance.

➤ **Business-Logic & High-Fidelity User Simulation**

Beyond technical exploits, simulate abuse of real application flows and business logic to reveal high-impact gaps.

- *User Behavior Emulation*: Automated workflows that mimic legitimate and malicious user journeys.
- *Contextual Attack Generation*: Use app metadata and business rules to craft targeted, realistic scenarios.

➤ **IoT, Edge & OT Attack Libraries**

Expand simulation coverage to embedded and cyber-physical systems with resource-aware scenario models.

- *Lightweight Payloads*: Specialized modules for low-power devices and gateway-level exploits.
- *Physical–Cyber Cascade Modeling*: Show how device compromises can affect physical processes.

➤ **Regulatory, Ethical & Compliance Support**

Ensure simulations support policy, audit, and legal requirements for safe, compliant exercises.

- *Policy Templates & Consent Workflows*: Pre-flight checks and authorization flows to prevent collateral damage.
- *Compliance Mapping & Audit Logs*: Map emulated tactics to standards (e.g., NIST/ISO/ATT&CK) and produce audit-ready evidence.

➤ **Adaptive, Self-Updating Platform**

The dashboard will continuously evolve by ingesting threat intelligence and exercise results.

- *Live TTP Feeds*: Automated updates of attack techniques and payload signatures.
- *Self-Tuning Playbooks*: Simulation parameters adapt based on past detection success and environment feedback.

➤ **Educational/Lab Scope**

Focus: Lightweight, sandboxed simulations for classroom use.

Description: Provide Demo Mode scenarios, step-by-step visualizations, simplified reports, and grading/assessment support so students learn attack flow without risking production systems.

➤ **SOC Readiness & Validation Scope**

Focus: Measure SOC detection and response capabilities.

Description: Run controlled adversary emulations against SOC toolchains (SIEM, EDR) and produce detection coverage metrics (e.g., detected TTPs, MTTD proxies) and actionable remediation guidance.

➤ **Enterprise Breach & Attack Simulation (BAS) Scope**

Focus: Large-scale, scheduled red-team campaigns for enterprise networks.

Description: Support multi-target campaigns, distributed execution, scheduling, role-based access, and enterprise reporting with compliance mapping and executive summaries.

Chapter 2

CONCEPTS AND METHODS

1.1 Definition :

Cyber-attacks are increasing rapidly in frequency, automation, and complexity. Modern attackers continuously search for weaknesses in networks, servers, APIs, applications, user access, and cloud configurations. Traditional security approaches are reactive and tool-based, making it difficult for security learners and organizations to understand how attacks evolve step-by-step. As technologies evolve, hackers also adopt new exploitation techniques and AI-driven approaches, making traditional manual testing insufficient. Therefore, there is a strong need for a centralized, intelligent platform that visually simulates the entire attack sequence and helps users understand real adversary behavior.

2.2 Core concept

The core concept of the **Red Team Simulation Dashboard** is to create a unified platform that visually represents each stage of a cyber-attack based on the cyber kill chain model. Instead of viewing separate raw outputs from tools like Nmap, SQLMap, and Metasploit, this dashboard integrates all tool outputs into a single GUI, providing real-time visualization of reconnaissance, exploitation, payload delivery, privilege escalation, and data exfiltration.

The system converts raw tool logs into structured JSON data, displays live attack status indicators, and provides graphical charts and attack-flow diagrams. It simplifies the offensive security workflow, making it easier for students, trainers, and security analysts to understand how attacks operate in real scenarios. The dashboard also generates automated reports that can be shared for academic use, training evaluations, or cyber-lab documentation.

Thus, the proposed Red Team Simulation Dashboard combines automation, integration, and visualization to make offensive security more understandable, safe, and accessible for learners and professionals.

Chapter 3

LITERATURE SURVEY

"Red Team Cutting Edge: the other guy Centralised Users Interface for Red Teams Instruments" The people who wrote this paper are R. S. M. G., G. D. C. Cavalcanti, and D. Caivano, respectively. L. A. F. de Lima, Colombia. Barbosa, E. F., de Lima, A. M. de Lima, together with R. A. de A. Fernandes In this paper, we give away Red Team Knife (RTK), an integrated interface bringing together several popular red-shirting tools, which incorporate SQLMap, Feroxbuster, and Nmap as well. Making such instruments available to both cybersecurity experts and people with no experience performing threat assessments is an objective. [1]

"Visual Assessments for Cyber Red Teaming" The writers are Justin Hernandez, Dr. Benjamin Turnbull, and Joseph Yuen The creation of an automatically operated cyber reactive teaming system that develops and evaluates multiple levels of attacks using models to create infrastructure photographic evidence of a structure is covered in this research investigation.[2]

"Top Red Team Tools, Commercial and Open-Source in 2025" A short description of the different open-access and commercial red-teamed instruments that the fact are set to become accessible in 2025 serves as provided in this article. It talks about their attributes, strengths, and role in creating practical imminent danger visualisations. [3]

"A Case Presentation of the Red Team: analysing Sticky Note to the Complete Cloud Compromise" Writers: Dimitrov Filip, This example case study demonstrates how an emergency team assessment, initially starting with a straightforward troublesome notification, identified problems that resulted in an unprecedented cloud computing infrastructure breach. It reiterates how crucial thorough checks for security have become. [4]

"Unique Attack Vectors and Red Team Operations" In summary, the current paper concentrates on the most prominent techniques of attack associated with the red group activities, which contain telemarketing, infection through malware promotion, as well as hacking of networks. It sheds the light on the strategies alongside devices employed within red team investigations.[5]

D. Caivano, G. D. C. Cavalcanti, R. S. M. G. de Lima, L. A. F. de Lima, E. F. Barbosa, A. M. de Lima, and R. A. de A. Fernandes present Red Team Knife (RTK), an integrated user experience invented to integrate several well-known colour red tools like Nmap, Feroxbuster, and SQLMap. It's possible that seasoned

electronic security professionals and those new to performing investigation ought to remain able to use these tools, according on the platform. The authors highlight the ways streamlining procedure execution, simplifying paperwork, followed by producing standardised, more easily analysed observations can each be achieved by integrating tool outputs into a single interface. They also go over how crucial it is to create papers as well as visual representations to aid in instructing and making informed choices.[6]

An approach that uses visual analytics to improve red team operations is presented in "Visual Analytics and Optimisation for Cyber Red Teaming" by Jonathan Yuen, Dr. Benjamin Turnbull, and Justin Hernandez Yuen et al. The research paper is concentrated on model-supported network recordings of an organisation, which enable the simulation along with assessment of multi-stage attacks. The dashboard enhances situational awareness and assists security teams in promptly identifying serious areas of security breaches by integrating multidimensional visualisations and security breach chain graphs. The authors point out that computerised analytics improve instructional videos for information security learners across addition to helping experts in red groups together.[7]

"Top Red Team Tools, Commercial and Open-Source in 2025," according to details complimentary commercial red teaming tools that will be accessible in 2025 are thoroughly reviewed at the following paper. With the objective to highlight the process by which integrating different tools can increase the realism and effectiveness of attack simulations, the authors look at their parameters, capabilities, and ease of use. According to the study, combining outputs from several tools into one overview can improve assessment, cut down on manual labour, and facilitate better education among both red and blue departments.[8]

Filip Dimitrov's Red Team Case Study: From Sticky Note to Full Cloud Compromise" Vladimir Dimitrov supplies an exhaustive examination of how a red team assessment went out of a small vulnerability—which is really a slippery reminder—to an exhaustive breaches of cyberspace infrastructure at hand. The study indicates the important thorough along with methodical methods for evaluation are for identifying invisible flaws. Nicholas Dimitrov emphasises where, especially in intricate online environments, the use of organised tools, thorough messages, particularly visualisation panels can enhance security comprehension, information gathering, together with management.[9]

"Key Attack Attack Vectors and Red Team Strategy" The biggest methods, methods, and processes (TTPs) use in red team operations—including such as malware collection, phishing incidents, and network attacks—are examined in the current investigation. The authors contend that creating realistic simulation dashboards

requires an awareness of common attack vectors. Security teams can better understand potential risks and create efficient response plans by utilising outputs from various tools and visualising each stage of the attack. In order increase the cybersecurity situation and support organisational management of decisions, the paper also highlights data collection and evaluation. [10]

MITRE Caldera (Adversary Emulation Framework) from Adam Pennington and Blake Strom The creation and deployment of MITRE Caldera, an artificially intelligent simulation of attackers program that synchronises modular plugins to replicate trustworthy burglary procedures followed by maps attacker tactics to the ATT&CK strategic comprehension base, are explained by Pennington and Strom. The work they perform demonstrates the manner in which employees can measure defensive competency, assess detection coverage, establish and execute repeatable red-teamed scenarios without asking for continuous manual scripting thanks to automation and standardised TTP mappings. They also stress the essential nature of plugin extensibility to continue keeping emulated applications up to date with some latest dangers as well safe, locked down implementation.[11]

Ahmed Alshamrani & Sahiti Myneni, Automated Attack Simulation in Cloud Environments Alshamrani and Myneni explore automated red-team simulations designed for cloud platforms. They propose ways to model cloud specific kill chains and simulate lateral movement across cloud services. Their research points out the unique challenges of cloud environments, such as short-lived hosts, API-based services, and permission models. They also support safe, policy-aware orchestration that avoids collateral damage while offering realistic test data for blue teams to enhance detection and response.[12]

Attack Simulation for IoT and Cyber-Physical Systems by José Manuel Fraile as well as Xabier Larrucea In internet of everything as well as online physical settings, what modelling is made difficult by device wide variation and even limited resources, Fraile together with Larrucea introduce structure for automated malicious attack mimicry. The authors contend whose services customised game design is essential for understanding the distinct defences at embedded technologies while also helping guide defenders on the physical-layer repercussions of cyber compromises, it suggest employing lightweight agent models and domain-specific scenario libraries to replicate device-level exploits and cascading effects. [13]

Adversary mimicry through Behaviour Graphs by Andrew Warfield and Peter Mattson By transforming defender decision-making processes as graph structures that encode probable strategies, acts, along with relationships, Warfield and Mattson present behavior-graphical methods to achieve competitor mimicking. The organisation emphasise that behaviour graphs enhance realism thanks to adding TTP sequences with

probabilistic branching to simulate proactive competitors; in addition, this technique empowers automated scenario manufacture offering highlights realistic, consequential paths using a network and helps analysts find choke points and detection gaps.[14]

Christopher S. Gates — Automating Offensive Security in Network Simulation With an emphasis on incorporating toolchains (scanners, exploit frameworks, and payload runners) within secure experimental spaces for exploratory purposes, Gates, the founder investigates automation methodologies to perform violent checks for safety in network simulation environments. He shows how artifacted graphing as well as scripted orchestration create repeatable workouts for helping big teams comprised and educational programs in validating defences. He additionally indicates standardised finished product formats and modular adapters to facilitate aggregation of results and analysis everywhere several different methodologies.[15]

Chapter 4

PROJECT PLAN

4.1 Methodology

The methodology for Red Team Simulation Dashboard is developed under Agile model architecture as shown in the below figure 4.1.



Fig 4.1: Agile model Architecture

The sequence phases in Agile model according to our project are mentioned below.

4.2 Brainstrom Analysis – Here requirement analysis are done based on following points

- ✓ Base paper for Red Team Simulation Dashboard
- ✓ Studying on Different Dashboard in the system
- ✓ Study of Operating System

4.5 Action Plan:



Fig 4.6 Action Plan

Summary of Key Milestones:

- June: Planning, requirement gathering, and market analysis.
- July: Core development and software architecture.
- August: Internal testing and alpha release.
- September: Beta testing and software optimization.
- October: Final testing, marketing preparation, and user documentation.
- November: Product launch and post-launch support.

This action plan ensures that the antivirus software is developed systematically, rigorously tested, and optimized for real-world use, leading to a successful launch and sustained performance in the cybersecurity market.

4.1 WORKING AND PROCESSES:

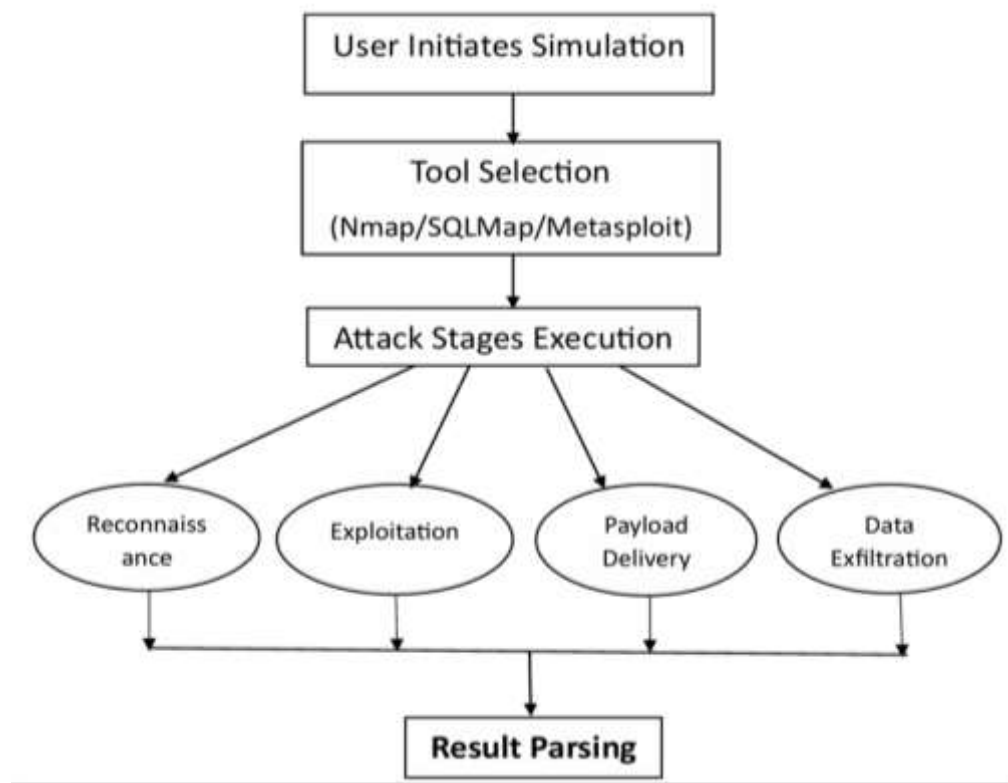


Figure 4.1 System Overview

Step 1: User Initiates Simulation

The user opens the dashboard and enters all the basic inputs such as target IP/Domain, network range, type of simulation mode (Demo Mode or Real Mode), which attack stages to perform (Reconnaissance / Exploitation / Payload Delivery / Data Exfiltration) and which tools need to be used. The user submits the request through the UI which is then forwarded to the backend API.

Step 2: Pre-Flight Verification & Authorization

The backend system checks whether the target is safe, legal and approved to attack. It also verifies whether the user has permission to run the simulation. If the target is not allowed or not verified, the system denies the simulation. If approved, the system generates a unique RUN ID to track entire testing process.

Step 3: Tool Selection & Mapping

Based on the user preferences, the orchestrator selects suitable penetration tools such as Nmap for scanning, SQLMap for injection, Metasploit for exploitation etc. The system maps each selected phase with the relevant tool. It forms an execution plan such as which tool should run first, in what sequence and under which conditions.

Step 4: Sandbox Environment Creation

Before starting the actual attack, the system creates a safe and restricted environment (container/VM) so that no real damage or data loss occurs outside the lab network. Limited permissions, limited access control and firewall rules are applied. So the attack runs in an isolated environment safely.

Step 5: Reconnaissance Execution

The first stage starts with information gathering. Tools like Nmap perform port scanning, service detection and fingerprinting of the target system. This reveals open ports, running services and possible entry points for exploitation. All raw data is streamed to dashboard in real time.

Step 6: Exploitation Attempts

Based on recon results, exploitation attacks are launched using modules of Metasploit or other exploit tools. The system tries to exploit known vulnerabilities of the target. Success or failure is recorded. If one exploit fails, system may attempt alternative exploit payloads.

Step 7: Payload Delivery & Post-Exploitation

If exploitation is successful, next step is payload delivery (like reverse shell, file drop, malware simulation etc.). The system performs post exploitation tasks like privilege escalation, credential harvesting, command execution, file browsing etc. Everything is performed within sandbox boundaries.

Step 8: Data Exfiltration (Controlled / Simulated):

The data exfiltration phase demonstrates how an attacker would remove valuable information from a compromised host while preserving safety and consent. In controlled runs the orchestrator either performs an actual, tightly scoped transfer of pre-approved test data to a designated lab sink, or simulates exfiltration by logging the intended actions and producing synthetic artifacts; in both cases the operation is confined to an isolated network path and is limited by size and rate thresholds. The

system records the transfer method (HTTP POST, SCP, DNS tunnel emulation, etc.), source and destination endpoints, byte counts, timestamps, and cryptographic hashes of any files involved, and it captures packet-snippet evidence (pcap) for forensic analysis. Crucially, pre-flight policies prevent real exfiltration against non-consented targets and outbound connections are whitelisted, ensuring no collateral leakage of sensitive production data. Demo mode replaces real artifacts with dummy files, while instructor-approved real-mode runs include explicit consent and additional logging and redaction controls.

Step 9: Real-Time Telemetry & Visualization:

Throughout the simulation the dashboard provides immediate situational awareness by streaming structured events and log snippets to the frontend via WebSockets or Socket.IO. Small, frequent event messages (for example {run_id, phase, tool, status, timestamp, short_snippet}) update the kill-chain diagram, timeline, attack graph, and terminal-like log window; larger artifacts and pcap slices are provided through deferred download links. The UI supports pause/rewind controls for instructors to annotate specific moments, quick links from highlighted events to stored evidence, and severity badges that call attention to high-impact occurrences (e.g., privilege escalation). To ensure reliability, emitted events are acknowledged and persisted server-side so the UI reflects durable state rather than ephemeral feeds; sequence numbers and retransmission logic detect and repair dropped messages to preserve ordering and completeness.

Step 10: Result Parsing & Normalization:

Raw outputs from heterogeneous tools (Nmap XML, SQLMap JSON, Metasploit console text, web crawler logs) are ingested by a modular Result Parser that tokenizes, extracts, and maps artifacts into a unified schema. The pipeline applies tool-specific extraction rules and heuristic fallbacks to yield canonical fields such as phase, tool, target, artifact_type, artifact_value, timestamp, and confidence.

Parsed artifacts are enriched with contextual metadata — mapping service banners to likely CVEs, linking commands to MITRE ATT&CK techniques, and assigning preliminary severity or confidence scores — and malformed outputs are retained for manual review. This normalization produces a consistent, queryable dataset that enables correlation across phases and supports downstream analytics, reporting, and instructor review.

Step 11: Data Storage, Correlation & Analytics

All normalized logs are stored in database. The system then correlates every stage from Recon to Exfiltration to show a full attack story line. Analytics are generated like which phase took more time, which exploit worked, which vulnerability was most critical etc.

Step 12: Report Generation & Export:

Upon run completion the system assembles comprehensive, customizable reports for different audiences: executive summaries for managers, step-by-step timelines and evidence links for instructors, and forensic packages for analysts. Standard exports include JSON (complete machine-readable run data), CSV (flattened findings for spreadsheets), PDF (formatted human-readable report with visuals), and a zipped evidence bundle (pcaps, transcripts, and hashes). Reports contain an executive summary, per-phase timelines, key artifacts and hashes, the reconstructed attack graph/kill chain visual, detection assessments, and recommended mitigations mapped to exploited vulnerabilities. Template-driven redaction and verbosity controls let instructors hide sensitive details for student copies while retaining full forensic evidence in secure archives.

Step 13: Cleanup and Audit Logging

To ensure safety, the system terminates running tools, deletes temporary files, removes shells, clears sandbox and resets environment to safe state. An audit log is stored so that future reviewers know who ran which attack, on which date and with what results.

Chapter 5

SOFTWARE REQUIREMENT SPECIFICATION

5.1 System Design: The System of Web Vulnerability Scanner system is designed by using the following hardware and software's Minimum Hardware Specification:

- ☐ Hard Disk more than 500GB
- ☐ Processor – 2.2 GHz
- ☐ Minimum Ram 4 GB and more
- ☐ Monitor, Keyboard and UPS

- **Software Specification:**

- ☐ Coding Language: Python
- ☐ Development Kit: Python 3.8.
- ☐ Front End

5.2 Development

During each sprint, the development team works on the prioritized features and continuously integrates modules into the **Red Team Simulation Dashboard**. Each increment focuses on building realistic attack simulation components and improving visualization. For example, development tasks include:

- Integration of common Red Team tools (Nmap, SQLMap, Metasploit, etc.)
- Development of Kill Chain visualization UI
- Result parsing and normalization of raw tool outputs
- Real-time logs and dashboard updating system
- Report generation and export functionalities (JSON/CSV/PDF)

Throughout the development cycle, the team follows **continuous integration practices**. Automated testing and frequent code commits ensure that newly implemented features do not introduce errors, performance issues, or security risks into the system. Each sprint ends with a working, testable increment of the dashboard,

which can be demonstrated to stakeholders (students, instructors, or cybersecurity experts) for feedback and improvement.

5.3 Quality Assurance:

In Agile development, Quality Assurance is continuous and iterative across all sprints. For the Red Team Simulation Dashboard, quality assurance plays a critical role because the system deals with real attack simulations, live event visualizations, and integration of multiple offensive security tools. Every feature must be tested thoroughly to ensure the dashboard behaves safely, accurately, and consistently in different environments (local labs, virtual machines, isolated networks).

The major testing types include:

- **Unit Testing:**

Validate individual components such as parser modules, tool integration adapters, and visualization elements to ensure each one performs correctly.

- **Security Testing:**

Ensure the dashboard itself cannot be exploited—checking API security, data access control (RBAC), sandbox escape prevention, and secure handling of logs and artifacts.

- **Performance Testing:**

Evaluate how the system responds when multiple attack stages run simultaneously, when high-volume logs are generated, and when multi-user simulations occur at the same time.

- **Real-World Testing:**

Execute controlled attack scenarios using real tools (e.g., Nmap, SQLMap, Metasploit) to verify the dashboard displays accurate kill-chain progression, event timing, risk severity, and final reports.

5.4 Deployment:

At the end of each sprint, the Red Team Simulation Dashboard will be delivered to users (students, trainers, or other testers) as an incremental feature update or sprint release. Each deployment focuses on controlled, safe, and progressive improvement of core offensive simulation capabilities.

This deployment may include:

- **New attack modules integration**

e.g. adding tools like Dirsearch, Hydra, or payload plugins.

- **Updated visualization components**

e.g. improved kill-chain stage graphics, timeline charts, or stage-wise progress bars.

- **Bug fixes and stability improvements**

from issues identified during sprint testing.

- **UI/UX enhancements**

to make dashboards more user-friendly and easier to interpret during live simulations.

Deployment will follow secure release procedures to ensure that every new feature is stable, safe to execute, and consistent with the controlled environment guidelines used for red team simulation.

Chapter 7

SOFTWARE TESTING

7.1 TEST CASES

7.1.1 Performance Testing

Performance testing for the Red Team Simulation Dashboard evaluates how the platform behaves under normal and peak simulation loads while maintaining responsiveness and real-time visualization. Key areas include: orchestration latency (time from user action to tool invocation), telemetry throughput (events/sec delivered to the frontend), UI render time for large logs/pcap previews, database write/read latency for parsed events, and resource consumption of sandboxed executions (CPU, memory, network). Test scenarios should include single-run, concurrent multi-run (multiple users starting simultaneous simulations), and long-running campaign tests. Acceptance criteria include bounded UI update latency (e.g., <500ms for small events), stable resource usage without memory leaks, and graceful degradation (e.g., rate-limiting) when throughput limits are reached.

7.1.2 System Testing

System testing validates end-to-end functionality across all modules: frontend, orchestration engine, tool adapters, sandbox environment, parser pipeline, storage, and reporting. Test cases should cover typical workflows (start run → stages execute → results parsed → report generated), edge cases (missing tool binaries, malformed tool output), permission and RBAC flows (student vs instructor vs admin), and integration points (SIEM export, webhook notifications). System tests also include compatibility checks across supported platforms (lab VMs, container runtimes) and browser testing for the UI. The outcome must confirm that the system meets functional requirements and that cross-component communication works reliably under expected operational conditions.

7.1.3 Recovery & Fault Tolerance

Recovery testing ensures the dashboard can recover from failures during simulations—e.g., tool crashes, sandbox VM failures, network interruptions, or partial parser errors—without losing critical evidence or leaving persistent, unsafe side effects. Tests should simulate abrupt process termination, network partitioning between orchestrator and agent nodes, and disk-space exhaustion. The system must demonstrate automatic rollback or cleanup (terminate stray processes, revoke temp credentials), safe state persistence (partial results preserved with proper flags), and restart behavior that allows either safe resume or clean re-run. Audit logs must capture failure causes and recovery actions to support forensic review.

7.1.4 Security Testing

Security testing validates that the platform itself resists misuse and compromise. This includes authentication and authorization checks (RBAC enforcement), input validation and injection testing on API endpoints, secure handling and storage of artifacts (encryption at rest, access controls), and sandbox escape attempts. Penetration testing exercises should attempt to abuse orchestration inputs to run unauthorized commands, access other tenants' artifacts, or exfiltrate data beyond lab sinks. The dashboard must also be tested against common web vulnerabilities (OWASP Top 10) and demonstrate safe logging (no secrets in plaintext), strong session management, and proper CORS/CSP policies.

7.1.5 Stress & Scalability Testing

Stress tests push the system beyond normal operating points to identify breaking points and failure modes. Scenarios include high-frequency event bursts (many tools producing logs simultaneously), mass-concurrent runs by many users, and large-volume artifact ingestion (many pcaps/files). Measure queue backlogs, parser throughput, indexing lag, and UI responsiveness under stress. The system should implement backpressure and graceful degradation (e.g., persist heavy artifacts to cold storage and stream lightweight summaries to UI) so that critical control paths remain available, and operators can scale components (add parser workers, scale DB indices) to recover.

7.1.6 Unit & Integration Testing

Unit tests target small modules such as tool adapters, parsing rules, JSON schema validators, and utility functions to ensure deterministic behavior. Integration tests validate combined behavior of subsystems: orchestrator ↔ sandbox agent, parser ↔ enrichment modules, and API ↔ frontend components. Mocking external tools (simulate Nmap/SQLMap outputs) helps validate parsing and visualization logic without running real exploits. Test coverage should include success/failure paths, edge inputs, and regression checks for parser rules when new tools are added.

7.1.7 Black-Box & Acceptance Testing

Black-box tests treat the dashboard as a closed system and verify outputs solely based on inputs and observable behavior. Example cases: feed a simulated scan scenario (predefined mock outputs) and verify that the dashboard visualizes the correct kill-chain states, generates accurate timeline and metrics, and produces expected report files. Acceptance testing should involve representative end users (students, instructors, SOC analysts) to validate usability, clarity of visualizations, and adequacy of generated reports for training or operational use.

7.2 APPLICATIONS

- **Red/Blue Team Training & Education** — Use the dashboard for classroom labs, capture-the-flag practice, and guided red-team exercises where students observe end-to-end attack progression and learn remediation steps.
- **Penetration Testing Support** — Aggregate tool outputs into a single report and visualization that help professional pen-testers document attack chains, justify findings, and prepare remediation roadmaps.
- **SOC Readiness & Detection Validation** — Run controlled emulations to measure whether detection stacks (SIEM/EDR) raise appropriate alerts and to compute detection metrics (simulated MTTD, coverage).
- **Breach & Attack Simulation (BAS) for Small Orgs** — Provide lightweight adversary emulation capabilities for organizations that cannot afford enterprise BAS platforms but want repeatable, auditable simulations.
- **Incident Rehearsal & Tabletop Exercises** — Generate reproducible scenarios to run debriefs and post-incident reviews; provide evidence packages for lessons learned and policy updates.
- **Research & Tool Development** — Serve as a modular experimentation platform for academics and developers building novel attack emulation techniques, behavior graphs, or parsing algorithms.
- **Compliance Demonstrations & Audit Evidence** — Produce standardized, exportable reports and audit trails mapping simulated TTPs to frameworks (e.g., MITRE ATT&CK, NIST) for governance and compliance exercises.

Testing Module: Input Files				Test by: Rucha Bhalerao	
Test Cases ID: 001				Test Date : 10/3/2025	
Test Priority: 1				Test Execution by: Akshada Gundal	
Description: Input file module perform selection of the file scanning and detection Vulnerabilities				Test Date : 1/4/2025	
Step	Test Steps	Test Data	Expected Result	Actual Result	Status
1.1	Tap on “Open the application”	Tap on “Scan File”	Open the window of browse file	Browse file open	Pass
1.2	Select Website		Error Should Find	Error should Successfully found and fixed	Pass
1.3	Press back		Should go to the home page again	Back to initial stage	Pass
2.1	Open settings	Scan Depth	Unable Meta Defender		Pass
2.2	Open Settings	Scan type	Virus code should given	Virus code displayed	Pass
3.1	Tap on “Start Scanning”	Live Output			Pass
3.2	Scanning info		Form should display in Vulnerabilities	Form display in Vulnerabilities	Pass
4.1	Report		Report generated properly or not	Generated	Pass
4.2	Recomendation				Pass

Chapter 8

CONCLUSION AND FUTURE WORK

8.1 Conclusion

The proposed Red-Team Attack Simulation Dashboard delivers a modern, automated, and centralized approach for visualizing cyber attack chains in a structured, easy-to-understand manner. By integrating multiple penetration testing tools such as Nmap, SQLMap, Metasploit etc. into a single unified panel, the system enables students, beginners, and professionals to clearly observe how an attacker progresses through reconnaissance, exploitation, privilege escalation, and data exfiltration. Real-time visualization, logs, and stage-wise representation allow learners to connect every attack phase logically instead of seeing raw outputs separately. This makes learning faster and decision-making smarter. The dashboard supports easy report generation, safe simulation environment, and controlled execution of attacks, thus reducing risks and making red teaming more systematic and accessible.

Future Work

Future versions of the Red-Team Simulation Dashboard can be enhanced using AI-driven prediction models to suggest the most likely attack path and automatically map attacks to MITRE ATT&CK techniques. The dashboard can be extended to support Cloud platforms (AWS / Azure / GCP), IoT devices, and hybrid networks for real-industry simulation usage. Further improvements include integration with SIEM / SOAR tools so that the dashboard can also measure detection capability of Blue Teams. Additional future work involves adding customizable kill-chain builders, API-based attack module installation, and ML-powered risk scoring to prioritize dangerous actions. The system may also support multi-user mode with role-based access (Instructor, Student, Analyst) to allow classrooms, SOC teams, and cyber ranges to use it simultaneously.

8.3 Final Thoughts

The Red-Team Attack Simulation Dashboard represents more than a tool; it is a pedagogical and operational bridge between theoretical knowledge and hands-on practice. By converting fragmented, technical outputs from multiple offensive tools into a coherent, stage-by-stage narrative, the dashboard reduces cognitive load for learners and accelerates comprehension. Students progress from passive observers to active experimenters:

they can run controlled scenarios, observe cause-and-effect relationships in real time, and immediately correlate a specific exploit to its downstream impacts. For instructors, this means less time spent explaining low-level logs and more time on higher-order concepts such as attacker intent, defense-in-depth tradeoffs, and mitigation strategies.

Beyond the classroom, the dashboard has practical value for organizations seeking to validate controls and improve security posture. Security teams can use the platform to rehearse attack scenarios, identify detection gaps, and generate audit-ready evidence that informs remediation prioritization. The unified reporting and traceability features make it easier to translate technical findings into management-level recommendations, helping stakeholders understand risk and authorize corrective investments. Moreover, because the system supports both demo and controlled real-mode operations, it can serve diverse needs—from academic labs and capture-the-flag events to SOC drills and pre-deployment security rehearsals.

Looking forward, the dashboard's greatest potential lies in its extensibility and role in a broader security ecosystem. As threat intelligence, AI-driven scenario generation, and automated defensive orchestration mature, the dashboard can act as a central hub that both emulates attackers and measures defenders—closing the loop between offense and defense. By fostering reproducible scenarios, collaborative debriefs, and sharable lesson artifacts, it can help build institutional knowledge across teams and cohorts. Ultimately, this platform contributes not only to individual skill development but also to organizational resilience: better trained personnel, documented exercises, and measurable detection improvements that together reduce the likelihood and impact of real incidents.

Chapter 9 References

- [1] R. S. M. G. de Lima et al., "Red Team Cutting Edge: Centralised UI for Red Team Tools" — introduces RTK, all of which combines Feroxbuster, Nmap, SQLMap, and Nmap into an individual desktop application to streamline Red Team execution for novices as well as experienced users.
- [2] "Visual Assessments for Cyber Red Teaming" by J. Hernandez, B. Turnbull, and J. Yuen — highlights an automated red team technology that creates visual evidence of infrastructure while participating multi-stage assault scenarios.
- [3] CyberNX's article, "Top Red Team Tools, Commercial and Open-Source in 2025," examines and talks about the features, benefits, drawbacks, along with capabilities of several red-teaming applications the fact that will be widely accessible in 2025.

- [4] "A Case Presentation of the Red Team: From Sticky Note to Full Cloud Compromise" by Filip Dimitrov —reveals the need for more thorough research by showing the means by which a minor earliest vulnerability transformed into a full cloud hacking attempt.
- [5] The article "Unique Attack Vectors and Red Team Operations" focusses on typical TTPs that the red teams are using to comprehend malicious user actions, including phishing, infection with spyware uploading, particularly cyberspace breaching.
- [6] D. Caivano et al., "Red Team Knife (RTK): Unified Offensive Tool Interface" — Describes how RTK streamlines instruction, storytelling, and execution by combining the various offensive features into a single user experience.
- [7] J. Yuen et al., "Visual Analytics and Optimisation for Cyber Red Teaming" — Talks concerning how to improve situational awareness of red working teams by using dashboards with graphics and graph-first attack attack chain perspectives.
- [9] Vladimir Dimitrov, "Red Team Case Study: Sticky Note to Cloud Collapse" — Shows how simple flaws which have been missed might result in a full-scale system breach in intricate environment.
- [10] "Red Team Strategy and Key Attack Vectors" — focusses on establishing the main strike routes along with describes how hostile visualisation aids with reaction strategy planning for corporate entities.
- [11] Adam Pennington while Blake Strom, "MITRE Caldera Framework" — Presents an automated attacker reproduction computing system that runs compatible red team scenarios by employing ATT&CK mappings to execute.
- [12] "Automated Attack Simulation in Cloud Computing Environments" by Ahmed Alshamrani and Samantha Myneni — identifies special dangers in API-based and dynamic-host architectures and suggests red team simulators tailored to the context of cloud computing.
- [13] "Attack Simulation for IoT and CPS" by José Manuel Fraile and Xabier Larrucea — presents simple IoT attack simulation environments for researching individual device security risks alongside.
- [14] Andrew Warfield and Peter Mattson, "Adversary Emulation through Behaviour Graphs" — This paper simulates complex perpetrator behavioural paths with probabilistic evolving using behavioural graph mathematical techniques.

[15] Christopher S. Gates, "Automating Offensive Security in Network Simulation" — discusses intelligent wiring of fraudulent tools as well as detection devices surrounding unsafe lab settings for demonstrations that can be repeatedly performed.