Multi-Agent AI Enhancing Security, Threat Response, and Orchestration (MAESTRO)

Scaling Cyber Operations with Artificial Intelligence

CS/CY-4820

Authors/Students:
Asher Antrim, Adrianna Holst, Carson Levenson,
Gabriela Mallack, Blake Mead, and Elliot Viaud-Murat
Cedarville University (CU)

Academic/Faculty Advisor:
Professor Nicholas Parry (CU)

Mentor:
Matthew Clark
MBClark Engineering LLC

21 February 2025

Abstract

In the cybersecurity field, a significant skills gap exists between senior-level operators and novice operators. The goal of this project is to develop an Artificial Intelligence Cyber Offensive Assistant that enables novice cyber operators to perform at the level of an expert. This assistant is called Multi-Agent AI Enhancing Security, Threat Response, and Orchestration (MAESTRO). Through our research and development, we built a program integrating local or cloud-based state-of-the-art Large Language Models (LLMs) that will give accurate responses to assist a cyber operator in developing tactics. We utilize a Multi-Agent System (MAS) and Knowledge-Augmented Generation (KAG), as well as rigorous user testing, to increase confidence in our system. This research addresses ways to improve accuracy and consistency in LLM responses to confidently use artificial intelligence to scale operators in the cyber operation sector.

List of Acronyms

AI	Artificial Intelligence
CKC	Cyber Kill Chain
CTF	Capture the Flag
CVE	Common Vulnerabilities and Exposures
ISC2	International Information System Security Certification Consortium
KAG	Knowledge Augmented-Generation
MAESTRO	Multi-Agent AI Enhancing Security Threat Response and Orchestration
LLM	Large Language Model
MAS	Multi-agent System
NVD	National Vulnerability Database
RAG	Retrieval-Augmented Generation
TTP	Tactics, Techniques, and Procedures
VM	Virtual Machine

Background

Cyber Operator Skill Gap

As the threat of adversaries grows exponentially year after year, the need for qualified cybersecurity professionals becomes increasingly critical. The complexity of adversaries' tactics and techniques necessitates the expertise of qualified professionals who are capable of adapting, mitigating risk, and outthinking the enemy. However, there is a shortage of expert cybersecurity professionals to address the ever-growing pressures of our adversaries. According to the world's leading cybersecurity professional association, International Information System Security Certification Consortium (ISC2), "While active workforce growth was stable, the workforce gap increased by 19% year-on-year to 4.8 million." This highlights the significant negative impact of cybercrime in America and the world. Considering the current and future state of decreasing cybersecurity professionals with an increase in cybercrime, the MAESTRO Team developed an Artificial Intelligence Assistant that uses a Multi-Agent System (MAS) in hopes of elevating non-elite cybersecurity professionals.

LLM vs. AI

Artificial Intelligence (AI) is a broad field that encompasses any machines or systems designed to perform tasks that would generally require human intelligence such as reasoning, problem-solving, learning, or perception. It includes various sub-fields such as robotics, machine learning, and natural language processing. On the other hand, Large Language Models (LLMs) are a specific type of AI focused on understanding, processing, and generating human language. LLMs are trained on massive datasets of text and use deep learning techniques to predict and generate text based on user prompts. Prompts are "specific inputs [from a user] that help guide your large language model in generating the desired outcome" (Multimodal 2024). LLMs focus on language-based tasks, leveraging both user prompts and datasets that they are originally trained on.

Multi-Agent System (MAS)

A Multi-Agent System (MAS) is a system composed of multiple interacting agents designed to solve complex problems that are difficult or impossible for a monolithic system to handle. An agent is an isolated entity that performs a specific task. MASs have specialized agents that can collaborate, share information, and effectively divide tasks to achieve a common objective more efficiently. The main advantages of a MAS are scalability and robustness (He 2025). MAS manages large-scale problems by distributing workload across multiple agents. The system can easily scale up because agents separate the tasks into smaller, more manageable tasks in light of the complexity of the main objective. Each agent has a specific role and is responsible for performing a particular task it specializes in, leading to more efficient and effective performance (LangChain 2025).

Retrieval Augmented Generation (RAG)

Retrieval Augmented Generation (RAG) is a technique that enhances LLMs by combining them with an external knowledge base to improve response accuracy and relevance. Instead of solely relying on fixed training datasets, which can lead to outdated or incomplete information, a RAG system retrieves relevant documents or data from an up-to-date and accurate knowledge base to supplement its responses. This approach mitigates hallucination and outdated or incorrect information in the model's response by ensuring that its answers are grounded in reliable sources. A RAG also remains adaptable to new knowledge without requiring frequent model retraining (RedHat 2025).

Multi-Agent Knowledge-Augmented Generation (KAG)

Knowledge-Augmented Generation (KAG) is a framework that integrates structured reasoning from knowledge graphs with the LLM's natural language processing capabilities (Kumar 2025). This enables LLM-based systems to handle complex, domain-specific tasks more effectively. Unlike traditional RAG methods, which retrieves relevant documents based on similarity only, KAG incorporates logical reasoning and structured knowledge graphs to provide more precise and contextually relevant responses. The framework enhances factual accuracy and enables more effective handling of intricate tasks that LLMs struggle with, making it useful for specialized domains.

LangChain

LangChain is a framework designed to facilitate the development of LLM based systems with external data sources, interfaces, and multi-step reasoning workflows. The primary advantages of LangChain are modularity and flexibility. It allows for the seamless orchestration of different components, such as memory for context retention, RAG for enhanced accuracy, and interfaces for executing tasks beyond text generation (LangChain 2025). This makes LangChain particularly useful for building intelligent assistants and autonomous agents that require complex reasoning and real-time information access.

Hypothesis

We aim to develop a reliable, consistent approach to using LLM-based systems to scale novice cyber operators, enabling them to perform at a higher skill level with greater efficiency and accuracy.

Methodology

To aid in advancing cyber operations, this research focuses on how to improve the efficiency and scalability of cyber operators with varying skill sets. We implement MAESTRO through a Knowledge-Augmented Generation (KAG) system based on a Multi-Agent architecture. This approach dynamically distributes tasks to specialized components that leverage Retrieval-Augmented Generation (RAG) to enhance response accuracy and relevance. By integrating structured retrieval with Large Language Model (LLM) reasoning, the system improves efficiency, ensures more precise answers, and scales easily with additional agents, making it well-suited for complex tasks that require accurate responses.

MAESTRO was built around the concept of the Cyber Kill Chain (CKC). The CKC, developed by Lockheed Martin, "identifies what the adversaries must complete in order to achieve their objectives" (Lockheed Martin 2025). The Unified Cyber Kill Chain is a combination of the MITRE ATT&CK and Lockheed Martin CKC, which we used to develop the states that the assistant tool will visit. In this context, state is a portion of a program in which specific actions take place to fulfill a specific task, and the program will not proceed to the following state unless the conditions are met. For example, if the program begins in the Reconnaissance state, it will help gather as much information about the target as possible. Once that is accomplished, the system will advance to the next state, Weaponization, as seen in Figure 1. The phases that were chosen for MAESTRO are Reconnaissance, Clarify, Enumeration, and Pivot.

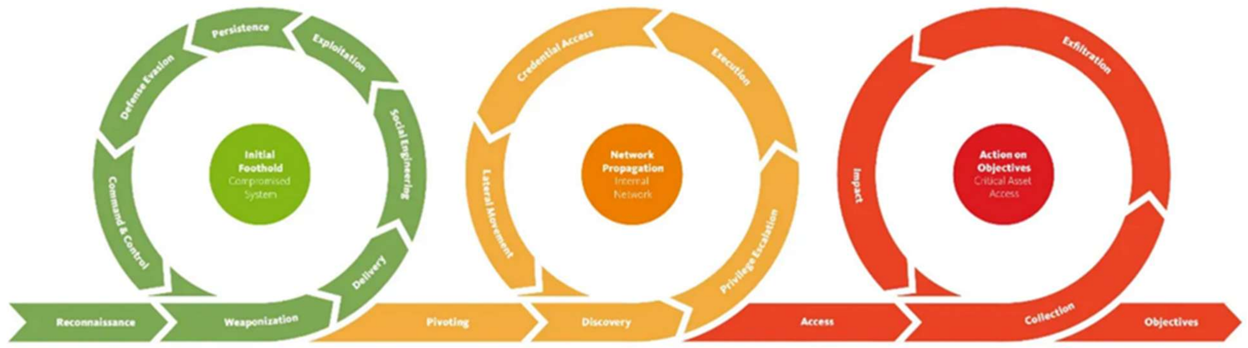

Figure 1: Unified Cyber Kill Chain

Since the tool is meant to leverage the skills of the cyber operator, it can assist at any phase. For example, if a user has extensive information about the target system, the assistant will move to the next phase rather than beginning in Reconnaissance. When an operator is unsure on how to proceed within the current state, the MAESTRO tool helps guide them to “completing” a state. This may be done by prompting the user to understand their objective, providing tips on how to proceed, or providing written code and instructions on how to execute an exploit. The goal is to help the operator know what to do next.

Information Retrieval

Information retrieval is an important component of building confidence and consistency in MAESTRO, and a key aspect of the Knowledge-Augmented Generation (KAG) architecture. The integration of a knowledge-based system helps improve the quality of the response from the LLM, as the responses generated are verified and refined through these trusted data sources. The datasets give the LLM a path in which to respond and help prevent divergence from what is being asked. To obtain trustworthy information, several databases were used. These databases were gleaned online from trusted sources such as the National Vulnerability Database (NVD), Collegiate Red Team Competition Team, and known Tools, Techniques, and Procedures (TTPs) from various sources. Each database can be used to cross-reference information as a method of verifying an LLMs proposed solution or as a source of information in generating responses. Each agent in the MAS will have access to its own pertinent database or sets of databases, ensuring that the response being generated can be trusted by the operator.

The databases being accessed provide useful information for the agent. The NVD CVE database provides known reported vulnerabilities. This database is constantly being externally updated; therefore, the most recent vulnerabilities are available for an operator to use in creating an exploit. Other useful databases are ones of known TTPs, and resources used by Cyber Competition teams to complete tasks such as Capture the Flag (CTF) competitions. The methods used by these teams serve as guidelines that the agent can use in developing an answer. These databases do not have to be publicly available, however. The way we have built our system allows for the use of proprietary databases. We have been able to use a database that our own Cedarville University Collegiate Cyber Defense Competition team has made based on their competitions. As the system continues to be built out, each agent will be associated with a database in which it will access to generate more accurate responses. For example, figure 2 below illustrates a simplified version of the agent’s usage of databases, with the question being posed to the agent shown on the left. Depending on the task, the questions vary, but the agent will access the appropriate databases before the answer is given. With the information gathered from the databases, and with the known objective, it will generate a response which is filtered and then returned.

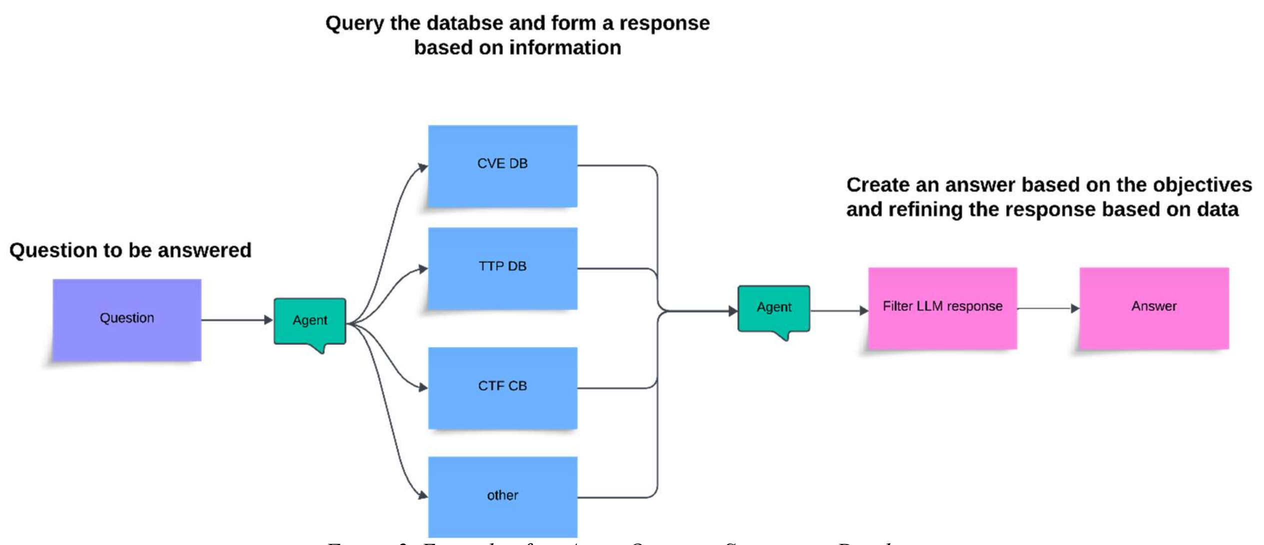
Query the databse and form a response based on information
Create an answer based on the objectives and refining the response based on data
Question to be answered
Question
Agent
CVE DB
TTP DB
CTF CB
other
Agent
Filter LLM response
Answer

Figure 2. Example of an Agent Querying Supporting Databases

With LangChain, there is the ability to use functions that are called "tools" for each agent. A tool is a mechanism that allows the agent to perform actions other than just returning text. These actions include things like querying databases, running code, calling APIs, or retrieving real-time information. Tools are very useful, and in this instance, they have been used for querying databases as well as assisting in writing exploit code. Different tools can be used for different agents, depending on the needs of the agent that is calling them.

In this project, there are several tools that have been implemented so far. These tools and brief descriptions of them can be seen in table 1. The three agents that have been implemented so far are Reconnaissance, Pivoting, and Exploit agents. The Reconnaissance Agent's goal is to gain intelligence about the victim machine. To do this, it uses the cve_database tool as well as the prompt_user tool. The cve_database tool pulls CVE information directly from the database, while the prompt_user tool prompts the operator for clarification. If the agent decides it does not have enough information to go on, it can prompt the operator to provide more context and details so it can give an accurate response. The Pivot and Exploit Agents both use the prompt_user tool, and the Exploit Agent uses the cve_database tool, as well. The Pivot Agent is used to transition the operator from the Reconnaissance phase of the Cyber Kill Chain (CKC) into the Exploit phase, and the Exploit Agent's purpose is to assist the operator in exploiting the target system based on the intelligence gained from the Reconnaissance Agent. In addition to the prompt_user and cve_database tools, the Exploit Agent uses an exploit_code tool. This tool is used to assist the agent in creating accurate exploit code using code templates. The tool retrieves the code template best suited for the task at hand. For example, if the operator is hoping to perform a buffer overflow attack, then the exploit_code tool will retrieve the buffer overflow code template for the agent to use. This way, the task for the agent is simple and straightforward, helping it provide a more accurate response and build further confidence in our program.

Table 1. Available Tools for Agents

Tools Available	**Description**
prompt_user	Prompts operator for clarifying information if the agent does not have enough to go on.
cve_database	Retrieves relevant CVEs from the NVD CVE database.
exploit_code	Retrieves relevant code template for what is being prompted. For example, this tool will retrieve buffer overflow code template if a buffer overflow exploit is desired.

Using tools alongside the agents has proved very helpful. Instead of requiring the operator to provide specific information about their target, the Large Language Model (LLM) within each agent can filter the information down to what is necessary. All that the operator must do is query the agent, asking to perform an action in one of the steps of the MAESTRO CKC. The agent decides what keywords to use from the operator's query and then uses the tools to retrieve the most relevant information. If it needs further context or clarification, it can prompt the operator for that information using the prompt_user tool.

As the system continues to be built out and the scenarios become more complex, there is still a risk of LLM hallucination. Providing datasets is one way to combat response hallucination, but other strategies are also necessary. Prompt engineering and breaking tasks into smaller components both play a role in improving the accuracy and correctness of answers.

Prompt Engineering

Prompt engineering is a method in which the given prompt is formatted or written in a way that will return the best output. There are several techniques involved in prompt engineering: instructive prompts, system prompts, question-answer prompts, contextual prompts, and mixed prompts (Giray 2023).

Instructive prompts guide the LLM to perform a specific task. These could be tasks such as writing a summary of a concept or providing an analysis on a certain topic. System prompts give the LLM a starting point for how it should answer any future prompts. For example, we have used system prompts in our project to tell the agent what its role is. MAESTRO has been told that it is a chatbot assisting a cyber operator in a Collegiate CTF competition, with the goal of helping the operator understand vulnerabilities that can be exploited on the victim machine.

Question-answer prompts help the LLM to structure what it is being asked to do, essentially providing boundaries over what is being prompted.

Contextual prompts help provide more context for the LLM. One of the pitfalls a lot of users fall into when using AI is not providing enough context. Context is key; the more concise the context provided, the better the LLM can give an accurate response. The challenge is discovering the right amount of context without providing too much, as excessive information could also lead to inaccurate responses.

Finally, mixed prompts are a mixture of the various prompts mentioned above. It may be appropriate to combine several of the prompt types to get the best answer possible. In this project, we have mostly combined the system prompts with instructive prompts and/or contextual prompts so that we can provide the LLM with a specific task to do and give it the right context to perform that task well. Using prompt engineering greatly helps the LLM to provide consistent, accurate responses.

Task Decomposition

Another way to decrease the chances of response inaccuracies is by breaking down the tasks the Large Language Model (LLM) performs. A clearer example is in the Exploit Agent, seen below in figure 3. Once the LLM receives the appropriate information to move to the Exploit Agent, it will access the Common Vulnerabilities and Exposures (CVE) database via the cve_database tool to find a potential vulnerability applicable to the scenario. Within the tool, shown in figure 4, it narrows down the CVEs from the database and returns a specific number of CVE IDs. To prevent the LLM from generating inaccurate CVE information, the program itself will retrieve the descriptions of each CVE. The CVE information is then returned to the LLM, which can either prompt the user for more information, call a tool to help write the exploit code, or move to a different agent.

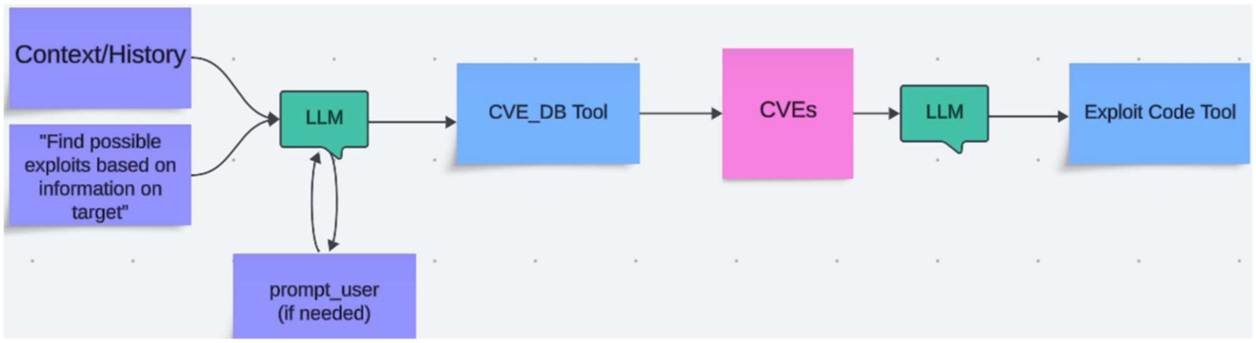

Figure 3. Exploit Agent CVE Database usage

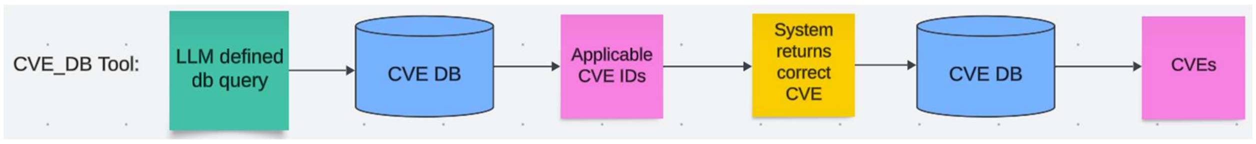

Figure 4. CVE database tool pipeline

Multi-Agent Architecture Implementation

Initial System Design

For our initial Multi-Agent System (MAS) design, we implemented a hierarchical Multi-Agent Architecture. The architecture consists of a supervisor agent that manages multiple specialized worker agents. The supervisor receives the initial prompt and determines which worker agent is best suited to handle the given request. The chosen worker agent processes the relevant portion of the task based on its specialization and returns the result to the supervisor. The supervisor then either delegates the next step to another agent or finalizes the response if the task is complete. Each worker agent functions as its own Retrieval-Augmented Generation (RAG) system, equipped with access to one or multiple domain-specific knowledge bases. It can prompt the operator for clarifications or additional inputs to refine their responses.

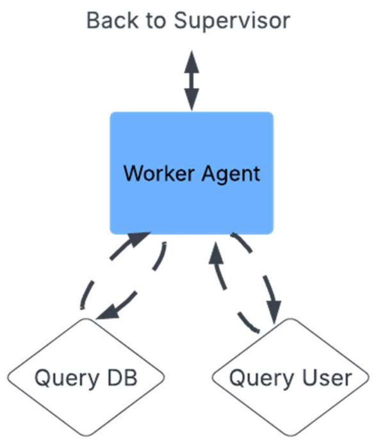

Figure 5. RAG based worker agent

In our initial implementation, there was one specialized agent for each step of the first cycle of the Unified Cyber Kill Chain (Reconnaissance to Pivoting), excluding Social Engineering, Command and Control, and Defense Evasion. The purpose of this design was to provide task specialization through multiple specialized RAG agents, as well as simple scalability and modularity as each agent is only connected to the supervisor. This design allowed new agents to be easily added without needing to restructure the entire system. While this approach had some benefits, we encountered challenges related to context retention and inter-agent coordination, which led to adjustments and refinements in our architecture.

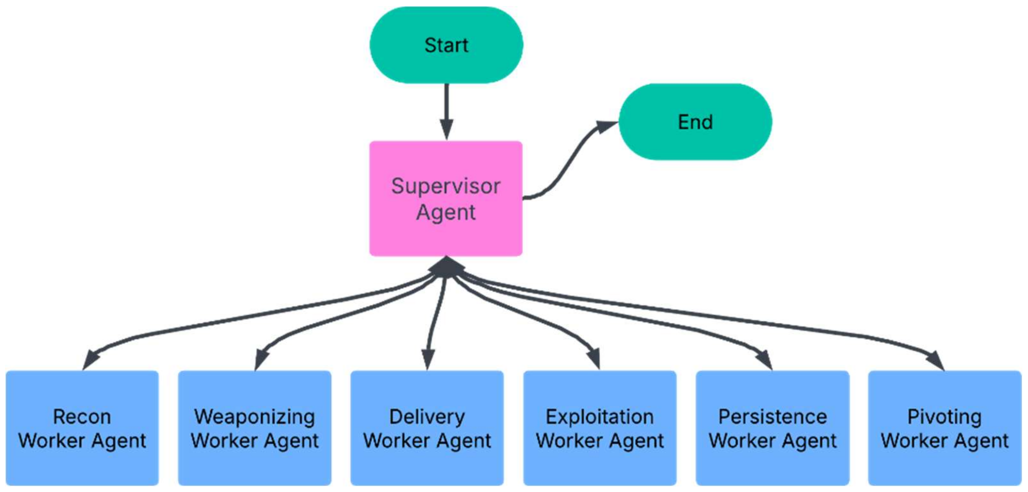

Figure 6. Architecture (First Iteration)

Context Management

One of the primary challenges encountered with the initial architecture was context loss, in which ineffective memory implementation prevented agents from retaining critical information across transitions. Another challenge was context misalignment, in which differences arose between agents' perspectives due to abrupt context switches.

To address context loss caused by excessive agent transitions, we merged multiple worker agents into fewer, broader specialized agents. Instead of having a separate agent for each step of the Unified Cyber Kill Chain, we combined them into three worker agents: Reconnaissance, Exploitation, and Pivoting. Additionally, to improve system memory, we began storing worker-user prompts and responses (which were not handled before), ensuring that all context was maintained throughout interactions.

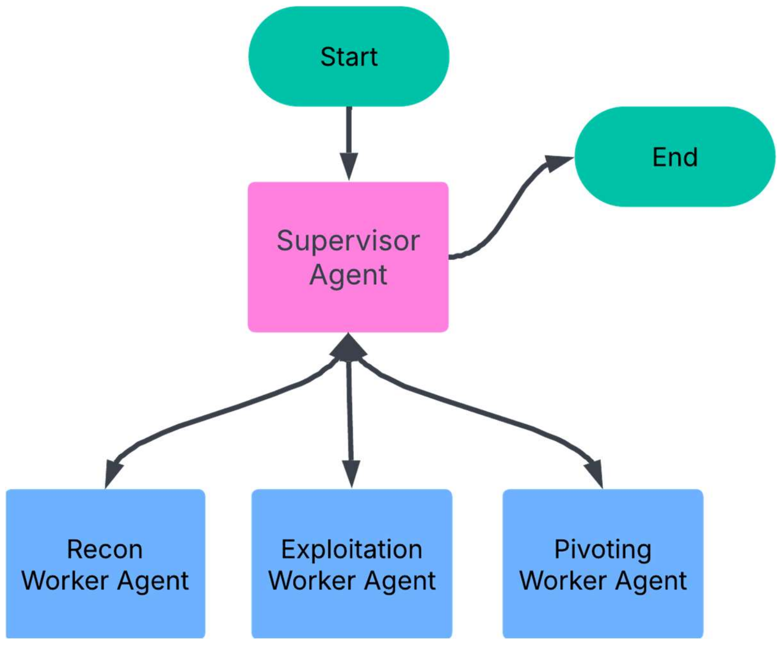

Figure 7. Architecture (Second Iteration)

By reducing transitions and implementing a more robust memory system, we significantly improved context retention and minimized context misalignment, leading to a more coherent and efficient multi-agent workflow.

Agent Transitioning

Although merging worker agents into broader specialized agents improved context retention and efficiency, it introduced a new problem: agents struggled to relinquish control and transition back to the supervisor. Even after completing the user's request, agents would continue processing and prompting the user unnecessarily instead of handing off control, leading to inefficiencies and undesirable actions.

To resolve this, we introduced a new sperate agent that would monitor conversation history to determine when a worker agent achieved its objective. This addition improved workflow and system efficiency, ensuring that specialized agents stayed focused on their given tasks and prevented tangents and repetitions once a task had been accomplished.

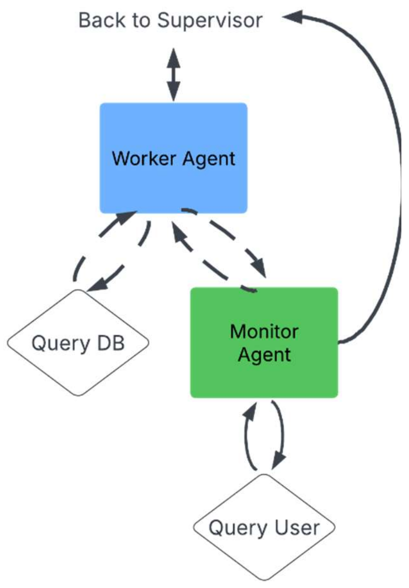

Figure 8. Monitoring Worker Agents

LLM Evaluations

An important factor for our program's reliability is the Large Language Model (LLM) used within the program. Because LLMs are prone to hallucination, returning inaccurate information, or not responding at all, choosing the best LLM is a critical step in ensuring the system's effectiveness. However, it is important to note that the LLM is not the cornerstone of the project's reliability. Once the system matures, the Multi-Agent System/Knowledge Augmented-Generation (MAS/KAG) architecture should handle instability or hallucination issues, regardless of the LLM's strength. However, it is still important to see which LLMs provide the most reliable and accurate responses, which is the goal of the tests described in this section.

To narrow down which LLM gave the most reliable response, a series of tests were developed. The tests created included tasks such as Common Vulnerabilities and Exposures (CVE) lookups, prompt refining, and security analysis, across a range of different LLMs. CVE lookups are a key part of the Retrieval-Augmented Generation (RAG) system, which allows the LLM to output up-to-date and relevant information from external databases. For example, the LLM may need to search for recent vulnerabilities, patches, or exploits to provide accurate insights in a cybersecurity context.

One of the most important evaluation metrics studied was the maximum context length and token count of each LLM. According to Microsoft, tokens "are words, character sets, or combinations of words and punctuation that are generated by Large Language Models (LLMs) when they decompose text" (Microsoft Learn 2024). These can be as short as a single character or as long as a word. The context length, measured in these tokens, determines how much information the model can process and generate at a given time. A longer context length allows the LLM to handle more complex, multi-step queries without losing track of earlier parts of the conversation.

This is crucial for tasks such as security analysis, where the LLM needs to maintain coherence over extended periods of time.

To assess each model's ability to handle complex scenarios, various large test queries were used. Performance in long queries was a key factor, as the machine's ability to remember context was vital in the scenarios tested. A strong memory allowed the AI to maintain its coherence across extended interactions. In the chart below, models such as Claude 3.5 Sonnet and GPT-4 were particularly notable due to their high token counts, where other LLMs such as Llama 3 and Mistral 7B struggled with retaining context over extended interactions, often requiring the tester to repeatedly remind it of the current situation.

Table 2. LLM Data Chart

Model Name	**Max Context Length (Tokens)**	**Input Tokens**	**Output Tokens**	**Number of Parameters**	**Efficiency in Detail**
Gemma 2[1]	8,192 tk	Up to 8,192	Variable	2 billion	Decent
Claude 3.5 Sonnet [2]	200,000 tk	Up to 100,000	Variable	175 billion	Excellent
Llama 3[3]	8,000 tk	Up to 8,000	Up to 2,048	8 billion	Great
Mistral 7B [4]	8,192 tk	Up to 8,192	Variable	7.3 billion	Great
Dolphin Mistral [5]	32,000 tk	Up to 32,000	Variable	7 billion	Great
GPT-4 (32K) [6]	32,768 tk	Up to 32,768	Up to 32,768	1.8 trillion	Excellent
Llama 3.2 [7]	128,000 tk	Up to 128,000	Variable	3.21 billion	Excellent

[1] Google. "Google/Gemma-2-9b." Hugging Face, December 13, 2024. https://huggingface.co/google/gemma-2-9b.
Haywood, Sloan, Genevieve Waren and Alex Warren. "Understanding Tokens - .NET." Microsoft Learn, December 21, 2024.
[2] Anthropic. "Claude 3.5 Sonnet," Anthropic, June 20, 2024. https://www.anthropic.com/news/claude-3-5-sonnet.
[3] Meta. "Meta-llama/Meta-Llama-3-8B" Hugging Face, December 6, 2024. https://huggingface.co/meta-llama/Meta-Llama-3-8B.
[4] "Mistral 7B." arXiv Preprint, October 2023. https://arxiv.org/pdf/2310.06825.
[5] LLM Explorer. "Dolphin 2.1 Mistral 7B by Cognitivecomputations: Benchmarks and Detailed Analysis. Insights on Dolphin 2.1 Mistral 7B." LLM Explorer, n.d. https://llm.extractum.io/model/cognitivecomputations%2Fdolphin-2.1-mistral-7b,60Q8GG9DUcdDnxoW5N4kS0.
[6] Context AI. "Llama 3 vs. GPT-4 Comparison." Context AI, 2024. https://context.ai/compare/llama3-8b-instruct-v1/gpt-4-0613.
[7] Meta. "Llama-models/Models/Llama3_2/MODEL_CARD.md." GitHub, n.d. https://github.com/meta-llama/llama-models/blob/main/models/llama3_2/MODEL_CARD.md.

One of the most notable challenges in testing our chosen LLMs was their handling of security-sensitive prompts. Some models, particularly GPT-4, were overly cautious when responding to exploit-related queries. This is most likely due to built-in safeguards designed to prevent misuse of AI, but it also limited the model's effectiveness in a cybersecurity scenario. This highlights a key tradeoff: while stricter security measures prevent potential misuse of the LLM, it also limits the desired results we seek from the machine. To address this, we could explore better prompt engineering techniques to guide the model towards more useful responses or even consider retraining a model with fewer built-in restrictions for the specific cases where security issues are handled externally.

Another crucial factor during the evaluations was the trade-off between cloud-based and locally hosted Large Language Models (LLMs). Given the high computational demand of larger models such as Claude and GPT4, cloud-based options are more practical, offering refined responses without any strain on local resources. However, locally hosted LLMs offer greater security and privacy, ensuring that any sensitive data remains on the host's machine without exposure to external servers. Ultimately, the choice between cloud-based and locally hosted LLMs depends on the shifting balance between performance and security, with cloud-based solutions excelling in efficiency and scalability, while local models provide increased control over data security.

Our team chose Claude 3.5 Sonnet as our LLM of choice for the project, for several reasons. The LLM's 175 billion parameter model made it a prime choice for handing long queries with very little hallucinations due to its 200,000 token context length. This, along with its ease of use and cloud-based nature, made the implementation easier than other locally hosted LLMs.

Testing Methodology and Evaluation

Introduction

For testing inside our program, we decided on three models to see how the variety of parameters within each model influences its performance in our program. In this testing, we ran each model through three different scenarios: scanning a system for recon information related to initial access, gaining access through a known vulnerability, and gaining public reconnaissance. The three models that were ran through these tests were Claude 3.5 Sonnet, GPT-4o, and Llama 3.2 (locally). With the amount of context that our program is providing the LLM, the context window of each model plays a huge role in making sure that the response is on-topic and relevant. The cloud-based models are better at dealing with a large amount of context, but with a large amount of computer power, one would be able to run a local model with comparable performance. The Llama 3.2 model used in this test is the 3 billion parameters version running on an Nvidia RTX 3080 12 GB with acceptable response tokens per second compared to cloud-based models.

Subjectivity in Testing

Evaluating LLM performance in cybersecurity tasks is inherently subjective. In this context, “subjective” means that the evaluation relies on expert judgment and qualitative assessments rather than standardized, quantifiable metrics. Cybersecurity operations frequently involve nuanced, context-dependent decision-making where a single “correct” answer may not exist. Consequently, the analysis focuses on how well each model’s responses align with best practices, the clarity and actionability of the guidance provided, and their adaptability to varying operational scenarios. This subjective approach mirrors the complexity and variability of real-world cybersecurity environments.

Dynamic User Prompting vs. Static Responses

A key differentiator in the evaluation is each model's ability to interact with the user during the operation. Notably, Claude Sonnet 3.5 supports dynamic, interactive user prompting within our program at the time of this paper. Throughout the testing scenarios, Claude can actively request additional information and clarification after each response it provides, allowing its guidance to be continuously tailored based on the operator's evolving context. In contrast, GPT-4o and Llama 3.2 (locally) provide single, static responses that do not accommodate follow-up queries. This limitation can affect their adaptability, especially when iterative feedback is crucial for effective decision-making and brings more value to the program. The future development of this project involves bringing dynamic user prompting to other models, but with the current state of the demo, it is not integrated for GPT-4o and Llama 3.2 (locally).

Evaluation Scenarios and Findings

The assessment of the models consisted of three core cybersecurity scenarios:

1. **Initial Network Scanning Guidance**
    a. **Objective:** Provide initial reconnaissance steps to help an operator gather information about a target machine.
    b. **Test Prompt: "Given an IP address and written permission from the network owner to perform scanning, what are the initial steps I can use to gather information about the machine and identify potential vulnerabilities?"**
    c. **Observations:**
        i. *Claude Sonnet 3.5* requests the target IP, uses interactive prompting to guide the operator through gaining additional details (e.g., open ports, services, and versions), and integrates CVE lookups to suggest potential vulnerabilities.
        ii. *GPT-4o* offers clear, sequential instructions (e.g., suggesting a nmap -A <IP_Address> scan) but provides unnecessary verbose output that does not provide any value to the operator.
        iii. *Llama 3.2* delivers a concise response that is suitable for local deployment yet lacks depth because of the limited context and slow response times.

2. **Vulnerability Exploitation and Initial Access**

a. **Objective:** Guide the operator through exploiting a known FTP vulnerability to gain initial access.
b. **Test Prompt:** “The target system is running an outdated version of an FTP server known to be vulnerable to a remote code execution exploit. What steps and commands should I use to safely exploit this vulnerability and gain initial access?”
c. Observations:
   i. *Claude Sonnet 3.5* tailors its exploitation strategy by guiding the operator through getting more details such as running a scan to get the FTP server software and version, thereby looking up in the CVE database any known vulnerabilities that are related to the version of FTP the operator found in the scan.
   ii. *GPT-4o* recommended a similar strategy to Claude by recommending a scan to get the software and version of the FTP server that is running.
   iii. *Llama 3.2* provides initial steps but does not elaborate on what exactly the operator should do which may necessitate additional research by the operator if they are not familiar with this attack already.

3. External Foot printing and Public Reconnaissance
a. **Objective:** Recommend best practices and tools for public reconnaissance of an organization’s external infrastructure.
b. **Test Prompt:** “I have been assigned to perform public reconnaissance on a target organization’s external infrastructure. What are the best practices, commands, and tools I should use to gather detailed information about their public IP addresses, domains, and network footprint?”
c. Observations:
   i. *Claude Sonnet 3.5* initially prompts the operator to confirm authorization to complete this situation and requests further details as needed by recommending tools and sites that the operator should leverage such as Shodan.io, DNS Lookup, and whois. Claude also gave the commands for the operator to run so that the operator could substitute the domain or IP address of the target into these commands.
   ii. *GPT-4o* started by warning the operator to only complete this attack if they have authorization to do so. Then proceeds to recommend websites and commands for the operator to use but does not give explicit copy-paste commands or even URLs.
   iii. *Llama 3.2* offers a very simple yet effective basic suggestion for the operator to complete. However, because of the long response times and the limited token responses, it does not bring any value to the operator. All the steps the model recommends should be knowledge that the operator is already familiar with.

Overall Comparative Analysis

- **Claude Sonnet 3.5:**

  - *Strengths:*
    - **Interactive prompting:** Actively requests user input for additional details, ensuring tailored and context-specific guidance.
    - **Comprehensive responses:** Incorporates real-time CVE lookups and structured step-by-step guidance.
  - *Limitations:*
    - The step-by-step nature of its responses can sometimes feel overly rigid, leaving little room for alternative or supplementary techniques.

- **GPT-4o:**
  - *Strengths:*
    - Provides detailed, context-aware instructions with a large context window.
  - *Limitations:*
    - Lacks the ability to engage in interactive prompting, which can result in less adaptive guidance, especially for complex, evolving scenarios. The model also can be too verbose with information that does not help the operator complete their task. This can potentially confuse newer operators and just overall bloat the response making it harder to find the actual information that is important to the operator.
- **Llama 3.2 (Locally):**
  - *Strengths:*
    - Offers clear and succinct instructions that can be beneficial in offline, local deployments.
  - *Limitations:*
    - Produces responses with limited context and slower response times, making it less effective in iterative, time-sensitive situations. This model also often requires more steps by the operator to find the command, code, or more details about the steps they would like to take.

Key Takeaways

In summary, subjective evaluation indicates that while all models can support cybersecurity operations effectively, the ability to interactively prompt the user—exemplified by Claude Sonnet 3.5—provides a significant advantage. This dynamic interaction allows for continuously tailored guidance, which is particularly valuable in cybersecurity scenarios where operational contexts are constantly evolving.

User Testing

As we continue to develop MAESTRO, it will undergo user testing. This provides the opportunity for users of all skill levels to provide feedback to refine the program and better serve and support operators. With the integration of a Large Language Model (LLM), the responses provided can vary. Thus, one major challenge is evaluating our program by analyzing quantitative data. To combat this, we developed a baseline to serve as a reference point for other user testing. Baselines offer several advantages, including providing an understanding of the current state of MAESTRO, identifying potential shortcomings, and establishing a standard for user assessment. Our baseline was created by testing one user with one year of cybersecurity experience.

After creating the baseline, users with varying levels of experience, ranging from very little to over 20 years of cybersecurity experience, will be contacted. The participant skill range will result in a better assessment of MAESTRO's ability to respond to various user prompts. Each tester will assess the same version of the machine allowing for greater consistency in the assistant.

User testing will consist of participants performing an exploit followed by the completion of a questionnaire. To ensure reliable results from the user testing, the exploit and questionnaire will be the same. They will be given little information on the details of the exploit. This will force the user to rely on their own knowledge and MAESTRO. For the first round of user testing, the users will be asked to perform a Metasploit attack on a TryHackMe Virtual Machine (VM), which is an online resource that provides cybersecurity education and access to VMs for hands-on-learning exercises. Their objective is to simulate an initial compromise over the target machine via the Server Message Block protocol. After running through the scenario, each user will answer the survey.

After the users test the program, we ask for feedback, ways to improve the program, and expectations versus program response. The data collected during the user testing questionnaire will focus on three areas: performance, efficiency, and accuracy. The user will provide the time it took to form the exploit and the number of times they relied on assistance or guidance from the assistant.

The user will rate on a scale of one to five the efficiency of the assistant in comparison to surfing the internet for an answer. Third, the assistant's accuracy can be broken into two areas: the assistant’s understanding of the user’s prompt and the assistant’s response. It is crucial to have an assistant that accurately grasps the user’s needs through one’s provided prompt, as this greatly impacts the effectiveness of the tool. Additionally, if the assistant provides incorrect commands or guidance, users can spend more time than desired on a task.

User testing will play a critical role in assessing the accuracy of MAESTRO and guiding necessary improvements. The assistant will be tested by a well-rounded group of cybersecurity operators. Through refining the assistant in response to user feedback, the program will become more human-centric. Developers will work to address and remediate gaps discovered. Furthermore, the data collected will evaluate the accuracy of MAESTRO.

Future Work

Our program is engaged in ongoing research aimed at enhancing its ability to assist operators across each step of the Cyber Kill Chain (CKC). As we continue to develop our program, we have set two major goals. First, we will continue to develop agents that are in line with the CKC. As we create more agents, we will pair them with a purpose-built database optimized for their specific task and data requirements. The new agents will undergo rigorous testing to ensure it provides valuable expertise to operators using the program. For example, we plan to validate the system through additional real-world cyber-attack scenarios. This includes walking through a Metasploit attack with the agents and other specific real-world attacks for each individual agent that will push the program to its limits. Extreme agent testing rolls into our second goal which involves continually evaluating our program. We plan to expand our pool of users to gather a wide range of cyber expertise and receive extensive feedback on various aspects of the program, such as usability and overall performance. By doing so, we can continue to refine and enhance the program, reinforcing its trustworthiness as a resource for cybersecurity professionals of all skill levels. Our program also aims to explore local LLM implementations to enhance data privacy and reduce the reliance on external cloud services. Together these goals are the cornerstone of our strategy to deliver a reliable, secure, trustworthy, and user focused tool for cybersecurity professionals.

Conclusion

In the Forbes article "The Importance of Time and Speed in Cybersecurity," Genes Yoo discussed how cybercriminals scanned the internet and compromised vendors within minutes, while "it took weeks for the vendor to release a patch after detecting that malicious activities were targeting their product globally" (Yoo 2024). Forbes highlights the need for quick action during cyber defense remediation, which also applies to cyber operators. Air Force Lieutenant General Jack Shanahan, director of the Defense Department's Joint Artificial Intelligence Center, affirms this truth by offering one way to increase the speed of operations: Artificial Intelligence (Vergun 2019). As a result of the effective integration of a Large Language Model (LLM), MAESTRO can help close the skill gap between novice and senior-level cyber operators. MAESTRO is being developed to become a reliable resource through a Knowledge-Augmented Generation (KAG) architecture with trusted knowledge sets that can be easily replaced with preferred datasets. Confidence in our LLM's final response to operators has been strengthened through the integration of the most accurate LLM. Additionally, datasets further enhanced and tailored the LLM's response. By developing a system in which accuracy is addressed, we can reliably assist and help increase the number of cyber operations performed. Ultimately, the assistant aims to accurately scale a novice cybersecurity professional’s ability, enabling them to efficiently complete cyber operational tasks.

Bibliography

16x Prompt. “Claude 3.5 Sonnet Vs GPT-4o: Context Window and Token Limit,” 16x Prompt, n.d. https://prompt.16x.engineer/blog/claude-sonnet-gpt4-context-window-token-limit.

Anthropic. “Claude 3.5 Sonnet,” Anthropic, June 20, 2024. https://www.anthropic.com/news/claude-3-5-sonnet.

Context AI. "Llama 3 vs. GPT-4 Comparison." Context AI, 2024. https://context.ai/compare/llama3-8b-instruct-v1/gpt-4-0613.

Council of Economic Advisers. “CEA Report: The Cost of Malicious Cyber Activity to the U.S. Economy.” The White House, February 16, 2018. https://trumpwhitehouse.archives.gov/articles/cea-report-cost-malicious-cyber-activity-u-s-economy/.

Giray, Louie. “Prompt Engineering with ChatGPT: A Guide for Academic Writers.” Annals of Biomedical Engineering, June 7, 2023. https://doi.org/10.1007/s10439-023-03272-4.

Google. “Google/Gemma-2-9b.” Hugging Face, December 13, 2024. https://huggingface.co/google/gemma-2-9b.

Haywood, Sloan, Genevieve Waren and Alex Warren. “Understanding Tokens - .NET.” Microsoft Learn, December 21, 2024. https://learn.microsoft.com/en-us/dotnet/ai/conceptual/understanding-tokens.

He, Junda, Christoph Treude, and David Lo. “LLM-Based Multi-Agent Systems for Software Engineering: Literature Review, Vision and the Road Ahead.” ACM Transactions on Software Engineering and Methodology, January 13, 2025. https://doi.org/10.1145/3712003.

ISC2 “Employers Must Act as Cybersecurity Workforce Growth Stalls and Skills Gaps Widen.” ISC2, September 11, 2024. https://www.isc2.org/Insights/2024/09/Employers-Must-Act-Cybersecurity-Workforce-Growth-Stalls-as-Skills-Gaps-Widen.

Kumar, Vaibhav. “Knowledge Augmented Generation (KAG) by Combining Rag with Knowledge Graphs.” Association of Data Scientists, January 10, 2025. https://adasci.org/knowledge-augmented-generation-kag-by-combining-rag-with-knowledge-graphs/.

LangChain. “Introduction to Langchain.” LangChain, n.d. https://python.langchain.com/docs/introduction/.

LangGraph. “Multi-Agent Systems.” LangGraph, n.d. https://langchain-ai.github.io/langgraph/concepts/multi_agent/.

LLM Explorer. “Dolphin 2.1 Mistral 7B by Cognitivecomputations: Benchmarks and Detailed Analysis. Insights on Dolphin 2.1 Mistral 7B.” LLM Explorer, n.d. https://llm.extractum.io/model/cognitivecomputations%2Fdolphin-2.1-mistral-7b,60Q8GG9DUcdDnxoW5N4kS0.

Lockheed Martin. “Cyber Kill Chain." Lockheed Martin, n.d. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html.

Meta. “Llama-models/Models/Llama3_2/MODEL_CARD.md.” GitHub, n.d. https://github.com/meta-llama/llama-models/blob/main/models/llama3_2/MODEL_CARD.md.

Meta. “Meta-llama/Meta-Llama-3-8B” Hugging Face, December 6, 2024. https://huggingface.co/meta-llama/Meta-Llama-3-8B.

"Mistral 7B." arXiv Preprint, October 2023. https://arxiv.org/pdf/2310.06825.

Multimodal. “LLM Prompting: How to Prompt LLMs for Best Results”. Multimodal, 2024.https://www.multimodal.dev/post/llm-prompting#:~:text=LLM%20prompting%20is%20creating%20specific,provide%20accurate%20and%20relevant%20responses.

NetDocuments. "Maximum Length." NetDocuments Software Inc., June 11, 2024. https://support.netdocuments.com/s/article/Maximum-Length.

Red Hat. “What Is Retrieval-Augmented Generation?” Red Hat, May 15, 2024. https://www.redhat.com/en/topics/ai/what-is-retrieval-augmented-generation.

Vergun, David. “Cyber Ops to Gain Speed, Accuracy from Ai.” U.S. Department of Defense, September 5, 2019. https://www.defense.gov/News/News-Stories/Article/Article/1953183/cyber-ops-to-gain-speed-accuracy-from-ai/.

Yoo, Gene. “Council Post: The Importance of Time and Speed in Cybersecurity.” Forbes, August 12, 2024. https://www.forbes.com/councils/forbestechcouncil/2021/01/22/the-importance-of-time-and-speed-in-cybersecurity/.