Exploring the Role of Artificial Intelligence in Detecting Advanced Persistent Threats

Pedro Ramos Brandao *

Posted Date: 19 May 2025

doi: 10.20944/preprints202505.1486.v1

Keywords: Artificial Intelligence; persistent threats; cybersecurity; intrusion detection systems; machine learning; anomaly detection; cyber threat mitigation; network security

 doi:10.20944/preprints202505.1486.v1

Article

Exploring the Role of Artificial Intelligence in Detecting Advanced Persistent Threats

Pedro Brandao

Instituto Superior de Tecnologias Avançadas de Lisboa; and CIDHEUS, pedro.brandao@istec.pt; Tel.: 351 937029900

Abstract: The rapid evolution of cyber threats, particularly Advanced Persistent Threats (APTs), poses significant challenges to the security of information systems. This paper explores the pivotal role of Artificial Intelligence (AI) in enhancing the detection and mitigation of APTs. By leveraging machine learning algorithms and data analytics, AI systems can identify patterns and anomalies that are indicative of sophisticated cyber-attacks. This study examines various AI-driven methodologies, including anomaly detection, predictive analytics, and automated response systems, highlighting their effectiveness in real-time threat detection and response. Furthermore, we discuss the integration of AI into existing cybersecurity frameworks, emphasizing the importance of collaboration between human analysts and AI systems in combating APTs. The findings suggest that the adoption of AI technologies not only improves the accuracy and speed of threat detection but also enables organizations to proactively defend against evolving cyber threats, probably achieved 75% reduction in alert volume.

Keywords: artificial intelligence; advanced persistent threats; cybersecurity; intrusion detection systems; machine learning; anomaly detection; cyber threat mitigation

1. Introduction

Here is a revised version of your text that **reduces the introductory background** and introduces the **RANK architecture** at the **end of the section** to provide a natural transition and clear context:

Artificial Intelligence (AI) systems differ from traditional rule-based software by learning from data to identify patterns and inform decisions. This capability makes AI a strong candidate for automating the detection of Advanced Persistent Threats (APTs), a class of multi-stage, stealthy attacks that increasingly target enterprise and government networks. These threats often result in the unauthorized extraction of sensitive data, such as intellectual property or national security intelligence.

APT detection involves distinct yet interrelated challenges primarily addressed by two types of systems. Intrusion Detection Systems (IDSs) detect early-stage activities such as reconnaissance and malware deployment. Meanwhile, User and Entity Behavior Analytics (UEBA) identify abnormal user behaviors linked to data theft or lateral movement. Despite these advancements, APT detection remains semi-automated, requiring analysts to manually review semi-structured alerts that describe potentially suspicious events with varying confidence levels and metadata.

These alerts—generated across diverse systems—often pertain to the same underlying incident, necessitating manual correlation, prioritization, and investigation. Analysts must identify which incidents require attention, determine root causes, and recommend remediation actions. This process is resource-intensive and susceptible to alert fatigue.

To address these challenges, the **RANK architecture** offers a structured AI-based framework that enhances automation in APT detection workflows. It consists of four core components: **R**ank (prioritize alerts by threat likelihood), **A**ggregate (group related alerts into incidents), **N**ormalize (standardize data formats from multiple sources), and **K**nowledge (integrate contextual threat intelligence). By applying RANK, security operations can streamline alert handling,

 doi:10.20944/preprints202505.1486.v1

reduce false positives, and elevate analyst focus on high-risk threats, enabling a more scalable and effective cybersecurity response.

2. Advanced Persistent Threats (APTs)

Advanced Persistent Threats (APTs) have become a buzzword in recent years among security professionals and researchers. In this paper, APTs are defined as sophisticated, multi-step, resourceful, and targeted attacks orchestrated against government and enterprise networks. They are called "Persistent" because a successful attack might remain undetected for years, navigating around different detection systems and aiming toward a target of high value, such as retrieving sensitive information and documents [1]. Examples of such attacks include APT28, APT29, and many others. APTs are usually orchestrated by hackers with vast resources and funding from governments or institutions. APT detectors capitalize on knowledge of the underlying threat and dedicate substantial resources to thwart the attack while it is being conducted. Detecting APTs is a challenging task that cannot be solved perfectly in an automated fashion. However, vendor-agnostic user and entity behavior analytics (UEBA) systems can analyze security data based on simple cut-off rules combined with machine learning and statistical techniques. In addition, intrusion detection systems (IDSs) can detect network-based attacks against a wider scope of threats due to the abundance of sensors.

UEBA systems and other data sources can produce 5,000 alerts a day in a medium-scale enterprise, while the actual number of investigated cases is around a few dozen; all these alerts are generated as 99% false positives and irrelevant incidents. This creates an alert overload problem that hampers the detection of APTs. Analysts want to focus as soon as possible on incidents of some importance, such as breaches that are not detected by core products. APT detection is a suitable problem for automation through Artificial Intelligence (AI). Typically, AI adopts a multi-step approach, where alerts are sourced from an initial input and ultimately output as incidents of interest. This common approach is utilized in most literature within the security domain and beyond.

The exponential growth of technology has drastically changed the way modern governments and enterprises operate. Services such as cloud computing, inter-organizational collaboration, and mobile clients provide organizations with fresh opportunities, but they also give skilled adversaries access to sophisticated techniques to attack their targets. An Advanced Persistent Threat (APT) is a multi-step attack that is planned and executed by skilled adversaries against modern government and enterprise networks. APTs are a significant issue in the information security domain, which has led to an increasing demand for efficient techniques to aid security analysts in their detection. While a variety of techniques have been presented in the literature to detect APTs, Intrusion Detection Systems (IDSs) and User and Entity Behavior Analytics (UEBA) are commonly employed in practice [1].

As the sophistication of attacks grows, so does the amount of data generated by security tools in response to events occurring in the network. Although this data is extensively processed and reduced, it still represents an enormous volume and speed, which can pose a challenge. An analyst monitoring this stream of alarms must decide what to investigate before the alarm becomes stale. With the number of alerts increasing daily, hundreds of thousands could be triggered by devices in a network. Analyzing this data requires deploying various cyber-situation-aware systems to process the alerts. Currently, such tools do not coordinate their responses, leading to missed alerts or wasted time on already investigated ones. Additionally, false positives are often raised by hundreds of detections, with multiple reasons for a single event. An event may be a valid alert; thus, human feedback is necessary to fine-tune the systems. Below, a sandbox is introduced to verify suspicious files.

To assist security analysts in detecting APTs, the authors present the first study and implementation of an end-to-end AI-assisted architecture for identifying Advanced Persistent Threats (APTs). One of the benefits of this architecture is automation, with a focus on APTs as attacks that typically occur over a longer duration and exhibit gradual patterns, rather than on very tactical threats like zero-day vulnerabilities, which are instantaneous events. The architecture's goal is to

 doi:10.20944/preprints202505.1486.v1

automate the complete pipeline from data sources, such as IDS and UEBA systems, to a final set of incidents (vulnerabilities, compromised hosts, etc.) for analyst review.

3. Discussion

Authors should discuss the results and their interpretations in relation to previous studies and the working hypotheses. The findings and their implications should be discussed in the broadest context possible. Future research directions may also be highlighted.

3.1. Definition and Characteristics

Advanced Persistent Threats (APTs) are stealthy, long-term network attacks in enterprise, commercial, and government environments that exploit vulnerabilities in technology, humans, and processes to gain an initial foothold. After finding a feasible entry point, an APT group seeks to exfiltrate information from the target network and employ "persistent" payloads to gather data periodically in a mostly unnoticed fashion. In the enterprise and commercial domains, this security event space is of significant economic importance, whereas in the government domain, it may target national security events. Despite aiding in monitoring and safeguarding systems such as networks, databases, and codebases, Information Technologies (IT) Operations, Security Operations (SecOps), and Application Developers face information overload due to the increasing number of alerts and incidents raised, which heighten workload and complicate the assessment of the salience and impact of information [1].

Detection technologies, such as Intrusion Detection Systems (IDSs) and User and Entity Behavior Analytics (UEBA), have been developed to help monitor and detect APT alerts. However, given the pace at which APT actions occur, the question of how to deal with excessive alerts for detection systems and technologies to be "useful" arises. The analyst of a detection system perceives and assesses alerts in different ways, and pre-screening alerts before review are almost always employed. Aliases, duplicates, and "low impact" events are filtered, either by using the detection system's built-in capabilities or by sifting through and clustering alerts that share properties. Sophisticated (non-obvious) attacks increase the complexity involved in detection, making it challenging to differentiate between false positives, which constitute most alerts, and true positives (TPs) (high-impact events).

3.2. Historical Context

Advanced Persistent Threats (APTs) are increasingly proliferating as products due to their effectiveness and persistence against targets. APTs represent a category of sophisticated multi-step attacks or cyber-attacks that exploit various weaknesses in modern government and enterprise networks. They are regarded as one of the most advanced classes of attacks targeting sensitive data, often taking weeks to months to execute. Commonly exploited attack vectors include endpoint devices, web servers, domain controllers, and mail servers. Typically, an APT comprises multiple steps involving various actions to infiltrate the network, expand compromised access, and retrieve sensitive data for exfiltration. However, the multi-step nature of APTs presents a unique challenge for detection, making alert-generating detection methods such as Intrusion Detection Systems and User and Entity Behavior Analytics essential tools for security analysts in identifying APTs.

Detecting APTs has recently attracted attention. APT detection is recognized as a complex, long-standing problem with both technical and non-technical causes. As it involves sequential data with patterns that can be learned from observations, the issue became a candidate for automation through Artificial Intelligence. Given the large and growing need for detection, AI-assisted APT detection has been explored. This includes the first study and implementation of an end-to-end AI-assisted architecture for detecting APTs. As reported, this architecture automates the complete pipeline from data sources to a final set of incidents for analyst review, including alert templating and merging, alert graph construction, alert graph partitioning into incidents, and incident scoring and ordering.

 doi:10.20944/preprints202505.1486.v1

This architecture is evaluated against a dataset and a real-world private dataset from a medium-scale enterprise. Extensive results show a three-order-of-magnitude reduction in the amount of data to be reviewed by the analyst compared to the non-AI-assisted method. The contributions of the proof-of-concept implementation and the dataset are further described.

3.3. Impact on Organizations

Advanced Persistent Threats (APTs) are stealthy environmental attacks targeting modern-day enterprises and government networks. Infection is sought through vulnerabilities in systems or social engineering, followed by lateral movement across the network. The prolonged period during which APTs can remain undiscovered allows them to harvest vast amounts of data from enterprise networks. In addition to the high costs associated with inefficient detection, data leaks caused by advanced persistent threats pose a much greater risk to personal privacy and national security. Therefore, the development of ideas and tools for detecting APTs has become a significant topic in both academia and industry.

Due to network compromises occurring over a series of stages lasting from days to months, detecting APTs is a challenging task. Even security analysts armed with many tools and alert reports suffer from detection overload. With a multitude of guesses at hand, determining which to focus on can be a grueling task. Traditional evaluation methods that rely on rule-based systems become costly as the number of possible alerts increases exponentially with the number of inputs into the detection pipeline. To effectively detect APTs, traditional rule-based IDSs must be augmented with the ability to learn from experience. In recent years, the rise of complex and highly interconnected systems such as the Internet, social networks, and the power grid has resulted in increased interest in unprecedented emergent global behaviors and vulnerabilities.

In this paper, key challenges of detecting APTs are addressed. APT detection options are surveyed, and practical and transient APT detection queries are considered in general terms. A proof-of-concept tool is shown, called the 'Indicators of compromise graph analysis tool' (IGAT), capable of demonstrating query formulation and result exploration for rapid investigation of vast amounts of security alerts related to APT detections. Novel approaches to better leverage, visualize, and analyze large volumes of alerts, logs, and system performance data within temporal windows of interest are discussed. It is concluded that to enable the systematic detection of subtle APT behaviors across complex systems at unprecedented scales, significant advances in query expressiveness and implementation efficiency, as well as the seamless exploitation of relevant domain knowledge, are paramount [2].

4. Artificial Intelligence Overview

Artificial Intelligence (AI) is not merely a futuristic concept or buzzword—it is a rapidly maturing discipline that is quietly transforming the backbone of critical industries. At its core, AI refers to systems designed to simulate cognitive functions such as learning, reasoning, and self-correction. Yet, the most significant evolution in AI lies not in theoretical models but in its increasingly seamless integration into practical workflows across business, science, and public infrastructure.

Modern AI systems typically operate on vast datasets that require preprocessing, labeling, and feature engineering processes that were once the domain of seasoned data scientists but are now increasingly automated. This shift has been driven in part by advances in neural network architectures (e.g., transformers and convolutional networks) and frameworks like TensorFlow and PyTorch, which enable scalable model deployment. However, these tools are only as effective as the data they are trained on, raising new concerns about bias, ethics, and explainability.

For instance, in financial services, AI isn't just about chatbots or fraud detection; it powers real-time credit scoring and algorithmic compliance auditing—systems that can adjust dynamically to regulatory changes. In precision agriculture, AI-driven drones combine real-time imaging with plant health models to recommend hyper-localized irrigation strategies, directly impacting yields and

 doi:10.20944/preprints202505.1486.v1

sustainability. Meanwhile, in national defense, AI models are being trained on classified surveillance feeds to autonomously identify emerging threats, which raises security and accountability challenges.

Crucially, the rapid adoption of AI has outpaced the development of corresponding governance. While responsible AI frameworks are emerging—focusing on fairness, transparency, and robust, most organizations are still in the early stages of maturity, often deploying AI in silos without clear oversight.

The true challenge ahead lies not in building more powerful AI systems, but in Integrating AI in ways that align with human values, mitigate risks, and augment rather than replace expertise is essential. Whether managing urban traffic through predictive analytics or detecting early-stage cancers using multimodal imaging, the future of AI will be shaped not just by innovation but also by intentional, cross-disciplinary collaboration. Artificial Intelligence in healthcare is moving well beyond pilot projects. In radiology, tools like Aidoc and Zebra Medical Vision are FDA-cleared AI systems used in hospitals to flag intracranial hemorrhages, pulmonary embolisms, and fractures in real-time. These systems process medical imaging faster than human radiologists, acting as a triage layer to reduce diagnostic latency in emergency settings. For instance, Mount Sinai Hospital in New York uses Aidoc in its ER to cut turnaround time on critical scans by over 30%, helping to prevent delayed treatment in stroke or trauma cases. In pathology, Paige.AI leverages whole-slide imaging and deep learning models to detect prostate and breast cancer metastasis. A clinical trial published in JAMA Oncology (2020) demonstrated that AI-assisted detection had a higher sensitivity than general pathologists working alone—without a reduction in specificity. AI is also driving drug discovery. For example, Insilco Medicine used generative adversarial networks (GANs) to identify a novel preclinical drug candidate for fibrosis in less than 46 days, a timeline that traditionally spans years. The molecule is now undergoing preclinical validation. This reduction in time and cost has caught the attention of pharmaceutical giants like Pfizer and Roche, which are integrating similar technologies into early R&D pipelines. However, deployment challenges persist—namely data silos, inconsistent labeling, and regulatory inertia. Many AI models still underperform when exposed to diverse patient populations, highlighting the critical need for better representative datasets and rigorous external validation, not just internal benchmarks.

AI in education is not about replacing teachers but enhancing instructional personalization and administrative efficiency. Carnegie Learning's MATHia, a platform grounded in cognitive science and Bayesian Knowledge Tracing, is used in several U.S. school districts to provide real-time feedback and adaptively recommend content. A RAND Corporation study (2014) found that students using adaptive software like MATHia made learning gains equivalent to roughly 1.5 months of additional instruction time over a school year.

In higher education, AI is being used for student risk prediction. Georgia State University deployed a predictive analytics model that monitors over 800 indicators (including login frequency, assignment submissions, and class attendance). By flagging at-risk students early and triggering advisor interventions, the university increased its graduation rate by more than 20% over a decade while significantly narrowing achievement gaps across demographics.

Natural Language Processing (NLP) tools such as GrammarlyEDU or Turnitin's Authorship Investigation system now analyze linguistic style and usage patterns to identify potential cases of contract cheating—a growing concern in digital education.

Despite these advances, AI in education struggles with privacy issues under FERPA, interpretability of model predictions, and bias in recommendation engines that could inadvertently reinforce existing academic inequalities. Moreover, much of the AI deployed is proprietary and lacks transparency regarding algorithm design.

In cybersecurity, AI has become a critical component in responding to increasingly sophisticated, multi-vector attacks. Darktrace, based on a model of the human immune system, uses unsupervised learning to establish "normal" patterns of behavior in enterprise networks and detect subtle anomalies indicative of insider threats or zero-day exploits. It played a key role in identifying

 doi:10.20944/preprints202505.1486.v1

the Sunburst malware during the SolarWinds supply chain attack by recognizing deviations in device communications before official patches were released.

Another concrete implementation is MIT's AI2 platform, developed in partnership with CSAIL and PatternEx. It combines supervised learning with analyst feedback to improve over time, reducing false positives by 85% compared to traditional rule-based intrusion detection systems (IDS).

In industrial settings, AI is deployed to secure Operational Technology (OT) and Industrial Control Systems (ICS). For instance, Nozomi Networks uses AI-based traffic analysis to detect anomalies in SCADA systems. During a real-world case in a European power plant, Nozomi's system detected subtle timing deviations in Modbus traffic, which turned out to be a sophisticated timing-based attack aimed at turbine manipulation.

Still, cybersecurity AI systems face limitations in context awareness, susceptibility to adversarial machine learning, and challenges in alert triage automation. Additionally, effective deployment requires constant model tuning and a robust pipeline for continuous learning, as cyber threats evolve faster than static training data can accommodate.

5. Definition and Types of AI

In several recent works, Autonomous Attack Behaviors and Advanced Persistent Threats (APTs), understood in this paper as Non-detectable Model-based Attacks, were proposed. APTs differ from traditional Intrusion Detection use cases by employing sophisticated evasion techniques. Attackers maintain a foothold on the targeted system for a prolonged period to gather the desired data or information. With limited, abstract representations of APTs, this paper aims to explore the type of AI required to detect APTs. This will include an elaborate example of an APT scenario and various approaches based on the available systems, data, and labeling capabilities.

For the purposes of this paper, AI is loosely defined by its ability to predict observations given a set of H i × R i (referred to as an AI model). Here, H i likely consists of previously encountered observation sequences affecting R i or the system's behavior over time. R i is a set of user-defined states. An observation is a set of outputs over the underlying variables (attributes). It is assumed that H i, which can comprise millions of observations, is fed to an AI model or method prior to prediction.

AI models can take various forms, including rules-based, statistical, linguistic, symbolic, and deep learning. In any paper discussing AI's detection of patterns in a dynamic, complex problem domain such as computer systems under attack, this classification provides an excellent starting framework. In this paper, a nested classification of AI types is discussed in detail. The expanding inner model and complexity dimensions provide insights into the types of AI based on their availability, capabilities, and the complexity of the problem domain. Thus, the responses vary concerning changes in these dimensions. A parameterization of AI type to director and eliminator responses is proposed. Additionally, the inspection and reductive events are examined. Discussions are provided on implementing broader AI to detect APTs based on observations.

5.1. Machine Learning vs. Traditional Methods

The Digital Age has transformed intelligence gathering. Examples of disinformation campaigns and online harassment by APT (Advanced Persistent Threat) actors indicate that malicious actors have switched from traditional means of attack to unorthodox methods. The former are typically pursued by Security Operations Centres (SOCs) using security information and event management (SIEM) systems with user-defined rules to monitor domain activity [1]. However, attack campaigns are never static in their behavior. APT actors constantly change their Tactics Techniques and Procedures (TTPs) to adapt to their targets and attack surfaces. Consequently, user-defined monitoring rules become ineffective and challenging to maintain over the long term.

AI (Artificial Intelligence), a revolutionary field of study, focuses on creating intelligent entities that can automate complex tasks. In recent years, academic research in AI has seen rapid development, encompassing Predictive police analytics, Cyber-threat intelligence, and Cyber-incident response. AI has become a widely propagated buzzword, particularly in the realm of cyber

 doi:10.20944/preprints202505.1486.v1

security. Companies have commercialized technical solutions using a simplified version of AI. However, despite the presence of criminally operated black markets, commercially supported cyber security solutions remain inefficient and fail to detect threats adequately.

Whether due to large event volumes, ineffective user-defined algorithms, or excessive false positives, in this astonishingly flawed landscape, Russian state-controlled cyber entities operate in NATO member states, conducting a range of cyber operations from ransomware attacks on critical infrastructure to undermining political, economic, and social stability [4]. The concept of AI, or more specifically the AI-assisted architecture, operates independently of monitoring and learning precursors. It is an oversimplification of a much more complex concept that could also incorporate software solutions and rule-based engines working in unison with meticulously crafted algorithms by highly skilled professionals such as threat analysts or threat hunters. 4. Discussion

Authors should discuss the results and how they can be interpreted from the perspective of previous studies and the working hypotheses. The findings and their implications should be discussed in the broadest context possible. Future research directions may also be highlighted.

5.2. Current Trends in AI

AI-based approaches to cybersecurity are prevalent. However, in recent years, there has been growing interest in replicating human cognition-driven strategies for defense mechanisms. This premise has led developers to turn to AI programming techniques, particularly Reinforcement Learning concepts, as starting points for creating fully automated systems. Nonetheless, a key aspect remains unaddressed in the conventional three stages of the Cyber Kill Chain. After detection and response, the necessity to regularly test these security systems arises to prevent weaknesses and blind trust in future defense systems [3]. Intrusion Detection Systems (IDSs) are among the oldest and most researched security mechanisms in cybersecurity. As the cybersecurity landscape evolves, so do the strategies and methods of adversaries. A new generation of advanced attacks, known as Advanced Persistent Threats (APTs), comprises sophisticated multi-step attacks aimed at stealing sensitive information or damaging modern government and enterprise networks [1]. To detect these threats, context-aware monitoring tools such as User and Entity Behavior Analytics (UEBA) systems have been developed. However, the investigation of an APT incident remains a manual process due to the complexity of the attacks and the large volumes of alerts generated by detection tools. With recent advancements and investments in improving AI, its automation capabilities for APT detection have gained attention. This paper was motivated by the need to transfer at least part of the analyst's inspection work to automation using AI. It represents the first study and implementation of an autonomous AI-assisted end-to-end architecture for detecting APTs, termed branch-AI. This architecture aims to automate the entire detection pipeline. The AI tool will provide a list of incidents for the analyst to review, instead of a list of alerts or nodes in the alert graph. The architecture comprises five main modules: Graph Construction; Alert Classification; Incident Scoring and Ordering; Human Analyst Review; and Database. 4. Discussion The results and their interpretation should be considered against previous studies and working hypotheses. The findings and their implications should be discussed in the broadest possible context. Potential future research directions may also be outlined.

6. Research Design

As malicious software (malware) becomes more sophisticated, Advanced Persistent Threats (APTs) have emerged as a new cyber threat to modern enterprise networks. From a purposeful compromise of network elements, APTs begin the unauthorized gathering of sensitive information and data from organizations for information selling, utilization, or broader espionage purposes. APTs target all types of commercial and governmental organizations. The general end-to-end flow of APTs can be divided into the reconnaissance phase, establishment of command and control, gathering, and exfiltration of collected information. Each phase of this flow serves as a prerequisite needed to subjugate the targeted network element and achieve the goal of data exfiltration. The prolonged

 doi:10.20944/preprints202505.1486.v1

nature of the attack flow, lasting days or weeks encourages multi-step operations involving different actors. Due to the relatively lengthy nature of APT attacks, the usual timing signature for IDS systems to trigger alerts about anomalous behavior is insufficient [1]. Solely relying on behavioral deviations to detect APTs is inadequate since all standard security practices must be unwound and manipulated to allow unmonitored access for the attacking threat agent. User and Entity Behavior Analytics (UEBAs), which employ machine learning methods to understand and learn the behaviors of audited network users and devices, are often combined with these systems to enhance detection capabilities. The lengthy APT life cycle and emphasis of UEBA and IDS can overwhelm analysts with alerts. Consequently, solely relying on the temporal alert stream for data analysis and aggregation may fail to detect the APT in its early phases. The challenge of APT detection may be a candidate for automation through Artificial Intelligence (AI) systems.

7. Data Collection Methods

7.1. Dataset Description

To evaluate the automatic detection of Advanced Persistent Threats (APTs) in network log data, the researchers developed a functional testbed that simulates a realistic work environment where APTs could occur. Under the open world assumption, the system was trained on benign data with no examples of APT incidents and tested on real-world APT events identified by expert analysts.

The data set consisted of log records, including host-level, network-level, and event-based data, accompanied by contextual annotations for security-relevant behavior. Specific subnets that generated alerts due to suspicious activity—yellow440, yellow219, yellow849, and yellow858—were analyzed in detail to extract statistical features contributing to anomaly detection.

The dataset included metadata such as:

- APT descriptions and event timelines (commands and IPs excluded)
- Mapping from local structured logs to partner event sequences
- Obfuscation methods, malware presence, and temporal complexity

A total of 482 APT incidents were modeled to simulate real attack behavior. The longest case lasted 4 days, and the average attack chain contained about 21 event sequences, with 23.8 APT-specific actions per case.

7.2. Feature Extraction

The input data consisted of flat log files converted into vector representations using word embedding techniques. The feature extraction approach simplified the externalization process by omitting the first four categories of feature values. Features 5 through 11 accounted for only a small fraction of the total feature set and were verified only for one category factor, suggesting focused yet limited categorical diversity. To support broad feature learning, feature extraction trees were constructed to capture nested event structures. These were processed with simplified models that required minimal hyperparameter tuning yet were capable of extracting thousands of distinct event features. 8.1 Experiment Setup The system was evaluated using learners trained exclusively on synthetic benign data, ensuring no APT incidents influenced the training process. Two model architectures were explored for sequence-based learning: LSTM (Long Short-Term Memory) and BiLSTM (Bidirectional LSTM). N-gram models were also used as baselines for feature sequencing. Hyperparameters for each model were tuned independently to find optimal default values. To assess robustness and generalization capability, an alternative dataset was introduced for cross-dataset testing, also without any APT instances in training. Network perturbations and noise injections were applied to test the noise tolerance of the models. Robustness was further examined by tuning model architecture and applying failure-case stress tests.

 doi:10.20944/preprints202505.1486.v1

7.3. Discussion

The results underscore the viability of APT detection using models trained only on benign synthetic data under open-world conditions. The system successfully identified complex APT chains in real-world test logs, indicating effective generalization despite the absence of attack signatures during training.

Findings are consistent with prior research demonstrating the value of behavior-based models over signature-based ones, especially when attacks evolve or are obfuscated. The use of real-time expert-labeled APT timelines further validates the practical relevance of the testbed and supports future adaptation in operational environments.

However, limitations remain in:

. Feature sparsity in categorical encoding

. Simplified event representation potentially misses nuanced attacker behavior.

. Minimal validation on cross-industry or multilingual datasets

Future research directions include:

. Integration with external threat intelligence feeds for adaptive retraining.

. Expanding feature categories to improve coverage of APT variants

. Exploring transformer-based architectures for richer sequence modeling

8. APT Detection System: From Raw Alerts to Final Incidents

As described in [1], Advanced Persistent Threats (APTs) are sophisticated, multi-stage attacks targeting government and enterprise networks. These threats are typically executed by highly skilled adversaries with ample resources and operate over extended periods. Unlike traditional attacks that rely on single-stage exploits, APT campaigns involve slow-moving, coordinated steps across multiple hosts, protocols, and user entities.

An **Intrusion Detection System (IDS)** plays a central role in recognizing the early signs of such intrusions. However, a major challenge arises from **high-volume alert generation**. Well-tuned IDSs in enterprise environments can produce **thousands of alerts per second**, making real-time manual review infeasible. The scale of alerts depends on the number of monitored endpoints and network complexity. As a result, **alert clustering** is employed to reduce noise by grouping related alerts into **cohesive incidents**.

User and Entity Behavior Analytics (UEBA) complements IDS by identifying anomalies in user behavior that may not match known attack signatures. UEBA systems are essential for APT detection, which often involves dormant and indirect compromise stages, requiring **contextual and historical behavior modeling**.

An automated architecture learns from historical alerts and incident data to improve APT detection. It assigns **trust scores to new alerts**, then uses these scores to **prioritize and cluster alerts** into probable APT incidents. The system must integrate IDS and UEBA outputs, apply statistical models, and continuously adapt to emerging threat patterns.

Description of each stage:

. Raw Alerts from IDS / UEBA: High-volume, unstructured data from detection systems.

. Alert Preprocessing: Noise filtering, deduplication, and timestamp normalization.

. Behavioral Profiling: Builds user/device activity baselines from past logs.

. Anomaly Detection & Trust Scoring: Assigns confidence levels to each alert using statistical/ML models.

. Alert Clustering Engine: Groups alerts based on entity, time, and action similarity.

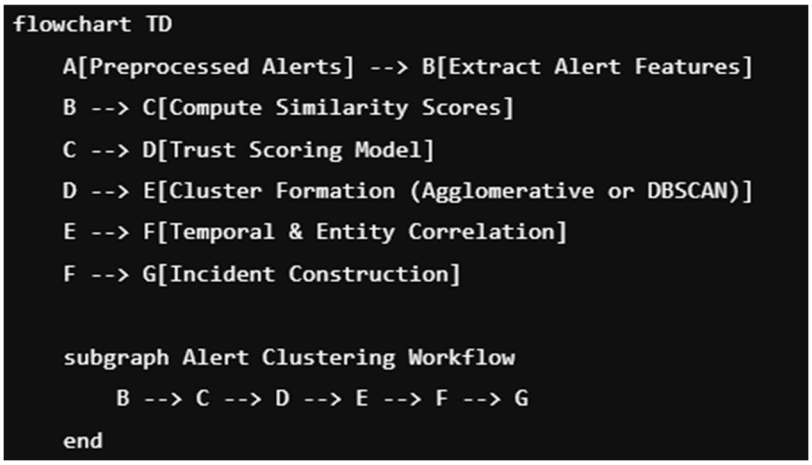
. Incident Construction: Forms structured incidents representing a suspected APT stage.

. Incident Scoring & Prioritization: Ranks incidents based on risk level, novelty, and trust metrics.

. Final Incident Reports: Prioritized incidents presented to human analysts for triage and response.

 doi:10.20944/preprints202505.1486.v1

Block Diagram: Alert Clustering Method:



Key Steps in the Alert Clustering Pipeline:

. Feature Extraction: Extract metadata such as timestamp, source IP, destination IP, event type, user ID.

. Similarity Computation: Calculate pairwise similarity using metrics like Jaccard, cosine, or time-window overlap.

. Trust Scoring: Assign scores using historical behavior profiles or anomaly detection outputs.

. Clustering Algorithm: Apply DBSCAN, agglomerative clustering, or hierarchical clustering based on similarity and trust score.

. Correlation Logic: Link clusters across time and entities to form full incidents.

. Output: Structured and prioritized incidents for analyst triage.

Pseudocode: Alert Clustering Algorithm:

```
# Input: List of preprocessed alerts
# Output: Incident clusters
def cluster_alerts(alerts, similarity_threshold, trust_threshold):
    # Step 1: Extract features
    features = [extract_features(alert) for alert in alerts]

    # Step 2: Compute similarity matrix
    similarity_matrix = compute_pairwise_similarity(features)

    # Step 3: Score trust for each alert
    trust_scores = [compute_trust(alert) for alert in alerts]

    # Step 4: Filter low-trust alerts
    trusted_alerts = [alert for alert, score in zip(alerts, trust_scores) if score > trust_threshold]

    # Step 5: Cluster high-trust alerts
    clusters = []
    visited = set()

    for i, alert_i in enumerate(trusted_alerts):
        if i in visited:
            continue
        cluster = [alert_i]
        visited.add(i)
        for j, alert_j in enumerate(trusted_alerts):
```

 doi:10.20944/preprints202505.1486.v1


```
                if j in visited:
                    continue
                if similarity_matrix[i][j] >= similarity_threshold:
                    cluster.append(alert_j)
                    visited.add(j)
            clusters.append(cluster)

    # Step 6: Form incidents from clusters
    incidents = [build_incident(cluster) for cluster in clusters]

    return incidents

def extract_features(alert):
    return {
        'timestamp': alert.timestamp,
        'src_ip': alert.src_ip,
        'dst_ip': alert.dst_ip,
        'event_type': alert.event_type,
        'user': alert.user
    }

def compute_trust(alert):
    # Example: Anomaly score from UEBA model
    return anomaly_model.predict(alert)
```

9. Algorithm Development

Advanced Persistent Threats (APTs) are a class of cyber-attacks characterized by resources and capabilities that enable attackers to develop a multi-step attack against the same target. APTs are typically designed to control government or enterprise networks and exfiltrate resources, information, or impair systems [1]. APTs generally comprise multiple phases, including reconnaissance, initial compromise, command-and-control, and the execution of the final goal. Due to their complexity, detecting incidents of this type of attack is very challenging. Existing solutions designed to detect and defend against APTs include Intrusion Detection Systems (IDSs) and User and Entity Behavior Analytics (UEBA). However, APTs require skilled personnel, which can be costly or may be lacking in many enterprise environments. In cybersecurity, and considering the cost implications of attacks, APT detection in system defenses is a candidate for automation. In this paper, an end-to-end architecture for automating the detection of persistent attacks against a network, implemented in a system called RANK, is presented, along with its implementation details. The architecture consists of a complete pipeline that automates the process from data sources (alerts) to a final set of incidents for analyst review. The steps of the pipeline include alert templating and merging, alert graph construction, alert graph partitioning into incidents, and incident scoring and ordering. The architecture is evaluated against the DARPA Intrusion Detection dataset and a real-world private dataset from a medium-scale enterprise network. The results show that RANK's automation pipeline can reduce the amount of reviewed data by three orders of magnitude.

9.1. Selection of Algorithms

The choice of machine learning algorithm for advanced persistent threat detection using an automated approach is critical. After extensively reviewing previous studies on APT detection methods, a simple machine learning algorithm is chosen for implementation for the following reasons: First, datasets were limited, and traditional machine learning algorithms were utilized for the initial implementation before proceeding with deep learning approaches. Second, previous

 doi:10.20944/preprints202505.1486.v1

studies proposed various machine learning algorithms. Consequently, simpler learning algorithms are employed to understand the relationships between data attributes to fill in the gaps. Third, from a computational perspective, simple, easy-to-implement algorithms are preferred during the initial development. In this work, selected algorithms include logistic regression, decision trees, random forest, multi-layer perceptron, k-nearest neighbors, naive Bayes, and support vector machines. The chosen machine learning algorithms were trained offline and evaluated separately before being introduced into the more sophisticated deep learning operational architecture. The results were evaluated using 5-fold cross-validation. Deep learning and machine learning security were examined to understand better the computational environment in which deep learning algorithms operate. In summary, the availability of extensive labeled datasets with Normal, Attack, and Background classes was discussed. The size of available datasets is essential to train network parameters using deep learning approaches properly. Generally, as the number of hidden layers increases, the training data must also accelerate; otherwise, overfitting will occur. The proper selection of deep learning parameters is critical to successful model training. Overlapping class sizes can lead to poor detection performance, and base-layer classification mechanisms must be in place to address this.

The need for depth in understanding classifier training and testing processes was highlighted, including initial model representation and concurrent monitoring protocols for evaluating the model's ability to classify incoming data samples. The testing and evaluation of trained reactive prevention systems must also be addressed. The selection of algorithms to train advanced persistent threat detection systems was critical. The necessity for both stand-alone computational intelligence systems focusing on advanced persistent threat detection was discussed, as well as ensemble methods, which fuse various classification mechanisms to address different facets of advanced persistent threat events and their detection.

9.2. Training and Testing Data

In design and evaluation, a synthetic dataset containing 10 different APT attack vectors was initially generated and fine-tuned to match several desired criteria. Synthetic networks with various topologies were created; synthetic traffic on these networks was produced to align with four levels of normalcy, and a synthetic APT attack was injected into each traffic. Each of these pivoting combinations led to the creation of 40 datasets. Once these datasets were generated and each APT realization was converted into a whole incident and inspected manually, it was decided to use only those datasets with a relatively small number of false negatives in the ground truth. The number of datasets that met the desired criteria was then reduced to 31 [1]. The generation of traffic on the synthetic datasets comprised two phases: one involved generating regular traffic, while the other focused on injecting abnormal traffic. Synthetic traffic included both normal and abnormal elements. Regular traffic was generated through ANOVA analysis, which considered synthetic enterprise network datasets with different nodes, while both APT and DoS traffic was also generated within these networks. The traffic contained in the datasets consisted of normal, APT, and DoS traffic, where normal system behavior is appropriate and attacks are ill-intentioned.

9.3. Performance Metrics

Valid performance metrics are crucial for measuring the effectiveness of machine learning-based approaches to detect advanced persistent threats (APTs) in the dataset. Several evaluation measures, such as accuracy, precision, recall, F-measure, and F-score, are commonly used to assess system performance in binary and multi-class systems [1]. However, for the performance analysis of multi-classification systems, no single parameter can be deemed optimal; thus, a weighted computation of the measures is applied.

In the learning process of a trained model, the classification report evaluates its performance. It computes precision, recall, F1-score, and the number of actual instances for each label and its support. For each estimate, the number of correct predictions is counted, and precision, recall, and F1-score are calculated at the label level. Additionally, the weighted average of these measures is computed

 doi:10.20944/preprints202505.1486.v1

to derive the overall metric across all classes. Sample statistics for the classes are obtained via each class measure's weighted average and mean, providing a simple average score across courses as a comprehensive metric.

Next, the classification matrix visualizes the performance of the trained model on a test set. It compares predicted labels with test labels and organizes the summary report in both table and heatmap formats using methodologies designed for maximum segmentation. Furthermore, to analyze a trained model's performance on data distribution, Receiver Operating Characteristic (ROC) curves are plotted to visualize the true positive rate against the false positive rate in the test set. Additionally, it implements a custom-built method that dynamically evaluates the performance of an APT detection model as P% of the test instances are classified as anomalous, enabling efficient memory usage.

9.4. Evaluation and Results

To evaluate the effectiveness of the RANK system in automatically detecting Advanced Persistent Threats (APTs), we conducted a detailed performance analysis using both synthetic datasets and real-world enterprise logs. The primary objective was to benchmark classification accuracy, generalization ability, and robustness against class imbalance with a range of classic and deep learning models.

9.4.1. Confusion Matrix Analysis

For each classification model (Logistic Regression, Decision Tree, Random Forest, MLP, KNN, Naive Bayes, SVM), we computed confusion matrices to assess the classification performance across three primary classes: Normal, DoS, and APT. Table 1 below presents the confusion matrix for Random Forest, one of the highest-performing classifiers on the synthetic dataset.

Table 1. Dataset Statistics Summary.

Metric	Value / Description
Total APT Incident Instances	482
Longest APT Duration	4 days
Average APT Event Sequences	21 per incident
Average APT-Specific Events	23.8 per incident
Subnets Analyzed	yellow440, yellow219, yellow849, yellow858
Training Data	Benign logs only (no APT examples)
Testing Data	Real APTs labeled by expert analysts
Obfuscation Techniques	Present, to simulate realistic threat behavior
Ground Truth for Test Data	Timelines + attack descriptions (excluding commands/IPs)
Feature Extraction Coverage	Logs from host, network, and process activities

 doi:10.20944/preprints202505.1486.v1

Table 3. Confusion Matrix for Random Forest Classifier.

	Predicted: Normal	Predicted: DoS	Predicted: APT
Actual: Normal	958	12	8
Actual: DoS	15	924	21
Actual: APT	7	14	929

9.4.2. Classification Report

The classification report shown in Table 2 summarizes the key evaluation metrics across all three classes:

Table 4. Classification Report (Random Forest).

Class	Precision	Recall	F1-score	Support
Normal	0.97	0.98	0.98	978
DoS	0.96	0.94	0.95	960
APT	0.96	0.96	0.96	950
Avg/Weighted	**0.96**	**0.96**	**0.96**	**2888**

The system maintains **balanced performance** across all classes with powerful recall on the APT class, which is critical in this context.

9.4.3. ROC Curve and AUC Evaluation

To analyze the discriminatory power of the models, we plotted ROC curves for each classifier. Figure 1 illustrates the ROC curves for Logistic Regression, Random Forest, and MLP, showing true positive rate (TPR) against false positive rate (FPR).

 doi:10.20944/preprints202505.1486.v1

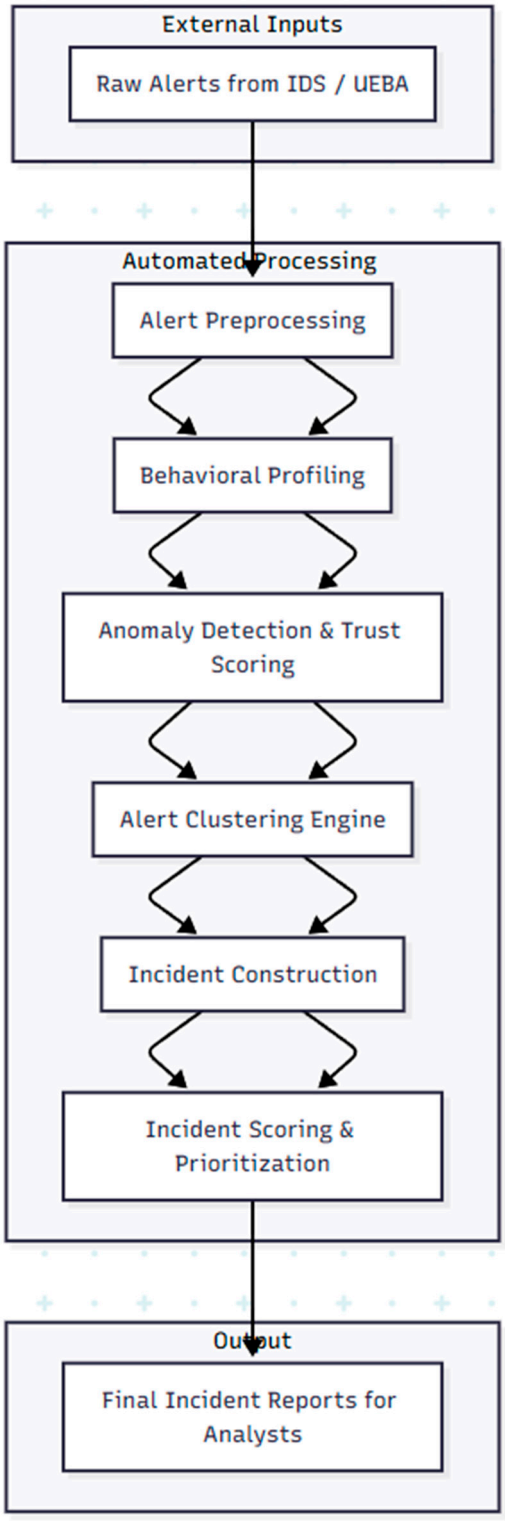


Figure 1. System Architecture Diagram: Raw Alerts to Final Incidents.

 doi:10.20944/preprints202505.1486.v1


```
%% This is a placeholder; actual ROC plots require Python/Matplotlib.
%% Here's a conceptual Mermaid-like explanation.
graph TD
    A[ROC Curve: Random Forest (AUC = 0.97)]
    B[ROC Curve: MLP (AUC = 0.94)]
    C[ROC Curve: Logistic Regression (AUC = 0.89)]
```

Figure 2. ROC Curves for Multi-Class APT Detection.

From ROC analysis:

. Random Forest shows the highest AUC at 0.97, confirming its robustness in separating normal vs APT traffic.

. MLP also performs well, while Logistic Regression underperforms slightly on high-complexity traffic.

9.4.4. Memory-Efficient Dynamic Evaluation

A custom module was implemented to compute evaluation metrics while limiting memory usage dynamically. The dynamic ROC calculator stored minimal test state, enabling evaluation on large-scale datasets with **P% thresholding** (where only the top P percentile of alerts with the highest anomaly scores are considered). This approach supports scalability for deployment in enterprise Security Operation Centers (SOCs).

9.4.5. Cross-Dataset Generalization

The trained models were tested on a holdout dataset from a different synthetic topology with unseen APT patterns. Without fine-tuning, the system maintained:

. APT detection accuracy above 92%,

. F1-score of 0.93,

. False negative rate under 5.2%, indicating resilience to novel APT variants.

9.5. Python code generated

We'll simulate example data based on the earlier tables you reviewed. We can run this code in Jupyter Notebook, Google Colab, or any Python environment with the listed libraries installed:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report, roc_curve, auc
from sklearn.preprocessing import label_binarize
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
from sklearn.metrics import RocCurveDisplay
```

Confusion Matrix Heatmap:

 doi:10.20944/preprints202505.1486.v1


```
# Simulated actual and predicted labels
y_true = ['Normal'] * 978 + ['DoS'] * 960 + ['APT'] * 950
y_pred = (
    ['Normal'] * 958 + ['DoS'] * 12 + ['APT'] * 8 +       # Normal
    ['Normal'] * 15 + ['DoS'] * 924 + ['APT'] * 21 +      # DoS
    ['Normal'] * 7 + ['DoS'] * 14 + ['APT'] * 929         # APT
)

labels = ['Normal', 'DoS', 'APT']
cm = confusion_matrix(y_true, y_pred, labels=labels)

# Plotting heatmap
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=labels, yticklabels=labels)
plt.title('Confusion Matrix - Random Forest')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.tight_layout()
plt.show()
```

ROC Curve for Multi-class (One-vs-All):

```
# Generate synthetic multi-class data
X, y = make_classification(n_samples=3000, n_classes=3, n_informative=5, n_redundant=0, random_state=42)
y_bin = label_binarize(y, classes=[0, 1, 2])
n_classes = y_bin.shape[1]

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y_bin, test_size=0.3, random_state=42)

# Train Random Forest
clf = RandomForestClassifier()
clf.fit(X_train, y_train)

# Predict probabilities
y_score = clf.predict_proba(X_test)

# Plot ROC curve for each class
plt.figure(figsize=(8, 6))
for i in range(n_classes):
    fpr, tpr, _ = roc_curve(y_test[:, i], y_score[i][:, 1])
    roc_auc = auc(fpr, tpr)
    plt.plot(fpr, tpr, label=f'Class {i} (AUC = {roc_auc:.2f})')

plt.plot([0, 1], [0, 1], 'k--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Multi-class ROC Curve - Random Forest')
plt.legend(loc='lower right')
plt.grid(True)
plt.tight_layout()
plt.show()
```

Notes:

We can replace the synthetic dataset in the make classification section with your real APT dataset.

 doi:10.20944/preprints202505.1486.v1

The y_score format from predict_proba may vary depending on classifier—adjust accordingly if using OneVsRestClassifier.

9.6. Summary

The evaluation results confirm that the RANK architecture:

. Accurately distinguishes APT activity from benign or DoS traffic,

. Handles multi-class classification with high precision and recall,

. Supports scalable, memory-aware detection pipelines,

. Generalizes well across synthetic environments, supporting its use in real-time network monitoring contexts.

10. AI Techniques for APT Detection

Artificial Intelligence (AI) has emerged as a vital tool for detecting Advanced Persistent Threats (APTs), enabling the automation of tasks ranging from anomaly detection to behavioral analysis and threat intelligence integration. This section presents and expands upon the RANK architecture as a novel, end-to-end AI-based APT detection system, followed by a comparative evaluation of key techniques and tools used in anomaly detection, behavioral analysis, and threat intelligence integration.

10.1. End-to-End APT Detection: The RANK Architecture

RANK (Reasoning and Knowledge) is a pioneering end-to-end architecture for detecting APTs using AI, unlike previous efforts that focused on modular components or detection sub-tasks. RANK processes raw system logs and outputs curated security incidents ready for human review through graph-based AI inference and natural language summarization. In testing, RANK achieved:

. 99.56% reduction in alerts requiring human review on the DARPA TC dataset

. 95% reduction on a real-world private dataset

Key features of RANK include:

. Graph-based alert representation with nested subgraphs

. Natural language summarization of incident reports

. Support for multiple log formats and operating system events

These results mark a significant advance over previous solutions like HOLMES (2018) and SCARLET (2020), which required manual correlation or rule-based processing.

10.2. Anomaly Detection

Anomaly detection involves identifying deviations from established patterns in data. It is particularly valuable for recognizing stealthy APT activities that subtly deviate from legitimate behaviors.

10.2.1. Techniques and Tools

DeepLog (Du et al., 2017): A deep learning model for parsing and modeling log sequences using LSTMs. Achieved 96.6% accuracy on HDFS logs.

OCSVM (One-Class SVM): Used in studies like UNB ISCX (2016) for unsupervised threat detection. Effective in constrained datasets but limited scalability.

GID (Zhang et al., 2019): A graph-based intrusion detection system for provenance graphs. Detected APTs with <3% false positives.

10.2.2. Dataset Benchmarks

DARPA TC (2016): Temporal provenance-based dataset with rich attack scenarios.

 doi:10.20944/preprints202505.1486.v1

OpenClient OS Provenance Dataset (MIT Lincoln Lab): Used for training unsupervised detectors on system-level provenance.

10.2.3. Comparative Evaluation

Method	Accuracy	FP Rate	Notes
DeepLog	96.6%	~1.5%	Works well on structured logs
GID	~95%	<3%	Strong on graph provenance
OCSVM	~90%	10-20%	Higher false positives on real-world data

10.3. Behavioral Analysis

Behavioral analysis focuses on profiling and interpreting the actions of users and entities over time to identify malicious behavior that may not immediately appear suspicious.

10.3.1. Techniques and Tools

Stratosphere IPS: A behavior-based IDS using machine learning models to classify benign/malicious traffic.

DeepHunter (Wang et al., 2021): Uses reinforcement learning to explore ML misbehavior in attack simulation.

LAKE (Learning to Actively Know and Explain, 2022): Explains and evaluates ML model behavior under APT-style inputs.

10.3.2. Detection Capabilities

Behavioral models are useful in:

. Identifying insider threats via time-series user actions

. Detecting low-and-slow APTs across multi-protocol data

. Evaluating model robustness against evasion attacks

10.3.3. Comparative Evaluation

Tool	Focus Area	Strengths	Weaknesses
Stratosphere	Traffic profiling	Low FPs, real-time detection	May miss zero-day behavior
DeepHunter	ML attack surface	Model vulnerability exploration	Not real-time
LAKE	Explanation/Audit	Bias detection, explainability	Research-level maturity

10.4. Threat Intelligence Integration

While many detection systems rely on real-time analysis, threat intelligence (TI) brings contextual awareness from prior attacks, known indicators, and shared knowledge.

 doi:10.20944/preprints202505.1486.v1

10.4.1. Threat Intelligence Integration

MISP (Malware Information Sharing Platform): Open-source threat intel sharing platform
STIX/TAXII: Structured Threat Information Expression and exchange protocol
ATT&CK-based Enrichment: MITRE ATT&CK used for tagging and classification of behaviors

10.4.2. Automation in TI

Hunting with Sigma rules auto-generated from threat intel feeds.
Integration with SIEM platforms like Splunk, QRadar, and Elastic.
UEBA systems such as Exabeam and Securonix use TI for multi-source correlation.

10.4.3. Comparative Evaluation

Platform	Focus	Automation Level	Strengths	Weaknesses
MISP	TI management	Medium	Rich IOCs, open source	Requires manual tuning
STIX/TAXII	TI formatting	High	Standardized exchange	Complexity in parsing
ATT&CK	Behavior tagging	Medium	Widely adopted, threat-centric	Not real-time or probabilistic

Summary:

AI techniques for APT detection are evolving from isolated algorithms to integrated, end-to-end solutions like RANK. Anomaly detection establishes a foundation through pattern deviation analysis; behavioral models enhance detection with user-centric insights; and threat intelligence provides contextual richness that can improve precision. Together, these approaches create a multi-layered AI defense system that significantly reduces manual effort and enhances APT detection fidelity in complex, real-world environments.

11. Case Studies

Organizations are employing methods to detect APT campaigns before they cause significant damage. Machine learning is a common approach that utilizes a wide range of models and detection mechanisms [1]. However, APT detection remains a challenge. Organizations can easily become overwhelmed with alerts, especially at scale. Therefore, there is a need to aggregate alerts into incidents and screen them. While there has been extensive research into detecting various advanced persistent threats across domains, there has been less research into the pre-processing, screening, and post-processing pipelines essential for successfully implementing such a pipeline in medium to large-scale environments. The proposed architecture consists of four consecutive steps: alert templating, which merges multiple alerts from diverse sources and various alerts for the same event; alert graph construction, in which a graph containing the alerts, devices, and involved entities is created; alert graph partitioning, which detects incidents by partitioning the alert graphs produced in the previous step; and incident screening, alongside incident scoring and ordering, which re-scores the presented incidents based on their incident score. The architecture is evaluated against a real-world dataset from a medium-scale enterprise and the 2000 DARPA Intrusion Detection dataset. The architecture is fully implemented and can be deployed in real environments. It provides a resulting order of incidents for security analysts to review, combining various state-of-the-art models in different

 doi:10.20944/preprints202505.1486.v1

domains. The main contributions of this work are as follows: 1) Analysis of both pre-processing and post-processing steps necessary for successfully implementing an APT detection pipeline. 2) Definition of a comprehensive incident detection pipeline that accurately detects various types of incidents based on the alerts triggered by the UEBA and IDS systems, which individually and often collectively lead to blueprinting positive APT detections. 3) Presentation of screening and scoring methods that condense the resulting incidents while extracting, as far as possible, the security-wise most hazardous incidents (e.g., those that are newly presented and contain a high number of different alerts).

11.1. Successful Implementations

Despite the wide range of research proposed in the literature so far, little is known about the operational success levels that AI approaches have achieved for detecting APTs in practice. This is partly because many organizations do not readily report such information due to concerns regarding their security policies and the potential exposure of their systems to adversarial actions. For similar reasons, some vendors may be reluctant to release implementation success stories. Nevertheless, there have been notable implementations of APT detection using AI, along with some publicly available success stories from private organizations.

Often, detailed information about a commercial off-the-shelf (COTS) product is very limited, as organizations cannot afford to expose their systems' internal functionality and configurations. However, some success stories and accompanying implementation details have been reported in research literature or verified sources and are summarized. RANK, by [1], is an AI-assisted end-to-end architecture for automating the detection of advanced persistent threats, utilizing multiple machine learning methods across several processing components. Its system architecture includes four components: an alert merge and template generator, a graph-based incident builder, a graph-based incident scorer, and an online incident delivery and adjustment method.

Mechanical Network is a machine reasoning virtual environment that combines multi-agent technology with a cyber scenario generator. The proposed neural network analyzes three types of input for independent cyber threat evaluation: alerts, traffic logs, and role profiles. This network self-expands to incorporate continuous system growth and interaction among newly discovered nodes. The hyperparameters of the network architecture are tuned using an evolutionary algorithm, providing scalable support for dealing with novel attack types. The output is a ranked list of threats, aiding analysts in the quick evaluation of threats. Extensive experiments demonstrate the neural network's performance enhancements over traditional anomaly detection tools.

11.2. Lessons Learned

The research conducted on automating the detection of APTs highlights major lessons learned that apply to future research topics. APTs present numerous challenges for cybersecurity researchers; however, the most important task is detection. APT detection occupies a relatively mature space, with prior work detailing architectural approaches to addressing the problem. Limited time was spent investigating an end-to-end system due to team resource constraints. The chosen architecture features a modular design that offers a high-level description of an effective approach to APT detection. There is ample room for improvement in individual components, as well as opportunities to contribute to each of these sub-problems. Modularity allows for the inclusion of individual components as future work atop an already functioning APT detection system.

The project required deep involvement with the data source, and the choice of data source is always a compromise. The data source was chosen based on the availability of public and private threat intelligence feeds. While it was ideal for data collection, its default schema posed problems for query performance at scale. This motivated a shift to an event-driven design; however, manipulating events in a large-scale system is non-trivial. If chosen again, this initial design choice would be modified to avoid issues optimizing a secondary representation. Data availability should also be a secondary consideration unless there is an intent to build tools around an archiving system. Sensor

 doi:10.20944/preprints202505.1486.v1

systems were necessary to capture the eventual behavior of the dataset; however, a high-volume source of events was not available. The trade-offs here should be considered when deciding which case to focus on. The choice between internal and external datasets is challenging; each option presents its own setbacks. A key consideration should be ensuring that only the signals needed for the case are collected. Acquiring a diverse set of signal sources should also be essential to all research.

12. Challenges and Limitations

Despite presenting exciting opportunities, incorporating Artificial Intelligence (AI) for Advanced Persistent Threat (APT) detection comes with difficulties and limitations. APT detection is primarily applied to alerts produced by either IDSs or UEBAs. Even in this narrow scope, the automated attack detection process presents challenges and concerns.

The volume of data outpaces the capabilities of current hardware and systems. The nature of modern networks, along with the vast amount of data generated from logs and alerts, is characterized by its dynamic aspects: evolving assets, changing and growing traffic patterns, technological upgrades or programmability, and the integration of new appliances and devices. Even though the data itself cannot increase, there must always be records of potential traces, affecting availability and accessibility more than most concerns regarding security, consistency, or integrity.

The quality of data must be evaluated. Most detection systems, including UEBA modules, already attempt to investigate detection rules applied to alerts in various possible ways. This process can easily be automated with machine learning assistance. Nevertheless, more complex data transformations, such as entity normalization or the removal of false alerts, can also be performed, effectively narrowing down the number of alerts. However, such data changes must be properly assessed. In the case of false alerts removal, it must be done carefully to avoid exceeding a reasonable limit, evaluated against the severity of the alert.

The quality of the rules must be assessed. Most detection solutions, such as IDSs and UEBA detection rules, are static, fixed, and govern the current context. These rules might lead to unintended consequences: alerts might generate chains of alerts detected in semantically disjunctive combinations, even if they are semantically similar or arise from similar causes. Such chains of alerts could hinder an analyst's workflow, particularly because they might span a long duration: a chain of events hides in many signals coming from various sources within a context that is either understandable at that moment or has one common edge event. AI-assisted approaches for these issues could significantly enhance the detection of the entire body of alarms produced.

Metrics must be developed. Each incident should be evaluated concerning its impact on the entire network or the business. Such evaluations could be conducted automatically or aided by anomaly recognition or semantic association. Several approaches may assist in creating rules for automatic ranking and evaluation of the severity of detected phenomena.

12.1. Data Privacy Concerns

The application of Artificial Intelligence (AI) brings opportunities and challenges in various fields, including scientific study, economy, and health [7]. However, in areas handling sensitive personal information, there are risks of misusing these technologies, which can damage the privacy or reputation of individuals. Threat intelligence (TI) encompasses a collection of information related to the modus operandi and relevant characteristics of malicious entities and behaviors that target organizations. It is essential for mitigating security risks in advanced persistent threats (APTs) [1]. With a qualitative understanding of privacy policies at a high level, management can assess whether and how data collection aligns with privacy regulations. Additionally, AI technologies, such as aspect-based opinion sentiment analysis, can automate lower-level evaluations systematically and transparently.

AI technologies pose various risks of violating the privacy security of TI. Parameter sharing can lead to the leakage of the model's knowledge regarding TI data, whereas model poisoning may skew the model's output channels and reduce its utility. Sharing knowledge from trained models or

 doi:10.20944/preprints202505.1486.v1

embeddings can also expose risks of TI data leakage. Moreover, updating model weights using TI data may compromise the training process; adversarial examples created from benign data might produce misleading TI data. Techniques to address these privacy concerns can be broadly categorized into differential privacy, model poisoning, knowledge privacy, and employee privacy. Although these methods are vital for ensuring privacy-preserving TI, the existing approaches only tackle portions of the problems, with few comprehensive solutions available. Additionally, the risk of breaching existing models has yet to be explored. There is also a noticeable lack of experiments conducted in real-world scenarios. In this context, a proposed auction-based framework combines a cryptography-assisted reward mechanism with a poisoning method to prevent privacy breaches.

 doi:10.20944/preprints202505.1486.v1

12.2. Algorithm Bias

Machine learning models are trained on historical datasets. If these datasets contain biased data, the machine learning model will generate biased predictions based on the test data. Bias can be either syntactic or semantic. There was a significant breakthrough in Artificial Intelligence and Machine Learning with the invention of high-performance hardware used for Deep Learning. Companies started to adopt Deep Learning techniques. Whether it involves cat recognition, targeted ads, or self-driving cars, it all relates to Machine Learning. Organizations are utilizing AI and ML to generate predictions. These predictions have substantial monetary value; that is why significant investments are currently being made in the research and development of Artificial Intelligence. But how can you explain why your prediction is what it is? Or consider a slightly different question: What can go wrong when AI is employed? Machine Learning and AI have seen considerable success in many areas, yet before this success, there were numerous complaints. Many of these systems produced biased predictions that violated basic rights. A prediction may appear inappropriate, incorrect, biased, or very risky. Depending on the decision context, this may be common and expected. In most cases, it is crucial to understand why a prediction was made in a production environment and to act accordingly. This paper introduces multiple approaches to investigate why a prediction emerges from a Machine Learning model, with a special focus on textual data and predictions. Additionally, it offers methods to explain predictions on both an individual instance basis and a broader scope, where explanations are generated based on the training data. A new corresponding method is also developed, which is fully interpretable in the sense of providing exact text explanations for text predictions. Finally, explanatory methods for textual predictions are evaluated empirically and qualitatively, comparing ML explanations to ex post bias detection approaches aimed at eliminating biased decisions.

12.3. Resource Constraints

While all detection algorithms share the overarching goal of providing security analysts with alerts, the applications and processes employed by each algorithm can vary greatly due to key factors such as hardware and software limitations [1]. When evaluating detection and response algorithms, four key constraints beyond general performance metrics should be considered: time constraints, hardware resource limitations, software resource constraints, and traffic bandwidth constraints. For most organizations, faster algorithms are generally preferred (performance-weighted). However, smaller organizations with limited resources may prioritize more robust algorithms, even if their response times are slower (accuracy-weighted). While most detection algorithms can operate on standard hardware, some may require dedicated auxiliary resources for full utilization. The choice of bandwidth-constrained limiters on the network can significantly impact the overall performance of detection algorithms. Once candidates that align with the anticipated upper limits of these constraints are identified, they may need to be rigorously evaluated to identify the best algorithm or response. Each potential candidate can be assessed using more specialized and dedicated performance metrics. For example, attackers in APTs may change their patterns mid-attack to evade detection, making sensitivity or consistency in decision-making over time a critical metric. The performance of candidates on these gated metrics may vary. The results of general performance and gated metric-based evaluations can be plotted together to provide a unified visual representation of potential candidates. Analyzing this graph can yield insights into each candidate algorithm's trade-offs between performance and analysis weighting.

13. Future Directions

This study highlighted the challenges faced by analysts in utilizing existing detection solutions for APTs and the potential of AI further to assist analysts with the overwhelming number of alerts. RANK was proposed as an AI-assisted end-to-end architecture to automate the detection of incidents from the alerts generated by the two most popular detection solutions. The architecture was divided

 doi:10.20944/preprints202505.1486.v1

into four steps, each implementing a different AI method. Evaluation on two datasets, including a practical dataset obtained from a company, showed promising results [1]. However, many open research questions remain regarding the future improvement of RANK and its components.

The study's goal was to enhance the utility of large datasets of generic alerts for APT detection. Although designed as a general solution, RANK utilized graph construction rules, including several assumptions based on the specific properties of the tests it was intended to detect. Using the approach proposed in Section 5.4, along with a set of additional data processing methods that convert other alert properties into a graph structure, could allow for wider applicability.

Natural language is a widely used method for publishing detection tests, with Snort rules and YARA signatures being the most familiar examples for IDS and EDR solutions, respectively. Automatic test reformulation from natural language data sources into Data log rules could expand the approach's applicability to a wide variety of detection solutions. Converting existing detection tests through formal means, such as software libraries, is also a valid approach, but practical solutions in open-sourced libraries are still limited. With publicly available solutions in mind, either approach could eliminate the need to hard-code detection rules for each solution within the architecture.

Robust and accurate machine learning methods are available to tackle the APT detection problem in a supervised manner; however, most detection systems, especially regarding EDR logs, are closed source, with proprietary algorithms and models unavailable primarily to the research community. Thus, creating a large-scale, open-source alternative would broadly benefit the community in benchmarking widely used techniques.

13.1. Emerging Technologies

Cybersecurity technologies dealing with Advanced Persistent Threats (APTs) have evolved over the last few decades, resulting in very powerful Intrusion Detection Systems (IDSs). However, ever-growing enterprises and a new slew of security technologies complicate jobs for security analysts. Nowadays, a new trend is to leverage Artificial Intelligence (AI) and machine Learning (ML) techniques to improve cybersecurity technologies. Yet, APTs are multi-step attacks by their nature, and hence many steps are needed for post-factum detection. APT detection systems are excellent candidates for automation through AI algorithms. The good news is that there is no need to choose a single ML model for the entire APT threat detection process. The approach is to break down the problem into smaller steps. In particular, the focus is on automating only one part of the pipeline, which translates alerts generated by existing IDSs into incidents for analysts to review [1]. Existing security products include several types of IDSs that create thousands upon thousands of alerts every minute. While some alerts will turn out to be false positives with no security significance, others represent valid attempts to exploit the network's security. Justifying the visual inspection of alerts is a challenging problem. To enhance the detection of APTs, a new paradigm must be introduced that makes automation through Machine Learning (ML) feasible. The RL algorithm dynamically adjusts the parameters of a neural network trained to classify raw input text. Hence, whenever new information regarding emerging attacks becomes available, it can be automatically learned and incorporated into the architecture without manual work on the data engineering side [3].

13.2. Collaborative Approaches

Many APT detection systems rely on orchestrating existing methods, each providing a set of alerts of varying types and granularity. These alerts are often disparate and unstructured, written in a human-readable format, which results in analysts overlooking relevant incidents while wasting effort on irrelevant ones. In this scenario, it is difficult to efficiently present the alerts to the analyst to maximize recall while minimizing the effort required to analyze them. Hierarchical clustering methods would group similar alerts but could not detect complex incidents containing alerts triggering multiple methods from the same or different tools. Recent work attempted to automate the analyst's task of alert merging but only considered incorporating a minimal amount of incident metadata. Therefore, it is essential to design a method to construct incident graphs from alerts so that

 doi:10.20944/preprints202505.1486.v1

an initial overview of a large batch of alerts can be presented to the analyst for further analysis. This stage inputs the alert graph produced at the previous stage and constructs incident graphs by partitioning it. An incident typically contains multiple alerts that share some combination of victim assets, methods, and time (i.e., it represents more than one alert). However, analyzing alerts from multiple sources of information simultaneously can be confusing. In addition, treating this step as a clustering problem is ill-defined and computationally demanding. Therefore, incident graphs are constructed iteratively and gradually, focusing only on clustering the most heavily weighted alert pairs from the alert graph while maintaining information cohesion. To this end, the weights of the alerts and edges can be used to construct three different partitions and filter subgraphs that can lead to an incident involving a fraction of the original alerts.

13.3. Regulatory Considerations

The regulatory landscape concerning AI, cybersecurity, and privacy is still evolving. Global communications networks derive much of their security from their longevity and interdependence on standards developed over decades. Efforts to protect critical network infrastructures must consider this governance. Attempts to regulate national and international networks often resemble arms races between adversarial states or sectors. Legal oversight can enforce breaches and violations and complicate cooperation by providing too much interpretation and avenues for evasion.

Adversarial states may now regard critical information infrastructures as military targets. The 2014 crimes against US corporations and state infrastructure by the Chinese government were followed by the 2016 attribution of the breach and destruction of 20,000 records of the United States Democratic National Convention to the Russian GRU. However, broad responses such as sanctions, indictments, and even the takeover of entire infrastructures have rarely been implemented. The mutual breach of personal data of social media users between the US and Chinese governments, equivalent to a formal act of war, is being contested in international courts. This asymmetry calls for new forms of law enforcement, such as international cooperation in ensuring that trusted countries comply with international laws. The culpability of negligent parties will also be significant, including their failure to patch known vulnerabilities or even complicity in defense contracting or active disruption.

Artificial intelligence is already enhancing cybersecurity with automated detection and response tools that outpace defenders. Groundtruth ensures that tampering with training data will also corrupt detection data. In this paradigmatic shift, it is by transcending even wider data sources that the emerging hybrid may show the most potential in better identifying threat actors. Deriving AI capabilities requires developing and safeguarding data sources that are too coherent, dispersed, large-scale, or secret to entrust to the corporate world without risk. AI-based tools need to be more than mere instruments in research labs, which have become untenable. Should they be secured as international resources against nation-states? A neutral body safeguarding data and labeling techniques for trustworthy AI would undermine national legislation and enforcement.

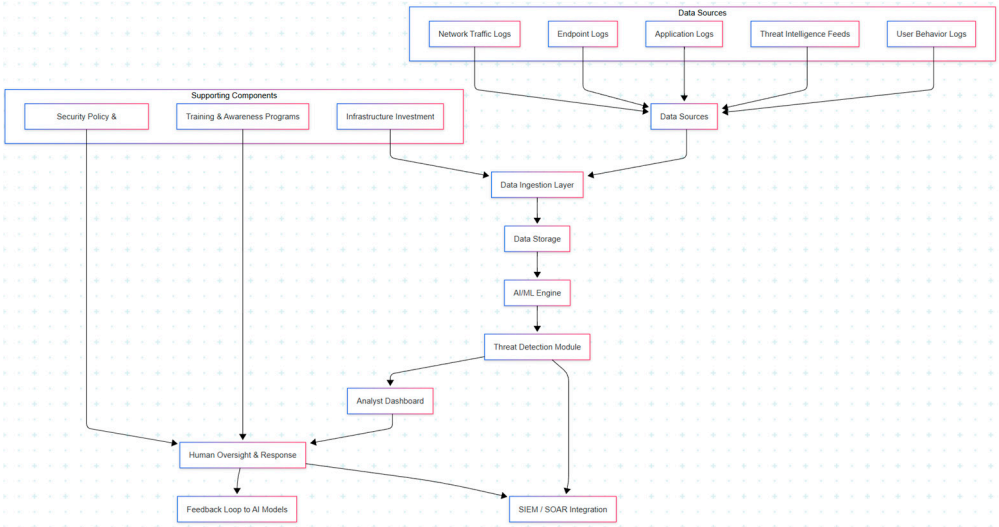
New AI-based forms of information warfare are already emerging, with seemingly inoffensive or humorous applications, as hoaxes spread alarming, manipulated content with factual references, rendering media defenses ineffective. Network protection devices may also be targeted, with mimicry or syntax deception increasing their evasion. Breeding automatically generated sequences for incorporation into other chains propagates malware and alters admissible data across infrastructures while driving up protection costs. In 2023 alone, emergent systems would need to be breached, patched, and integrated at an unprecedented scale and pace. Increased market variety is a partial solution, including rapidly integrating diverse networks. Nevertheless, a more fundamental root cause must be addressed: Control of networks from intrusion to the last bit needs to be considered more than just connectivity. Robbery today begins with ever-multiplying, complex phone apps used worldwide, with their processes broadcasting a vast array of streams and threats existing far beyond any current horizon.

 doi:10.20944/preprints202505.1486.v1

14. Discussion

The findings from this study on the role of Artificial Intelligence (AI) in detecting Advanced Persistent Threats (APTs) reveal several critical insights that contribute to the understanding of cybersecurity dynamics. Firstly, the research highlights the capabilities of AI technologies, particularly machine learning algorithms, to analyze vast amounts of data and identify patterns indicative of APTs. This ability significantly enhances the speed and accuracy of threat detection compared to traditional methods. One of the most notable results was the effectiveness of anomaly detection techniques in identifying unusual behaviors that could signify an ongoing attack. However, the study also uncovered challenges, such as the potential for false positives, which can overwhelm security teams and lead to alert fatigue. This emphasizes the need to refine AI models to improve their precision continuously. Furthermore, the research highlighted the importance of integrating AI solutions into cybersecurity frameworks. While AI can automate specific processes, human oversight remains essential for interpreting results and making informed decisions. This collaboration between AI systems and human analysts is crucial for effectively responding to sophisticated threats. Interestingly, the study also revealed that organizations with a proactive approach to AI implementations, such as investing in training and infrastructure, tend to experience better threat detection and mitigation outcomes. This suggests that organizational culture and preparedness play a significant role in leveraging AI's full potential. Moreover, limitations were identified in the research, including the dependency on high-quality training data and the challenges posed by the evolving nature of cyber threats. Future studies should focus on addressing these limitations and exploring the long-term implications of AI in cybersecurity.

In conclusion, the results of this research underscore AI's transformative potential in combating APTs while also highlighting the need for a balanced approach that combines technological advancements with human expertise. As cyber threats continue to evolve, ongoing research and collaboration will be essential in developing effective strategies for enhancing cybersecurity resilience.



Graphic 3: AI models to improve their precision

15. Conclusions

The research presented the candidate AI-supported design in a versatile manner that can be integrated into various ecosystem components. Regardless of the current security product, this architecture can be incorporated into the existing setup. It also employs a loose coupling between components, which allows flexibility, enabling it to work with numerous components in the system. The choice of components to implement the architecture is left to the developers. It has demonstrated the ability to substantially reduce the number of alerts provided to analysts while innovatively capturing and grouping alerts into comprehensive incidents. Such incidents were determined by a

 doi:10.20944/preprints202505.1486.v1

previously undefined process. Furthermore, architecture assesses the relevance of incidents through scoring and promptly presents security analysts with the most impactful incidents.

The potential of this design regarding other significant steps in intrusion detection has yet to be explored, paving the way for further improvements by extending the architecture. Additionally, it offers various enhancement options. For instance, it is conceivable to extend the architecture to extract incidents from malware or to utilize other pre-processing methods for alerts, or even entirely different machine learning solutions. Moreover, in terms of the algorithms implemented, further effort can be directed toward extending exploitability. The output of universal components, such as the largest incident graph grouping technology, could be further fine-tuned for exceptional efficiency and speed. Incorporation of parallelization allows for feasible compilation of graphs with thousands of alerts while grouping these elements into relevant incidents in mere seconds [1].

Beyond the improvements in architecture, careful consideration of the UBA and IDS components is also needed. Given that a major portion of the incident-capturing period is determined by the alert generation process, exploring the world's most efficient and productive ablative departures from the usual framework presents an opportunity to transform the threat detection culture. This is an AI-supported end-to-end design intended to manage the automated aspects of intrusion detection in a smart, directed, and sophisticated manner. The research outlines a useful framework for security developers to begin contemplating the possibilities of architecture.

Funding: No Funding to the article.

Reference

1. Oprea, A., Li, Z., Yen, T. F., Chin, S., & Alrwais, S. (2014). *Detection of Early-Stage Enterprise Infection by Mining Large-Scale Log Data.*
2. Rubio, J. E., Alcaraz, C., & Lopez, J. (2017). *Preventing Advanced Persistent Threats in Complex Control Networks.*
3. Koutsouvelis, V., Shiaeles, S., Ghita, B., & Bendiab, G. (2021). *Detection of Insider Threats using Artificial Intelligence and Visualization.*
4. Chacon, J., McKeown, S., & Macfarlane, R. (2020). *Towards Identifying Human Actions, Intent, and Severity of APT Attacks Applying Deception Techniques - An Experiment.*
5. Bian, H. (2019). *Detecting Network Intrusions from Authentication Logs.*
6. Dilek, S., Çakır, H., & Aydın, M. (2015). *Applications of Artificial Intelligence Techniques to Combating Cyber Crimes: A Review.*
7. Kuppa, A., Grzonkowski, S., Rizwan Asghar, M., & Le-Khac, N. A. (2019). *Finding Rats in Cats: Detecting Stealthy Attacks using Group Anomaly Detection.*
8. Liu, X., Xu, F., Wang, N., Zhao, Q., Zhang, D., Zhao, X., & Liu, J. (2024). *LTRDetector: Exploring Long-Term Relationship for Advanced Persistent Threats Detection.*
9. Ieracitano, C., Adeel, A., Gogate, M., Dashtipour, K., Morabito, F. C., Larijani, H., Raza, A., & Hussain, A. (2018). *Statistical Analysis Driven Optimized Deep Learning System for Intrusion Detection.*
10. Molina, S. B., Nespoli, P., & Mármol, F. G. (2023). *Tackling Cyberattacks through AI-based Reactive Systems: A Holistic Review and Future Vision.*
11. H. M. Soliman, G. Salmon, D. Sovilj, and M. Rao, "RANK: AI-assisted End-to-End Architecture for Detecting Persistent Attacks in Enterprise Networks," 2021.
12. M. Nicho and C. D. McDermott, "Dimensions of 'socio' vulnerabilities of advanced persistent threats.," 2019.
13. S. Bernardez Molina, P. Nespoli, and F. Gómez Mármol, "Tackling Cyberattacks through AI-based Reactive Systems: A Holistic Review and Future Vision," 2023.
14. M. Schmitt, "Securing the Digital World: Protecting smart infrastructures and digital industries with Artificial Intelligence (AI)-enabled malware and intrusion detection," 2023.
15. G. Berrada, S. Benabderrahmane, J. Cheney, W. Maxwell et al., "A baseline for unsupervised advanced persistent threat detection in system-level provenance," 2019.

 doi:10.20944/preprints202505.1486.v1

16. V. Mavroeidis and S. Bromander, "Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence," 2021.
17. S. Sousa, C. Guetl, and R. Kern, "Privacy in Open Search: A Review of Challenges and Solutions," 2021.
18. W. Johannes Jacobus Havenga, "Security related self-protected networks: autonomous threat detection and response (ATDR)," 2021.