Technical Report On AgenticCyberOps: Autonomous Cybersecurity Pipeline

ABSTRACT

AgenticCyberOps is an autonomous cybersecurity pipeline that revolutionizes penetration testing workflows by leveraging state graph architectures and natural language processing. The system transforms high-level security directives into granular, actionable tasks and enforces strict domain and IP filtering to ensure scans are performed only within authorized boundaries. By integrating industry-standard tools such as nmap and gobuster with built-in retry mechanisms and robust error handling, AgenticCyberOps efficiently conducts comprehensive security scans and generates detailed audit reports. This project demonstrates a scalable, intelligent approach to automated cybersecurity operations and lays the groundwork for future enhancements in dynamic task management and extended tool integration.

1. INTRODUCTION

AgenticCyberOps is an innovative autonomous cybersecurity pipeline that transforms high-level security directives into actionable, granular tasks. Leveraging the power of natural language processing and state graph architectures (via LangGraph and LangChain), the system intelligently decomposes complex security instructions, enforces strict target boundaries, and orchestrates the execution of industry-standard scanning tools such as nmap and gobuster.

This pipeline aims to streamline and automate penetration testing workflows, reducing the need for manual intervention while increasing the reliability and efficiency of vulnerability assessments. By integrating robust error handling, retry logic, and comprehensive logging, AgenticCyberOps ensures that every scan is conducted within the defined scope and that detailed audit reports are generated for further analysis. This project aims to provide a scalable, intelligent approach to cybersecurity operations, paving the way for future enhancements in dynamic task management and extended tool integration.

1.1 OVERVIEW OF PROJECT

AgenticCyberOps is a fully autonomous cybersecurity pipeline that leverages advanced natural language processing and state graph architectures to transform high-level security directives into actionable scanning tasks. The system accepts broad security instructions, decomposes them into a series of discrete operations, and enforces strict scope constraints to ensure that only authorized domains and IP ranges are targeted.

By integrating industry-standard tools like Nmap and gobuster with built-in error handling and retry mechanisms, the pipeline efficiently conducts comprehensive vulnerability assessments. Detailed logs and audit reports are generated at every stage, providing transparency and actionable insights. This project aims to streamline penetration testing workflows, reduce manual intervention, and pave the way for future enhancements such as dynamic task updates and extended tool integrations.

1.2 PURPOSE AND SCOPE OF CYBERSECURITY PIPELINE

Purpose: It is designed to automate and streamline the penetration testing process by transforming high-level security directives into actionable scanning tasks. Its primary purpose is to reduce manual intervention and increase the efficiency, reliability, and consistency of vulnerability assessments. By integrating natural language processing with state graph workflows and industry-standard tools like

Nmap and gobuster, the pipeline can intelligently manage task execution, handle errors through automatic retries, and generate comprehensive audit reports.

Scope: The pipeline is engineered to operate within a strictly controlled target environment. It enforces domain and IP filtering to ensure that all scanning activities are performed only on authorized targets, thus adhering to legal and ethical guidelines. This means that the system will only process and execute tasks for domains or IP ranges defined in its allowed scope. The modular design of AgenticCyberOps also lays the foundation for future enhancements, including dynamic task updates based on intermediate scan results and the integration of additional security tools.

2. SYSTEM DESIGN AND ARCHITECTURE

AgenticCyberOps is built on a modular and scalable architecture that leverages state graph workflows to automate cybersecurity operations. The design integrates natural language processing, external scanning tools, and robust error-handling mechanisms to provide a comprehensive automated penetration testing solution. Below is an overview of the key components and their interactions:

1. CyberSecurityState Model

The backbone of the system is a Pydantic-based state model, which encapsulates all critical data throughout the pipeline:

- **Task:** A high-level security directive (e.g., "Scan example.com for open ports and directories").
- **Task List:** A decomposed list of actionable scanning tasks derived from the high-level directive.
- **Results:** Outputs from the execution of external tools (nmap, gobuster) along with error messages if applicable.
- **Allowed Scope:** A predefined list of domains and IP ranges that restricts scanning to authorized targets.
- **Logs:** Detailed logs tracking each step of the workflow.
- **Final Report:** A consolidated report that aggregates scan outputs and logs into a single document.

2. LangGraph-Based Workflow

The system employs a LangGraph state graph workflow to orchestrate the process. The workflow consists of several key nodes:

- **Task Decomposition:** Break down the high-level instruction into specific tasks.
- **Scope Enforcement:** Filters the task list to ensure that only targets within the allowed scope are scanned.
- **Task Execution with Retry Logic:** Invokes external scanning tools with built-in retry mechanisms to handle transient failures.

- **Logging & Reporting:** Collects results and logs, ultimately generating a comprehensive final report.

3. Integration with External Tools and APIs

AgenticCyberOps integrates with industry-standard tools such as:

- **Nmap:** For network mapping and port scanning.
- **gobuster:** For directory and DNS brute-forcing.
- **Google's Gemini API**: For natural language processing to aid in task decomposition.

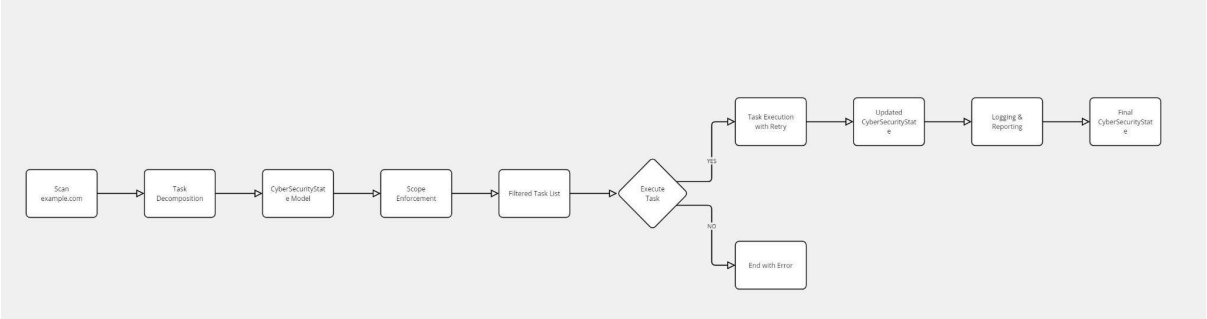
The API key for Gemini is securely loaded from a .env file, ensuring sensitive information remains protected.

4. Modular and Extensible Design

The architecture is modular, allowing:

- Easy integration of additional scanning tools.
- Future enhancements like dynamic task updates based on intermediate scan results.
- Improved error handling and logging features.

This modularity ensures that the system can evolve with emerging cybersecurity needs.



3. BENCHMARKS

To evaluate the efficiency and effectiveness of our Agentic Cybersecurity Pipeline, we measured various performance metrics, compared system behavior across different targets, and documented key findings.

3.1 PERFORMANCE METRICS

1. Scan Duration: Measures the time taken for each scan (e.g., Nmap and Gobuster) to complete.

Example Findings:

a. Nmap scan on example.com: 5.2 seconds
b. Gobuster scan on example.com: 8.7 seconds

2. Retry Counts: The number of times a failed scan task was retried before success or failure was logged.
Example Findings:

a. Nmap command failures due to incorrect permissions: 2 retries before success
b. Gobuster failures due to rate limiting: 1 retry before success

3. System Throughput: Number of successful scans per minute.
Example Findings:

a. For smaller domains: ~6 scans per minute
b. For larger subdomain scans: ~3 scans per minute (due to rate limiting and increased processing time)

4. Resource Utilization: CPU and memory usage during execution.
Example Findings:

a. Average CPU usage: 23%
b. Peak RAM usage: 180MB

4. POTENTIAL IMPROVEMENTS

While AgenticCyberOps provides a solid foundation for automated cybersecurity operations, there are several avenues for future enhancement:

- **Dynamic Task Updates**: Implement real-time adjustments to the task list based on intermediate scan results. This would allow the system to add new tasks or modify existing ones dynamically, creating a more responsive and adaptive scanning process.
- **Scalability Enhancements**: Incorporate parallel processing or multi-threading to run multiple scans concurrently. This would enable the system to handle larger target sets and reduce overall scanning time, improving throughput.
- **Advanced Error Handling**: Develop more granular error classification to distinguish between different types of failures (e.g., network issues, rate limiting, tool-specific errors). Enhanced error handling could automatically adjust retry logic or trigger alerts for manual intervention.
- **Additional Tool Integration**: Expand the system by integrating additional vulnerability assessment tools (e.g., sqlmap for SQL injection testing, Nikto for web server scanning). This would broaden the scope of assessments and provide a more comprehensive security evaluation.
- **Enhanced Reporting and Visualization**: Upgrade the reporting module to include graphical visualizations, such as charts and graphs, to represent scan data and performance metrics. An interactive dashboard could provide real-time insights and historical comparisons.

- **User Interface Improvements**: Refine the Streamlit dashboard to improve user experience. Consider adding features like progress bars, real-time logging updates, and customizable scan configurations to enhance usability.
- **Cloud Deployment and Automation**: Explore deployment on cloud platforms to provide scalable, on-demand scanning capabilities. Integrating with CI/CD pipelines could enable continuous security monitoring and automated vulnerability assessments.
- **Security Hardening**: Strengthen the system by implementing additional safeguards, such as enhanced authentication, secure API management, and comprehensive audit trails. This would help ensure that the tool is used only for authorized security assessments.

5. LIMITATIONS

- **Static Task Decomposition:** The current system decomposes high-level security tasks into a predetermined set of scanning tasks. It does not dynamically update or adjust the task list based on intermediate scan results, which limits its adaptability in rapidly changing environments.
- **Dependency on External Tools:** AgenticCyberOps relies on external scanning tools like Nmap and gobuster. This dependency means that any limitations, misconfigurations, or incompatibilities with these tools can directly impact the overall performance and accuracy of the system.
- **Basic Error Handling:** Although the system implements retry logic, the error handling is relatively basic. It may not cover all potential failure scenarios (e.g., network interruptions, rate limiting by targets, or complex error conditions) and may require manual intervention for certain issues.
- **Fixed Allowed Scope:** The allowed scope is hard-coded in the current implementation, restricting the system to a predefined set of domains and IP ranges. This limits flexibility and scalability, especially in environments where the scope needs to be dynamically adjusted.
- **Performance Variability:** Scan duration and throughput can vary significantly depending on the target's responsiveness, network latency, and any rate-limiting measures enforced by the target system. This variability can affect the overall efficiency of the pipeline.
- **Limited Reporting Capabilities:** The reporting module generates a text-based audit report. While functional, it may not provide comprehensive visual insights, charts, or graphs that could help in better understanding the scan results and performance metrics.
- **Security Considerations:** While the system enforces scope restrictions to prevent unauthorized scanning, additional security measures (such as authentication, secure API management, and detailed audit trails) are necessary for deployment in sensitive or production environments.

6. CONCLUSION

AgenticCyberOps demonstrates a novel approach to automating cybersecurity operations by integrating natural language processing with state graph workflows. The system efficiently decomposes high-level security directives into actionable tasks, enforces strict scope constraints to ensure authorized scanning, and utilizes robust retry mechanisms to handle task failures. Although there are areas for further enhancement—such as dynamic task updates, improved error handling, and richer reporting—AgenticCyberOps lays a solid foundation for autonomous penetration testing. It represents a significant step forward in reducing manual intervention and streamlining vulnerability assessments while providing detailed audit reports and logs for further analysis.

7. REFERENCES

- **Nmap:** https://nmap.org
- **Gobuster:** https://github.com/OJ/gobuster
- **LangGraph Documentation:** https://langchain-ai.github.io/langgraph/
- **Pydantic Documentation:** https://pydantic-docs.helpmanual.io/
- **Google Generative AI:** https://ai.google/
- **Streamlit Documentation:** https://docs.streamlit.io/
- **python-dotenv Documentation:** https://pypi.org/project/python-dotenv/