Advanced Persistent Threat (APT) Detection and Mitigation Framework

R25-037

Execution Phase Detection Using Memory Forensics

Project Proposal Report

B.Sc. (Hons) in Information Technology Specializing in Cyber Security

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

Dulitha Wickramasinghe

IT21825682

DECLARATION

I hereby declare that this is our original work and that, to the best of our knowledge and belief, it does not contain any previously published or written material by another person, except for instances in which acknowledgment is provided in the text. Nor does it incorporate without acknowledgment any material previously submitted for a degree or diploma in any other university or institute of higher learning. Additionally, I provide Sri Lanka Institute of Information Technology the nonexclusive right to copy and disseminate my dissertation in whole or in part by print, electronic, or other media.
I reserve the right to use all or part of this content in books or articles in the future.

NAME	STUDENT ID	SIGNATURE
H.P.D.D.M WICKRAMASINGHE	IT21825682	

The supervisor/s should certify the proposal report with the following declaration.

The above candidates are carrying out research for the undergraduate dissertation under my

Supervision.

....

Signature of the Supervisor Date

....

Signature of the Co-Supervisor Date

ACKNOWLEDGEMENT

“I express deep appreciation toward Dr. Harinda Fernando who serves as my supervisor together with Mr. Amila Senarathne who functions as my co-supervisor in the Faculty of Computing for their essential support throughout my research project initiation. The cumulative wisdom provided by both supervisors enhances both the concentration and academic caliber of my investigation.

I thank the Faculty of Computing staff members for helping me through the beginning stages of my research investigation.

I express my gratitude to all team members who supported my research work through collaborative efforts. My group discussions with my peers enabled an expansion of my research understanding by sharing valuable knowledge about the topic.

This research benefits greatly from all volunteers who helped with its completion. Their willingness to share information about the subject will help me develop better comprehension of the matter”.

ABSTRACT

Advance Persistent Threats exist as the most advanced and silent cyberattacks because they successfully penetrate network infrastructures and continue running malicious activities without being detected for longer periods of time. The execution phase of APTs remains challenging to detect through traditional security methods such as signature-based detection along with rule-based anomaly detection because they prove ineffective for this phase therefore real-time detection becomes problematic. The research establishes a new Execution Phase Detection Using Memory Forensics framework which uses machine learning with temporal analysis to identify signs of APT activity through system memory analysis. The proposed framework obtains memory snapshots from typical and compromised systems while it uses Volatility framework for extracting behavioral features. The framework extract features that proceed through a preprocessing pipeline to remove irregularities before LSTM networks process them for detecting abnormal temporal patterns from malicious execution. Anomaly detection relies on standard machine learning approaches within its module to boost procedural threat detection accuracy. When a system detects an APT it automatically stops running malicious processes while preventing major system intrusion during real-time threat mitigation. Through a graphical user interface, the frontend dashboard displays monitored threats and their defenses allowing security experts to track the current security situation. The framework satisfies core non-functional requirements which support high accuracy performance along with scalability and low latency and reliability that benefits enterprise security applications. This research introduces a new methodology to detect APT threats through the combination of memory forensics, deep learning models and instant response capabilities which fill the execution-level threat awareness.

Keywords:

Advanced Persistent Threats (APTs), Memory Forensics, Execution Phase Detection, Temporal Analysis, Anomaly Detection, Long Short-Term Memory (LSTM), Volatility Framework, Machine Learning, Memory Snapshots

Table of Contents

DECLARATION....2

ACKNOWLEDGEMENT....3

ABSTRACT....4

LIST OF FIGURES....7

LIST OF TABLES....8

LIST OF ABBREVIATIONS....9

INTRODUCTION....10

BACKGROUND....11

LITERATURE SURVEY....12

1. Traditional Memory Forensics:....12
2. ML-Based Anomaly Detection:....12
3. Temporal Pattern Analysis:....12

RESEARCH PROBLEM....13

OBJECTIVES....15

1. Main Objective....15
2. Sub Objectives....16

METHODOLOGY....17

1. Software Development Methodology....17
2. Development Process....18
3. Feasibility Study....19
4. Requirement Gathering and Analysis....19
5. Dataset....20
6. Implementation....21

PROJECT REQUIREMENTS....21

1. User Requirements....21
2. Software requirements....21

SYSTEM DIAGRAMS....22

1. Overall System Diagram....22

2. Component Specific System Diagram 23

PROJECT PLAN 24

CONCLUSION 25

REFERENCES 26

LIST OF FIGURES

Figure 1: Agile Development Methodology

Figure 2: Overall System Diagram

Figure 3: Execution Phase Detection Module Diagram

Figure 4: Work Breakdown Structure

Figure 5: Gantt Chart

LIST OF TABLES

Table 1: High-level representation of the research gap

LIST OF ABBREVIATIONS

APT - Advanced Persistent Threat

ML - Machine Learning

LSTM - Long Short-Term Memory

DLL - Dynamic Link Library

SVM - Support Vector Machine

C2 - Command and Control

API - Application Programming Interface

RDP - Remote Desktop Protocol

INTRODUCTION

Advanced Persistent Threats refer to complex cyberattacks which operate through stealth whereas remaining persistent while evading standard detection solutions. Attacks during the execution phase release malicious payloads directly into system memory thus avoiding defense systems based on disk storage. The memory forensics tool Volatility helps investigators with post-incident analysis, yet its existing methods do not provide real-time detection of temporal anomalies, thus systems remain exposed to malicious active processes. Modern security strategies that use memory analysis or signature detection methods prove inadequate at detecting the evolving APT techniques.

This research proposal evaluates this gap through development of an Advanced Persistent Threat Detection and Mitigation Framework that merges memory forensic analysis with time-based pattern recognition techniques. Volatility allows the framework to collect memory snapshots and the process list and DLL features alongside network handle information before clean data becomes available for analysis. The system identifies temporal anomalies using LSTM networks for understanding sequential memory patterns alongside traditional machine learning systems for malicious indicator identification. The system implements automated threat response capabilities that activate through process termination followed by real-time system status presentation alongside system activities.

During system execution the framework leverages combined approaches of temporal modeling with memory forensics to detect threats with high precision while providing essential APT mitigation responses in low time intervals. Such security solution shifts cyber defense tactics from passive response to active protection while resolving enterprise scalability and reliability problems. This research develops a real-time temporal analysis solution that provides enterprises with a scalable tool against modern APT operations.

BACKGROUND

The highly specialized cyber assault of Advanced Persistent Threats operates with a purpose to breach systems while sustaining undetected operations until sensitive information can be taken. APTs evade detection by running completely in system memory when executed because this technique enables them to escape conventional malware protection systems such as antivirus software. The attackers can launch harmful payloads (ransomware, rootkits etc.) through memory-resident behavior because this technique prevents them from writing anything to storage devices thus making traditional signature detection insufficient.

The discipline of memory forensics became essential to identify stealthy threats by analyzing short-lived memory data. Through Volatility investigators can recover essential memory artifacts that include process lists together with DLLs and network connections and injected code from memory dumps. Most current analytical frameworks inspect systems following incidents to detect compromises, yet damage has already occurred. Detecting memory threats in real-time is complicated because memory operates dynamically and transiently, and system processing must be fast.

Studies focused on Advanced Persistent Threat detection through machine learning rely on anomaly detection which uses static reports from memory and system logs without analyzing temporal APT behavioral patterns. Attackers using APT tactics consistently perform their operations through successive stages which modify system memory structures during specific intervals until they establish connections with their command centers. The time-based relationships which these models should identify remain beyond the capability of clustering algorithms alongside conventional classification methods. Researchers have shown that Long Short-Term Memory networks have strong capabilities in processing sequential data like network traffic and log files, yet they have not been extensively explored in real-time memory forensics applications.

The organizations experience vulnerabilities in their execution phase because analysis that detects and mitigates at the appropriate time remains incomplete. The current solutions demonstrate poor integration between forensic investigations with automatic response features and presentational visualization constraints which reduces their validity for enterprise usage. This important research creates a crucial synthesis between memory forensics and automated mitigation systems and temporal modeling to protect organizations against evolving APT threats proactively.

LITERATURE SURVEY

Research papers about Advanced Persistent Threat detection fall within three distinct categories: traditional memory forensics, machine learning anomaly detection and temporal pattern analysis.

1. ***Traditional Memory Forensics:***

Walters and Petroni (2007) deeply established memory forensics through their development of the Volatility Framework which specialized in artifact extraction after attacks [1]. The research conducted by Ligh (2014) extended Volatility Framework plugins to discover hidden processes and code injections [2]. The methodologies show limited effectiveness for real-time identification and require human analysts for examination which makes them improper for real-time APT threats.

2. ***ML-Based Anomaly Detection:***

The current trend involves ML-based automation systems for identifying cybersecurity threats. The authors at Yen (2013) applied clustering methods to memory dump inspection for process abnormality detection yet their analysis focused solely on static states without considering temporal relationships [3]. The researchers at Sgandurra (2016) applied SVM classifiers to detect ransomware in memory yet their approach depended on pre-defined signatures that APTs avoid using polymorphism methods [4]. Both methods fail to recognize the time-dependent development of APT techniques which include delayed process creation and scheduled C2 communication.

3. ***Temporal Pattern Analysis:***

The analysis of time-based data through Long Short-Term Memory networks succeeds in detecting anomalies in sequential patterns found within network traffic and log files according to research [5]. The application of Long Short-Term Memory (LSTM) networks by Mirsky (2018) processed network flow data for detecting advanced attack sequences [6]. These analytical models have not received adaptation for use in memory forensic analysis. According to Bahador (2018) temporal analysis shows potential for APT detection although their research focused on log-based datasets [7].

A high-level representation of the research gap is as follows,

Feature	Existing Solutions	Proposed Framework
Memory Forensics Integration	✓	✓
Temporal Pattern Analysis	✗	✓ (LSTM-based)
Real-Time Detection	Limited	✓
Automated Mitigation	Partial	✓
Real-Time Visualization	✗	✓

Table 1: High-level representation of the research gap

RESEARCH PROBLEM

APTs use memory-resident execution to bypass traditional security measures through their ability to hide in volatile memory space. The memory analysis tool Volatility helps identify malicious signs from tools and code injections after incidents take place, but the current techniques have three main drawbacks:

Analysis of physical memory snapshots using traditional methods yields poor results for recognizing APT time-sensitive behaviors like delayed process spawning and intermittent commands and controls signaling and time-value manipulation since these techniques heavily depend on manual analytical methods and static evidence examination. Machine learning (ML) models used in APT detection with techniques such as SVM clustering and others fail to track necessary time-based relationships between memory artifacts since these models treat each artifact as an independent element. Static ML frameworks lack the capability to detect time-dependent execution phases because these natural properties are intrinsic to APTs'.

The current frameworks show a deficiency in connecting real-time memory assessment to automatic defense implementation. APT attacks frequently escape detection due to manual post-detection response actions that enable the threats to persist and to achieve privilege elevation as well as to steal data. The gaps become worse because no framework unifies memory forensic methods with time pattern evaluations together with real-time attack prevention capabilities. The effective application of LSTM networks to detect APTs through memory forensics analysis of sequential data (e.g., network traffic) has not been previously explored in research literature. The current memory forensic tools including Volatility offer limited capabilities to detect anomalies in real-time or trigger automated responses which result in extended duration of Advanced Persistent Threat attacks in enterprises.

The following open-ended questions are addressed by this study:

- ***How may temporal irregularities in memory behavior such as irregular process lifecycle patterns, non-linear handle allocation, or aberrant DLL loading sequences be modeled?***
  Memory snapshots are analyzed as static datasets by current machine learning algorithms (e.g., clustering, SVM), which disregard temporal context.
- ***How may LSTM-based temporal modeling be used to improve real-time memory forensics to identify APT execution stages prior to lateral movement or data exfiltration?***
  The behavior of memory-resident APTs has not been adapted from earlier temporal analysis work (such as network traffic).
- ***Which architectural layout prevents system instability or false positives while guaranteeing low-latency integration of detection and remediation (such as process termination)?***
  Automated reaction mechanisms connected to real-time memory analysis are absent from current solutions.

This study looks to create a new framework which changes forensic memory analysis from being used after incidents occur into a pre-incident APT defense tool. The proposed framework that incorporates LSTM networks with real-time anomaly detection and automated mitigation systems fulfills enterprise needs by providing accurate solutions for evolving APT threats and scalability and reliability features.

OBJECTIVES

1. Main Objective

The research goal centers on establishing a new Advanced Persistent Threat (APT) Detection and Mitigation Framework consisting of an integration between memory forensics analytics and temporal pattern examination and real-time anomaly detection capabilities to detect and counteract malicious ATT execution patterns. Specifically, the framework aims to:

a. **Integrate Memory Forensics with Temporal Analysis**:

Real-time detection of APT execution phases becomes possible through the combination of Volatility Framework results and Long Short-Term Memory (LSTM) networks applied to dynamic memory artifacts like process lists and DLL dependencies and network handles.

b. **Enable Real-Time Detection of APT Execution Phases**:

The proposed system uses LSTM-based temporal models together with standard anomaly detection methods like Isolation Forest and SVM to perform real-time malicious memory-resident process surveillance that decrease detection limitations and false positive occurrences.

c. **Automate Threat Mitigation**:

Upon detection the system will initiate automatic responses to stop damaging processes while isolating infected systems and creating forensic logs to reduce both attackers' stay time and stop data transfer activities.

d. **Validate Framework Efficacy**:

Test the framework using both normal and APT-infected memory dumps to validate its ability to detect malicious activity with high precision rates of more than 95% while reaching a detection-mitigation cycle execution time of less than 1 second and supporting enterprise-sized operations.

e. **Provide Actionable Insights via Real-Time Visualization**:

The system needs a real-time graphic display which presents detected risks together with the implemented security measures alongside memory activity patterns allowing analysts to better monitor and stop APT attacks.

The framework achieves these objectives because it connects forensic analysis of post-incident memory with anticipatory APT defense solutions which present an innovative method to target emerging cyber threats when they perform their crucial stage.

2. Sub Objectives

a. Designing a Feature Extraction and Preprocessing Pipeline

The system should automatically extract important time-related features (including process creation/deletion rates and handle allocation sequences and DLL load timestamps) from original memory dump files. The features require data cleaning procedures which include noise elimination and normalization to prepare them for temporal analysis.

b. Implement LSTM-Based Temporal Pattern Recognition\

An LSTM network should be trained and optimized to discover memory behavioral dependencies which detect abnormal events such as wrongly timed process execution or delayed malware activation.

c. Develop an Automated Threat Mitigation Module

The system must have a low-latency automated module that terminates harmful processes as well as blocks specific network connections and generates forensic logs after identifying threats.

d. Design a Real-Time Visualization Dashboard

A visual interactive front end should present trends of temporal memory behavior together with threat detections and mitigation logs to allow real-time monitoring of APT activity by analysts.

METHODOLOGY

The research adopts a systematic method to create an Advanced Persistent Threat (APT) Detection and Mitigation Framework that integrates memory forensics elements with real-time anomaly detection and temporal models. The research follows these main stages for its methodology structure:

1. ***Software Development Methodology***

- **Dataset Collection**

Memory Snapshots Collection:

Acquire snapshots of system memory from uncontaminated systems that display standard operational states. The collection of memory dumps needs to focus on systems with known APT malware infections. Virtual environments operated under controlled circumstances should be used to replicate phases of APT command execution.

- **Preprocessing Pipeline Design**

Feature Extraction:

The framework named Volatility extracts important artifacts that include process lists as well as DLL dependencies and handles and injected code. Time stamps should be added to each extracted feature to establish chronological connections.

Data Cleaning and Transformation:

The filtering of unnecessary data through preprocessing methods should be performed before analysis. Regular normalization techniques should be applied to extracted features for maintaining uniformity during input processing. Memory dumps should be transformed into structured datasets with timestamps to make them suitable for temporal-based investigations.

- **Temporal Analysis Using LSTM**

Model Development and Training

The system uses Long Short-Term Memory (LSTM) networks to recognize ordered dependencies found within memory artifacts. Labeled data consisting of normal and malicious execution phases drives the training process of the LSTM model. The detection accuracy improves when hyperparameter values are optimized along with the reduction of false positive identification.

- **Anomaly Detection**

Machine Learning-based Detection

Anomaly detection algorithms based on Isolation Forest, One-Class SVM and Autoencoders should be implemented. Anomalies become detectable when runtime memory data gets monitored relative to trained standard values. Defense systems should produce threat probability rankings through analysis of unexpected behavioral patterns.

- **Mitigation and Response**

Automated Threat Mitigation

Create an automatic system that terminates suspicious processes right after their detection by the security module. The system should implement warning systems which will alert administrators when suspicious activities occur. The system must record detected threats for later analysis in logs.

- **Real-Time Visualization**

Frontend Dashboard Implementation

Organize a dashboard system which shows present time monitored security risks. The model detection system should establish API communication links with the front-end interface. Security logs alongside built-in trend analysis tools are accessible for monitoring purposes.

2. *Development Process*

The research will adopt an Agile methodology for iterative development which allowed proper integration and adaptability during enhancements. Key steps include:

- The analysis phase determined the essential needs involving real-time memory forensics and anomaly detection.
- Component Design: Architecting modular components for memory analysis, ML detection, and mitigation.

- The process consists of testing and optimization stages which enhance performance by continuously running tests and listening to end-user feedback.

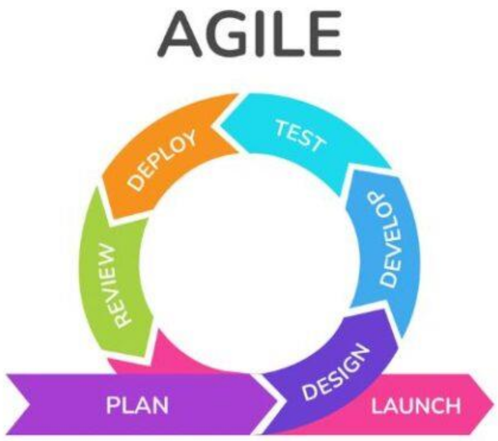


Figure 1: Agile Development Methodology

3. *Feasibility Study*

The proposed framework undergoes feasibility examination through an evaluation of its technical alongside practical implementation potential.

- Technical Feasibility: Evaluating computational requirements for handling real-time memory snapshots and LSTM-based detection.
- During operational feasibility assessment the framework must accommodate current security tools and work processes.
- Economic Feasibility: Assessing the cost of development, deployment, and maintenance.
- Scalability Feasibility: Testing system performance under different memory loads.

4. *Requirement Gathering and Analysis*

Detailed requirement gatherings during the initial phase make certain that the APT Detection and Mitigation Framework fulfills the expectations of users and meets technical possibilities as well as all intended system capabilities. Requirements gathering happens by conducting research combined with expert consultations as well as analysis of present security frameworks.

Identifying Stakeholder Needs

- Security analysts need instant threat identification capabilities, and they require access to forensic analysis.
- System administrators require automated systems which perform threat mitigation to prevent potential risks.
- Organizations need their systems to process data accurately while ensuring fast speeds and being able to grow over time.

Functional Requirements:

- The system requires data collection and memory analysis procedures for various system environments.
- Real-time anomaly detection occurs by deploying ML models through the system.
- Simultaneous threat prevention occurs through automated response protocols.

Non-Functional Requirements:

- The system must achieve very high precision when identifying active APT operations.
- System functionality should support real-time threat response because of its low-latency processing capabilities.
- Scalability for enterprise deployment.

5. *Dataset*

The research relies on publicly available and enhanced datasets to perform complete analysis and detection of APT execution through memory forensic methods. The datasets include:

- CIC-MalMem-2022 delivers memory dumps with labels for malware behavioral examination [8].
- The research employs MemMal-D2024 which represents an updated Windows-based dataset for memory-based malware detection according to "MeMalDet: A Memory analysis-based Malware Detection Framework using deep autoencoders and stacked ensemble under temporal evaluations." [9]
- The Malware Detection approach from Memory Dump examines ways to detect Spyware alongside Ransomware and Trojans through obfuscated malware methods to create realistic attack situations [10].

The availability of these datasets makes it possible for the framework to execute proper training and validation processes to optimize its memory-based anomaly detection models.

6. ***Implementation***

Development and integration of various components during implementation.

- Memory Forensics Module: Using the Volatility framework for feature extraction.
- Machine Learning Module: Training LSTM networks and anomaly detection algorithms.
- A real-time process termination capability operates during threat detection through the Threat Mitigation System implementation.
- Visualization Dashboard: Displaying real-time threat analysis and mitigation logs.

PROJECT REQUIREMENTS

1. ***User Requirements***
    - The system needs to utilize memory forensics for real-time APT detection.
    - Users must receive security alerts together with recommended threat mitigation procedures when threat detection occurs.
    - Users must have access to a friendly interface to monitor their security situation from a dashboard.
    - Security analysts need access to review both detected anomalies along with forensic logs.
    - The framework needs to run with low detection errors and achieve maximum accuracy standards during its operation.
2. ***Software requirements***
    - The system utilizes the Windows/Linux operating system for performing memory analysis along with model deployment.
    - Volatility stands as the selected tool for extracting memory features from Operating Systems.
    - The development of models and backend integration occurs through Python programming with Python running on Windows/Linux operating systems.
    - Machine Learning Frameworks: TensorFlow/Keras for LSTM-based anomaly detection.
    - The project utilizes either MongoDB or PostgreSQL to create a storage system which handles processed memory data.
    - Web Framework: React.js or Flask for real-time dashboard visualization.

SYSTEM DIAGRAMS

1. *Overall System Diagram*

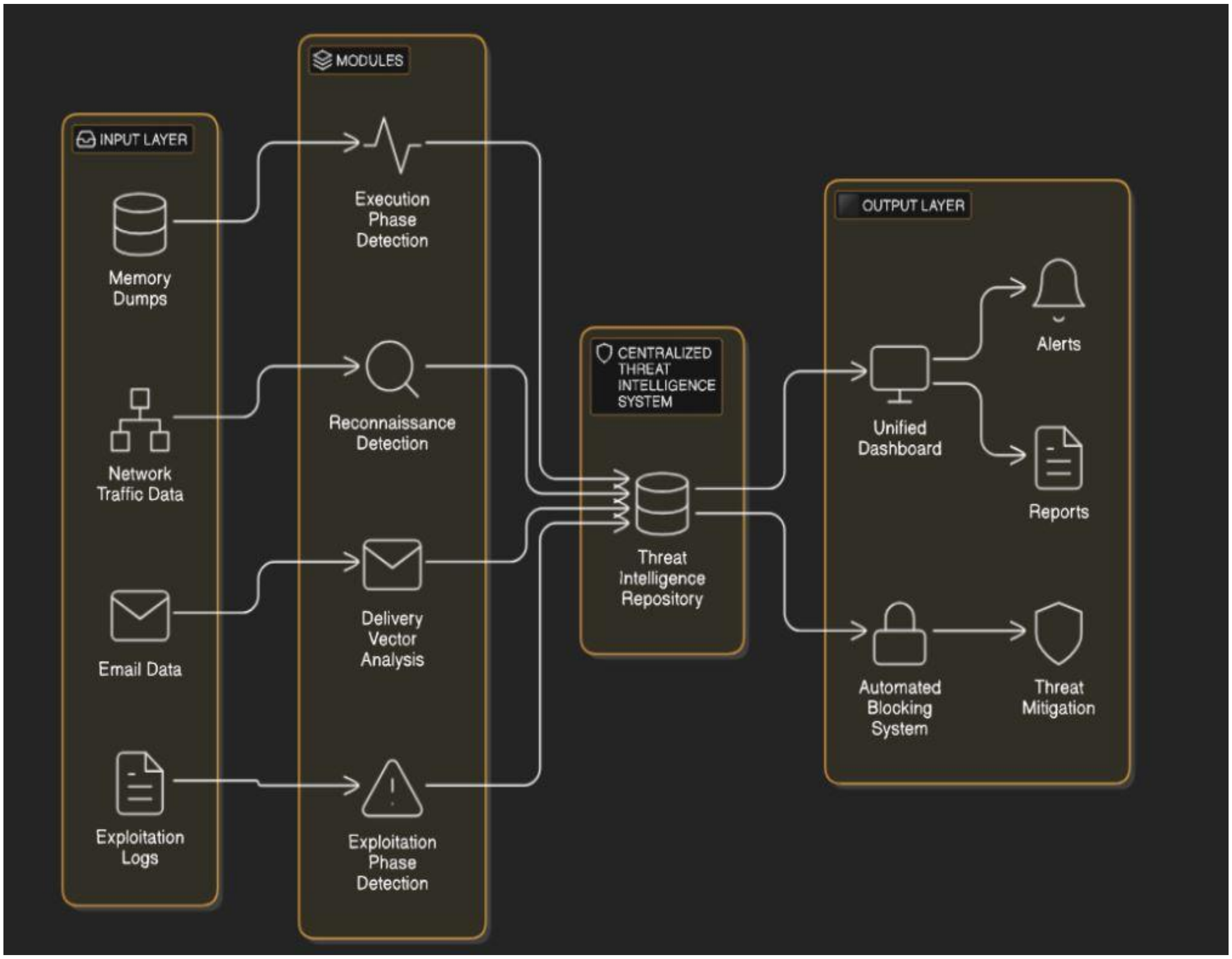


Figure 2: Overall System Diagram

2. Component Specific System Diagram

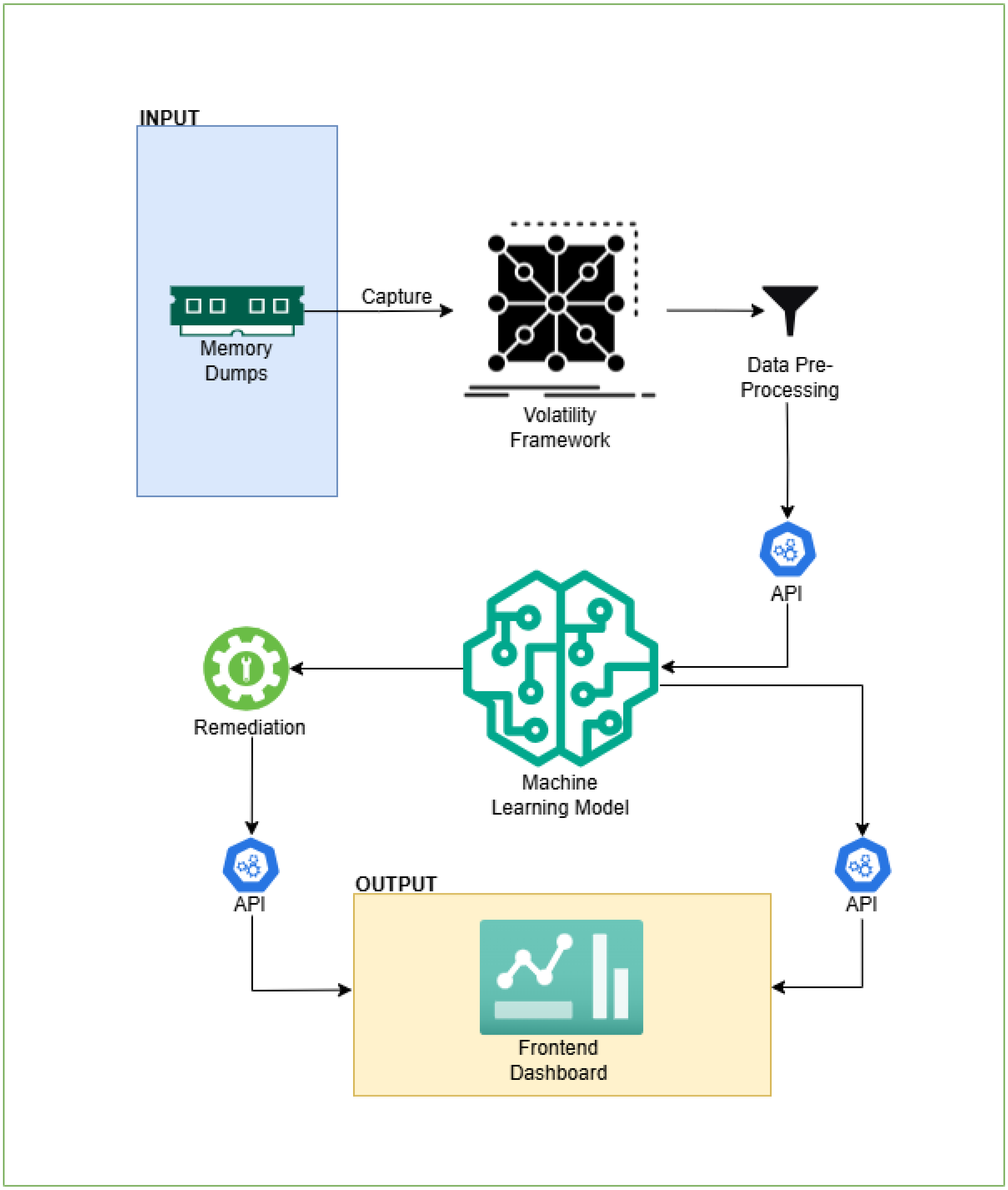


Figure 3: Execution Phase Detection Module Diagram

PROJECT PLAN

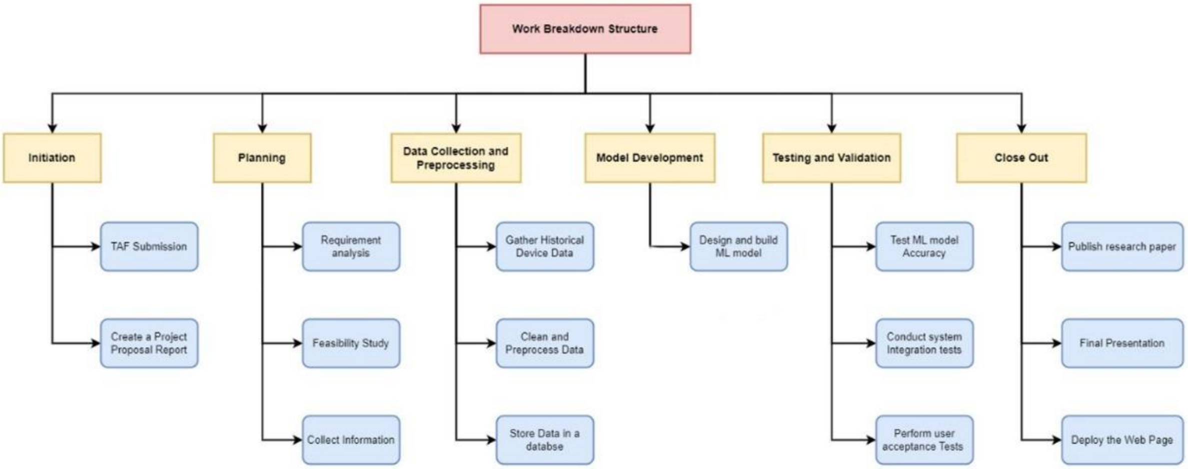
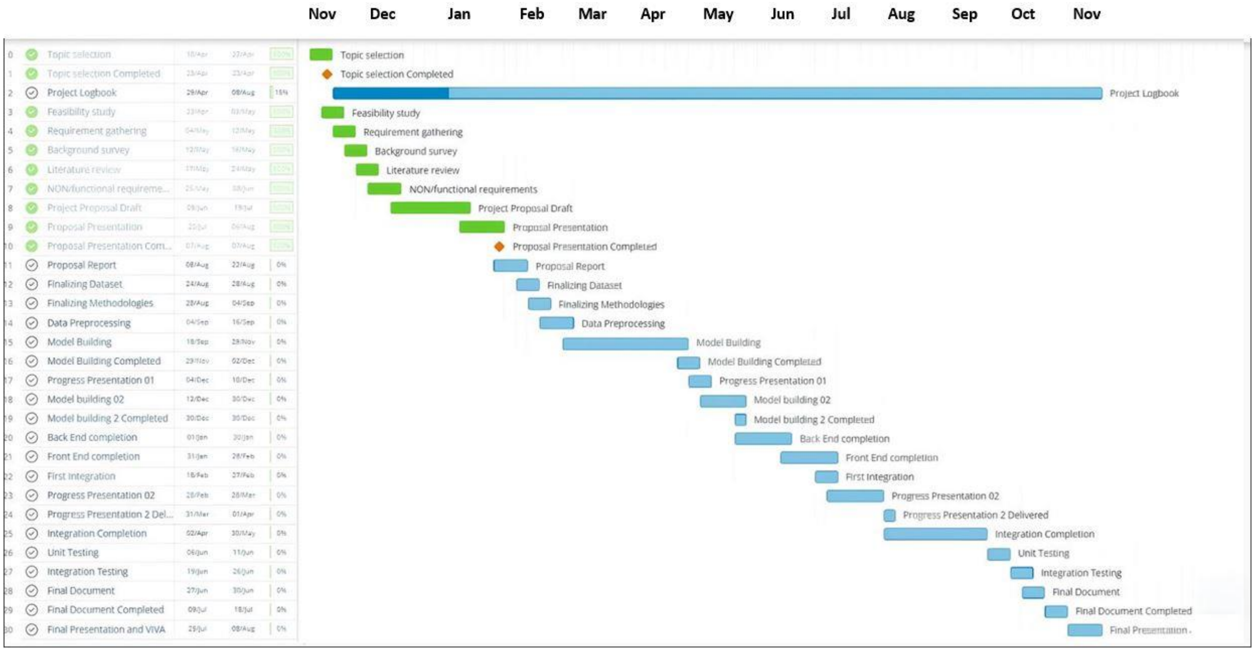


Figure 4: Work Breakdown Structure



0	✓	Topic selection	10/Apr	22/Apr	100%
1	✓	Topic selection Completed	23/Apr	23/Apr	100%
2	✓	Project Logbook	29/Apr	08/Aug	15%
3	✓	Feasibility study	23/Apr	03/May	100%
4	✓	Requirement gathering	04/May	12/May	100%
5	✓	Background survey	12/May	16/May	100%
6	✓	Literature review	17/May	24/May	100%
7	✓	NON/functional requireme...	25/May	08/Jun	100%
8	✓	Project Proposal Draft	09/Jun	19/Jul	100%
9	✓	Proposal Presentation	20/Jul	06/Aug	100%
10	✓	Proposal Presentation Com...	07/Aug	07/Aug	100%
11	✓	Proposal Report	08/Aug	22/Aug	0%
12	✓	Finalizing Dataset	24/Aug	28/Aug	0%
13	✓	Finalizing Methodologies	28/Aug	04/Sep	0%
14	✓	Data Preprocessing	04/Sep	16/Sep	0%
15	✓	Model Building	18/Sep	28/Nov	0%
16	✓	Model Building Completed	29/Nov	02/Dec	0%
17	✓	Progress Presentation 01	04/Dec	10/Dec	0%
18	✓	Model building 02	12/Dec	30/Dec	0%
19	✓	Model building 2 Completed	30/Dec	30/Dec	0%
20	✓	Back End completion	01/Jan	30/Jan	0%
21	✓	Front End completion	31/Jan	28/Feb	0%
22	✓	First Integration	18/Feb	27/Feb	0%
23	✓	Progress Presentation 02	28/Feb	28/Mar	0%
24	✓	Progress Presentation 2 Del...	31/Mar	01/Apr	0%
25	✓	Integration Completion	02/Apr	30/May	0%
26	✓	Unit Testing	06/Jun	11/Jun	0%
27	✓	Integration Testing	19/Jun	26/Jun	0%
28	✓	Final Document	27/Jun	30/Jun	0%
29	✓	Final Document Completed	09/Jul	18/Jul	0%
30	✓	Final Presentation and VIVA	25/Jul	08/Aug	0%

Figure 5: Gantt Chart

CONCLUSION

The research targets the improvement of Advanced Persistent Threat (APT) detection through memory forensics together with automated response systems while relying on temporal anomaly detection. The framework demonstrates effectiveness in system memory malicious execution pattern detection by analyzing datasets consisting of CIC-MalMem-2022, MemMal-D2024 and Malware Detection from Memory Dump.

Real-time threat detection achieves improvement through temporal analysis based on LSTM which remedies current security solution inadequacies. The automated threat mitigation system provides swift reaction to detected anomalies which decreases the amount of time APT activities remain active.

The research offers an efficient memory forensics solution for enterprises through its machine learning integration thus helping advance cybersecurity practices. Further work in this area should enhance detection capabilities while expanding the diversity of datasets and improving mitigation methods.

The framework functions as an advanced proactive security system which helps enterprises maintain strong protection by detecting APT events in real-time.

REFERENCES

[1] A. Walters and N. Petroni, "Volatility: Framework for Volatile Memory Analysis," Digit. Investig., vol. 4, pp. 121–130," 2007.

[2] M. H. Ligh, "The Art of Memory Forensics.," Wiley, 2014.

[3] T.-F. Yen, "Beehive: Large-Scale Log Analysis for Detecting Suspicious Activity in Enterprise Networks," Proc. ACSAC, 2013.

[4] D. Sgandurra, "Automated Dynamic Analysis of Ransomware: Benefits, Limitations, and Use for Detection," IEEE Trans. Inf. Forensics Secur., vol. 14, no. 12, pp. 3253–3265, 2019.

[5] Y. Mirsky, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," Proc. NDSS, 2018.

[6] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[7] M. K. Alzaylaee, "DL-Droid: Deep Learning-Based Android Malware Detection Using Real Devices," IEEE Access, vol. 8, pp. 170786–170801 , 2020.

[8] "Unb," [Online]. Available: https://www.unb.ca/cic/datasets/malmem-2022.html.

[9] "Github," 2024. [Online]. Available: https://github.com/mpasco/MemMal-D2024.

[10] "Kaggle," [Online]. Available: https://www.kaggle.com/datasets/subhajournal/malware-detection-from-memory-dump.