530 PROJECT FINAL REFLECTION

Interactive Network Incident Response Simulator: A SOC-Style Training Tool for DDoS, Data Breach, and Ransomware Scenarios
Md Shafiqul Islam
Fall-2025

PROJECT DESCRIPTION

This project builds a hands-on, web-based simulator that allows learners to step into the role of a Security Operations Center (SOC) analyst. All activity within the simulator is fully synthetic and safe. Therefore, no real attacks are executed yet the alerts, log excerpts, and visualized system metrics are realistic enough to let students practice detecting incidents, prioritizing response actions, and guiding recovery in a controlled environment.

The simulator implements three core incident scenarios that reflect common and high-impact network security events:

- DDoS (volumetric network and transport attacks)[3]: The dashboard displays elevated inbound TCP and UDP traffic rates, rising CPU and memory utilization on the target server, and service degradation. These signals allow learners to practice identifying availability issues and selecting appropriate mitigation strategies.

- Data breach (credential misuse and suspicious exfiltration): Learners observe repeated authentication failures, logins from unusual geographic locations, abnormal access patterns, and unexpected outbound data transfers. These indicators mirror real-world signs of compromised credentials or data theft.

- Ransomware-style infection (lateral movement and file-access issues): The simulation surfaces anomalous SMB write activity, unexpected process execution, and sudden file-access errors, enabling learners to work through containment, eradication, and recovery decisions.

Each scenario unfolds as a structured sequence of synthetic events, including alerts, logs, and metric updates. At defined decision points, learners choose from a constrained set of realistic response actions, such as blocking an IP address, isolating a host, rotating credentials, or restoring from backup. The system evaluates these decisions using structured scoring rubric, provides immediate feedback, and generates concise, NIST-aligned[1] post-incident recommendations so learners can understand what actions were effective, which were risky, and why.

In addition to the interactive simulations, the project includes an integrated Learning Center that provides background instruction on TCP/IP behavior, common attack patterns, and response rationale. This instructional layer supports learners in understanding what they are observing during the simulations and why certain actions score higher than others.

Overall, this project is designed as an educational visualization and training tool. The emphasis is on learning, practicing incident-response[2] decision-making, and receiving timely, explanatory feedback, rather than on teaching offensive techniques or reproducing real attacks.

Target Audience

The Interactive Network Incident Response Simulator is designed primarily for undergraduate students studying networking or cybersecurity, as well as early-career IT and security professionals who need hands-on incident-response[2] practice. The simulator is intentionally approachable for learners with basic networking knowledge, while still offering sufficient depth to support skill development in realistic incident-response workflows.

In addition to individual learners, the project is suitable for use by instructors, course designers, and academic or outreach programs that focus on cybersecurity education. The scenario-based structure, scoring system, and feedback make it well suited for classroom demonstrations, guided labs, and assessment-based exercises. Educators can reuse and adapt the scenarios as interactive teaching materials to reinforce core network security concepts and incident-response best practices.

Audience Skill Level

The intended audience is expected to have a foundation in networking and operating systems. Learners should be familiar with core concepts such as IP addressing, ports, and the general differences between TCP and UDP, along with an understanding of common operating system concepts like services, processes, and basic system behavior. No prior experience with offensive security tools, advanced forensics, or real incident-response operations is required.

Learning objectives

Objective 1 – Learners will be able to reliably spot and interpret common incident indicators — for example, recognizing volumetric spikes that suggest a DDoS, spotting repeated failed logins or unusual outbound transfers that point to credential misuse, and identifying SMB write anomalies that suggest ransomware activity.

Objective 2 – Learners will be able to apply a NIST-aligned[1] incident-response process (Preparation → Detection & Analysis → Containment → Eradication → Recovery → post-incident) to choose appropriate containment and remediation steps during each scenario, as measured by their simulation scores and decision logs (target: consistently make the correct sequence of containment/remediation choices across scenarios).

Objective 3 – An additional objective is that learners can recommend practical mitigations and policy actions — such as network segmentation, multi-factor authentication, and a tested backup/restore process — and justify those recommendations.

Project Impact

The Interactive Network Incident Response Simulator contributes meaningful value to network security education by addressing a common gap between theoretical instruction and practical incident-response decision-making. Many learners understand attack concepts, protocols, and defenses in isolation but struggle to apply that knowledge when faced with evolving incidents, incomplete information, and competing priorities. This project responds to that challenge by offering a realistic yet safe environment where users can practice interpreting network signals, select response actions, and understand the consequences of those actions within a structured incident-response framework.

For the intended audience of undergraduate students and early-career IT or security professionals, the simulator provides hands-on exposure to incident types that are both common and impactful in real organizations, including DDoS attacks, credential misuse, and ransomware infections. By presenting incidents as timelines composed of logs, alerts, and system metrics, the project helps learners develop the ability to connect TCP/IP behavior and application-level events to operational security decisions. The use of NIST SP 800-61[1] as the organizing framework further strengthens its relevance by reinforcing industry-recognized response practices rather than ad hoc reactions.

The project is carefully tailored to the skill level of its audience. It assumes only foundational networking knowledge and avoids advanced offensive techniques or deep forensic analysis that could overwhelm learners. Instead, it emphasizes clear indicators, realistic decision points, and explanatory feedback. The integrated Learning Center supports this approach by providing background instruction on attack patterns, indicators, and response rationale, allowing learners to build understanding before and during simulation use. This structure helps fill an existing educational gap by gradually building understanding in a way that supports both comprehension and application.

Beyond individual learners, the project offers broader usefulness to instructors and the network security education community. Its data-driven design allows scenarios to be modified or extended without changing core application logic, making it suitable for classroom demonstrations, lab exercises, and assessment activities. Overall, the simulator demonstrates how interactive, decision-focused training tools can enhance network security education by moving beyond static examples and enabling learners to develop practical, transferable incident-response skills.

EXPECTED OUTCOMES, EVALUATION, AND ASSESSMENT

Tangible deliverables:

- A fully functional local web application (Flask + frontend) that runs three realistic cyber incident scenarios. Each scenario includes decision points, scoring, and feedback to help users learn through simulation.
- A scenario data package (JSON/CSV) containing all scenario definitions, sample logs, and optional pre-generated pcap visualizations for instructors to use or customize.
- Comprehensive documentation, including:
    - A user guide explaining how students can navigate and interact with the scenarios.
    - An instructor guided detailing how to run, modify, and assess scenarios.
    - A README describing code setup, dependencies, and architecture.
    - A final project report summarizing the research background, implementation methods, and learning outcomes.
- A short demo showing the main user interactions and decision flows for each scenario.

Evaluation and assessment criteria:

- Technical accuracy: Simulated network indicators and log data correctly reflect real TCP/IP [5]and application-layer behavior.
- Learning effectiveness: Scenarios clearly communicate learning goals, provide timely and meaningful feedback, and show measurable improvement in user performance over multiple attempts.
- Usability: The interface is intuitive, easy to navigate, and the setup process can be reproduced without issues.
- Documentation quality: Guides and code comments are clear, complete, and helpful for both learners and instructors.

Performance metrics:

- Scenario detection accuracy: Percentage of indicators correctly identified by pilot users.
- Average user score: Mean score achieved across all scenarios.
- Response efficiency: Difference between the user's time-to-containment and the ideal benchmark defined in the rubric.

PROJECT EVALUATION AND OUTCOMES

Additions Made Since the Last Submission:

Since the intermediate submission, the project was expanded beyond the original scenario, decision, and score workflow to improve instructional clarity, usability, and overall learning effectiveness. The most significant addition was the introduction of a dedicated Learning Center, which provides structured background knowledge before learners begin simulations. This module covers TCP IP[4] concepts and log sources, definitions of the three incident types, common attack progression patterns, key indicators, response playbooks, and relevant real-world examples. The Learning Center directly addresses the need for stronger instructional context and helps learners better understand what they observe during simulations.

Another important addition was the inclusion of a scenario background and help feature within the simulation screen. This allows learners to access scenario specific context without leaving the workflow, supporting quick recall of incident behavior and best practices while decisions are being made. Educational feedback was also significantly enhanced. Feedback now explains why a selected option is effective or risky, rather than only indicating correctness. A Detailed Analysis view was added to compare all response options side by side, explain scoring tiers, and highlight tradeoffs between choices. This encourages reflective learning rather than answer memorization.

Several user interface and usability improvements were also implemented to support smoother learning flow. These included clearer severity indicators, improved modal behavior for readability, and more intuitive navigation between screens. Together, these changes transformed the project from a basic simulation tool into a more complete instructional training platform.

Expected Outcomes and Learning Objectives for the Target Audience:

The simulator is designed for cybersecurity students and early career SOC trainees. Based on the project proposal and intermediate objectives, the expected learning outcomes include the ability to recognize common incident indicators, apply NIST aligned[1] incident response reasoning, make informed decisions under pressure, and learn through meaningful feedback rather than trial and error.

Specifically, learners are expected to identify signals associated with DDoS attacks, credential misuse, data exfiltration, and ransomware behavior using logs and system metrics. They should also be able to reason through response actions using the NIST SP 800 61 lifecycle and understand why certain actions occur in specific sequences. Finally, learners should develop critical thinking skills by weighing tradeoffs and improving decision quality across repeated attempts with the help of explanatory feedback.

Assessment of Whether Intended Outcomes Were Met

Overall, the project meets most of its intended outcomes and shows clear improvement since the intermediate submission. Indicator recognition, NIST aligned[1] reasoning, and instructional feedback quality were met strongly. Learners are consistently exposed to realistic signals and guided toward appropriate responses through structured scenarios and explanations.

Assessment depth and measurement rigor were met partially. While the scoring rubric functions correctly and supports meaningful evaluation within scenarios, the system does not yet measure long term learning gains or aggregate performance across users. Persistent learner tracking, instructor analytics, and exportable assessment artifacts were not fully implemented and are identified as areas for future development.

Metrics and Indicators Demonstrating Achievement of Objectives

Several observable indicators within the current system demonstrate how learning objectives were achieved. For indicator recognition, the simulator presents chronological events logs with timestamps, severity, and source information, along with live system metrics such as CPU usage, memory consumption, and network traffic. Learners must interpret evolving patterns rather than isolated snapshots, reinforcing real world analysis skills.

For NIST aligned reasoning, scenario progression follows investigation, containment, eradication, and recovery phases. End of scenario recommendations are grouped by NIST lifecycle stages, reinforcing structured response thinking. The Learning Center and scenario background content further reinforce this structure before and during simulations.

Decision making under pressure is supported through decision points that present multiple plausible actions, including options that appear tempting but introduce operational risk. Scoring tiers reward actions that balance speed, correctness, and safety. The Detailed Analysis view strengthens reasoning by comparing options and explaining why some choices are stronger than others.

Instructional feedback is delivered through immediate explanations following each decision and through option comparison views that allow learners to understand higher scoring alternatives. This supports iterative improvement and deeper understanding.

Outcomes and Objectives Not Fully Met

Some outcomes were not fully met due to scope and time constraints. Persistent learner progress tracking was not implemented because session data is stored in memory and resets when the server restarts. Instructor dashboards and analytics were also not implemented, as they require persistent storage, reporting views, and additional metrics collection. Formal learning evaluation studies were only partially addressed. While the simulator provides strong instructional support, it does not include built-in pre and post testing or longitudinal measurement needed to formally quantify learning gains.

Strengths and Weaknesses Analysis

Strengths
The project’s primary strength lies in its instructional design and realism. The Learning Center, scenario background help, and detailed option analysis strongly support learning transfer. The simulation closely reflects real SOC workflows through the use of logs, metrics, and evolving incidents. Alignment with NIST SP 800 61 reinforces professional incident response practices and terminology. The JSON driven scenario design allows easy modification and future expansion, and the feedback system enables repetition-based learning through clear explanations.

Weaknesses
The main weaknesses relate to scalability and assessment depth. The lack of persistent storage limits long term tracking and classroom scale deployment. Assessment is currently limited to scenario scoring and is not connected to broader competency measurement or instructor reporting. Scenario realism also depends on continued refinement of authored content, and validation has relied mostly on manual testing rather than automated test coverage.

Personal Impact

Working on the Interactive Network Incident Response Simulator had a meaningful impact on how I understand both network security and cybersecurity education. Before starting this project, my understanding of incident response was largely conceptual, based on reading standards, learning attack categories, and studying defensive controls in isolation. Building this simulator required me to think more deeply about how incidents unfold over time and how security analysts make decisions with incomplete information, limited time, and competing priorities.

The project pushed me to move beyond simply identifying attacks and instead focus on the reasoning behind response actions. Designing realistic decision points forced me to consider operational tradeoffs, such as balancing containment speed with service availability or preserving evidence while limiting damage. This shift helped me better appreciate why incident response is as much about judgment and communication as it is about technical knowledge. Aligning the scenarios with the NIST SP 800 61 lifecycle also strengthened my understanding of structured response processes and their importance in real organizational environments.

From a development perspective, the project influenced how I approach building educational tools. Adding features like the Learning Center and detailed feedback made me realize that effective learning tools must guide users before, during, and after interaction, not just test them. I became more aware of how design choices, interface clarity, and explanation quality directly affect learning outcomes. Overall, this project reinforced my interest in applied network security and showed me how technical systems can be designed to support learning, reflection, and skill development in a realistic and responsible way.

Personal Learning Goals

Through this project, I aim to strengthen both my technical and educational design skills. Specifically, I want to:

- Build a deeper, hands-on understanding of how network behaviors in the TCP/IP[4] stack appear in real-world incident logs—for example, how patterns in metrics reveal transport-layer floods or abnormal connections.
- Learn how to design interactive, educational simulations that turn raw technical data into clear, teachable moments.
- Enhance my full-stack development abilities by building a robust Flask-based backend and an engaging, responsive frontend, while creating teaching resources that instructors can easily reuse.
- Develop clear, structured documentation and assessment rubrics to make the tool ready for classroom use and adaptable for CyIO or other cybersecurity outreach initiatives.

Skills and Knowledge Gained

This project significantly deepened my understanding of network security by moving my learning from theoretical knowledge to applied reasoning. Through designing and implementing the Interactive Network Incident Response Simulator, I developed practical skills in recognizing how network, application, and host-level indicators appear during real incidents. Creating synthetic logs, system metrics, and event timelines strengthened my ability to interpret traffic patterns, authentication behavior, and file activity in the context of DDoS attacks, data breaches, and ransomware infections. This process helped me better understand how abstract networking concepts translate into observable signals used during incident response.

From a technical perspective, I gained experience building an interactive security training system that models real workflows. I learned how to manage session state, evaluate user decisions, and implement structured scoring and feedback mechanisms that remain consistent across multiple scenarios. Working with JSON-based scenario definitions improved my understanding of data-driven design and reinforced the importance of separating application logic from instructional content. These skills are directly applicable to building maintainable security tools and simulations.

The project also strengthened my understanding of incident response frameworks. Aligning scenarios and feedback with the NIST SP 800 61 lifecycle helped me internalize the reasoning behind response sequencing, including why detection, containment, eradication, and recovery must occur in a structured order. Designing decision points forced me to consider tradeoffs related to timing, operational impact, and evidence preservation, which broadened my perspective beyond purely technical fixes.

My personal learning goals are focused on improving practical incident-response reasoning and understanding how to design effective security training tools. These goals were largely achieved through the iterative development process and the addition of features such as the Learning Center and detailed feedback explanations. One area where learning goals were only partially met relates to formal assessment and analytics. While I gained insight into how scoring and feedback can support learning, I did not fully implement persistent tracking or aggregated performance analysis, which limited deeper exploration of learner progress over time. This limitation was primarily due to scope and time constraints rather than a lack of conceptual understanding.

Overall, this project expanded both my technical expertise and my ability to think critically about how network security knowledge is taught and applied. It strengthened my confidence in analyzing incidents, designing response workflows, and building tools that support learning through realistic, decision-focused practice.

Lessons Learned

One of the most important lessons learned during this project was the value of focusing on clarity over complexity. Early versions of the simulator attempted to include too many indicators, logs, and signals at once, which made scenarios harder to follow and reduced their instructional value. Through iteration and testing, I learned that effective incident-response training depends on highlighting the most meaningful signals and guiding learners toward interpreting patterns rather than overwhelming them with data. This reinforced the idea that realism must be balanced with usability in educational tools.

Another key lesson involved the importance of structure and sequencing in both system design and incident-response workflows. Aligning scenarios with the NIST SP 800 61 lifecycle helped ensure that response actions followed a logical progression, and it also revealed how easily learners can make mistakes when steps are taken out of order. Designing feedback around these sequencing errors taught me how critical it is to explain not just what went wrong, but why timing and order matter in real incidents.

From a development standpoint, the project highlighted the need for clean separation between application logic and educational content. Using JSON-driven scenarios made it easier to refine and extend the simulator, and it showed how flexible design choices early on can reduce rework later. I also learned the importance of frequent, small-scale testing. Validating event flow, scoring consistency, and interface behavior at each stage prevented larger issues from accumulating as the project grew.

Finally, the project reinforced the importance of being willing to revise or remove ideas that are technically interesting but do not improve learning outcomes. Features like the Learning Center and detailed option analysis emerged from recognizing gaps in user understanding rather than from initial design plans. These experiences will inform my future work by encouraging a more learner-focused mindset, careful scope management, and a stronger emphasis on iterative refinement when building security-focused educational systems.

Future Impact

The Interactive Network Incident Response Simulator has clear potential to grow beyond its current scope and provides continued value for network security education and training. One immediate area for future impact is expanding the simulator to support persistent learner tracking and instructor level analytics. Adding database backed session storage would allow long term progress tracking, comparison across attempts, and the generation of reports that instructors could use for assessment and feedback. This would make the tool more suitable for classroom deployment and structured lab evaluation.

Another area for future development is expanding scenario variety and realism. Additional scenarios could be introduced to cover incidents such as insider threats, misconfiguration driven breaches, or cloud specific failures. Existing scenarios could also be extended with greater variability, such as alternative attack paths or randomized indicators, to reduce predictability and better reflect real world uncertainty. Because the simulator is driven by JSON based scenario definitions, these extensions can be implemented without changing the core application logic.

The project could also be enhanced through deeper integration with established security frameworks and taxonomies. Mapping scenario elements to frameworks such as MITRE ATT&CK or adding explicit references to compliance and policy considerations would strengthen its relevance for professional training. In addition, support for multiuser or team-based simulations could allow learners to practice collaboration and communication, which are critical skills in real SOC environments.

From a broader perspective, this project demonstrates how interactive, decision focused simulation can complement traditional network security education. With continued refinement, it could serve as a reusable teaching tool for undergraduate courses, cybersecurity workshops, or self-guided learning. The design approach and lessons learned from this project can also inform future work on building educational systems that emphasize reasoning, tradeoffs, and structured response thinking rather than rote memorization.

Explanation of Submitted Materials

This section provides an organized overview of all components included in the final project submission, making it easy for the instructor and TA(s) to access, review, and assess each element individually. The submission includes a comprehensive final project report, a recorded demonstration video, complete and well-documented source code, scenario data files, and supporting documentation. All files are clearly labeled, organized, and accessible either through Canvas (primary grading location) or the project's public GitHub repository.

All programming code included in this submission is fully documented with in-line comments explaining functionality and design decisions. Detailed installation and execution instructions, along with a complete list of required dependencies and external libraries, are provided in the project README. No personal login credentials, API keys, or external services are required to access or run the project

PROJECT CONTENT DELIVERABLES

Item 1: Final Project Report

- Item Description:
  Comprehensive final project report.
- File name: CPRE-5300_FINAL_PROJECT_REPORT.PDF
  Uploaded directly to Canvas.

GitHub :
https://github.com/shafiqul92/soc-incident-response-simulator/blob/main/CPRE-5300_FINAL_PROJECT_REPORT.PDF

Item 2: Demonstration Video

- Item Description:
  Final project demonstration video providing a complete walkthrough of the *Network Incident Response Simulator*. The video demonstrates scenario selection, Learning Center navigation, incident event progression, decision-making and scoring, detailed feedback analysis, and NIST SP 800-61 aligned recommendations.
- File name: CPRE-5300_PROJECT_DEMO.mp4
  Canvas Studio: Uploaded to Canvas Studio.

GitHub (backup copy):
https://github.com/shafiqul92/soc-incident-response-simulator/blob/main/CPRE-5300_PROJECT_DEMO.mp4

Item 3: Project README

- Item Description:
  Project README file containing installation instructions, dependency requirements, execution steps, usage guidance, project structure explanation, and troubleshooting information.
- File name: README.txt (uploaded to canvas)

GitHub:
https://github.com/shafiqul92/soc-incident-response-simulator/blob/main/README.md

Relevant information:
The README provides step-by-step instructions for running the application locally, including dependency installation and server startup. No login credentials or API keys are required.

Item 4: Source Code (Text Version – Canvas Submission)

- Item Description:
  Plain-text versions of the complete source code submitted via Canvas for grading and inspection. These files contain the full backend and frontend implementation of the *Network Incident Response Simulator* in .txt format, allowing reviewers to examine the code without running the application.
- File name/path/link:
  Canvas submission (text files):
  - app.txt – Backend Flask application
  - index-3.txt – Frontend HTML template
  - app_js-3.txt – Frontend JavaScript logic
  - style_css-3.txt – Frontend CSS styling

Item 5: Project Summary Document

- Item Description:
  Project summary document providing a structured overview of the *Network Incident Response Simulator*. This document summarizes the project motivation, objectives, system architecture, core features, scenario design, learning outcomes, and overall significance of the simulator as a SOC-style educational tool.

- File name: PROJECT_SUMMARY.md
  GitHub:
  https://github.com/shafiqul92/soc-incident-response-simulator/blob/main/PROJECT_SUMMARY.md

- Relevant information:
  The document is written in Markdown format and can be viewed directly in a web browser on GitHub without downloading. It serves as a high-level companion to the full final project report and helps reviewers quickly understand the scope and design of the project.

SOURCE CODE AND READMEs

Item 1: Project README (Installation and Usage Documentation)

- Item Description:
  Comprehensive README file providing step-by-step installation instructions, execution commands, usage guidance, project structure overview, dependency listing, and troubleshooting information. This file serves as the primary entry point for setting up and running the project.
- File name/path/link:
  https://github.com/shafiqul92/soc-incident-response-simulator/blob/main/README.md

- Relevant information:
  - Lists of all required software and libraries
  - Provides commands to install dependencies (pip install -r requirements.txt)
  - Explain how to start the Flask server and access the application (http://localhost:5050)
  - No login credentials required
  -

Item 2: Backend Source Code (Flask Application)

- Item Description:
  Core backend source code implementing the Flask web server, REST API endpoints, session management, scenario loading from JSON files, scoring logic, and NIST SP 800-61 aligned recommendation generation.
- File name/path/link:
  https://github.com/shafiqul92/soc-incident-response-simulator/blob/main/app.py

- Relevant information:
  - Fully commented with in-line explanations for major functions and logic blocks
  - Executed locally using Python (python app.py)
  - Runs on port 5050 by default

Item 3: Frontend Source Code (HTML, JavaScript, CSS)

- Item Description:
  Complete frontend implementation consisting of HTML templates, JavaScript logic, and CSS styling that

together provide the user interface, user interactions, data visualization, and responsive design of the simulator.

- **File name/path/link:**
    - HTML templates: https://github.com/shafiqul92/soc-incident-response-simulator/tree/main/templates
    - JavaScript logic: https://github.com/shafiqul92/soc-incident-response-simulator/tree/main/static/js
    - CSS stylesheets: https://github.com/shafiqul92/soc-incident-response-simulator/tree/main/static/css
- **Relevant information:**
    - HTML rendered using Flask's Jinja2 templating engine
    - JavaScript handles API communication, scenario progression, decision submission, and Chart.js integration
    - CSS provides responsive layout, animations, severity indicators, and modal styling
    - No build tools or compilation required (pure HTML/CSS/JavaScript)

Item 4: Code Archive (Text Versions for Review)

- **Item Description:**
  Plain-text copies of the core source code files provided in .txt format for ease of review and grading without requiring execution of the application.
- **File name/path/link:**
  https://github.com/shafiqul92/soc-incident-response-simulator/tree/main/code
- **Relevant information:**
    - Includes text versions of backend, frontend HTML, JavaScript, and CSS source files
    - Files preserve original formatting and in-line documentation

Item 5: Python Dependency Specification

- **Item Description:**
  Python dependency file listing all required backend libraries with version information to ensure reproducibility.
- **File name/path/link:**
  https://github.com/shafiqul92/soc-incident-response-simulator/blob/main/requirements.txt
- **Relevant information:**
    - Dependencies installed using pip install -r requirements.txt
    - Requires Python 3.7 or higher
    - No additional configuration or API keys required

DEPENDENCIES, RESOURCES & TOOLS

- **Python:** 3.7+ (3.8+ recommended)
- **pip:** required to install dependencies
- **Python packages (via requirements.txt):**
    - Flask==3.0.0
    - Flask-CORS==4.0.0
- **Browser:** Any modern browser (Chrome/Firefox/Edge/Safari) with JavaScript enabled
- **External JS library:** Chart.js loaded via CDN (cdn.jsdelivr.net) in the HTML (no local installation required). Internet required only for first load; afterward it may be cached.
- **Optional tools (only for development/modification):**
    - Git (for cloning) or GitHub ZIP download

    - IDE/editor such as VS Code/PyCharm
    - Python virtual environment (recommended)
- **OS compatibility:** Windows 10/11, macOS, Linux (tested/supportable across platforms)
- **Configuration:** Default server port is **5050**; can be changed in app.py if needed.
- **Credentials/API keys:** None required (standalone local execution; no external services or databases).
- **Data sources:** Synthetic scenario data stored locally in JSON files; aligned with NIST SP 800-61 and CISA-inspired playbooks.

References

[1] P. Cichonski, A. Millar, T. Grance, and K. Scarfone, Computer Security Incident Handling Guide, NIST SP 800-61 Rev. 2, Gaithersburg, MD, USA: National Institute of Standards and Technology, 2012. Available: https://csrc.nist.gov/pubs/sp/800/61/r2/final

[2] A. Nelson, S. Rekhi, M. Souppaya, and K. Scarfone, Incident Response Recommendations and Considerations for Cybersecurity Risk Management, NIST SP 800-61 Rev. 3, Gaithersburg, MD, USA: National Institute of Standards and Technology, Apr. 2025. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r3.pdf

[3] L. Zhang, V. Vasavada, S. R. Kakarla, A. Stavrou, and E. Osterweil, "Design Patterns in the Volumetric DDoS Defense Solution Space," IEEE Commun. Surveys & Tutorials, submitted 2024. Available: https://web.cs.ucla.edu/~lixia/papers/IEEE-DDoS-SolutionSpace.pdf

[4] E. Cadzow, "Exploring the Shift Toward TCP DDoS Attack Vectors," Corero Network Security, Jul. 11, 2023. Available: https://www.corero.com/tcp-ddos-attack-vectors/

[5] "IP traceback," Wikipedia, accessed Dec. 2025. Available: https://en.wikipedia.org/wiki/IP_traceback

[6] P. Manso, J. Moura, and C. Serrao, "SDN-Based Intrusion Detection System for Early Detection and Mitigation of DDoS Attacks," arxiv, Apr. 15, 2021. Available: https://arxiv.org/abs/2104.07332

ADDITIONAL INFORMATION

Github Repository: https://github.com/shafiqul92/soc-incident-response-simulator