AI-Driven Adaptive Security and Recovery Pipeline

BITS ZG628T: Dissertation

by

Dabhade Pooja Bhanudas
2023MT93089

Dissertation work carried out at
BITS PILANI

Submitted in partial fulfilment of **M.Tech. Software Engineering** degree programme

Under the Supervision of

Rupam Kumar Kundu

Dell Technologies, Bangalore

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE

PILANI (RAJASTHAN)

(March 2025)

ABSTRACT

Introduction:

In an era of rapidly evolving cyber threats and increasing system vulnerabilities, ensuring robust security and resilience for software systems is paramount. Current cybersecurity solutions often focus on specific aspects such as endpoint protection, centralized log management, or malware detection, leaving gaps in handling diverse attack vectors and ensuring seamless recovery. This project aims to address these limitations by developing an AI-driven Adaptive Security and Recovery Pipeline (ASRP) that integrates monitoring, detection, quarantine, and resolution mechanisms for comprehensive protection against a wide range of attacks, including digital, ransomware, system hardware failures, and software vulnerabilities.

Objective:

The primary objective of the project is to design and implement a modular, scalable, and AI-powered pipeline capable of:

1. Monitoring systems for suspicious activities across hardware, software, and networks.
2. Detecting fraudulent or malicious activities in real-time.
3. Quarantining compromised binaries and isolating affected systems autonomously.
4. Resolving incidents by automatically replacing compromised components with verified, secure versions, ensuring minimal downtime and system integrity.

Proposed Methodology:

The project will consist of the following four stages:

1. **Monitoring:**
    - Utilize AI-based anomaly detection models (e.g., LSTMs) to monitor system logs, network traffic, and hardware performance.
    - Integrate real-time dashboards for visualizing system health and potential threats.
    - **Tools:** Elastic Stack (ELK) for log monitoring, Grafana for dashboards, and Prometheus for performance metrics.
2. **Detection:**
    - Implement supervised and unsupervised machine learning models to classify and prioritize threats based on severity and potential impact.
    - Develop a hybrid detection system combining signature-based and behavioural analytics.
    - **Tools:** TensorFlow or PyTorch for ML model implementation, Snort or Suricata for signature-based detection, and OpenAI API for behavioural analysis.
3. **Quarantine:**
    - Introduce a sandboxing mechanism for isolating suspected binaries and processes.

- Maintain a secure repository of verified binaries using digital signatures and hash verification or certificate-based validation to ensure data integrity. Additionally, blockchain integration can be considered as an advanced option for tamper-proof storage and management.
- **Tools:** Firejail or Cuckoo Sandbox for process isolation, HashiCorp Vault for secure repository management, and OpenSSL for digital signature verification.

4. **Resolution:**
    - Employ containerization (e.g., Docker) for rapid deployment of fresh, secure components.
    - Implement automated rollback mechanisms using immutable snapshots and transaction logs.
    - Use AI-driven decision-making to prioritize and execute recovery actions.
    - **Tools:** Docker or Kubernetes for containerization, AWS Backup for snapshots, and AI-based decision engines like IBM Watson for recovery actions.

Key Differentiators:

1. **Holistic Approach:** Unlike existing solutions, the ASRP addresses a broad spectrum of attack types, including hardware failures.
2. **Autonomous Recovery:** Integrates self-healing mechanisms with minimal human intervention.
3. **Predictive Capabilities:** Incorporates machine learning models for early detection and prevention of potential zero-day attacks.
4. **Digital Signatures and Certificate Validation:** Ensures integrity and authenticity of critical data and binaries. Blockchain integration is also available as an advanced option for added security.
5. **Customizable Framework:** Offers a modular design to accommodate future expansions and organization-specific requirements.

Research Areas:

The project spans multiple research domains, including:

- Cybersecurity: Intrusion detection, ransomware resilience, and incident response.
- Artificial Intelligence: Anomaly detection, threat classification, and autonomous decision-making.
- Machine Learning: Supervised/unsupervised learning and ensemble methods.
- System Resilience: Self-healing systems and disaster recovery.
- Digital Signatures and Certificate Validation: Ensuring data integrity and secure backup storage. Blockchain: Advanced option for tamper-proof backup solutions.

Expected Outcomes:

1. A fully functional prototype of the Adaptive Security and Recovery Pipeline.
2. Demonstration of the pipeline's effectiveness in detecting, isolating, and recovering from diverse attack scenarios.
3. Comprehensive documentation and performance evaluation, comparing it with existing cybersecurity solutions.

Conclusion:

This project aims to bridge the gaps in current cybersecurity systems by delivering an innovative, AI-powered solution that enhances system resilience and security. By combining advanced monitoring, detection, quarantine, and recovery capabilities, the Adaptive Security and Recovery Pipeline will set a new standard in automated cybersecurity and recovery systems.

Signature of the Student:

Name: Dabhade Pooja Bhanudas

Place: Pune, Maharashtra

Date: 20th March 2025

Signature of the Supervisor:

Name: Rupam Kumar Kundu

Place: Bangalore, Karnataka

Date: 20th March 2025

Contents

1. Modules in AI-driven Adaptive Security and Recovery Pipeline:

Stage 1: Monitoring

Objective:
Simulate system monitoring for suspicious activity (e.g., checking file changes in a directory where application files are stored).

Implementation:
- Monitor a directory of any application which we want to monitor for any unauthorized changes (e.g., creation of an encrypted file).
- Log events like file modifications, deletions, or new processes related to application.

AI Integration:
- Use an anomaly detection model (e.g., using Long Short-Term Memory (LSTM) networks, Isolation Forests, or Autoencoders) to identify unusual patterns in file activities or system behaviours that deviate from normal baselines.
- Example: Train an AI model to recognize typical file access patterns in the Notepad directory. Flag abnormal activities such as rapid file encryption (ransomware-like behaviour).

Stage 2: Detection

Objective:
Detect suspicious patterns indicating an attack.

Implementation:
- Use a lightweight anomaly detection algorithm (e.g., comparing hash values of application binaries).
- Trigger alerts when unauthorized modifications to files or binaries are detected.

AI Integration:
- Implement machine learning models (e.g., Random Forests, Gradient Boosting, or SVM) for classifying activities as malicious or benign based on features extracted from:
    - File metadata (size, type, etc.).
    - System events (process creation, network connections).
    - Behavioural patterns (frequency of file modifications).
- Example: Use a pre-trained malware classification model to analyse and score file or process activities.

Stage 3: Quarantine

Objective:
Isolate and block malicious files or processes.

Implementation:
- Move suspicious files to a quarantine directory.
- Replace compromised binaries with secure ones from a pre-verified repository.

AI Integration:
- Apply reinforcement learning to decide the best action for isolating threats (e.g., blocking processes vs. quarantining files).
- Example: The system could dynamically adjust quarantine thresholds based on live threat assessments using AI.

Stage 4: Resolution

Objective:
Automatically recover from the detected attack.

Implementation:
- Restore original files and binaries from the verified repository.
- Restart the application and verify the integrity of the restored components.

AI Integration:
- Employ AI models to:
    - Predict which recovery steps are necessary based on attack severity.
    - Validate the restored application by simulating expected behaviour and ensuring its integrity.
- Example: Use a model trained on past recovery logs to recommend whether to simply replace the file or take additional steps like restarting related services or notifying admins.

2. Study on how this project stands out from existing solutions:

There are some already existing systems which are part of the cybersecurity ecosystem, designed to protect organizations against evolving threats. Each focus on specific aspects of security, ranging from detection and prevention to response and remediation.

2.1 IBM QRadar:

Category:
Security Information and Event Management (SIEM)

Purpose:
Log analysis, threat intelligence, and correlation of security data.

Key Functions:
- Centralized Log Management: Collects logs from various sources (firewalls, servers, applications) and correlates them.
- Threat Detection: Uses rule-based and AI/ML-driven analytics to detect anomalies or potential threats.
- Incident Response: Provides insights for security teams to respond to incidents faster.
- Compliance Management: Ensures compliance with regulations like GDPR, HIPAA, or PCI-DSS.

Usage Scenario:
Large enterprises with diverse IT infrastructures needing centralized security monitoring and compliance reporting.

2.2 CrowdStrike Falcon:

Category:
Endpoint Detection and Response (EDR) and Managed Detection and Response (MDR)

Purpose:
Protects endpoints (desktops, laptops, servers) from threats like malware, ransomware, and zero-day exploits.

Key Functions:
- Endpoint Protection: Real-time monitoring and detection of malicious activities on endpoints.
- Threat Hunting: Uses AI to detect sophisticated threats and provides tools for human threat hunters.
- Incident Response: Identifies and isolates compromised endpoints to prevent lateral movement.
- Cloud-Based: Operates through lightweight agents that report to a cloud platform for analysis.

Usage Scenario:
Organizations seeking robust, real-time endpoint security with AI-driven insights and fast remediation capabilities.

2.3 SentinelOne:

Category:
Endpoint Protection Platform (EPP) with AI and EDR capabilities

Purpose:
Autonomous detection, prevention, and remediation for endpoint attacks.

Key Functions:
- AI-Powered Threat Detection: Uses ML models to detect and block known and unknown threats in real time.
- Behavioural Analysis: Tracks application and system behaviours to identify anomalies and predict attacks.
- Automated Remediation: Automatically isolates and rolls back systems to a pre-attack state.
- Ransomware Mitigation: Protects against ransomware by detecting encryption patterns and terminating malicious processes.

Usage Scenario:
Businesses needing an all-in-one, autonomous solution for endpoint protection and fast recovery.

To differentiate my project from existing solutions like IBM QRadar, CrowdStrike Falcon, and SentinelOne, I am focusing on filling the gaps and introducing novel features. This project stands out by providing a comprehensive, adaptive, and unified solution that combines the strengths of existing tools while addressing their limitations. Focus on automation, predictive capabilities, and hardware-software integration for a truly innovative approach. Here's a list of potential enhancements my project can offer:

AI-Powered Autonomous Quarantine and Recovery:

Current Limitation:
- Most tools rely on manual intervention for certain remediation steps (e.g., deciding what to isolate or rollback).

Enhancement:
- Autonomous Quarantine: Automatically isolate compromised binaries or system components using predictive analytics.
- Intelligent Recovery: Implementing self-healing mechanisms that dynamically replace infected binaries from a secure, blockchain-backed repository.

Cross-Platform Integration:

Current Limitation:
- Existing tools are often tied to specific environments (e.g., IBM QRadar for centralized logs, SentinelOne for endpoints).

Enhancement:
- To Build a system that seamlessly integrates with multiple platforms and environments (on-premises, cloud, hybrid) and coordinates across all levels of infrastructure (databases, endpoints, applications, and servers).

Zero-Day Attack Prediction and Prevention:

Current Limitation:
- Many tools focus on detecting attacks after they happen or blocking known threats.

Enhancement:
- Introduce predictive models for zero-day vulnerabilities by leveraging large-scale data and transfer learning.
- Use adversarial learning to simulate potential zero-day exploits and prepare proactive defences.

End-to-End Visibility with Multi-Layered Dashboards:

Current Limitation:
- Dashboards in existing tools often focus on specific aspects (logs, endpoints).

Enhancement:
- Design multi-layered dashboards providing real-time visibility across:
- Hardware health.
- Software logs.
- Network activity.
- Threat status and resolution progress.

AI-Powered Threat Attribution:

Current Limitation:
- Tools focus on stopping threats but don't always identify their origins.

Enhancement:
- Implement AI models for threat attribution to track and identify the source of the attack (e.g., insider threats, supply chain attacks).

Blockchain for Integrity and Auditability:

Current Limitation:
- Lack of tamper-proof mechanisms to verify the integrity of quarantined data or logs.

Enhancement:
- Use blockchain for storing:
- System state snapshots.
- Integrity-verified backups of binaries.
- Immutable audit trails for post-incident investigations.

AI-Guided Recovery Prioritization:

Current Limitation:
- Existing systems treat all threats as equal in urgency.

Enhancement:
- Develop an AI-guided prioritization engine that assesses the severity and business impact of threats and recommends a response order.

Modular and Extensible Design:

Current Limitation:
- Many existing tools are rigid and don't allow seamless expansion for new attack types.

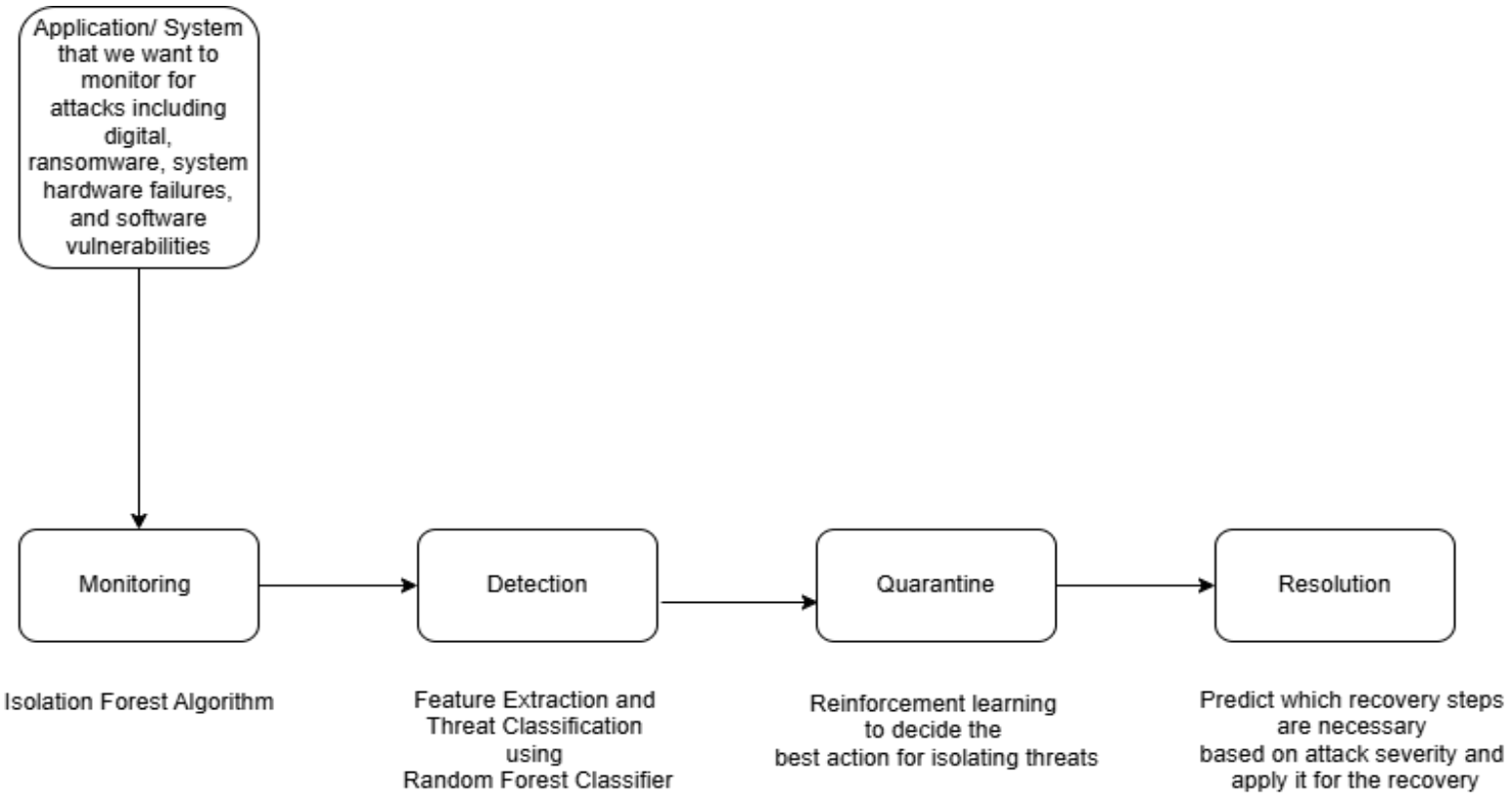
Enhancement:
- Designing a system as a modular framework where:
- New AI models or attack-specific detection modules can be plugged in.
- Organizations can customize based on their specific needs.

Self-Adaptive Threat Models:

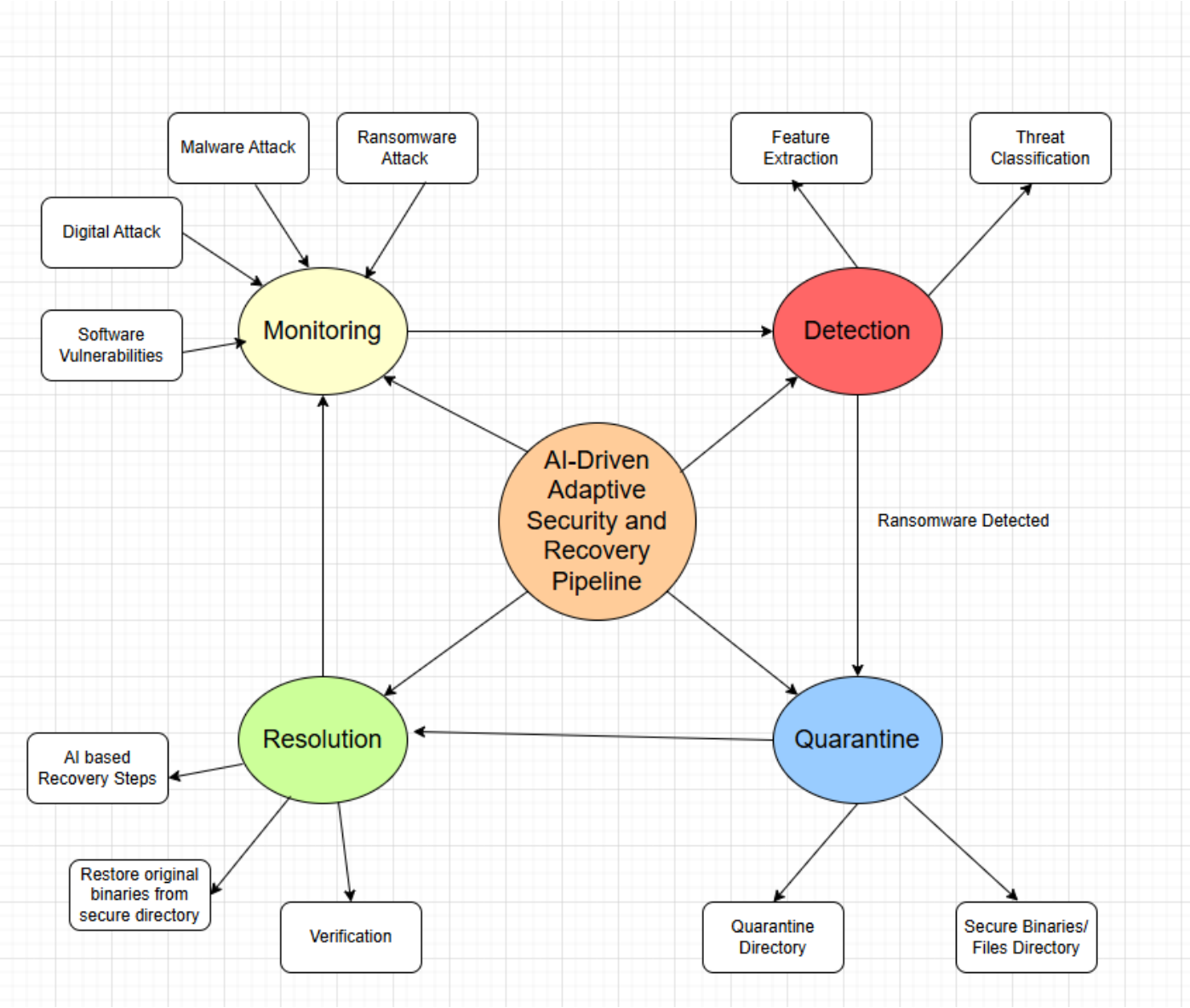
Current Limitation:
- Static models may fail to adapt to new threat patterns.

Enhancement:
- Incorporating self-learning AI models that continuously evolve by incorporating new threat data into their training.

3. Functional Block diagram/ description of Pipeline:



4. Modules in AI-driven Adaptive Security and Recovery Pipeline:



5. Algorithms used in each step of pipeline:

4.1 Isolation Forest algorithm:

The Isolation Forest algorithm is a machine learning method specifically designed to detect anomalies (unusual patterns) in a dataset. It works by isolating data points that behave differently from the majority

The Basic Idea:

- Imagine your data points are trees in a forest.
- To isolate a tree (data point), you chop through branches (features of the data) until the tree is cut off from the forest.
- Trees that are very different from the others (anomalies) can be isolated quickly with fewer cuts.
- Trees that are similar to the others (normal points) need more cuts because they're deeply connected to the rest of the forest.

How It Works:

1. Random Splitting:

- The algorithm builds a forest of "isolation trees" by splitting the data repeatedly at random.
- For each split, it picks a feature (e.g., file size, modification frequency) and a random value to divide the data.

2. Isolation Depth:

- The more splits (cuts) it takes to isolate a data point, the more "normal" it is.
- If a point is isolated in just a few splits, it's likely an anomaly.

3. Anomaly Score:

- Each point gets a score based on how quickly it was isolated across all the trees.
- A high score means the point is likely an anomaly.

Why It's Great:

- Efficient: It's faster than other anomaly detection methods because it doesn't need to calculate distances or probabilities for every data point.
- Scalable: Works well even with large datasets.
- No Assumptions: It doesn't assume the data follows a specific distribution (like normal distribution).

Real-Life Example:

- You monitor file sizes in a directory.
- Most files are between 90 KB and 110 KB.
- Suddenly, a file of 5000 KB appears.
- The Isolation Forest quickly isolates this file because it's very different from the others, flagging it as an anomaly.

Key Parameters:

- Number of Trees: More trees give better results but take longer to compute.
- Contamination: This is the proportion of data you expect to be anomalies (e.g., 5%).

In simple terms, the Isolation Forest is like a detective looking for suspicious behaviour in a crowd. It focuses on how quickly it can "single out" someone acting differently from the rest. If it's too easy to isolate, it's flagged as suspicious!

4.2 Alternatives to Isolation Forest Algorithm:

Several algorithms can be used for anomaly detection, each with its strengths and weaknesses. Here are some popular alternatives:

4.2.1. One-Class SVM (Support Vector Machine):

How it works:

- Models a boundary around normal data points and flags anything outside this boundary as an anomaly.

Pros:

- Effective in high-dimensional spaces.
- Good for small datasets.

Cons:

- Computationally expensive for large datasets.
- Sensitive to outliers in the training set.

4.2.2. DBSCAN (Density-Based Spatial Clustering of Applications with Noise):

How it works:

- Groups points into dense regions and marks points in low-density areas as anomalies.

Pros:

- Non-parametric (no need to specify a distribution).
- Handles clusters of arbitrary shapes.

Cons:

- Requires tuning parameters like eps (distance threshold) and minPts (minimum points to form a cluster).
- Struggles with high-dimensional data.

4.2.3. Autoencoders (Neural Networks):

How it works:

- Compresses data into a smaller representation and reconstructs it. Anomalies reconstruct poorly because they don't fit the learned patterns.

Pros:

- Works well with complex, high-dimensional data.
- Can adapt to various use cases with enough training data.

Cons:

- Requires a lot of data and computational resources.
- Longer training time.

4.2.4. LOF (Local Outlier Factor):

How it works:

- Compares the density of a point to its neighbours. Points in low-density regions are flagged as anomalies.

Pros:
- Captures local patterns.
- No assumptions about data distribution.

Cons:
- Computationally expensive for large datasets.
- Sensitive to the choice of parameters like k (number of neighbours).

4.2.5. PCA (Principal Component Analysis) for Anomaly Detection:

How it works:
- Projects data into a lower-dimensional space. Points with high reconstruction errors are flagged as anomalies.

Pros:
- Simple and interpretable.
- Works well for linear data patterns.

Cons:
- Struggles with nonlinear patterns.
- Sensitive to scaling and noise.

4.3 Why Choose Isolation Forest for Monitoring Phase?

- Efficiency:
Isolation Forest is computationally efficient because it isolates anomalies using random splits rather than calculating distances or densities.
Ideal for real-time monitoring in systems with frequent updates.

- Scalability:
It works well with large datasets, which is critical for monitoring logs, file systems, or networks.

- Simplicity:
Requires fewer parameters to tune compared to methods like DBSCAN or Autoencoders.

- No Distribution Assumption:
Unlike PCA or One-Class SVM, Isolation Forest does not assume that the data follows a specific distribution, making it versatile for different datasets.

- Robust to High Dimensions:
Handles datasets with many features (e.g., file size, modification time, frequency) without suffering performance degradation.

- Interpretability:
Anomaly scores are intuitive and easy to understand. For example, a high anomaly score directly means the instance is more likely to be anomalous.

Key Trade-offs:

- Why Not Use Neural Networks (Autoencoders)?
Autoencoders are powerful but require large amounts of training data and computational power, which might not be feasible for lightweight systems or quick prototyping.

- Why Not Use Density-Based Methods (DBSCAN, LOF)?
These methods struggle with high-dimensional data and require precise tuning of parameters, which can complicate implementation in dynamic monitoring systems.

Conclusion
Isolation Forest is a fast, scalable, and robust algorithm that fits the requirements of the Monitoring Phase in the pipeline. It ensures the system can efficiently and effectively flag anomalies in real-time without excessive computational overhead.

4.4 Detection stage:

How It Works:
Training:
The model learns patterns from the training dataset, such as how large file sizes or recent modifications correlate with specific threats.

Prediction:
For unseen data, the model predicts the likelihood of each threat type based on learned patterns.

Evaluation:
The classification report shows how well the model performs on unseen data.

Why Use AI Here?
- AI-powered models can handle complex patterns in data that traditional rules-based systems might miss.
- Over time, the model improves as it learns from new data, adapting to emerging threats like new types of malware or phishing attacks.

Input to the Detection Stage:
- size: The file's size in bytes.
- modification_time: The timestamp when the file was last modified.
- creation_time: The timestamp when the file was created.
- anomaly: Indicates the anomaly score (-1 means anomalous as flagged by the Isolation Forest algorithm).

How the Detection Stage Works:
Feature Extraction:
- The Detection Stage uses the size, modification_time, and creation_time features as input.
- The anomaly score is not used in this stage because the Detection Stage focuses on identifying the type of threat.

Threat Classification:
- The Random Forest Classifier (or another ML model) analyses the input data.
- Based on the patterns learned during training, the model assigns the file to a specific threat category (e.g., benign, ransomware, phishing).
For instance:
    - Large file size + recent modification = ransomware.
    - Sudden appearance of a new file + no recent modification = phishing.

Output of the Detection Stage:
- Predicted Threat:
The Detection Stage classifies the file as Ransomware, meaning it suspects that the anomaly is caused by a ransomware attack.

- Confidence:
This is the model's certainty in its prediction.
For example:
85% means the model is confident that the anomaly represents ransomware.

How to Interpret This Process:
- In the Monitoring Stage, the anomaly was flagged based on statistical irregularities (e.g., unusual size or timestamps).
- In the Detection Stage, the anomaly is classified into a specific threat category using a trained model.
- If the model's confidence is low (e.g., <50%), the output should be treated with caution, and further analysis (e.g., manual review) might be required.
- The stage adds context to the anomaly, helping you decide the next action:
    - If ransomware is detected, the pipeline moves to the Quarantine Stage.
    - If the anomaly is benign, no action is taken.

6. Design considerations/ Pre-requisites:

- Set Up an ELK Stack:
You can host an ELK stack on your local system, a cloud service (like AWS or Elastic Cloud), or a VM (e.g., Docker-based setup).

- Elasticsearch API Endpoint:
Ensure your Elasticsearch instance is accessible from Kaggle. (Use the public IP/endpoint if using a cloud setup.)

- Install Python Libraries:
Use Elasticsearch for communicating with Elasticsearch and Logstash if needed.

7. Future plan:

No.	Phases	Start Date – End Date	Work to be done	Status
1	Dissertation outline	18th January 2025 - 25th January 2025	Literature Review and prepare Dissertation Outline	COMPLETED
2	Design and Development	25th January 2025 – 10th March 2025	Design & Development Activity of different phases. Monitoring and Detection phases completed, Quarantine and Resolution stages are in progress	IN PROGRESS
3	Testing of work done and modifications	10th March 2025 – 20th March 2025	Software Testing, User Evaluation & Conclusion. Testing of Monitoring and Detection phases completed. Quarantine and Resolution stages are in progress.	IN PROGRESS
4	Mid-Semester progress report	20th March 2025 - 27th March 2025	Submit Dissertation to Supervisor & Additional Examiner for review and feedback	COMPLETED
5	Incorporating feedback after review	31st March 2025 - 17th April 2025	Incorporating and modifying report or work after examiner's review.	PENDING
6	Final Dissertation Report	17th April 2025 - 24th April 2025	Final Review and submission of Dissertation	PENDING