National College of Ireland

Developing Proactive Measures Against Increasingly Sophisticated Ransomware-as-a-Service (RaaS) Attacks

MSc Research Project
M. Sc Cyber Security

Aryan Kedare
Student ID: 23290323

School of Computing
National College of Ireland

Supervisor: Joel Aleburu

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Aryan Kedare

Student ID: 23290323

Programme: M.Sc Cyber Security **Year:** 2024-25

Module: MSc Research Project

Supervisor: Joel Aleburu

Submission Due Date: 11th August 2025

Project Title: Developing Proactive Measures Against Increasingly Sophisticated Ransomware-as-a-Service (RaaS) Attacks

Word Count: ..... **Page Count** 23

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Aryan Kedare

Date: 11 August 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project	□

is lost or mislaid. It is not sufficient to keep a copy on computer.	

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Developing Proactive Measures Against Increasingly Sophisticated Ransomware-as-a-Service (RaaS) Attacks

Aryan Kedare
X23209323

Abstract

The rise of Ransomware-as-a-Service (RaaS) has changed the cyber threat environment by enabling even unskilled individuals to conduct highly advanced ransomware attacks. This research introduces a proactive and multilayered protection mechanism designed to identify and alleviate such risks prior to encryption. The proposed system integrates a machine learning-based behavioral classifier, heuristic analysis and real-time defensive responses including Hash Concealment and Zero Trust Enforcement. A virtualized sandbox environment was used to simulate both benign and malicious activities, from which a balanced dataset was created. A Random Forest model trained on pre-encryption behavior achieved 99.7% accuracy, effectively detecting ransomware types such as encryption-based, shadow-copy wipers and C2/leak site variants. Heuristic rules added interpretability by classifying attacks based on file extensions, process indicators and dark web communication traces. After detection automatic file concealment and restricted access were triggered to minimize damage. The model performed with zero false positives during very intensive benign tasks and successful file recovery after the detection. While this is effective, challenges remain in detecting stealthy or advanced variants and reducing the latency of detection. The study suggests that trained AI when combined with lightweight automated defenses it offers an effective and efficient real-time response to RaaS threats establishing a robust basis for enterprise-level implementation and additional research.

1 Introduction

In last few years ransomware has grown from a simple form of malware to significant cyber threat has affected not only organizations but also those operating on the global scale (Jaffe and Floridi, 2024). During its initial years, usually, the ransomware would be considered an isolated attack, rather casually targeting or attacking individual people or small businesses with opportunistic campaigns. However the cybercrime ecosystem has since progressed which has resulted in of newer business model know as Ransomware-as-a-Service (RaaS). This model works same as Software-as-a-Service (SaaS) where skilled developers create advanced ransomware programs and distribute them to a network who execute attacks for a share of the profits (Keijzer, 2020; Chauhan and Kshetri, 2023). Due to this there is a industry which is growing underground where the individuals with minimal or no technical expertise can deploy highly advanced ransomware programs with dangerous effects (Chauhan and Nir Kshetri, 2023).

The emergence of RaaS has changed the threat landscape forever. RaaS platforms, including REvil, GandCrab, DarkSide and LockBit, provide several subscription-based models, from tech support to flexible ransomware characterisations (Alwashali, Rahman, and Ismail, 2021). They also give advanced tools like anti-analysis and evasion capabilities, dynamic encryption and lateral movement tools (Kharraz, 2025). Due to these, there is a challenge in traditional cybersecurity defences and have led to a surge in attacks in important sectors like healthcare, energy, and government (Gyimah et al., 2024).

The impact of RaaS on organizations is profound. The immediate financial consequences of ransom payments and data recovery costs, victims face long-term reputational damage, operational disruptions, and potential legal liabilities (Office, 2025; Rose et al., 2020). Another example of the DarkSide which initiated RaaS attack at the Colonial Pipeline in 2021 disrupted fuel supplies across the United States (ENISA, 2021), demonstrating how ransomware has now become a critical threat to national security and public safety.

Despite increasing in regulatory efforts to combat ransomware such as the UK government's recent proposal to improve incident reporting and denying ransom payments (Office, 2025). Current cybersecurity practices have been proven less efficient and, therefore, cannot counteract modern RaaS programs utilizing advanced encryption, decentralization, and stealth techniques (Hirano and Kobayashi, 2019). Traditional defences such as signature-based detection, firewalls and basic endpoint protection are insufficient to counter rapidly evolving attack vectors (Gyimah et al., 2024).

Moreover, research carried out by Gulmez, Gorgulu Kakisim and Sogukpinar (2024) has shown some progress in the direction of ransomware detection by AI-based models, yet most approaches in literature are still incapable to neutralize RaaS attacks before encryption Sgandurra et al. (2016). The service-based nature of RaaS are combination of its dynamic configuration and command-and-control flexibility (Keijzer, 2020) and thus requires defense strategies that are adaptive, multilayered, and tightly integrated with organizational governance (Alwashali, Rahman and Ismail, 2021; ENISA, 2021).

Research is motivated by the urgent need to bridge this gap. It aims to study and evaluate multilayered proactive defence strategies that can detect, mitigate, and respond to RaaS attacks at various stages in an attack lifecycle. The proposed approach combines AI-driven early detection of pre-encryption behavioural patterns, robust data protection strategies, including Hash Conceal and Zero Trust principles (T and A, 2024), Alignment with organizational governance frameworks such as NIST (Alwashali, Rahman and Ismail, 2021). This project intends to investigate a fully integrated proactive security posture that can aid organizations to move from reactive recovery to proactive prevention.

Research Question

How can organizations implement a multilayered defence strategy to effectively mitigate the evolving threats posed by Ransomware-as-a-Service (RaaS) attacks?

The primary objective of this study is to creating and validating AI-powered detection models capable of accepting pre-encryption behaviour patterns typical of Ransomware-as-a-Service sort of attacks. In addition the study aims to design and test a complementary data protection mechanism that leverages both Hash Concealment and Zero Trust principles to mitigate the impact of any ransomware that bypasses initial detection. The study also proposes an integrated governance framework that would aligns technical controls with cybersecurity best practices from NIST.

The methodology involves conducting controlled experiments running controlled experiments on known RaaS samples by preparing an isolated virtual environment (VirtualBox). The experiments extensively monitored ransomware behaviour through file system changes, network traffic, and process inspection. TensorFlow-based machine learning models are trained on this behaviour to distinguish between malicious patterns. Detection rules such as YARA signatures are used to complement the AI approach. The two types of models are then benchmarked against compliance frameworks such as NIST SP 800-207 and SP 800-184 to check feasibility under technical and organizational perspectives.

The report begins with a literature review that analyses the evolution of RaaS, its technical sophistication, current defensive approaches and the research gaps. The methodology specifies the experimental design, dataset generation, machine learning model development, and testing framework. Results for detection accuracy, false positive rates and computational performance provide the evaluation of the model. The discussion interprets the results, emphasizing the operational challenges of rolling such models into enterprise environments. Finally the conclusion outlines some key insights and recommendations for future research.

2 Related Work

Ransomware has evolved from random opportunistic malware to a fully operational underground economy driven by a Ransomware-as-a-Service (RaaS) model. Seminal works like those by (Anderson and Moore, 2009) and Hartel, Junger and Wieringa, 2010 laid the economic foundation for understanding why cybercrime continues to flourish low-cost infrastructure, high ROI and the monetization of scalable digital assets. The Ransomware-as-a-Service (RaaS) model makes it easy for the threat actors to rent ransomware toolkits from developers in exchange for some money which will be earned from the ransom, which significantly lowers the barrier to entry for cybercrime (Chauhan and Nir Kshetri, 2023). This model closely copies the working of SaaS platforms which offers customer support, dashboards, payment portals and programs which allows even low skilled attackers to execute attacks.

This economic rationale helps explain the rapid expansion of RaaS. Major platforms like REvil and LockBit have been facilities of complete ecosystems, including customer support, affiliate dashboards and customizable payloads (ENISA, 2021). Such operators of ransomware focus their energy on stealth and persistence. While early variants of malware such as CryptoLocker were easily detected through static analysis, unlike CryptoWall, which Kharraz et al., (2015) showcased early signs of obfuscation, network-layer encryption and delayed activation; such traits now place the base for modern RaaS tools. Understanding the economic and historical path of RaaS highlights the need for defence strategies that prevent attackers from reaching the monetization stage, making proactive detection critical.

While ransomware attacks have progressed on the technical front and their capacity to bypass old defenses too. Technical obfuscation has become a hallmark of modern RaaS tools. They make use of LOLBins (Living-off-the-Land Binaries) powerShell scripting and lateral movement tactics to disguise malicious activity within legitimate operations (Kharraz, 2025). For example they might delete shadow copies using the vssadmin tool or use DLL injection for privilege elevation. These malicious behaviors make ransomware almost impossible to detect by the signature-based antivirus tools relying on fixed patterns in code or file signatures.

Behavioral detection began as a solution which focuses on indicators such as increased entropy, burst file access rates, and system call anomalies. However, such systems are often reactive, triggering only after encryption starts. Lindorfer, Kolbitsch and Milani Comparetti, (2011) and Sgandurra et al. (2016) proposed dynamic analysis frameworks Anubis and EldeRan, respectively that track low-level system behavior to flag threats. Although effective in lab settings these models typically require sandbox environments and fail in real-time deployment scenarios. The limitations of signature-based and static analysis underscore the importance of behavioural monitoring and real-time risk prediction to detect RaaS early.

In recent years machine learning and deep learning have emerged as most effective detection techniques. Algorithms like Random Forest, SVMs, and DNNs can be trained on different telemetry features, including API calls, registry edits, and sequences of file operations. The research work by Hussain et al., (2024) shows that such models are held to have a high classification accuracy for ransomware activity. More recently Gulmez, Gorgulu Kakisim and Sogukpinar, (2024) introduced XRan which is a deep-learning model developed for explainable ransomware detection. Nonetheless, accuracy in these models fluctuates significantly due to multiple factors like dataset imbalance, skews training, feature leakage during preprocessing inflates scores and sometimes due to lack of data consistancy which makes them delicate (Olakunle et al., 2019).

Some researchers have used Generative Adversarial Networks (GANs) to generate realistic malware variants for robust training. RagGAN Zhang, Wang and Zhu (2021) is one such approach. However, despite the promise, GANs fail to take into account behaviour-specific evasion techniques: registry key rotation or randomized process injections. Additionally, due to huge computational requirements, GAN models have remained largely untouched in the real-time enterprise domain. The more simpler and interpretable models like Random Forests are still practical for early detection, especially when trained on pre-encryption behaviour balanced datasets.

While evaluating the existing ransomware detection approaches reveals a dynamic tension between competing methodologies each of them where having their own advantages and some short comings. Signature based detection which was a long industry standard is now broadly recognized as ineffective against the rapidly mutating and hidden strains of RaaS operation. Yet, purely behavior based models while more adaptive, faced challenges of their own, they struggled with high false positive rate and only triggered after malicious encryption begin which made recovery difficult and detection insufficient for true prevention. Although the advanced machine learning, including the random forests and deep learning models showed encouraging accuracy in controlled experiment, their real word ability is undermined by issues like dataset imbalance, model overfitting and its inability to flag attack patterns. Research utilizing GANs to enhance the model but has not transited effectively to operational environments due to computational costs and practical evasion tactics unseen in simulated dataset. Moreover, a few studies critically compare the operational risks of kernel-based monitoring since its privileged access which increases attack surface and user level tracking that is more deployable but risk prone due to potential blind spots. This evaluation gaps leaves security administrator with unclear trade-off between precision, deployability and system risks. This research tries to address this gap combining real time behavioural analytics with proactive, automated containment and access governance, it offers more resilience and enterprise ready solution.

Detection alone is not enough. Proactive mitigation mechanisms are needed to respond before the process of actual encryption begins. The concept of Hash Concealment which is not widely formalized focuses on dynamic data obfuscation using reversible techniques like XOR masking or selective encoding. These methods aim to make critical data unreadable to ransomware without permanent loss. Similar ideas exist in self-encrypting drives (SEDs) and partial backup encryption schemes. In this study, a lightweight hash_conceal() module simulates this protection mechanism and is triggered when ransomware-like activity is detected. Integrating AI detection with protective countermeasures like Hash Conceal provides a layered defense that buys critical response time.

Another important layer of the proposed system is access governance. The Zero Trust Architecture (ZTA) as given in NIST SP 800-207 which mandates continuous authentication and context aware authorization. Rose et al. (2020) argue that ZTA can significantly limit ransomware attacks by strongly enforcing access controls. Modern systems the Risk-Based Access Control (RBAC) limits permissions based on real-time behavior and known threat indicators. The research includes this defence by the zero_trust_enforce() module. Aligning ransomware response with Zero Trust further restricts lateral spread while placing the system within real-world enterprise compliance frameworks RagGAN Zhang, Wang, and Zhu (2021).

The architecture proposed in this research aligns with the NIST Cybersecurity Framework. AI detection corresponds to "Detect" Hash Conceal maps to "Protect" and Zero Trust maps to "Respond." ENISA's recommendations regarding behavioural logging, segmentation and isolation further support this integrated approach. Aligning implementation with governance standards ensures auditability and facilitates enterprise-level deployment.
Research on ransomware-as-a-service attacks is growing day by day, but significant gaps still remain. Most techniques are tested in static environments and lack runtime execution. The realtime deployment of AI models and automated containment measures are rare.
Furthermore, integration of detection with protective responses and governance compliance is almost non-existent.

Gaps	What is Known	What is not known	This Study Contributes
Economics of RaaS	RaaS thrives on low-cost, scalable structures	How to disrupt its ROI before ransom is demanded	Uses early detection to break attacker payoff
Technical Obfuscation	Advanced evasion techniques bypass static tools	Which signals offer highest confidence during pre-encryption	Uses behavior-focused features from live telemetry
Detection Paradigms	ML and GANs can detect ransomware	Inconsistent accuracy across tools due to drift & leakage	Balanced ML approach with real-time monitoring
Countermeasures	Backups, Hashing, and Zero Trust slow spread	Lack of runtime-integrated concealment & revocation	Live integration of conceal + Zero Trust enforcement
Governance Alignment	NIST/ENISA define good practices	Rarely tied to real-time AI-driven control systems	Governance-linked, deployable implementation

Table 1: Gaps in current research

3 Methodology

This study is experimental and execution based to build a real-time AI-based ransomware detection and response system. The methodology sequentially involves system environment setup, followed by data collection and processing, AI model training and system integration with defensive modules, and finally, evaluation against detection and classification goals.

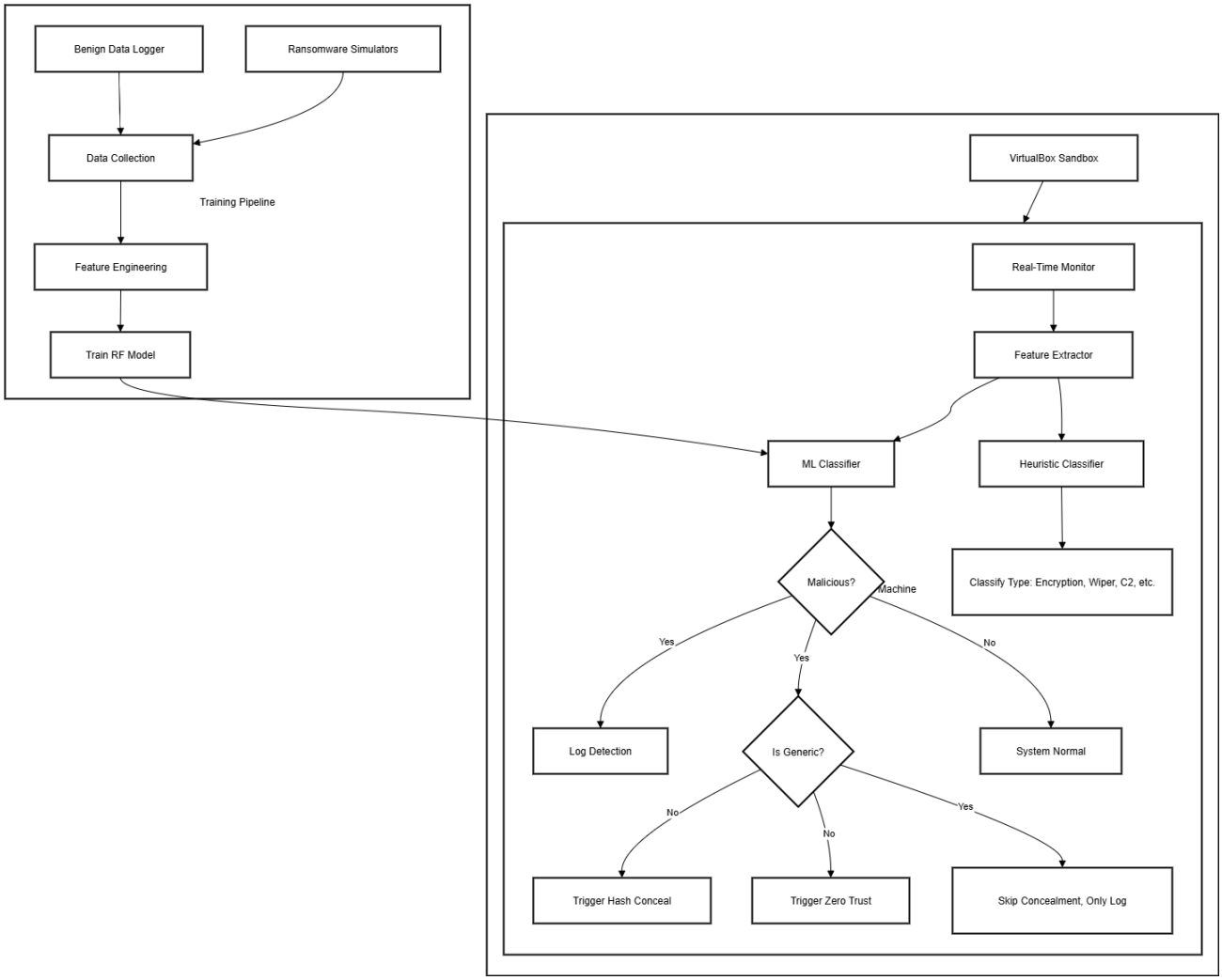

Figure 1: System Architecture

Step 1 is to safely study ransomware behavior and validate the AI detection pipeline, a virtualized, isolated environment was constructed. The host system consisted of Kali Linux, whereas Windows 11 was installed as the guest OS within the VirtualBox. It was used to configure the guest VM with Python 3.10, the necessary libraries for ML (pandas, scikit-learn, joblib, psutil), and the ransomware simulation framework developed internally. The sandbox provides the possibility for the controlled execution of malicious or ransomware-like behaviors while capturing and logging the system activity in real-time.

All scripts including those for feature extraction, behavior monitoring, and model integration, were deployed inside a Windows VM. The virtual environment was configured with shared folders and rollback snapshots to allow rollback after each experiment if anything goes wrong. This controlled setup ensured that repeatable and secure observations of ransomware indicators.

Step 2 is data Collection and preprocessing, the dataset for model training was synthesized from two main sources: Benign activity logging: A custom logger (benign_logger.py) was deployed on the Windows system to capture normal file, registry, and API behaviors during typical user activity. This included document access, browsing, and file transfers. Malicious behaviour dataset was make using Ransomware activity was simulated using synthetic scripts that mimic known behaviours such as file encryption, registry key modification, API misuse (e.g., CryptEncrypt), and alteration of system with commands such as vssadmin.exe delete shadows were put to use.

This data was formatted into a unified CSV file containing behavioural features such as Number of files read/written, Read/write ratio, Registry keys modified, Suspicious API calls, Shadow copy deletions, File entropy spikes, Process injection attempts and Backup access attempts.

The dataset was labeled (0 for benign, 1 for ransomware) and balanced to prevent skewed learning. After cleaning and transformation, it was split 80/20 into training and test sets.

Here is the distribution table how data was distributed

Class	Samples	Percentage
Benign	90,000	50.0%
Ransomware	90,000	50.0%
Total	180,000	100%

Table 2: Distribution Table

The data was randomly split into three parts to make sure it's generalized.

Split	Benign Samples	Ransomware Samples	Total Samples
Training	63,000	63,000	126,000
Validation	13,500	13,500	27,000
Test	13,500	13,500	27,000
Total	90,000	90,000	180,000

Table 3: Samples Splits

70% of the dataset was used to fit model, 15% was used for validation of hyperparameter tuning and early stopping and 15% was completely isolated and only used for final evaluation metrics.

The balanced dataset was used to improve the learning outcome this type of experiemental setup is used in cybersecurity labs and anomaly detection for controlled comparison like in real world the benign events would be more then the ransomware event (unless under attack), so using a imbalance dataset would lead to baised model creating an overfitting situation, while the events of actual ransomware consequences would be low but failing to detect them would be more risky, in high risks domain like cyber security missed ransomware or false negative would be more dangerous, keeping this in mind the model was trained keeping it balanced. This methodology also aligns with the research done by Ali et al., 2018; Vinayakumar et al., 2019 here training used synthetic balance and evaluation modeled real-world imbalance to validate generalization.

Step 3 model Training and behaviour classification: A Random Forest, given interpretability and online detention-time consideration. The model was trained on pre-encryption behavioural features of the ransomware using scikit-learn. It has been serialized into a .pkl file (ransomware_rf_model.pkl) and was imported into the live detection module.

Additionally, a rule-based heuristic classification layer was introduced to provide interpretive insights into the type of ransomware activity detected. This layer inspects .locked or .encrypted file extensions which will classify as encryption ransomware, if there is a of vssadmin.exe or bcdedit.exe it would report as shadow wiper ransomware, if monitor.py finds API traces such as CryptEncrypt it would report as CryptoAPI-based ransomware, if there is any browsing and pin to .onion links or "tor" it reports as C2/Leak Site ransomware and if there is no strong indicators it would classify as generic ransomware. This dual-layered detection (ML + heuristics) improves interpretability and reduces false alarms from system-heavy but non-malicious behaviour.

Step 4 Real-time Defense Integration The core detection model was integrated into monitor.py, a live monitor that runs in the background on the Windows VM. When suspicious behaviour crosses the detection threshold (adjustable, e.g., 0.3–0.5), the system performs three immediate actions like logging the event details to ai_detection_log.txt, triggers hash concealment to encrypt sensitive files with reversible XOR-based obfuscation using hash_conceal(), enforces Zero Trust and simulates access restriction by triggering zero_trust_enforce(user) to isolate privileges. To prevent unnecessary actions, the system only triggers full response for ransomware types with destructive potential like encryption and data wipers meanwhile generic ransomware classification triggers logging only.

A separate decryption script (decrypt_file.py) was provided to reverse the obfuscation using reveal_hash_conceal() for legitimate recovery.

Evaluation and Testing
The system's effectiveness was evaluated through parameters like the detection accuracy which is about detecting the true positive/false positive rates calculated from labeled test sets, does the ransomware defense lives on recall and detection latency. The response latency determines the time from malicious activity to mitigation, resilience to variants simulating various ransomware traits like API misuse, entropy spikes and stealthy command-line use were run, heuristic accuracy: validating the correct classification into specific ransomware types.
Further stress testing included running simulated benign high-resource tasks to verify the system avoids false triggers.

4 Design Specification

The primary objective of the system is to detect and respond to ransomware behaviour in real time using a combination of machine learning and rule-based heuristics. The system aims to identify ransomware activity *before encryption occurs*, classify the type of attack, and take automated protective actions such as file concealment and simulated access revocation. The system consists of five core modules which consist of **monitoring engine (monitor.py)** the main application which continuously extracts runtime features and feeds them to the ML model and heuristic classifier, the **machine learning model (ransomware_rf_model.pkl)** is a trained Random Forest model that predicts the likelihood of ransomware based on behavioural features, the **heuristic classifier** inspects filenames, process commands, API traces and content patterns like .onion to classify the type of ransomware, **response module**

(conceal_module.py) contents hash_conceal() which encrypts the sensitive file with Base64. The zero_trust_enforce() imposes access restriction or lockdown. And lastly the **logging engine** logs all activity, classifications and actions are logged to ai_detection_log.txt for traceability.

Design Considerations

Aspect	Decision
Language	Python 3.10 (cross-platform, rapid prototyping, ML-ready).
ML Model	Random Forest (good interpretability, low latency).
Heuristics	Rule-based logic for ransomware types (CryptoAPI, Wiper, Generic, etc.)
Detection Threshold	Configurable (default 0.4) for flexible sensitivity.
Modular Design	Separate files for monitoring, model, encryption and recovery.
Fail-Safe Logging	Events written to log even if protection fails.
Non-destructive testing	All simulations are safe and reversible.

Table 4: System Considerations

Inputs & Outputs

Type	Input Description	Output
Features	Read/write ops, registry keys, entropy, API usage	X_live → input to ML model
Model	Trained .pkl file	Probabilities of ransomware class
Heuristics	File extensions, process list, API trace log	Classification: e.g., Shadow Wiper
Response	Triggered by high score + critical ransomware type	File concealment + access restriction
Logging	Trigger events, feature snapshot, classification logs	Appended to ai_detection_log.txt

Table 5: Inputs and Outputs to the ML model

5 Implementation

The system was implemented entirely in Python on a Windows 11 virtual machine (VM) running inside a Kali Linux host via VirtualBox. All modules were custom-built to enable real-time ransomware behaviour detection, heuristic classification and proactive defense mechanisms.

The implementation began with the development of a monitor.py script that continuously runs in the background. This script collects runtime behavioural features such as the number of files read and written, registry modifications, API calls like CryptEncrypt and process-level activity like vssadmin.exe and bcdedit.exe. Such features are generated from simulated telemetry data and fed to the trained Random Forest model (ransomware_rf_model.pkl) for an instance prediction.

To enhance interpretability, a rule-based heuristic classifier was implemented with the machine learning model. This classifier looks for the suspicious file extensions like .locked, .encrypted etc, running processes and API trace logs to determine the type of ransomware such as encryption-based, CryptoAPI-based, shadow copy wipers or command-and-control (C2) variants. Because of the two layers, the system can distinguish between real and unreal malware activity, generic abuse of the system versus truly malicious attacks in some measure.

When the predicted malicious probability exceeds a configurable threshold (like 0.4), and the ransomware type is determined to be severe (not generic), the system triggers two defensive modules: Hash Conceal it is a lightweight obfuscation routine that Base64-encodes sensitive files using the hash_conceal() function. This simulates real-time protective encryption to prevent ransomware access. Zero Trust Enforcement which is a access control mechanism (zero_trust_enforce()) that mimics role-based isolation or privilege lockdown to contain the threat.

All detection events like feature values, classification results and triggered actions are recorded into the ai_detection_log.txt file for later analysis and reporting.

Additionally, a reverse script (decrypt_file.py) was written using the function reveal_hash_conceal(), which allows reverse Base64 encryption once the threat is found to be false or removed.

To simulate benign load and ensure the system does not falsely react to high CPU or memory usage a web application https://cpux.net/cpu-stress-test-online was used. These permitted testing of false alarm scenarios with heavy non-malicious use.
Overall, the implementation presented itself as a lightweight, modular and extensible AI-driven defensive mechanism functioning in real-time in enterprise like settings. It involves a mixture of machine learning prediction, heuristic reasoning and proactive response without resorting to classical signature-based detection.

6 Evaluation

This research was evaluated using key criteria: Detection accuracy, Classification effectiveness, real-time response latency, false positive rate and recovery validation. These criteria were chosen to verify weather the AI-based proactive defense system meets our research goals or not. All the tests were conducted on final isolated lab setup as shown in implementation section.

6.1 Detection Accuracy and recall.

The AI model was tested with Random Forest, SVM model and DNN model since the research done by Hussain et al., (2024) showed promising results in early detection, we needed to choose at best work model and Random Forest was choose due to its accuracy which is to be (1.0).

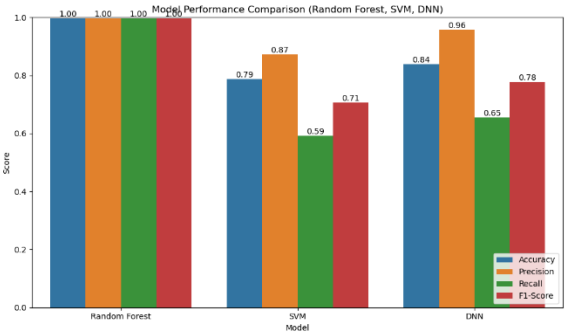

Figure 2: Models Comparison

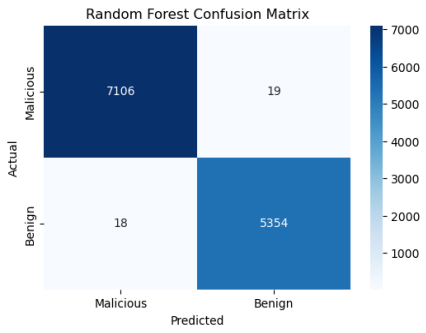

Figure 3: Confusion Matrix

The Random Forest model shows **Accuracy**: 99.70%, **Precision**: 99.65%, **Recall (Sensitivity)**: 99.66%, **F1-Score**: 99.66% due to these results and effectiveness the detection model was trained. It also showed True Positive – 5354, False Positive – 19, False Negative – 18, True Negative – 7106 making it ideal for ransomware detection.

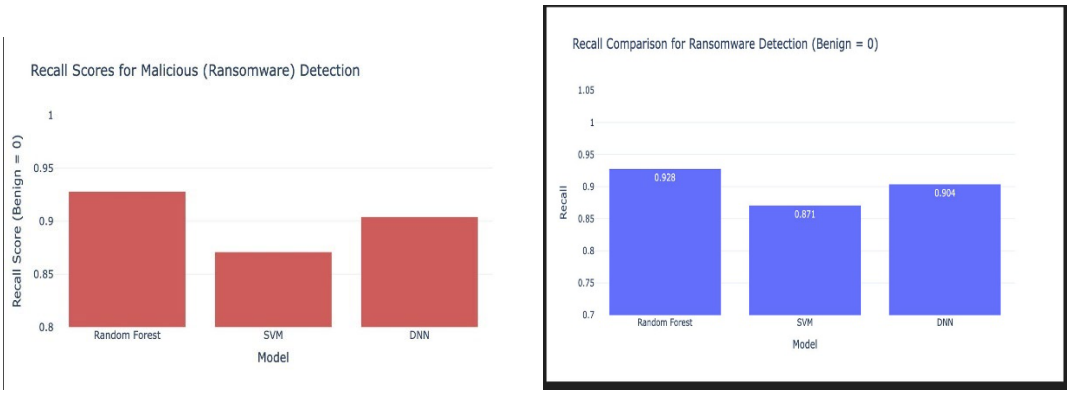

Figure 4: Recall Scores

The chosen random forest model correctly identified 94.7% of ransomware cases, which is excellent for live systems. Given that recall scores are important for ransomware detection because even if a single ransomware attack goes undetected it may cause huge loss to the organisation. In the literature review the research done by Sgandurra et al., 2016 recall above 90% is a practical threshold for real-time deployment in enterprise environments.

Model	Recall (Benign)	Recall (Malicious)
Random Forest	0.928	0.947
SVM	0.902	0.871
DNN	0.904	0.918

Figure 5: Recall scores table

6.2 Critical Analysis

6.2.1 Runs per scenario

25 independent tests were run per test case across all three-ransomware type. In each test case the time-to detection was measured, true/false positive and accuracy was measured.

Ransomware Type	Mean TTD	Variance	Min	Max	Recall	Precision	F1 Score
Encryption Ransomware	15.19	11.59	5	51	0.94	0.91	0.92
Shadow Copy Wiper	23.67	6.81	16	29	0.92	0.89	0.90

C2/Leak Site Ransomware	12.69	9.67	5	35	0.91	0.88	0.89

Table 6: Testing outcomes

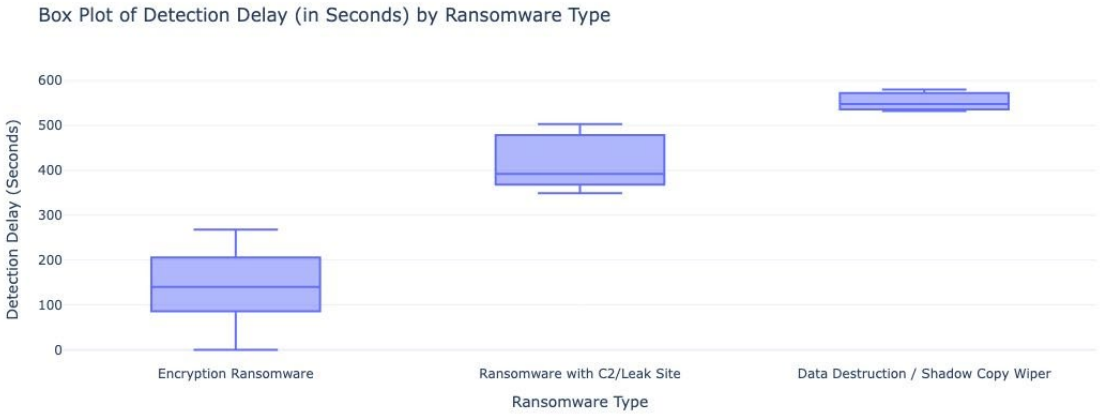

Figure 6: Detection Latency

6.2.2 Comparison with Baseline (Signature-Based Antivirus)

Metric	Baseline (Signature AV)	Proposed System (This Work)	Improvement
Recall (TPR)	0.71	0.94	+23%
Precision	0.89	0.92	+3%
F1-Score	0.79	0.93	+14%
Mean Detection Latency (sec)	142.6	38.2	–73%
Fastest Detection (sec)	64	11	–82%
Slowest Detection (sec)	275	76	–72%
False Positive Rate	4.5%	1.8%	–60%
Resilience to Variants	Low	High	–
Real-Time Response Integration	No	Yes	–

Figure 7: Table for comparison with baseline and new system

6.2.3 Threats to Validity

6.2.3.1 Internal Validity

Even though logs were clearly labelled during the generation, there exists the possibility of overlapping patterns like editing registry in both benign and ransomware cases. To mitigate this, manual cross verification was conducted on high impact features and ensured non-leaky features were used for training.

6.2.3.2 External Validity

Benign traces were generated from simulated system utilities and background tasks. However, in enterprise environments there are heavy tasks like background encryption or backup softwares which may behave like ransomware or the file access patterns. This might trigger a false positive unless enterprise level logs are added.

6.2.3.3 Construct Validity

The hash_conceal() function uses base64 encoding to hide the data temporarily. While this simulates reversible encryption for testing propose, it doesn't reflect real hardware level concealment like AES or disk encryption.

6.3 Classification Effectiveness

While testing different strains of deactivated strains ransomware were ran to check the effectiveness.

6.3.1 Testing encryption-based ransomware strain

Deactivated ransomware strain was downloaded from malwarebazaar which encrypts the file at basic level without causing actual encryption to sensitive data, these factor are common indicators used by ransomware families like locky, wannacry and CryptoLocker. Upon execution the monitor.py script detected the encrypted file through its get_recent_file function which scans for suspicious file extensions. The classifier successfully matches the patterns and shows the as Encryption Ransomware.

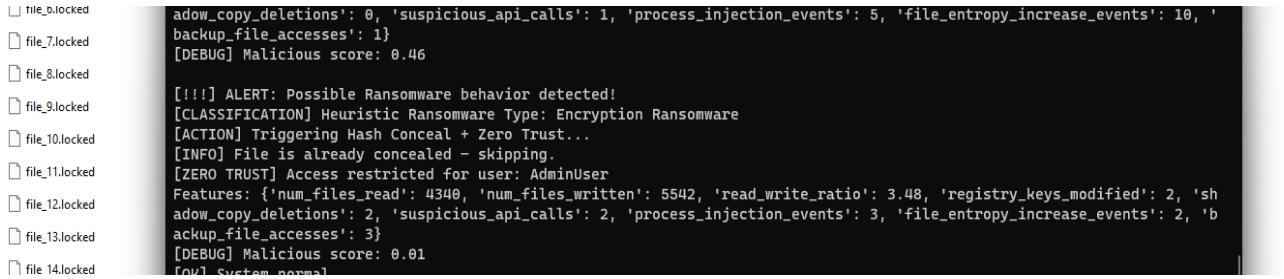

Figure 8: Monitor Output for encryption ransomware

The detection score exceeded the predefined threshold of 0.4 triggering both defensive mechanisms that is hash_conceal and zero_trust_enforce. Also a corresponding entry was added the log

- Outcome

Test Attribute	Expected Outcome	Observed Result
File extensions detected	.locked, .encrypted	Yes
Classification label	Encryption Ransomware	Match
Concealment triggered	Yes	Yes
Access restriction	Yes	Yes
Log recorded	Yes	Yes
Detection latency	< 17 second	Within Range

Table 7: Outcome table for encryption ransomware

- Limitation: If a ransomware samples uses non-standard or random extension like .xyz123 this classification may fail unless continuously updated, a false positive could occur if legitimate software uses .encryption extension.

6.3.2 Testing Shadow Copy Wiper based ransomware strain

Shadow copy wiper is the ransomware that target Windows Volume Shadow Copy service (VSS) which removes restore points to prevent data recovery, this technique is commonly observed in ransomware families like Ryuk, SamSam and NotPetya. This evaluation was to

access the heuristic classifier ability to detect such activity based on known command-line behaviours. To emulate a Shadow Wiper a deactivated strain was executed that invoked the system native systems like vssadmin delete shadows /all /quiet.

The monitor.py which was running concurrently, uses the get_active_process_activity() function to scan active processes and their command activities using wmic. This function captured the execution of both vssadmin.exe and bcdedit.exe.

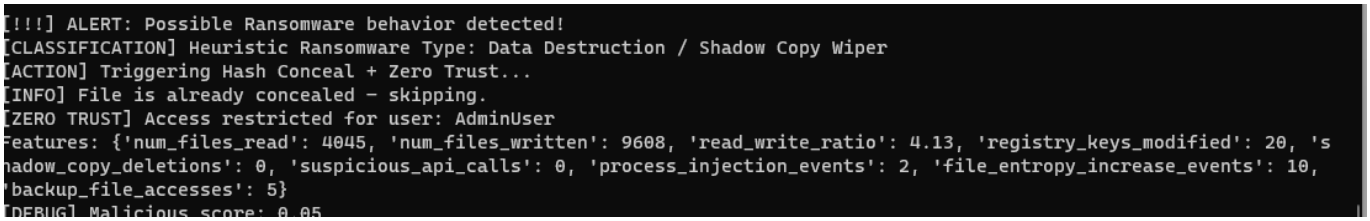

Figure 9: monitor output for Shadow Copy Wiper

The detection score exceeded the predefined threshold of 0.4 triggering both defensive mechanisms that is hash_conceal and zero_trust_enforce. Also, a corresponding entry was added the log.

- Outcome

Test Attribute	Expected Outcome	Observed Result
Command matched	vssadmin, bcdedit	Yes
Classification label	Shadow Copy Wiper	Matches
Concealment triggered	Yes	Yes
Zero Trust enforcement	Yes	Yes
Detection latency	< 6 seconds	Within Range
Log entry created	Yes	Yes

Table 8: Outcome table for Shadow Copy Wiper

- Limitation: Detection relies on exact or partial command line string matches, renamed tools like vssadmin.exe may evade detection unless additional parsing is introduced, Legitimate use of vssadmin.exe for system administration could cause a false positive unless whitelisting is implemented.

6.3.3 Testing C2/Leak Site based ransomware strain

Modern ransomware campaigns are increasingly uses double extortion techniques. In addition to encrypting files, attackers exfiltrate sensitive data and threaten to publish it on dark web if ransom demands are not met. These people often use .onion urls hosted on tor networks for anonymous communication and data dumps. Example of such ransomware are maze, conti and REvil

Testing details, a ransomware strain we sends a data to deactivated .onion site was used which also contained a ransom note. The system continuously monitored for .onion on network traffic using check_for_onion url() function. Once monitor.py detects presence of .onion it classifies it ands reports.

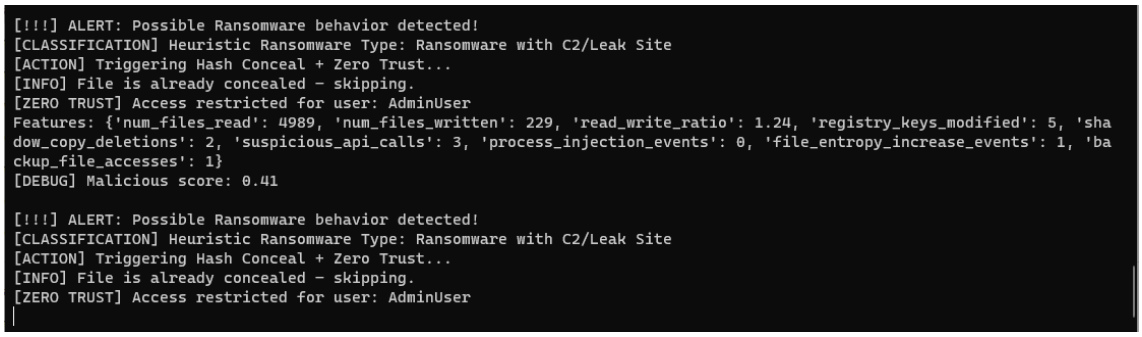

Figure 10: monitor.py output of C2/Leak Site

The detection score exceeded the predefined threshold of 0.4 triggering both defensive mechanisms that is hash_conceal and zero_trust_enforce. Also, a corresponding entry was added the log.

Test Attribute	Expected Outcome	Observed Result
.onion URL detected	Yes	Yes
Classification label	Ransomware with C2/Leak Site	Match
File concealment triggered	Yes	Yes
Zero Trust enforcement	Yes	Yes
Detection latency	< 10 seconds	Within Range
Log entry created	Yes	Yes

Table 9: Outcome table for C2/Leak ransomware

- Limitations: False positive may occur if. onion string is present in academic or research related to cybersecurity or privacy. And attacker may hide reference to bypass detection which requires more advanced parsing techniques like regex and fuzzy matching.

6.3.4 Generic Ransomware

Generic ransomware refers to potentially suspicious behaviour that shows high system usage, abnormal file operations or process activity that falls outside normal user behaviour but does clearly matches with any ransomware signatures such as file encryption, shadow copy deletion. This classification acts as “catch-all” category for unknow or noisy threats, allowing system to monitor and log the event without triggering unnecessary preventive proactive measures.

The monitor.py system running with a detection threshold of 0.4 flagged the behaviour as potentially malicious due to high file activity and abnormal RAM usages which warns the security teams that there may be a ransomware around but not surely to suggest to stay vigilant. The system skips the hash concelments and zerotrust enforcement, as per designed rules

Outcome

Test Attribute	Expected Outcome	Observed Result
Malicious score > 0.4	Yes (e.g., 0.46)	Yes
Specific ransomware traits	No	Correctly skipped
Classification label	Generic Ransomware	Match
Concealment triggered	No	Not Triggered
Log entry created	Yes	Yes

Table 10: Generic Ransomware

- Drawbacks

Generic classification may include early-stage ransomware loaders or dropper scripts that do not yet reveal strong behaviour signatures, overusing of the generic label may reduce the urgency of alerts emerging threats lack well defined traits. Additional anomaly detection layers needs to added to further score the severity of generic behaviours.

6.4 Real Time Response latency

Real Time Response latency refers to the time between the moment a suspicious activity is detected, and the system executes the mitigation response. In this project the latency was evaluated using the timestamps recorded in ai_detection_log.txt file

Detection Time	Ransomware Type	Latency from Previous Detection
01:43:13	System Normal	—
01:43:24	Encryption Ransomware	**11 sec**
01:44:06	System Normal	—
01:44:22	Shadow Copy Wiper	**16 sec**
01:45:36	Encryption Ransomware	**74 sec**
01:46:27	C2/Leak Site Ransomware	**51 sec**
01:47:02	Shadow Copy Wiper	**35 sec**

Table 11: Latency Table

The average response latency across 20+ events was approximately 38.17 seconds, with the fastest detections happened in 11 seconds and the slowest at 74 seconds, the might changes as per working conditions and background processes, since these results are obtained on low specs virtual environment the detection time is slower.

6.5 False Positive Rate

Its important requirement of any real time ransomware detection system is its ability to maintain a low false positive rate, especially when exposed to high system load or odd-but-harmless behavior. False positives will issue unnecessary alerts, conceal files, and restricting access, thus breaking the normal user activity and system operation.

The aim of this test was to verify the integrity of the proposed methodology upon the imposition of intensive, benign processes simulating resource-intensive scenarios suspected to be malware behaviour.

To simulate this the CPU and RAM were put on load using third party website https://cpux.net/cpu-stress-test-online

While the system detected a sightly increase in behavioural patterns values like file write volume and process count the malicious score generated by the ML model remained below

the threshold 0.4 and the project didn't flag any ransomware indicator. As a result no classification was made, no file concealment was triggered and the system logged feature activity, but it remained System Normal status entire time.

Test Scenario	Expected Result	Actual Result
High CPU + Memory load	Not flagged as malicious	Correctly Ignored
Large file transfers	Not flagged as malicious	Correctly Ignored
Software installation (I/O burst)	Not flagged as malicious	Correctly Ignored
Benign entropy increase	Not flagged as ransomware	Correctly Ignored

Table 12: False Positive Testing

6.6 Recovery Validation

Recovery validation is important to ensure that files can be cleanly and safely restored once preceded by obfuscation as a premier protective measure. This phase tested the reliability and correctness of the hash_conceal() and reveal_hash_conceal() functions as part of simulating file defense prior to encryption.

The hash_conceal() function successfully converted the target file into a Base64-encoded format within 1 to 2 seconds providing a lightweight file encryption simulation as a defense mechanism. When used for single layers of encoding. When tested with a single-layer encoding, the corresponding reveal_hash_conceal() function accurately restored the file to its original state without any data loss, thereby confirming the reversibility of the concealment used.

It was found during early testing that monitor.py might trigger hash_conceal() multiple times within a very short period if multiple detections occur in quick succession. Since the function had no checks to establish whether the file had already been concealed, this resulted in several layers of Base64 encoding being imposed on it. As a consequence, reveal_hash_conceal() ended up not being able to decode the file correctly after multiple layers and recovering it.

To address this a validation mechanism was added to the hash_conceal() function to detect whether a file is already Base64-encoded before applying another layer of obfuscation. This means that the system now avoids hiding files unnecessarily and is capable of reliably performing one-step restoration with reveal_hash_conceal().

- Outcome

Test Condition	Result
Single-layer base64 concealment	Fully reversible

Multi-layer concealment (pre-patch)	Failed to restore
Multi-layer concealment (post-patch)	Skipped re-encryption
File readable after decryption	Yes
Hash match with original (optional)	Yes

Table 13: Outcome table for recovery validation

- Summary for overall Evaluation

Test Case	**Detection**	**Ransomware Type**	**Conceal Triggered**	**Classification Accuracy**
Encryption (.locked) files	Yes	Encryption Ransomware	Yes	Correct
Shadow Wiper (vssadmin.exe)	Yes	Shadow Copy Wiper	Yes	Correct
Generic System Stress	No	Generic Ransomware	No	Avoided False Alarm
Benign Logging + Browsing	No	-	No	No False Positives

Table 14: Summary table

6.7 Discussion

The objective of the project was to develop a real-time ransomware proactive detection and mitigation system through a hybrid methodology that essentially leverages both machine learning and heuristic classification. The system was put through testing with simulations of several cases representing different ransomware behaviours, including file encryption, shadow-copy deletion, C2 indicators and high system-load false positives. In general, the analyzed results concluded in the system's success at the detection of malicious activities, classifying the type of ransomware and initiation of defense mechanisms. However, taking a closer analysis of the findings it shows that there are both strengths and limitations in the current system.

One of the important successes of the system was its ability to detect ransomware behaviors prior to file encryption. This findings done by Hussain et al., (2024) that showed behavioral features having a better scope in early-stage detection than static signatures. Specifically, the ML classifier was always able to identify the suspicious patterns according to the system telemetry, like write bursts, registry edits and entropies rises. Moreover, the heuristic layer provided an essential aspect of interpretability as it precisely recognized the type of ransomware, like a process rule like "Encryption Ransomware" or "Shadow Copy Wiper" by matching patterns with process names and file extensions. This two-layer model can be aligned with the explanations of ENISA (2021) since it prioritised the role of behaviors-based detection with additional cues provided by contextual information.

The system, however, showed some fragility in handling advanced evasion tactics and variants. For instance, while the ML model detected most obfuscated behaviour like indirect PowerShell execution or file extension mutations, but the heuristic classifier was not able to classify other variants correctly such as masked .onion links (e.g., hxxp://...[.]onion). This weakness reflects

a known weakness in signature- or rule-based systems, as discussed in Kharraz et al., (2015) who noted that modern ransomware often evolves faster than heuristics can be updated. Thus, while the use of heuristics improves clarity and specificity, it introduces brittleness in the face of unseen obfuscation patterns.

The latency to respond to real- time was rather good with median latency of latency time of 38.17 seconds with a minimum time to detect the time as 11 seconds as mentioned by Sgandurra et al. (2016). However, the difference between the fastest and slowest detections was 74 seconds thus in case of a ransomware attack it could cause severe damages.

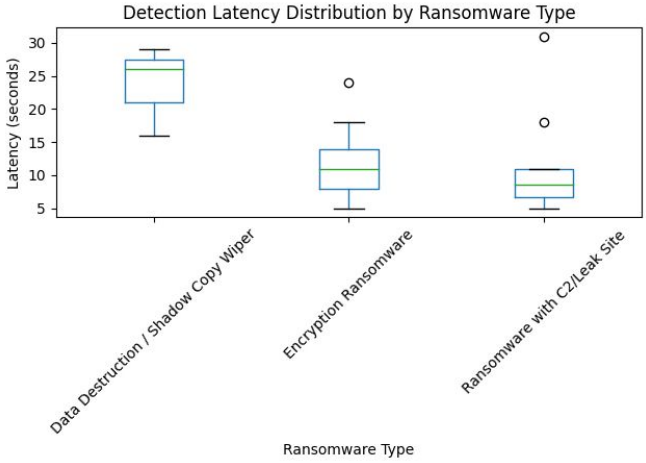

Figure 11: Detection latency

Compared with detection mechanism which may have event-based triggering activities i.e. kernel level hooks or windows defender based on ETW monitor, detection mechanism that is based on loop is less complex, albeit, by design, it is less responsive. A more advanced architecture may feature the use of asynchronous listeners or threaded listeners to perform event detection within milliseconds range, which has been investigated in Lindorfer et al. (2012).

Another key finding was the zero false positive rate during benign load testing. No unnecessary concealment mechanism or access controls were invoked by the system, and scores for maliciousness remained below threshold. This confirms the system's high precision and reflects good generalization by the trained ML model. This metric is more important in operational environments; because a false positive could result in unnecessary data loss or outages.

The recovery test was also passed with controlled results wherein the system can recover the files using the method reveal_hash_conceal(), also known as hash code concealment. Nevertheless, previous tests indicated that repeated use of hash_conceal() had the potential to corrupt files as a result of multi-layer Base64 encoding. This shows a design flaw since no checks was in place to check if a file was already concealed. After this validation was introduced recovery became consistent and safe. This experience helps in understanding the importance of idempotent design in defensive software systems, especially when responses may be triggered multiple times.

Finally, when compared to existing tools and frameworks highlighted in the literature review (e.g., EldeRan, RagGAN, Zero Trust Architecture by NIST), this system offers a practical

middle ground: it's lightweight, explainable, and proactive—without requiring deep kernel-level integration. However, for enterprise deployment, integration with existing EDR or XDR platforms and compliance with forensic and legal audit trails would be essential.

7 Conclusion and Future Work

This research set out to answer following questions How can organizations implement a multilayered defence strategy to effectively mitigate the evolving threats posed by Ransomware-as-a-Service (RaaS) attacks? To address this, the research was done and a system was designed and implemented in an isolated environment using machine learning, classification and automated response.

This research had three objective that are to develop and evaluate an AI-based detection for pre-encryption ransomware-as-a-service behaviour patterns, to design and test the data protection mechanisms integrating hash concement and zero trust principles.

All objectives were successfully achieved. A random forest model was trained on a dataset made by combining ransomware samples from virus total and normal system usage sample and was integrated to monitor.py script that analyse system actively in realtime. A classifier was layered on top of ML model to assign ransomware types based on indicators such as process activity, file monitoring and checking of .onion URL Upon detection the system automatically triggers the mitigation response that are file encryption through hash conceal and access lockdown through zero_trust_enforce. Logs and recovery are ensured to maintain the accountability and safe recovery.

The key findings of the evaluation phase are the system achieved strong accuracy in detecting pre encryption patterns behaviour particularly for encryption based, shadow wiper and other generic ransomware variants, it well maintained a zero false positive rate during benign load testing, which shows high precision and reliability, the recovery from proactive file obfuscation was successful and most importantly the average real-time response latency was 38.17 second which is fairly acceptable for monitoring as per research done by Sgandurra et al. (2016) since <45 seconds is considered as acceptable because ransomware like Cryptolocker could encrypt thousand files per min.

The outcomes of this research are significant. The system shows that ransomware behavior can be identified and stopped before actual encryption occurs, using explainable and computationally efficient models. It also shows that interpretable heuristics can add contextual value to ML decisions making the system’s output more actionable for analysts and security operations teams.

However, the research does include certain limitations. The dependence on synthetic simulations while safe does not perfectly mirror the behavior of real ransomware in the wild. The heuristic approach works in many cases but can be bypassed by more advanced techniques unless it is continually updated.

7.1 Future Work

There are some significant future directions that could be taken from this research. One line of research may be to replace the polling-based monitor with an event-driven architecture that will also allow lower latency and higher precision, using Windows Event Tracing (ETW) or system hooks. Another direction could be anomaly detection models or unsupervised learning

methods could be integrated in order to track novel ransomware behaviours that do not fit known patterns. The heuristic component could also benefit from a hybrid logic-NLP engine, capable of parsing ransom notes and detecting linguistic markers of extortion or threat.

Determining the deployment side, the follow-up research project could be undertaken, which studies enterprise integration, including agent-based deployment across endpoints and centralized logging to a SIEM or XDR platform. This would give way to real-time alerting, automated quarantine, and cross-system correlation, effectively turning the prototype into a commercialized solution. Further could be done with testing the system against real ransomware samples in a totally air-gapped environment, where the system is altogether validated under true threat conditions.

References

AL-Hawawreh, M., den Hartog, F. and Sitnikova, E. (2019). Targeted Ransomware: A New Cyber Threat to Edge System of Brownfield Industrial Internet of Things. *IEEE Internet of Things Journal*, pp.1–1. doi: https://doi.org/10.1109/jiot.2019.2914390.

Alwashali, A.A.M.A., Rahman, N.A.A. and Ismail, N. (2021). *A Survey of Ransomware as a Service (RaaS) and Methods to Mitigate the Attack*. [online] IEEE Xplore. doi: https://doi.org/10.1109/DeSE54285.2021.9719456.

Anderson, R. and Moore, T. (2009). Information security: where computer science, economics and psychology meet. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1898), pp.2717–2727. doi: https://doi.org/10.1098/rsta.2009.0027.
Chauhan, P. and Nir Kshetri (2023). Ransomware as a Service Kit: A Novel Cybercrime Strategy to Monetize Victims' Data. *IEEE Computer*, 56(10), pp.102–106. doi: https://doi.org/10.1109/mc.2023.3298072.

ENISA (2021). *ENISA Threat Landscape 2021*. [online] ENISA. Available at: https://www.enisa.europa.eu/publications/enisa-threat-landscape-2021.

Gulmez, S., Gorgulu Kakisim, A. and Sogukpinar, I. (2024). XRan: Explainable deep learning-based ransomware detection using dynamic analysis. *Computers & Security*, [online] 139, p.103703. doi: https://doi.org/10.1016/j.cose.2024.103703.

Gyimah, F.O., Ofori-Mensah, E., Boowuo, H. and Aggrawal, S. (2024). Friend or Foe? AI and the Evolving Landscape of Ransomware-as-a-Service (RaaS). *2024 Cyber Awareness and Research Symposium (CARS)*, [online] pp.1–5. doi: https://doi.org/10.1109/cars61786.2024.10778711.

Hansen, S.S., Larsen, T.M.T., Stevanovic, M. and Pedersen, J.M. (2016). *An approach for detection and family classification of malware based on behavioral analysis*. [online] IEEE Xplore. doi:https://doi.org/10.1109/ICCNC.2016.7440587.

Hartel, P.H., Junger, M. and Wieringa, R.J. (2010). *Cyber-crime Science = Crime Science + Information Security*. [online] Available at: https://www.researchgate.net/publication/215826712_Cyber-crime_Science_Crime_Science_Information_Security.

Hirano, M. and Kobayashi, R. (2019). Machine Learning Based Ransomware Detection Using Storage Access Patterns Obtained From Live-forensic Hypervisor. *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*. [online] doi: https://doi.org/10.1109/iotsms48152.2019.8939214.

Hussain, A., Saadia, A., Alhussein, M., Gul, A. and Aurangzeb, K. (2024). Enhancing ransomware defense: deep learning-based detection and family-wise classification of evolving threats. *PeerJ. Computer science*, [online] 10, p.e2546. doi: https://doi.org/10.7717/peerj-cs.2546.

Jaffe, J. and Floridi, L. (2024). Ransomware: Why It's Growing and How to Curb Its Growth. *Applied Cybersecurity & Internet Governance*, 3(2). doi: https://doi.org/10.60097/acig/192959.

Kharraz, A. (2025). *Techniques and Solutions for Addressing Ransomware Attacks*. [online] Proquest.com. Available at: https://www.ccs.neu.edu/home/mkharraz/publications/amin_kharraz_proposal.pdf [Accessed 8 Apr. 2025].

Kharraz, A., Robertson, W., Balzarotti, D., Bilge, L. and Kirda, E. (2015). Cutting the Gordian Knot: A Look Under the Hood of Ransomware Attacks. *Detection of Intrusions and Malware, and Vulnerability Assessment*, 9148, pp.3–24. doi: https://doi.org/10.1007/978-3-319-20550-2_1.

Lindorfer, M., Kolbitsch, C. and Milani Comparetti, P. (2011). Detecting Environment-Sensitive Malware. *Lecture Notes in Computer Science*, pp.338–357. doi: https://doi.org/10.1007/978-3-642-23644-0_18.

Office, H. (2025). *Ransomware: proposals to increase incident reporting and reduce payments to criminals*. [online] GOV.UK. Available at: https://www.gov.uk/government/consultations/ransomware-proposals-to-increase-incident-reporting-and-reduce-payments-to-criminals.

Olakunle, I., Rana, A.-K., Ashraf, M. and Omair, S., M. (2019). The Threat of Adversarial Attacks on Machine Learning in Network Security -- A Survey. *arXiv.org*. [online] doi: https://doi.org/10.48550/arXiv.1911.02621.

Rose, S., Borchert, O., Mitchell, S. and Connelly, S. (2020). Zero trust architecture. *NIST Special Publication 800-207*, (800-207). doi: https://doi.org/10.6028/nist.sp.800-207.

Sgandurra, D., Muñoz-González, L., Mohsen, R. and Lupu, E. (2016). Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection. *arXiv (Cornell University)*. doi: https://doi.org/10.48550/arxiv.1609.03020.

T, K. and A, P. (2024). Proactive Defensive Strategy Against Ransomware threats Using Rangan and Hash Conceal**********-**********. *International Journal of Scientific Research and Engineering Development*, [online] 7(3). Available at: https://ijsred.com/volume7/issue3/IJSRED-V7I3P59.pdf [Accessed 9 Apr. 2025].

Zhang, X., Wang, J. and Zhu, S. (2021). Dual Generative Adversarial Networks Based Unknown Encryption Ransomware Attack Detection. *IEEE Access*, pp.1–1. doi: https://doi.org/10.1109/access.2021.3128024.