

Leveraging Generative Artificial Intelligence for Memory Analysis

Author: William Copeland, wcopeland1981@hotmail.com

Advisor: Tim Proffitt

Accepted: May 28, 2024

Abstract

The increasing sophistication of malware poses significant challenges for traditional memory analysis techniques in digital forensics. This research explores the potential of leveraging Generative Artificial Intelligence (AI) models, specifically OpenAI's GPT-4 Turbo and Anthropic's Claude 3 Opus, to enhance malware detection in memory. By combining the data extraction capabilities of the Volatility Framework with the predictive power of Generative AI, this study aims to develop an innovative approach for accurately identifying and classifying malicious activities in memory dumps. The research methodology involves collecting a diverse set of memory dump samples, preprocessing the data using Volatility plugins, and evaluating the performance of the AI models using quantitative metrics.

The findings highlight the potential of Generative AI models in effectively identifying malware, while also revealing limitations and areas for improvement. The implications suggest that Generative AI models can serve as valuable complementary tools alongside traditional malware detection methods, and future research recommendations include expanding datasets, developing domain-specific models, and integrating Generative AI capabilities into existing memory forensics workflows. This study lays the foundation for further exploration and advancement of Generative AI models in-memory analysis and malware detection.

1. Introduction

In digital forensics, memory analysis is crucial in uncovering valuable evidence and insights from computer systems. Memory dumps, snapshots of a computer's volatile memory at a specific time, contain a wealth of information about running processes, network connections, and system activities. However, malware's increasing sophistication and complexity pose significant challenges for traditional memory analysis techniques. Research has shown that malware has reached a level of sophistication, making detection and analysis extremely difficult for forensic investigators and incident responders due to malware authors employing various obfuscation and evasion techniques (Kolbitsch, et al., 2009).

Researchers and practitioners have turned to advanced tools and frameworks to address these challenges, such as the Volatility Framework. The Volatility Framework is a widely used open-source memory forensics tool that provides a collection of plugins for extracting and analyzing data from memory samples from different operation systems (The Volatility Framework, 2024). However, the effectiveness of memory analysis heavily relies on the ability to accurately identify and classify malicious patterns and behaviors in the extracted data.

The Generative Artificial Intelligence (AI) field has witnessed significant advancements in recent years, opening new data analysis and pattern recognition possibilities. Modern Generative AI models include OpenAI's GPT-4 Turbo and Anthropic's Claude 3 Opus. GPT-4 Turbo is a large multimodal model with a larger context window of 128,000 tokens, which accepts text or images as inputs to produce a text output. With broader general knowledge and advanced reasoning capabilities, the model can more accurately solve complex problems (Models, 2024). Claude 3 Opus is Anthropic's most powerful multimodal model, which delivers advanced performance and demonstrates a human-like understanding of text (Models Overview, 2024). With an extended context window of 200,000 tokens, the model can process large amounts of data to complete highly complex tasks (Long Context Window Tips, 2024). To utilize these Generative AI capabilities, a paid subscription is required to interact with the model via web interface or access to the model via Application Programming Interface (API).

Overall, both models are promising candidates for application in various domains, including cybersecurity and digital forensics.

This research explores the potential of leveraging Generative AI models for enhanced malware detection in memory. By combining the data extraction capabilities of the Volatility Framework with the predictive power of Generative AI, an innovative approach will be developed that can accurately identify and classify malicious activities in memory dumps, ultimately strengthening the capabilities of forensic investigators in combating sophisticated cyber threats.

2. Research Method

2.1. Quantitative Analysis Method

This research will utilize the accuracy and error rates method to evaluate the performance of the Generative AI models, specifically OpenAI's GPT-4 Turbo and Anthropic's Claude 3 Opus, in identifying malware with memory dumps. This quantitative analysis method provides a straightforward approach to measuring the overall proportion of samples correctly classified by the model, which aids in identifying areas of improvement for the model. Key metrics are calculated by comparing the models' predictions with the ground truth labels, such as accuracy and error rates, which consist of the False Positive Rate (FPR) and False Negative Rate (FNR), to assess the effectiveness of malware detection. Accuracy, as seen in Figure 1, is the measurement of the overall correctness of the models in identifying malicious and benign memory dumps.

$$\text{Accuracy} = (\text{True Positive (TP)} + \text{True Negative (TN)}) / (\text{Total Instances})$$

Figure 1: Accuracy Calculation

FPR, as seen in Figure 2, calculates the proportion of benign samples incorrectly classified as malicious.

$$\text{FPR} = \text{False Positive (FP)} / (\text{False Positive (FP)} + \text{True Negative (TN)})$$

Figure 2: False Positive Rate Calculation

FNR, as seen in Figure 3, calculates the proportion of malicious samples incorrectly classified as benign.

$$\text{FNR} = \text{False Negative (FN)} / (\text{False Negative (FN)} + \text{True Positive (TP)})$$

Figure 3: False Negative Rate Calculation

These metrics provide a means to evaluate the models' performance and help identify their strengths and weaknesses in the context of memory forensics.

2.2. Data Collection

The data collection phase involved acquiring diverse memory dump samples and categorizing them as malicious, benign, or unknown. To ensure the reliability and representativeness of the dataset, memory dumps from various sources, such as publicly available datasets, malware repositories, and real-world incidents, will be collected.

The Volatility Framework will extract valuable information from a system during memory acquisition, such as processes, load modules, handles, and network connections from each memory dump sample (The Volatility Framework, 2024). The Volatility plugin's output creates a rich set of artifacts and data points to serve as input for the Generative AI models. Three artifact categories of Volatility plugins were utilized during this research: processes, network connections, and suspicious activity. Figure 4 outlines the plugins for each category.

Volatility Framework Plugins		
Processes	Network Connections	Suspicious Activity
pslist	netscan	malfind
psscan		apihooks
cmdline		timeliner
dlllist		
ldrmodules		
handles		

Figure 4: Volatility Plugins

These plugins were chosen based on their relevance to memory analysis and their ability to provide valuable insights into potential malicious activities.

2.3. Data Processing

Several preprocessing steps must be performed to prepare the collected data from the memory dump. First, the output from the Volatility plugins must be cleaned, aggregated, and converted into a structured format, specifically JavaScript Object Notation (JSON). This standardized format will facilitate the data ingestion into the models and ensure consistency across different memory dump samples.

Next, a label is appended to the preprocessed data with the sample name, operating system, and the ground truth information for each memory dump. The ground truth labels will classify each sample as malicious or benign, providing a reliable reference for evaluating the performance of the models. Figure 5 is the ground truth table utilized for the samples.

	Malicious	Benign
Sample-1	True	False
Sample-2	True	False
Sample-3	False	True
Sample-4	True	False
Sample-5	True	False
Sample-6	False	True

Figure 5: Ground Truth Table

Furthermore, developing an appropriate system prompt is crucial for effectively utilizing the models. The prompt is designed to elicit relevant information from the models based on the preprocessed data, enabling the model to make a prediction and provide meaningful insights. Figure 6 is the system prompt utilized during the research.

You are an incident responder analyzing a memory dump from a potentially compromised Windows system using the Volatility framework. You will be provided with the consolidated output of the Volatility plugins run against the memory dump. Your task is to carefully analyze the provided Volatility plugin output to look for evidence that is of malware on the system.

Use the step-by-step instructions below to build a response to the user's input.

Step 1 - The user will provide you with data. Provide the following information:

<name> Provide the sample file name.

<operating system> Provide the operating system identified in the file.

<key> Provide the key to the label key pair in the data.

<answer> Inform the user if the dump is Malicious, Benign, or Unknown. Only provide one answer.

Sample Name: <name>

Operating System: <operating system>

Truth Label: <key>

Prediction: <answer>

Step 2 - Write out your analysis and reasoning based on the Volatility output. Cite specific lines from the Volatility output to justify your reasoning where applicable.

Figure 6: System Prompt

2.4. Model Evaluation

Evaluating the models requires a system prompt and ingesting the preprocessed data into the models to assess their performance using quantitative metrics. The OpenAI's GPT-4 Turbo and Anthropic's Claude 3 Opus models were employed for this purpose, leveraging their advanced data analysis capabilities to analyze the memory dump data.

The preprocessed data was ingested into OpenAI's GPT-4 Turbo and Anthropic's Claude 3 Opus models using their respective developer platforms, OpenAI's Playground (Playground, 2024) and Anthropic's Workbench (Workbench, 2024). These web-based interfaces provide a user-friendly environment for developers to submit data and interact with the models through their APIs. The platforms allow for configuring model parameters, such as temperature and token limit, and provide a means to input the system prompt and data. Upon submission, the models process the input and generate a response containing their analysis and classification of the memory dump samples. The model's

classifications are then compared against the ground truth labels associated with each memory dump sample to evaluate the models' performance and calculate the accuracy, FPR, and FNR metrics. By comparing the evaluation metrics between the models, the research will show insights that can be gleaned into the relative strengths and weaknesses.

In summary, the research method outlined in this section provides an approach to evaluating the effectiveness of each model. It aims to contribute to the advancement of integrating Generative AI into forensics techniques to combat evolving malware threats.

3. Findings and Discussion

3.1. Analysis of Quantitative Results

Before diving into the quantitative analysis of the model's performance, it is necessary to understand the nature of the dataset fed into the models. The input dataset consisted of extracted artifacts (running processes, network connections, and suspicious memory regions were extracted) from memory samples. These samples encompassed a variety of malware families, such as ransomware, trojans, and rootkits, as well as benign memory dumps from clean systems. Including malicious and benign samples was essential to accurately assess the models' ability to distinguish between the two classes.

It is important to note that the dataset used in this analysis was limited to six samples but contained very complex data. The dataset represented various scenarios, encompassing operating systems, system configurations, and malware behaviors. The complexity of the dataset posed a significant challenge for the models, as they needed to identify subtle patterns and indicators of compromise amidst a vast amount of memory data. By feeding the models with such a comprehensive and challenging dataset, the aim was to assess their ability to generalize and accurately detect malware in memory dumps.

The quantitative analysis of the models' performance yielded meaningful insights. The truth table and model predictions for each sample were used to calculate the accuracy, FPR, and FNR for both the GPT-4 Turbo and Claude 3 Opus models, refer to Appendix A – F. Figure 7, Accuracy and Error Rates, consolidates and provides the metrics for both models.

Open AI GPT-4 Turbo		Anthropic Claud3 Opus	
True Positives (TP)	4	True Positives (TP)	6
True Negatives (TN)	0	True Negatives (TN)	0
False Positives (FP)	2	False Positives (FP)	0
False Negatives (FN)	0	False Negatives (FN)	0
Total Samples	6	Total Samples	6

Accuracy	0.6666667	Accuracy	1
False Positive Rate	1	False Positive Rate	0
False Negative Rate	0	False Negative Rate	0

Figure 7: Accuracy and Error Rates

The accuracy, FPR, and FNR values range from 0 to 1, providing a standardized metric for evaluating the models' performance. Accuracy represents the overall correctness of the models in classifying memory dumps as malicious or benign. It is calculated by dividing the sum of true positives (correctly identified malicious samples) and true negatives (correctly identified benign samples) by the total number of samples. An accuracy closer to 1 is an indication of better overall performance.

In this analysis, the GPT-4 Turbo model achieved an accuracy of 0.6666667, which means it correctly classified approximately 66.67% of the samples. On the other hand, the Claude 3 Opus model achieved an accuracy of 1, indicating that it correctly classified all the samples in the dataset.

The False Positive Rate (FPR) represents the proportion of benign samples incorrectly classified as malicious. It is calculated by dividing the number of false positives (benign samples incorrectly identified as malicious) by the sum of false positives and true negatives (correctly identified benign samples). An FPR closer to 0 is desirable, as it indicates a lower rate of false alarms. The GPT-4 Turbo model demonstrated an FPR of 1, incorrectly classifying all benign samples as malicious. This high FPR suggests that the GPT-4 Turbo model tends to generate false positives, potentially leading to unnecessary investigation efforts and resource allocation. In contrast, the Claude 3 Opus model achieved an FPR of 0, indicating that it did not misclassify any benign samples as malicious, which is ideal for reducing false alarms.

The False Negative Rate (FNR) represents the proportion of malicious samples incorrectly classified as benign. It is calculated by dividing the number of false negatives (malicious samples incorrectly identified as benign) by the sum of false negatives and true positives (correctly identified malicious samples). An FNR closer to 0 is preferred, as it indicates a lower rate of missed malware. The GPT-4 Turbo and Claude 3 Opus models demonstrated an FNR of 0, indicating that they correctly identified all the malicious samples in the dataset. This perfect FNR suggests that both models can detect malware without missing any malicious instances.

However, it is crucial to consider the limitations of the dataset used in this analysis. The dataset consists of a limited number of samples, and it is vital to evaluate the models' performance on a more extensive and diverse dataset to assess their generalization capability. Additionally, the model's ability to detect unseen malware samples in real-world scenarios should be validated to ensure their effectiveness in categorizing sophisticated and evolving malware.

Refer to Appendix A- F to review the complete response from GPT-4 Turbo and Claude 3 Opus models for the sample data submitted.

3.2. Limitations

Despite the promising results, it is vital to acknowledge the limitations of the current research. One limitation is the size and diversity of the dataset used for evaluation. While efforts were made to collect a representative sample of memory dumps, the dataset does not encompass all possible variations and emerging malware techniques. Future research should expand the dataset and include a more comprehensive range of malware families and system configurations.

Another limitation was the constraint posed by the context window size of the GPT-4 Turbo and Claude 3 Opus models. The Claude 3 Opus model has a significantly larger context window, capable of processing up to 72,000 more tokens than the GPT-4 Turbo model. This difference in token capacity presented challenges when submitting the data generated by the Volatility plugins. Due to the extensive amount of data produced, the token limit of the model was exceeded. An iterative process was employed to

reconfigure the plugin selection to ensure that the data generated by the Volatility plugins did not exceed the token limit of the models. The objective was to find an optimal balance between collecting a comprehensive set of artifacts and reducing the overall data size. The process involved exchanging plugins multiple times and evaluating the impact on the token count. After each reconfiguration, the resulting data was submitted to OpenAI's Tokenizer (Tokenizer, 2024) and Hugging Faces Tokenizer tool (Tokenizer Arena, 2024), which calculated the token size of the data. This iterative process was repeated until the data for each memory dump sample was within the 128,000 tokens, ensuring the data did not exceed the context window for both models. As a result of this optimization, the initial set of ten plugins had to be narrowed down to a subset of six plugins. Figure 8 presents the refined list of plugins used in the final analysis.

Volatility Framework Plugins		
Processes	Network Connections	Suspicious Activity
pslist		malfind
psscan		apihooks
cmdline		

Figure 8: Volatility Plugins

This reduction allowed for the successful submission of the data to both models without exceeding their token capacities. However, it also meant that some potentially valuable information from the excluded plugins could not be incorporated into the analysis. This limitation highlights the trade-offs when working with AI models with different context window sizes.

Moreover, the models used in this research, GPT-4 Turbo and Claude 3 Opus, have inherent limitations. These models are based on natural language processing and may not be designed explicitly for malware detection in memory dumps. Further, fine-tuning and adapting the models to the specific domain of memory forensics could enhance their performance.

3.3. Areas of Improvement

Several areas for improvement can be identified based on the findings and limitations discussed. First, the preprocessing techniques applied to the memory dump data could be further refined. Exploring techniques to efficiently compress or summarize

the data generated by the Volatility plugins would enable the inclusion of more comprehensive information within the token limits of the models.

Secondly, the prompts used to interact with the models can be optimized. Developing more targeted and context-specific prompts could improve the models' ability to identify malicious patterns accurately. Prompt development is crucial in getting the most out of Generative AI models. Researchers can guide the models by carefully designing prompts to produce more accurate and relevant responses for malware detection and other tasks.

When optimizing prompts, several key factors should be considered. The prompt should have a clear objective, explicitly stating what the model should achieve (Prompt Engineering, 2024). Providing specific details or parameters relevant to the task can further guide the model's response. Using concise and unambiguous language reduces confusion and improves the model's understanding of the task. Including relevant context, framing the prompt as a question or directive, and specifying the role or persona the model should adopt can also help refine the response style and depth according to the assumed expertise (Prompt Engineering, 2024). Breaking complex tasks into clear, manageable steps and providing examples can lead to more structured and coherent responses (Prompt Engineering, 2024).

Prompt development is an iterative process that requires experimentation and refinement. Researchers should continuously evaluate the model's responses and adjust the prompts accordingly. Well-crafted prompts can significantly enhance the models' ability to identify malicious patterns accurately in the context of malware detection. By carefully designing prompts that align with each task's specific objectives and requirements, researchers and practitioners can harness the power of these models to solve complex problems and generate valuable insights.

Additionally, incorporating domain-specific knowledge and rules into the models could enhance their performance. By integrating expert knowledge and heuristics from

memory forensics, the models can be guided toward a more accurate and context-aware prediction.

Furthermore, exploring ensemble techniques that combine Generative AI models or integrating them with traditional malware detection methods could provide a more robust approach to identifying malware in memory dumps.

3.4. Reflection on the Research Objective

The research objectives set out at the beginning have been successfully addressed. The quantitative analysis method, utilizing accuracy and error rates, provided a means to evaluate Generative AI models' performance in detecting malware in memory dumps.

The data collection and preprocessing steps, leveraging the Volatility Framework and selected plugins, enabled the extraction of artifacts from memory dumps. The preprocessed data was input for the Generative AI models, facilitating their evaluation and analysis.

The findings and discussion section provided in-depth insights into the models' performance, limitations, and areas for improvement. The implications of the findings for enhancing malware detection in memory were explored, highlighting the potential benefits and challenges associated with integrating Generative AI models in memory forensics.

4. Implications and Recommendations

4.1. Implications of the Findings

The findings of this research have significant implications for improving malware detection in memory. The generative AI models' high accuracy and relatively low error rates demonstrate their potential as a valuable tool for memory forensics practitioners.

By leveraging the power of Generative AI, investigators can automate and accelerate the process of analyzing memory dumps, enabling them to identify potential malware infections quickly. The experimental results in this study show that the Claude 3 Opus model achieved a perfect accuracy score of 1 and an FNR of 0, indicating its ability to correctly identify all malicious samples without missing any (Figure 7). This level of

accuracy can considerably reduce the time and energy required for manual analysis, allowing for more efficient and effective incident response. Research has shown that AI-based technologies leveraged in digital forensic investigations can drastically save the time needed to evaluate and uncover potential security breaches (Fakiha, 2023). Moreover, the ability of generative AI models to learn and adapt to new malware patterns and techniques opens possibilities for proactive threat detection. As new malware variants emerge, these models can be continuously trained and updated to detect evolving threats, enhancing organizations' overall security posture. Research suggests that organizations must employ continuous learning and adaptive models to keep pace with the ever-evolving landscape of malware threats (Sindiramutty, 2023). However, it is crucial to consider the limitations and potential risks associated with relying solely on Generative AI for malware detection. False positives and negatives can have significant effects, such as wasted resources on benign samples or overlooking threats. In this study, the GPT-4 Turbo model's high FPR of 1.0 (Figure 7) highlights the need for caution and further refinement. Therefore, it is recommended that Generative AI models be used as a complementary tool to work alongside traditional malware detection methods and human analysis.

4.2. Recommendations For Future Research

Based on the outcomes of this study, several recommendations for future research can be made to enhance the effectiveness and applicability of Generative AI models in memory forensics. These recommendations focus on addressing the limitations identified in the current study and exploring new avenues to further advance the field of AI-assisted memory forensics.

4.2.1. Expanded Dataset

One of the critical limitations of the current study is the limited size of the dataset, which consisted of only six memory dump samples. While this dataset allowed for an initial evaluation of the Generative AI models' performance, future research should aim to collect a more extensive and diverse dataset to enhance the model's generalization ability and robustness. Expanding the dataset to include hundreds or thousands of memory dump samples from various sources would provide a more comprehensive

representation of real-world scenarios. By curating a dataset encompassing a wide range of malware families and benign samples from different operating systems and configurations, future research can assess the models' performance in more realistic and diverse settings.

In addition to expanding the dataset size, future research should also consider the quality and labeling of the memory dump samples. Ensuring that the dataset is accurately labeled and includes detailed metadata about the malware types, system configurations, and collection methods can significantly facilitate the training and evaluation of Generative AI models. Collaboration with malware experts and leveraging established malware repositories can help ensure the reliability and representativeness of the dataset.

4.2.2. Develop Domain-Specific Models

Another recommendation is to develop domain-specific Generative AI models tailored specifically for the memory forensics domain. While general-purpose language models have shown promise in this study, their performance can be further enhanced by incorporating domain-specific knowledge and features. Developing domain-specific models involves fine-tuning the pre-trained language models on a large corpus of memory forensics data, such as memory dump samples, analysis reports, and relevant literature. By exposing the models to the specific terminology, data structures, and patterns prevalent in memory forensics, researchers can create specialized models that capture the intricacies and nuances of this domain.

4.2.3. Integrate With Existing Tools and Workflows

Future To maximize the impact of Generative AI models in forensics, future research should focus on seamlessly integrating these models into existing tools and workflows practitioners use. Integrating these models will require careful consideration of the input and output formats, data preprocessing requirements, and user experience. However, integrating these models into existing tools poses challenges related to compatibility, performance trade-offs, and the need for standardized data formats and APIs.

Developing user-friendly interfaces and APIs should enable practitioners to incorporate models into their workflows easily. This may involve creating plugins or extensions for popular forensics tools that provide access to the models' capabilities. By removing the complexities of the underlying AI models, these interfaces can empower practitioners to leverage the benefits of Generative AI without requiring extensive knowledge of machine learning or natural language processing.

Furthermore, research needs to be conducted on how integrating models can optimize the automation of data preprocessing and analysis pipelines. Automating tasks such as data extraction, formatting, and normalization through Generative AI integration can significantly streamline the forensics process, reduce manual effort, and improve the consistency and reliability of the analysis.

Lastly, researchers must understand the risks and limitations of relying on the models once integrated into forensic workflows. While Generative AI models offer promising capabilities for enhancing forensics investigations, it is crucial to recognize and address the potential challenges and pitfalls that may arise from their integration into existing tools and processes.

One significant risk is the potential for biased or misleading results due to inherent biases in training data, which could lead to wrongful accusations or misinterpretation of the data. These biases can derive from various sources, including historical inequalities, human prejudices, or unrepresentative data samples. Ongoing research and collaboration among researchers, tool developers, and forensics practitioners are essential to mitigate the risk. Researchers must actively work to implement robust measures to identify, understand, and address biases in training data. This includes employing diverse and representative datasets and continuously monitoring and evaluating the model.

While integrating Generative AI models into forensics workflows holds immense potential, researchers must remain vigilant and proactively address the risks and limitations to ensure these powerful tools are responsible and effective use.

5. Conclusion

The increasing sophistication and complexity of malware pose significant challenges for traditional memory analysis techniques in digital forensics. This research addressed this problem by investigating the effectiveness of Generative AI models detecting malware in memory dumps. Through quantitative analysis, the study evaluated the performance of models using accuracy and error rates. The key findings of this research highlighted the potential of Generative AI models in effectively identifying malware in memory dumps, with both models achieving high levels of accuracy. However, the study also revealed the presence of false positives, emphasizing the importance of using Generative AI as a complementary tool alongside traditional methods.

Based on the recommendations and implications discussed, future research must focus on expanding datasets, developing domain-specific models, and integrating with existing tools and workflows. By addressing these areas, practitioners and researchers can leverage the power of Generative AI to enhance their organization's security posture and combat evolving malware threats more effectively.

References

- Fakiha, B. (2023, August 4). Enhancing Cyber Forensics with AI and Machine Learning. *Enhancing Cyber Forensics with AI and Machine Learning: A Study on Automated Threat Analysis and Classification*, 13(4), 701-707.
- Kolbitsch, C., Comparetti, P. M., Kruegel, C., Kirda, E., Zhou, X., & Wang, X. (2009). *Effective and efficient malware detection at the end host*. Retrieved 4 13, 2024, from https://usenix.org/legacy/event/sec09/tech/full_papers/sec09_malware.pdf
- Long Context Window Tips*. (2024, April 05). Retrieved from Anthropic: <https://docs.anthropic.com/clause/docs/long-context-window-tips>
- Models*. (2024, April 04). Retrieved from OpenAI: <https://platform.openai.com/docs/models/continuous-model-upgrades>
- Models Overview*. (2024, April 04). Retrieved from Anthropic: <https://docs.anthropic.com/clause/docs/models-overview>
- Playground*. (2024, April 04). Retrieved from OpenAI: <https://platform.openai.com/playground>
- Prompt Engineering*. (2024, April 05). Retrieved from OpenAI: <https://platform.openai.com/docs/guides/prompt-engineering>
- Sindiramutty, S. R. (2023, March 30). Autonomous Threat Hunting: A Future Paradigm for AI-Driven Threat Intelligence.
- The Volatility Framework*. (2024, April 04). Retrieved from Volatility Foundation: <https://volatilityfoundation.org/the-volatility-framework/>
- Tokenizer*. (2024, April 04). Retrieved from Open AI: <https://platform.openai.com/tokenizer>
- Tokenizer Arena*. (2024, April 04). Retrieved from Hugging Face: https://huggingface.co/spaces/Cognitive-Lab/Tokenizer_Arena
- Workbench*. (2024, April 04). Retrieved from Anthropic Console: <https://console.anthropic.com/workbench>

Appendix A

Sample 1

ASSISTANT

Sample Name: Sample-1
Operating System: Windows XP SP2
Truth Label: 1
Prediction: Malicious

Analysis and Reasoning:
The memory dump analysis reveals multiple indicators of malicious activity:

- 1. Suspicious Processes and Anomalies:**
- The process `a[1].php` with PID 2280, initiated by `cmd.exe`, is highly unusual as PHP files are typically not executed directly as processes in Windows environments. This is a strong indicator of a script being used as part of an attack or malicious activity.
- 2. API Hooking and Trampolining:**
- Multiple instances of API hooking and trampolining were detected, particularly in critical system processes and libraries such as `kernel32.dll`, `USER32.dll`, `OLEAUT32.dll`, and `ntdll.dll`. These hooks can be used by malware to intercept system calls and manipulate or snoop on data, indicative of rootkit or trojan behavior.
- 3. Memory Protections and Allocations:**
- The `malfind` plugin output shows several processes with memory regions having `PAGE_EXECUTE_READWRITE` permissions, which is unusual and typically indicative of code injection or execution of non-standard code segments.
- 4. Network Connections:**
- The `connscan` results show network connections to suspicious IP addresses and unusual ports, which could be related to command and control servers or data exfiltration activities.
- 5. File and Process Anomalies:**
- The presence of `IEXPLORE.EXE` with suspicious command-line URLs and unusual child processes like `a[1].php` suggests browser exploitation or redirection to malicious sites.
- 6. Import Address Table (IAT) Hooking:**
- The `apihooks` output indicates IAT hooking in `IEXPLORE.EXE` for functions like `LoadLibraryA`, `LoadLibraryW`, and others across various DLLs. This technique is often used by malware to intercept and manipulate the behavior of standard library calls.

Given these findings, the memory dump is classified as malicious, with evidence pointing towards the presence of malware that employs rootkit-like behaviors, code injection, API hooking, and potentially unwanted network communications.

Figure A1. OpenAI Playground Output

RESPONSE

Operating System: Windows XP SP2
Truth Label: 1
Prediction: Malicious

Analysis:
Based on the provided Volatility plugin output, there are several indicators that suggest the memory dump is from a system compromised with malware:

1. The malfind plugin output shows multiple instances of suspicious memory regions with RWX (Read, Write, Execute) permissions across processes like firefox.exe, thunderbird.exe, and IEXPLORE.EXE. Legitimate code rarely requires RWX permissions. This is often indicative of malicious code injection.
2. The apihooks plugin reveals numerous inline hooking of functions across system DLLs like ntdll.dll, USER32.dll, OLEAUT32.dll, etc. Extensive inline hooking, especially of low-level Windows API functions, is a common malware tactic to intercept and modify program behavior. The hooks are originating from modules like mozglue.dll and xul.dll which seem suspicious.
3. The apihooks output also shows hooks on sensitive functions like LdrLoadDll, NtCreateFile, etc. Hooking these functions allows malware to intercept and potentially interfere with module loading and file creation.
4. The plist output has a suspicious process named "a[1].php" (PID 2280) spawned as a child of cmd.exe (PID 3812). Executable PHP files are very uncommon and it being launched via cmd.exe is suspicious.
5. The cmdline for the cmd.exe process (PID 3812) shows it launching the suspicious "a[1].php" providing it an "INSTALL" argument. This looks like a malware installation command.
6. Legitimate Windows processes like AcroRd32.exe and IEXPLORE.EXE have network connections to suspicious IP addresses based on the connscan output. For example, IEXPLORE.EXE (PID 2276) has a connection to a.doginhispen.com which looks like a malicious domain.

In summary, the combination of RWX memory regions, extensive API hooking, suspicious process names and command-lines, and network connections to likely malicious domains strongly suggests this memory dump is from a system infected with malware. The malware seems to have hooked multiple processes and is likely capable of stealing data, downloading additional payloads, and performing other malicious actions.

Figure A2. Claudio 3 Workbench Output

Appendix B

Sample 2

ASSISTANT

Sample Name: Sample-2
 Operating System: Windows 10
 Truth Label: 1
 Prediction: Malicious

Analysis and Reasoning:

1. **Malfind Plugin Output**:**
 - The `malfind` plugin output shows multiple instances of processes with memory regions having `PAGE_EXECUTE_READWRITE` permissions, which is unusual and indicative of code injection or execution of non-standard memory pages. For example:
 - `MsMpEng.exe` (PID 2620) has several entries with `PAGE_EXECUTE_READWRITE` permissions, which is suspicious for a system process associated with Windows Defender.
 - `powershell.exe` (PID 8448) also shows similar characteristics, which is concerning given PowerShell's common use in scripting and automation, potentially for malicious purposes.
 - `SearchUI.exe` and `smartscreen.exe` also exhibit similar suspicious memory patterns.
2. **Netscan Plugin Output**:**
 - The `netscan` output shows network connections from `svchost.exe` and `powershell.exe` to external IP addresses on common HTTP and HTTPS ports. Notably, `powershell.exe` has connections to IP addresses that are not typically associated with benign behavior, especially given the context of other suspicious indicators.
3. **Cmdline Plugin Output**:**
 - The command line parameters for `powershell.exe` do not show any immediate red flags, but the presence of PowerShell in conjunction with the other findings raises the level of suspicion.
4. **General Observations**:**
 - The presence of `PAGE_EXECUTE_READWRITE` permissions in multiple processes, combined with external network connections from system processes to potentially harmful IP addresses, strongly suggests that the system is compromised. The use of PowerShell, a powerful tool capable of executing scripts and commands, further aligns with common tactics employed by attackers.

Given these observations, the memory dump analysis strongly points towards a malicious compromise of the system, likely involving code injection and unauthorized external communications.

Figure B1. OpenAI Playground Output

RESPONSE

Markdown Text

Sample Name: Sample-2
 Operating System: Windows 10
 Truth Label: 1
 Prediction: Malicious

Analysis:

Based on the Volatility plugin output provided, there are several indicators that suggest this memory dump is from a system infected with malware:

1. The malfind plugin output shows multiple processes with suspicious memory regions marked RWX (Read, Write, Execute), which is often a sign of injected malicious code. Processes like MsMpEng.exe, SearchUI.exe, smartscreen.exe, OneDrive.exe, powershell.exe, and Skype4Life.exe all have RWX memory regions with no clear legitimate purpose.
2. The plist and psscan plugins show a suspicious process named "jcdnrhug.tmp" (PID 4016) as a child process of svchost.exe (PID 4756). The random-looking name and spawning under svchost are indicators of potential malware.
3. The netscan plugin reveals multiple established TCP connections from processes like svchost.exe and Skype4Life.exe to various IP addresses on ports 80 and 443. While not necessarily malicious on its own, when combined with the other suspicious artifacts, this could indicate command and control or data exfiltration activity.
4. The cmdline plugin shows a PowerShell process (PID 8448) launched with no command line arguments, which is sometimes an indicator of malicious PowerShell activity.
5. Legitimate security processes like MsMpEng.exe (Windows Defender) and smartscreen.exe having RWX memory regions suggests potential tampering by malware to disable defenses.

In summary, the combination of the suspicious jcdnrhug.tmp process, RWX memory regions in multiple processes, unexpected network connections, and signs of PowerShell and security product tampering leads me to conclude this memory dump is from a system infected with malware. The truth label of "1" (malicious) matches my analysis.

Figure B2. Anthropic Workbench Output

Appendix C

Sample 3

ASSISTANT

Sample Name: Sample-3
 Operating System: Windows 11
 Truth Label: 0
 Prediction: Malicious

Analysis and Reasoning:

- Malfind Analysis:**
 - The Malfind plugin output shows multiple instances of suspicious memory regions with `PAGE_EXECUTE_READWRITE` permissions, which is unusual and indicative of code injection or unpacking of executable code in memory. For example:
 - PID 3696 (MsMpEng.exe) has several entries with disassembled code showing repeated `int3` instructions and other assembly snippets that are typical in shellcode or packed executables.
- Netscan Analysis:**
 - The Netscan output does not show direct signs of C2 communications or unusual network activity. However, the presence of multiple listening ports across various processes like `lsass.exe`, `svchost.exe`, and others could be benign but warrants further investigation given the other findings.
- PsList and PsScan Analysis:**
 - Both PsList and PsScan show processes with expected parent-child relationships and no hidden processes, which does not directly indicate malicious activity. However, the presence of `MsMpEng.exe` (Microsoft Defender) with suspicious memory sections could indicate that the malware is attempting to tamper with or bypass antivirus components.
- CmddLine Analysis:**
 - The CmdLine plugin output does not show any immediate red flags in the command-line arguments of processes. However, this does not rule out the possibility of malware that does not rely on suspicious command-line arguments to operate.

Given the presence of executable and writable memory pages with patterns indicative of shellcode or unpacked binary execution, particularly in critical system processes like `MsMpEng.exe`, the memory dump is classified as Malicious. This assessment is based on the analysis of memory artifacts and the typical behavior of malware trying to execute code directly in memory to evade detection. Further binary analysis and correlation with known malware signatures would be necessary to identify the exact nature and intent of the malicious code.

Figure C1. OpenAI Playground Output

RESPONSE

Sample Name: Sample-3
 Operating System: Windows 11
 Truth Label: 0
 Prediction: Benign

Based on the provided Volatility plugin output, there are no clear signs of malware on this Windows 11 system memory dump. Here is the analysis:

1. The pslist and psscan output shows expected system processes like System, Registry, smss.exe, csrss.exe, wininit.exe, services.exe, lsass.exe, svchost.exe, etc. No suspicious processes stand out.
2. The netscan output reveals common Windows ports in LISTENING state like 135, 139, 445, 3389 (RDP), 49664-49669 (RPC), etc. owned by expected processes like svchost.exe, System, lsass.exe, spoolsv.exe. No suspicious network connections observed.
3. The malfind plugin found a few small allocations with PAGE_EXECUTE_READWRITE protection in legitimate processes like MsMpEng.exe (Windows Defender), SearchHost.exe, WWAHost.exe, WaAppAgent.exe, OneDrive.exe. These are likely false positives and not malicious.
4. The cmdline output shows command lines for running processes which look normal, related to the Windows OS and installed applications like Edge browser, Windows Defender, Azure agent, etc. No suspicious process command lines found.

In summary, the Volatility analysis does not indicate the presence of malware on this Windows 11 system based on the provided output. The processes, network connections, memory allocations and command lines all appear to be benign and related to normal Windows operation and installed legitimate applications. No further signs of malicious activity were identified in this memory dump.

Figure C2. Anthropic Workbench Output

Appendix D

Sample 4

ASSISTANT

Sample Name: Sample-4
 Operating System: Windows 7 SP1
 Truth Label: 1
 Prediction: Malicious

Analysis and Reasoning:
 The memory dump analysis reveals several indicators of malicious activity:

1. **Malfind Plugin Output**:
 - The `malfind` plugin output shows suspicious memory regions in processes `explorer.exe` and `TrueCrypt.exe` with `PAGE_EXECUTE_READWRITE` permissions, which is unusual and indicative of code injection techniques commonly used by malware to execute arbitrary code.
2. **API Hooks**:
 - The `apihooks` plugin output lists multiple hooks in critical system functions within `explorer.exe` and `PGPdesk.exe`. These hooks redirect function calls to potentially malicious code, which is a common technique used by malware to intercept and alter the behavior of system functions for data theft, spying, or persistence.
3. **Unusual Network Activity**:
 - The `netscan` plugin output shows unusual network activity, such as listening on uncommon ports and connections to non-standard local addresses, which could be related to command and control communication or data exfiltration.
4. **Suspicious Processes and Modules**:
 - The presence of `RDDService.exe`, `PGPtray.exe`, and `TrueCrypt.exe` running from unusual paths or with suspicious command-line arguments could indicate that these processes are masquerading as legitimate software to hide malicious activities.
5. **Executable Pages**:
 - The presence of executable pages with `PAGE_EXECUTE_READWRITE` permissions in processes typically not involved in such activities further supports the hypothesis of malicious code injection.

Given these findings, the memory dump is classified as malicious. The combination of code injection, API hooking, unusual network activities, and the presence of suspicious processes strongly suggests that the system is compromised.

Figure D1. OpenAI Playground Output

RESPONSE

Markdown Text

Sample Name: Sample-4
 Operating System: Windows 7 SP1
 Truth Label: 1
 Prediction: Malicious

Analysis:
 Based on the provided Volatility plugin output, there are several indicators that suggest this memory dump is from a system compromised with malware:

1. The malfind plugin output shows that the explorer.exe (PID 1812) and TrueCrypt.exe (PID 2852) processes have suspicious memory regions with RWX (Read, Write, Execute) permissions. This is often a sign of code injection by malware.
2. The apihooks plugin reveals numerous hooked functions in explorer.exe and PGPdesk.exe related to loading libraries (LoadLibrary, LoadLibraryEx, FreeLibrary, etc). The hooks originate from the PGPmapih.dll module which is not a standard Windows DLL. API hooking is a common malware technique.
3. The apihooks output also shows inline hooking in the FTK Imager.exe process from the ProfUIS284ad32.dll and mfc90u.dll modules. While this could potentially be legitimate behavior, it warrants further investigation.
4. There are kernel-mode IAT hooks from the dump_atapi.sys driver to various functions in ataport.sys. Kernel-mode hooking is very suspicious and is usually associated with rootkits.
5. The pslist output includes some security-related processes like TrueCrypt.exe and PGP software which could potentially be used by an attacker. Their presence alone is not malicious but it provides additional context.

In summary, the memory dump contains multiple signs of malicious code injection, user-mode and kernel-mode API hooking, and suspicious processes. This strongly indicates the system was infected with malware when the memory dump was captured. Further analysis of the injected code and hooked functions is recommended to determine the specific malware and its capabilities.

Figure D2. Anthropic Workbench Output

Appendix E

Sample 5

ASSISTANT

Sample Name: Sample-5
 Operating System: Windows XP SP2
 Truth Label: 1
 Prediction: Malicious

Analysis and Reasoning:

1. **Malfind Analysis**:**
 - Multiple processes such as `csrss.exe`, `winlogon.exe`, `services.exe`, `lsass.exe`, `vmauthl.exe`, `svchost.exe`, `explorer.exe`, `rundll32.exe`, `vmtoolsd.exe`, `IEXPLORE.EXE`, `wmiprvse.exe`, `TPAutoConnSvc.e`, `alg.exe`, `TPAutoConnect.e`, `wscnfy.exe`, and `dwwin.exe` show evidence of memory regions with `PAGE_EXECUTE_READWRITE` permissions, which is unusual and indicative of code injection (Malfind output). This is a common tactic used by malware to execute arbitrary code in the context of another process.
2. **API Hooks**:**
 - The `apihooks` output shows multiple hooks in critical processes like `explorer.exe`, `vmtoolsd.exe`, and `IEXPLORE.EXE`. These hooks are intercepting functions in `mswsock.dll` and `deployPkg.dll`, which could be used by malware to manipulate network communications or system operations.
3. **Connection Scan**:**
 - The `connscan` results show `IEXPLORE.EXE` making numerous connections to various external IP addresses on common HTTP and HTTPS ports. This could indicate command and control (C2) communication or data exfiltration attempts.
4. **Process List and Scan**:**
 - The `pslist` and `psscan` outputs show all processes listed with their respective parent processes and creation times. The presence of typical system processes does not show anomalies in their hierarchy or creation times, but the combination of other suspicious indicators supports the conclusion of a compromise.

Given the evidence from memory permissions indicative of code injection, unauthorized API hooks, and unusual network activity, the memory dump from this Windows XP SP2 system is assessed as being compromised by malicious software.

Figure E1. OpenAI Playground Output

RESPONSE

Sample Name: Sample-5
 Operating System: Windows XP SP2
 Truth Label: 1
 Prediction: Malicious

Analysis:

Based on the provided Volatility plugin output, there are several indicators that suggest the memory dump is from a system infected with malware:

1. The malfind plugin output shows numerous processes with suspicious memory regions marked as PAGE_EXECUTE_READWRITE, which allows for both execution and modification. This is often a sign of injected malicious code. The processes with such regions include core Windows processes like csrss.exe, winlogon.exe, services.exe, lsass.exe, and svchost.exe.
2. The apihooks plugin reveals inline hooking in explorer.exe and IEXPLORE.EXE. The mswsock.dll!WSPStartup function is hooked in both processes, likely to intercept and manipulate network communications. The vmtoolsd.exe process also has suspicious hooks in the deployPkg.dll module.
3. The connscan plugin shows IEXPLORE.EXE (PID 420) establishing numerous outbound connections to various IP addresses on ports 80 and 443. This could indicate communication with a command-and-control server or data exfiltration.
4. The pslist and psscan plugins show a suspicious cmd.exe process (PID 2240) that has exited. This could have been used to execute malicious commands.
5. An unknown process, dwwin.exe (PID 2844), is launched from IEXPLORE.EXE. This is likely malware, as it's not a standard Windows process.

In summary, the combination of suspicious memory regions, API hooks, network connections, and unknown processes strongly suggests that this memory dump is from a system compromised with malware. Therefore, I predict this sample is Malicious.

Figure E2. Anthropic Workbench Output

Appendix F

Sample 6

ASSISTANT

Sample Name: Sample-6
Operating System: Windows 10
Truth Label: 0
Prediction: Malicious

Analysis and Reasoning:

1. **Malfind Analysis:**
 - Multiple entries from the Malfind plugin indicate the presence of suspicious memory regions with 'PAGE_EXECUTE_READWRITE' permissions, which is unusual and indicative of code injection or execution of non-standard memory pages. For example:
 - PID 3724 (MsMpEng.exe) has several entries with disassembled code showing typical shellcode patterns like `push`, `pop`, `jmp`, and `ret` instructions, which are common in executable shellcodes.
 - PID 1496 (WaAppAgent.exe) and PID 5896 (WindowsAzureGu) also show similar patterns with 'PAGE_EXECUTE_READWRITE' permissions, which are not typical for standard operations of these processes.
2. **PsList Analysis:**
 - The processes listed, such as MsMpEng.exe, WaAppAgent.exe, and WindowsAzureGu, are legitimate Windows processes but their behavior as indicated in the Malfind output suggests they might have been compromised or are being used to execute malicious code.
3. **NetScan Analysis:**
 - Connections to suspicious IP addresses were not directly evident from the provided data, but the presence of unusual memory patterns and permissions strongly suggests malicious activities that could include data exfiltration or command and control communication not captured in the netscan output.
4. **CmdLine Analysis:**
 - The command line parameters for processes do not show obvious signs of malicious activity, but the integrity of these processes is under question due to the findings from the Malfind plugin.

Given the evidence from the memory analysis indicating code injection or unauthorized code execution within system processes, the prediction is that the memory dump is indicative of a compromised (Malicious) state. This conclusion is primarily supported by the findings from the Malfind plugin, which is critical for identifying hidden and potentially harmful code execution within process memory spaces.

Figure F1. OpenAI Playground Output

RESPONSE

Sample Name: Sample-6
Operating System: Windows 10
Truth Label: 0
Prediction: Benign

Analysis:

Based on the provided Volatility plugin output, there are no clear signs of malicious activity on this Windows 10 system:

1. The pslist and psscan output shows expected system processes like smss.exe, csrss.exe, wininit.exe, services.exe, lsass.exe, etc. running from standard Windows locations. No suspicious processes are observed.
2. The netscan output reveals common Windows services listening on ports like RDP (3389), SMB (445, 139), RPC (135), DNS (53), etc. The established network connections are to Microsoft/Azure IP addresses which is normal for a cloud VM. No suspicious ports or connections detected.
3. The cmdline output does not contain any overtly malicious command line arguments. Standard Windows processes and some third-party applications like FTK Imager are seen.
4. The malfind plugin has flagged some small allocations with RWX permissions in legitimate processes like MsMpEng.exe (Windows Defender), WaAppAgent.exe (Azure agent), SearchApp.exe, etc. While not ideal, small RWX allocations in these processes are not a reliable indicator of malware by themselves. No injected code or shellcode is apparent.
5. No evidence of malware persistence (suspicious autoruns), process hollowing, code injection, or suspicious driver loading is found in the limited output provided.

In summary, while some non-ideal process properties are observed, there is insufficient evidence to declare this memory dump as definitively malicious based on the given Volatility output alone. The system appears to be a benign Azure cloud VM. Further analysis with additional plugins and manual review of suspicious artifacts would be required to completely rule out malware.

Figure F2. Anthropic Workbench Output