Journal of Applied Big Data Analytics, Decision-Making, and Predictive Modelling Systems

# Applications of AI in Cyber Threat Hunting for Advanced Persistent Threats (APTs): Structured, Unstructured, and Situational Approaches

**Nur Alam Farhad Shakil**[1], **Ripan Mia**[1] **and Imtiage Ahmed**[2]

[1]*Information Technology, Washington University of Science and Technology, Alexandria, Virginia, USA*
[2]*Department of Computer Science and Engineering, World University of Bangladesh, Bangladesh*

This manuscript was compiled on November 29, 2023

## Abstract

Advanced Persistent Threats (APTs) represent a severe threat in cybersecurity because of their stealthy, sophisticated, and long-term nature. APTs cannot be detected by traditional rule-based detection since APTs typically utilize zero-day exploits, polymorphic malware, and legitimate credentials to evade detection. Artificial Intelligence (AI) supports cyber threat hunting by analyzing large datasets, detecting subtle adversarial patterns, and interpreting anomalies in context within dynamic environments, though its practical deployment faces notable challenges. This article presents a theoretical examination of AI-based methods for APT detection and mitigation based on three complementary frameworks: structured, unstructured, and situational approaches. Structured methods utilize deterministic models, including graph-based anomaly detection and supervised learning algorithms, to recognize known attacker activity. Unstructured methods use unsupervised learning, natural language processing, and deep neural networks to find hidden patterns in heterogeneous data (e.g., logs, network traffic). Situational methods incorporate contextual awareness, threat intelligence, and reinforcement learning to tailor hunting methods to adaptive operational environments. The integration of these techniques can support proactive multi-stage APT campaign detection, though practical challenges such as data limitations and adversarial evasion may hinder consistent performance. Some of the most important challenges, such as adversarial machine learning, interpretability, and ethics, are considered to highlight the subtlety of applying AI in practice. Drawing on insights from cybersecurity, machine learning, and cognitive science, this paper presents a taxonomy to assess AI's role in addressing APTs, while highlighting practical challenges and future research directions to improve adaptive cyber defense.

**Keywords:** *Advanced Persistent Threats, Artificial Intelligence, Cyber Threat Hunting, Detection Methodologies, Machine Learning, Situational Awareness, Zero-Day Exploits*

**Received:** August 19, 2023 **Revised:** October, 16, 2023 **Accepted:** November, 20, 2023 **Published:** December, 07, 2023



## 1. Introduction

Advanced Persistent Threats (APTs) are the most powerful threats to modern-day cybersecurity [1]. They are initiated by well-resourced and sophisticated attackers who use covert, long-term campaigns in an attempt to access targeted networks and go undetected for extended periods of time. APT campaigns, as opposed to opportunistic attacks, are objective-driven and customized in nature, usually instigated by nation-states or organized crime groups with discernible goals like espionage or intellectual property theft. After entering a network, the APT attacker spreads horizontally, privileges are escalated, and they achieve persistence while keeping overt malicious signatures to an absolute minimum. APTs are able to evade normal security controls such as firewalls and signature-based antivirus tools due to their stealthy and adaptable nature. This cat and mouse game has shown the limitation of traditional reactive defensive methods and the need to have proactive threat hunting measures that can detect stealthy intrusions before malicious damage is caused.

Cyber threat hunting today is a proactive method in the detection and incapacitation of threats like APTs that evade regular monitoring and alerting. Threat hunting is the active process of filtering through system and network telemetry for as-yet non-incident-producing signs of malicious activity. It is something more than response to known indicators of compromise; it is the process of making hypotheses regarding potential attacker presence and searching for behavioral artifacts in logs, network traffic, and endpoint telemetry. As opposed to passive security monitoring, hoping for pre-configured signature or known-pattern-based alerts, threat hunting is active hunting of the unknown and the abnormal. Threat hunting combines the experience and instincts of security analysts to predict adversary behavior and uncover subtle indications of an imminent attack. By continuous construction and hypothesis testing on the basis of data at hand, threat hunting seeks to minimize enemy dwell time—the duration for which the attackers remain concealed within an infrastructure—limiting the extent of damage caused by APT attacks.

Sophercation and sheer size of existing IT environments present huge hurdles, however, to the effectiveness of manual threat hunting. Large enterprises produce enormous volumes of logs and network events every day beyond what a human analyst could possibly examine in depth. Additionally, APT actors make a special effort to blend their activity with legitimate traffic, rendering it difficult to detect malicious signs like looking for a needle in a haystack. To augment human hunters and address these challenges, organizations are resorting more and more to Artificial Intelligence (AI) and machine learning techniques [2]. AI holds the promise of expanding threat hunting to large volumes of data and recognizing patterns too elusive or complex for human analysis. By automatically scanning millions of events, AI systems can highlight suspicious anomalies, correlate apparently unrelated alerts, and even provide hypotheses for investigation. In short, AI is a force multiplier for threat hunting teams, performing the data processing and pattern matching that is boring and enabling human analysts to think at high-level reasoning and decision-making.

AI-powered cyber threat hunting encompasses a range of methodologies that can be grouped into broad categories of structured, unstructured, and situational methodologies. Structured methodologies depend on pre-defined attack behavior knowledge and are based on rule-based reasoning or supervised learning-based algorithms for the identification of known threat signatures and behaviors. Unstruc-

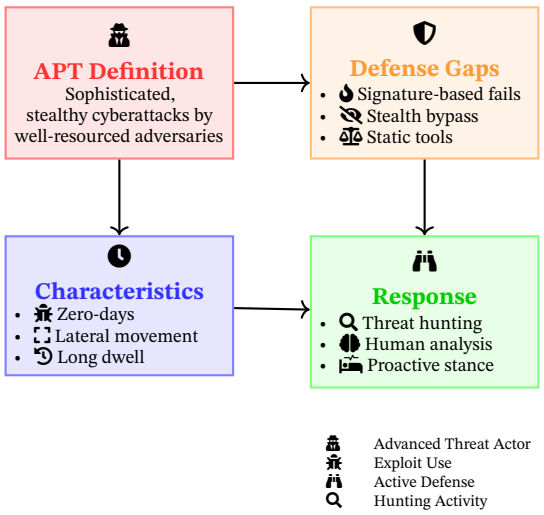


**Figure 1.** APT Lifecycle and Modern Defense Strategy Mapping

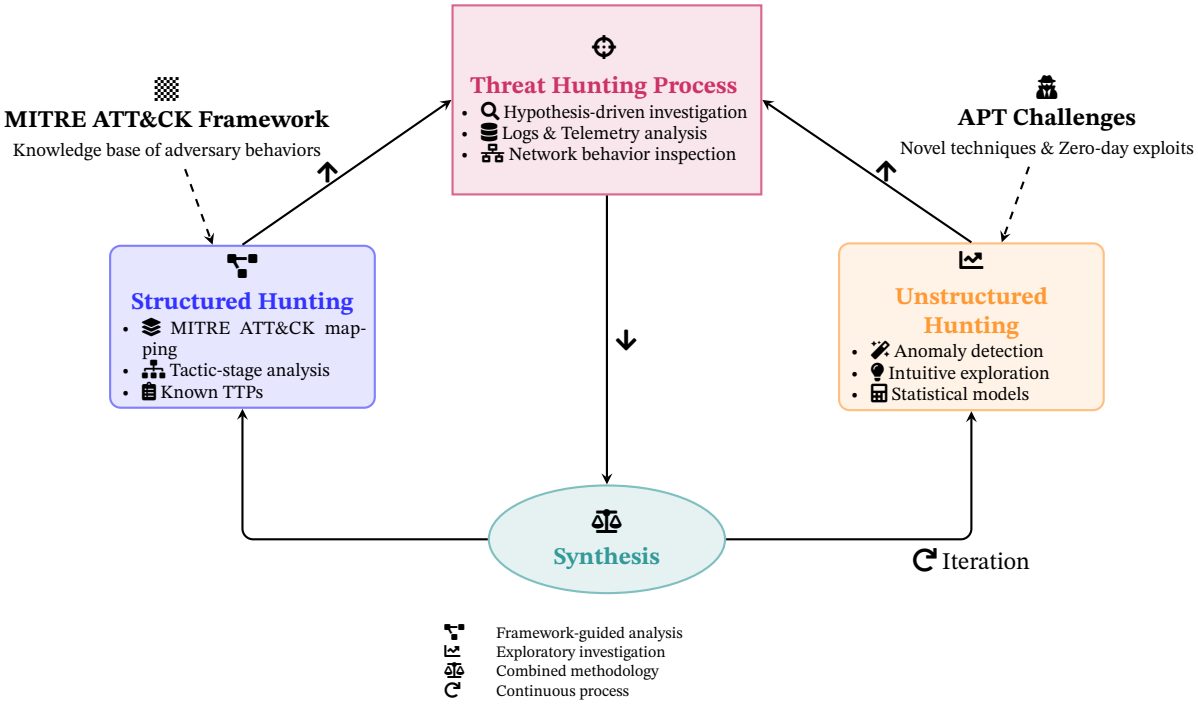


**Figure 2.** Threat Hunting Methodology: Structured and Unstructured Approaches Integrated Through MITRE ATT&CK Framework

tured methods, however, eliminate explicit hypotheses in advance; they instead utilize unsupervised machine learning and sophisticated data analytics to identify anomalies or data clusters without prior labeling, which typically uses statistical analysis and natural language processing (NLP) tools to decipher unstructured logs and evidence. Situational approaches focus on context awareness, combining information regarding the environment and present situations—like asset worth, user roles, and external threat intelligence—so that detection logic is dynamically tuned. All of these paradigms address a different aspect of the APT detection problem: structured paradigms are optimal for detecting attacks that follow familiar patterns, unstructured paradigms allow discovery of unknown or covert attack traces, and situational paradigms ensure that detection is dependent on context.

In recent years, research and practice in the industry have given rise to a variety of AI approaches to augmenting each of these paradigms of threat hunting. For instance, supervised machine learning-based classifiers have learned to identify malicious command strings or network traffic patterns common to APT malware, capturing expert knowledge of known attack methods in statistical models. Anomaly detection algorithms have been used on user behavior data to create baseline profiles and raise alarms for deviations that can signal a laterally moving or exfiltrating compromised account. More context-sensitive AI systems now include dynamic data—like correlating a detected security event against a known adversary tactics database or adjusting alert severity depending on the value of the affected system—to improve priority and interpretation of possible threats. These AI-based techniques are increasingly at the forefront of contemporary security operations centers (SOCs), where they act as sophisticated tools complementing human hunters by performing mundane analysis automatically and by delivering insights that otherwise would be hard to obtain.

As exciting as AI is for threat hunting scalability for APTs, leveraging these technologies is essential with a clear awareness of their strengths and weaknesses. The use of AI to threat hunting is not a silver bullet but does require cautious design to make sure models are precise, resistant to adversarial evasion, and understandable to

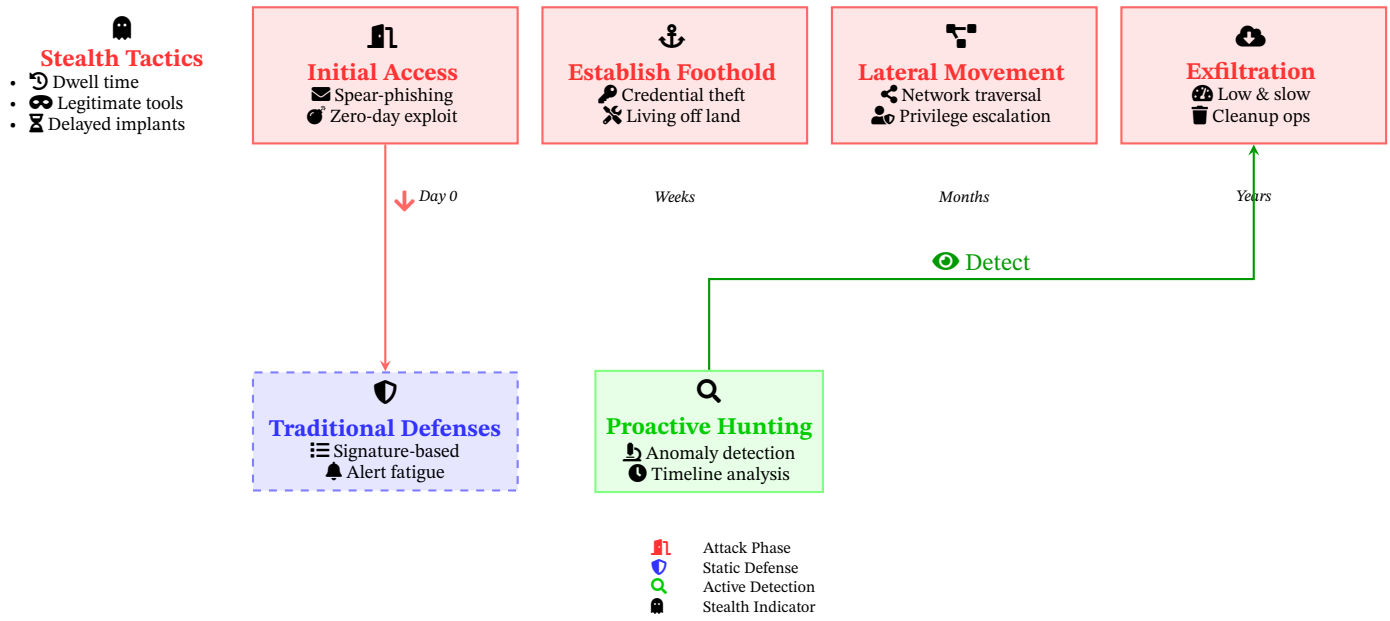


**Figure 3.** APT Lifecycle Visualization Showing Stealth Tactics and Defense Evolution

analysts [3]. Further, the adaptive and changing nature of APT tactics necessitates AI models to update continuously to novel patterns of malicious activity. This paper explains the cutting-edge uses of AI for APT cyber threat hunting, analyzing how structured, unstructured, and situational methodologies each play a part in detecting and analyzing hidden attackers. We explore not only the success that these AI methodologies have enabled but also the inherent difficulty of applying them in the wild—ranging from issues of data quality to the potential for attackers to deceive learning algorithms.

The rest of this paper is structured as follows. Section 2 presents cyber threat hunting basics and the nature of APTs that makes them so difficult to detect. Section 3 discusses structured methods, looking at rule-based systems and supervised learning models that use known indicators and labeled data to detect threats. Section 4 touches on unstructured approaches, with an emphasis on unsupervised learning, anomaly detection, and natural language processing methods for pattern discovery in data without necessarily having prior knowledge. Section 5 addresses situational approaches and context-aware AI, outlining the ways environmental and contextual data enhance APT detection through methods such as Markov decision processes for dynamic analysis. Section 6 discusses the limitation and disadvantage of using AI for threat hunting, including data inadequacies, false positives, and evasion techniques from attackers. Section 7 discusses future directions of this research field, including innovative research and developments that will further improve AI-based threat hunting. Section 8 concludes the paper by making some observations on the convergence of these methodologies and the direction of using AI against advanced persistent threats in the future.

## 2. Cyber Threat Hunting and APT Characteristics

Cyber threat hunting is a detective function that complements standard security monitoring by looking for threats that have evaded preventive and detective controls. In contrast to automated alerting processes using pre-defined signatures or rules, threat hunting is most typically initiated by human analysts developing hypotheses on possible security incidents. Hunters utilize their knowledge of attacker techniques and normal behavior on their networks to inform these hunts. A typical threat hunt engagement might start with a trigger or indicator—an out-of-hours network connection, a late-evening login suspicion, or a strange process operating on a priority server—that raises an eyebrow to the possibility of something being out of order [4]. From there, the analyst works through various streams of data (network logs, endpoint telemetry, authentication logs, etc.), dropping breadcrumbs like breadcrumbs in a criminal case that they would work through. It is an iterative and nonsequential process: hypotheses are ruled out or winnowed down as more data are considered. The ultimate aim is either to uncover a latent malicious presence or to be in a position to rule out absolutely that the anomaly is not hostile. Threat hunting really heightens the security team's situational awareness and understanding of their surroundings, so the subtle signs of compromise are not lost in the noise of baseline behavior.

A key component of an effective threat hunt is the employment of frameworks and methodologies to guide the investigation process. Perhaps the most popularly employed framework is the MITRE ATT&CK matrix, which provides an exhaustive catalog of tactics and techniques employed by attackers when conducting intrusions. By correlating seen activity or anomalies against recognized techniques (i.e., recognizing that a sequence of PowerShell commands maps to a recognized privilege escalation technique), hunters can rigorously identify which phase of an intrusion may be occurring and what to look into next. Standardized threat hunting methodologies typically consist of creating targeted hypotheses within such frameworks, sometimes referred to as intelligence-driven or hypothesis-driven hunts. For instance, if threat intelligence indicates that one particular APT group is known to have activity within the sector and has historically used credential dumping as a precursor to lateral movement with stolen credentials, then a hunter can look specifically for signs of anomalous credential access (such as LSASS memory dumps) followed by authentication activity between internal hosts. By having a playbook of a sort based on past experience with the tactics of an opponent, structured threat hunting can effectively reduce the search space and concentrate on high-risk indicators of attack.

Structure is good, but threat hunting also appreciates the flexibility of unstructured or exploratory analysis, particularly when confronted with innovative opponents. Hunters trust their judgment and experience to notice when "something just looks off" in the data, even though it may not be a well-defined mapping to an recognized attack pattern. It may entail exploratory scanning of big volumes of data with data science methods, visualization, or ad-hoc queries to detect outliers. For example, an analyst may graph attempts at authentication over time and observe a low-key but steady pattern of accesses to a prized database during unusual hours that would be worth in-

vestigating further. Unstructured hunting can uncover anomalies that structured approaches may not catch, especially new techniques that have not yet appeared in any threat intelligence database [5]. In reality, good threat hunters incorporate both methods: they apply formal methods to manage known threats and free-form investigation to find the unknown unknowns. This mutual support provides extensive protection against APTs that can span the use of thoroughly documented techniques through the development of completely new exploits.

Advanced Persistent Threats are the ultimate threat that necessitates strong threat hunting because of their unique nature. The term "advanced" is employed to describe the advanced toolkits and techniques applied by these actors. Zero-day exploits (unknown and unpatched vulnerabilities) are typically utilized by APT actors to provide initial access, or they execute highly targeted social engineering campaigns such as spear-phishing with very highly tailored lures that are convincing to single recipients. They can also use multi-step exploit chains to evade security layers, mixing custom malware with living-off-the-land tactics (abusing legitimate system tools and processes) to maintain a low footprint. The "persistent" refers to the attackers' will and capability to stay. Once inside a network, APT players typically create mechanisms for persistence—anything from hidden backdoor programs and scheduled tasks that reopen the door, to keeping hijacked credentials active and creating other user accounts to use at some future point in time. They don't smash-and-grab; they quietly spy and become indistinguishable from routine operations, occasionally running slowly enough so that any alarm based on volume does not trigger. They continuously work towards their objectives, which may be espionage, exfiltration of data, or pre-sabotage, month after month or week after week, and continuously evolving to remain alive in the environment and evade detection.

The second characteristic feature of APTs is their employment of stealthy lateral movement and privilege escalation. Once the entry is gained through the initial violation of a typically low-privileged machine or user account, the attackers systematically navigate the internal network and look for more valuable assets like domain controllers, databases, or file servers with sensitive information. They exploit trust relationships between computers—by using methods like pass-the-hash, pass-the-ticket, or token impersonation to jump laterally without triggering alarms. Lateral movement usually appears as regular administrative traffic if professionally executed; for instance, the use of valid credentials to access a server will not raise a credential abuse flag in most systems. APT actors usually have a complete understanding of how to exploit the internal infrastructure of the organization against itself [6]. They can pull password hashes from one box, crack them locally, and then use those credentials somewhere else, or employ admin tools (e.g., PsExec, WMI) to run commands remotely, all of which can look like normal IT traffic on the logs unless examined in context.

The "threat" aspect of APT highlights the intent and possible effect of these attackers. Different from typical cyberthieves who would use ransomware for quick cash, APTs typically go after strategic assets. They could steal sensitive intellectual property, private communications, or government secrets and do so covertly to maintain their access over a longer duration. APT data exfiltration is typically throttled and disguised. For example, instead of dumping gigabytes of information on the network simultaneously (something which could stress monitoring equipment), an APT will drip feed data slowly over a period, or encrypt and mask it in what appears to be normal traffic (like outbound HTTPS to cloud storage) so it can't be found. In other cases, APTs stage internal servers within an organization and compile stolen data on those, and then distribute it out in batch mode during opportune times. They are also apt to clean up after themselves or leave "sleepers" — inactive implants that can be turned on later — and therefore it is unremovable. The combination of these APT traits makes an enemy who is very hard to find and suppress. Legacy security solutions based on known signatures (e.g., virus signature or preconfigured intrusion detection signature) won't typically capture APT as the tactics are carefully crafted for a specific target and change their disguise over time. Behavioral detection signatures can also be bypassed if the steps taken by the attacker are within the confines that can be defined as normal for the compromised system or user. Thus, APT-targeted organizations have come to understand that mere automatic alerts are not enough; threat hunting has to be proactive to uncover the faint footprints of APT attackers. This often involves looking for small anomalies: a piece of equipment that is communicating with an IP block it never contacted before, an account that normally logs in from New York suddenly logging in abroad in the middle of the night, a system process forking a command shell (something that is unusual behavior for this process), or an unexpected blip in file access on a server after hours. Any one of them might be a needle in the haystack that suggests a greater malicious operation.

Most importantly, perhaps, APT threat hunting is dependent on context being understood about how things are going [7]. An action that is innocent in one context (an admin running a network scan as maintenance is being performed) might be suspicious in another (the scan being run by a non-admin user on an unauthorized system). The hunters need to use contextual data — asset criticality, user roles and usual behavior, temporal trends, and threat intelligence on known attackers — in order to eliminate false positives and identify true signs of compromise. For instance, if the computer of a developer displays a process compiling code in the middle of the night, this would be normal if there are looming deadlines; but if the same is displayed by a financial clerk's machine, it will be unusual. Similarly, knowing what malware family is being targeted by APT actors in the industry will alert hunters to look for a certain file hashes or process names in their environment. This union of low-level data (events, logs) and high-level context (what is occurring where, when, and who it is) is at the center of successful threat hunting.

## 3. Structured Approaches: Rule-Based and Supervised Learning

**Algorithm 1:** Rule-Based Threat Detection

**Input:** Incoming security events $E$, Rule set $R = \{r_1, ..., r_n\}$
**Output:** Generated alerts $A$
Initialize alert list $A \leftarrow \emptyset$;
**for** *each event* $e \in E$ **do**
  **for** *each rule* $r_i \in R$ **do**
    Extract condition $c_i \leftarrow r_i$.condition;
    **if** $e \models c_i$ ; `// Event matches rule condition`
    **then**
      $A \leftarrow A \cup$ GenerateAlert$(e, r_i)$;
      **if** $r_i.break\_on\_match$ **then**
        *break;*
      **end**
    **end**
  **end**
**end**
Apply correlation engine to $A$ ; `// Check multi-event patterns`
**if** $\exists a_1, a_2 \in A$ *matching correlation pattern* **then**
  $A \leftarrow A \cup$ GenerateMetaAlert$(a_1, a_2)$;
**end**
Update rules $R \leftarrow$ FetchLatestIoC() ; `// Periodic updates`
**return** $A$;

Structured threat hunting methods utilize monitored attacker activity and established patterns to inform the detection process [8]. For AI-based apps, "structured" is employed to mean that the system is

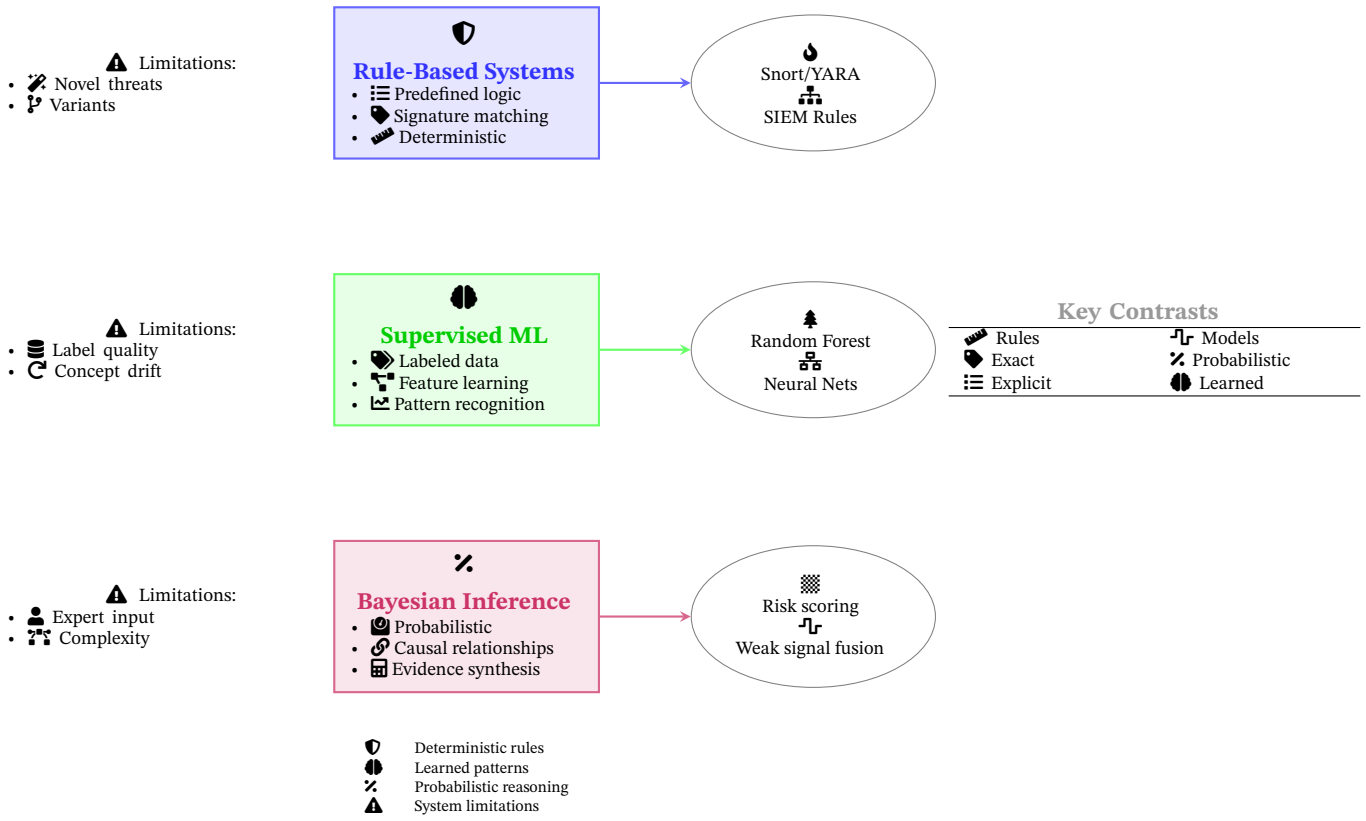


**Figure 4.** Structured Detection Approaches: Rule-Based Systems, Supervised Learning, and Bayesian Inference

**Algorithm 2:** Supervised Learning Threat Detection

**Input:** Training data $D = \{(\mathbf{x}_i, y_i)\}$, Detection threshold $\tau$
**Output:** Trained model $M$, Real-time predictions

**Training Phase:**;
Extract features $\mathbf{X} = [\phi(e_1), ..., \phi(e_m)]^T$ ; `// Feature engineering`
;
Train classifier $M \leftarrow \arg\max_M \mathcal{L}(M; \mathbf{X}, \mathbf{y})$;
Optimize hyperparameters via cross-validation;

**Detection Phase:**;
**for** *each new event* $e_t$ **do**
  Extract feature vector $\mathbf{x}_t \leftarrow \phi(e_t)$;
  Compute score $s_t \leftarrow M(\mathbf{x}_t)$ ; `// Using learned` $P(y = 1|\mathbf{x})$
  **if** $s_t > \tau$ **then**
    Trigger investigation alert;
    Log explanation $\sum w_i x_{t,i}$ ; `// Feature contribution analysis`
  **end**
**end**
Schedule model retraining with fresh data;
**return** *Alert decisions*;

executing either on human-authored rules or models educated with labeled examples of bad and good behavior. Both approaches rely on the assumption that we have some a priori insight into how bad behavior looks — perhaps rule-encoded explicitly or implicitly as a training set. In reality, systematic approaches will be behind threat detection mechanisms since they can quickly spot known attack techniques and Indicators of Compromise (IoCs). For example, if one knows that a specific malware installs a characteristic signature in system logs or a specific change of configuration is suggestive of an attacker attaining persistence, one can craft a rule or train a classifier to look for that same pattern on a thousand systems in real time. The strength of rule-based techniques lies in their accuracy and reliability when faced with established threats; however, the fact that they are knowledge-based can also turn against them for novel or evolving APT methodologies beyond the patterns they have been trained to detect.

Rule-based techniques are the most straightforward form of structured threat hunting. These are if-then rule-based systems defined by experts that issue an alert when specific conditions hold in the data. A classic example is the use of signature-based Intrusion Detection Systems (IDS) and intrusion prevention systems (IPS), such as Snort or Suricata, where a library of signatures (bytes patterns in network traffic or system calls sequences) is kept. When the incoming data matches a signature, an alert is issued. In the same way, on the host side, security tools can utilize YARA rules to search memory and files for distinct byte patterns characteristic of known malware employed by APT actors. In a threat hunting scenario, a rule could be written to search through logs for evidence of a given technique, i.e., one could look for any PowerShell command line that contains Base64-encoded data (since attackers will occasionally use encoded commands as a method of hiding their intentions), or write a rule to alert on any process creation of 'cmd.exe' by an application that's normally a Microsoft Office application (which is unusual and most typically linked with macro malware infection chains). Correlation rules in Security Information and Event Management (SIEM) solu-

tions are another example of rules logic: these correlations search for series of events which, together, indicate an attack, e.g., series of creating user accounts followed by administrative permission granted on the account and subsequent successful logon with it onto a key server. Each incident on its own may be harmless, but collectively they represent a recognized pattern of an attacker establishing a new backdoor admin user.

The advantage of rule-based hunting is determinism and interpretability. When a rule fires, an analyst can directly view why: the conditions are defined clearly, so the cause is evident (e.g., "Alert fired because an outbound connection to a watchlisted IP address was seen from a database server") [9]. This one-to-one mapping of detection to explanation is useful for forensic purposes and for compliance when analysts need to explain why they alerted on an incident. Additionally, rule-based detection contains few false positives if and only if rules are designed well enough to recognize strongly discriminant malware patterns (e.g., a rule on a known malicious file hash has virtually zero likelihood of mis-classifying benign activity). The failure of such a purely rules-based approach is monumental in the APT context. Rules are brittle and reactive in character: they need to be coded following the discovery of a technique, so new tactics used by a sophisticated attacker will not be recognized until the attackers become aware of them and modify their rules. APT attackers have time and freshness on their side — they can tailor their tools to the target environment, maybe tweak a malware sample by just a little so its signature no longer shows up in familiar patterns. If defenders depend only on static rules, they might be able to ignore these tweaked attacks. Also, it becomes cumbersome with so big a set of rules; there can be too many rules affecting performance and making it snowflake under the torrent of warnings, such as the additional false positives if any trigger benign behavior under edge conditions. So, rule-based methods although invaluable for identifying known threats and therefore part of a valuable structured hunting, need to be supplemented by more evolutionary AI methods if the unknown needs to be caught.

Supervised machine learning methods push structured threat hunting a step further in the form that they learn decision rules automatically from data. Rather than asking security professionals to hand-code each relevant pattern manually, supervised learning algorithms learn against labeled datasets with samples of bad (e.g., past APT intrusion behavior, simulated attacks, or known malware traffic) and good (normal system and user activity) behavior. The output of training is a model that is able to generalize from those examples to classify new behavior or events as likely malicious or benign. Essentially, the model combines lots of detection rules from the training set — perhaps revealing sophisticated combinations of features that are correlated with maliciousness, which a human would not have picked up on. For example, a model trained under supervision may learn to recognize that one rare sequence of API calls, invoked by an office printer process, is a legitimate indicator the device is compromised even though none of the API calls are suspicious in isolation. In deployment, the model can then mark that printer for inspection if it displays the learned sequence, acting as a self-motivated sentinel for the learned attack pattern.

Supervised learning methods commonly used in intrusion detection and threat hunting are decision tree-based methods, support vector machines, and more recently neural networks (deep learning models) [10]. One can train a single tree or set of trees (e.g., random forest or gradient boosting machines) to receive features of an event (e.g., login attempt features could be time of day, source IP reputation, data accessed, etc.) and produce a score or label showing whether the event is normal or part of an attack. These models divide the feature space according to training data, really creating a set of hierarchical rules, but rules based on statistical relationships in data rather than on expert opinion. Support vector machines might be trained on network traffic measurements to define a boundary in a high-dimensional space between attack traffic and good traffic. Deep learning models, including multilayer neural networks, have been applied to learn features automatically from raw data such as logs or packet bytes and even outperform human-crafted feature selection for complicated patterns in certain cases. For instance, a neural network autoencoder can be trained on patterns of good user behavior sequences and deployed as a classifier based on reconstruction error: if the new action sequence is not reconstructed well by the autoencoder (i.e., it's not close to what was observed while training), it may be marked as likely to be malicious. Although in theory the autoencoder is an unsupervised model to train, employing it as such for one-class classification obfuscates the difference between supervised detection in that we learn "normal" as the only training class and everything else as suspicious.

To illustrate a supervised learning detection mathematically, consider a simple binary classification scenario for identifying malicious events. We can define a feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ capturing properties of an event or sequence of events (such as counts of certain log keywords, durations, or statistical measures of network flows), and a label $y \in \{0, 1\}$ where 1 indicates malicious (attack) and 0 indicates benign. A logistic regression model, one of the simplest classifiers, would compute a weighted sum of features and apply a sigmoid function to estimate the probability that $y = 1$ given $\mathbf{x}$:

$$P(y = 1 \mid \mathbf{x}) = \sigma(w_0 + w_1 x_1 + \cdots + w_n x_n),$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid activation. The weights $w_i$ are learned from the training data by maximizing the likelihood of the correct labels (or equivalently minimizing a classification error loss). Once trained, this model will output a probability score for any new observation; security practitioners can set a threshold on this score to decide if an event is flagged for investigation (for example, flag if $P(y = 1 \mid \mathbf{x}) > 0.9$). More complex models like neural networks replace the simple linear combination $w_0 + \sum_i w_i x_i$ with deeper nonlinear transformations, but the core idea remains: the model encodes a decision rule derived from examples of attacks and normal operations.

One of the advantages of supervised models in the APT scenario is that they can capture subtle indicators by linking a number of weak signals together. APT activity would come as collections of anomalies that would be too faint to trigger a rule on their own. For example, the employee's console might display a temporary increase in CPU activity (most likely the result of executing malware), a few extra outgoing connections to odd locations than usual, and access to a file share the user hasn't yet touched. Any one of these anomalies can be dismissed as benign variation, but a supervised model of anomaly detection (or a classifier) might recognize that the simultaneous occurrence of each of these events all in rapid succession was an indicator of compromise in the training data. It would therefore flag this combination with high risk score, but a rule-based system examining each condition in isolation would miss it. This capacity to detect sophisticated patterns is particularly important in detecting APTs in the initial phases of their kill chain, where the indicators are subtle and spread out over numerous dimensions of a system. [11]

Bayesian inference models present another formal learning approach, merging domain knowledge with probabilistic reasoning to deal with uncertainty for threat detection. In a Bayesian model, one constructs a model that represents probabilistic relationships between different security events and the presence of an attack. For instance, suppose we denote by $A$ the proposition that "an APT is active in the network," and let $E$ represent a collection of observed evidence (such as "unusual admin login", "outbound traffic spike", "antivirus alert on a server"). Bayesian methods allow us to update our belief in $A$ upon observing $E$ using Bayes' theorem:

$$P(A \mid E) = \frac{P(E \mid A)\, P(A)}{P(E)}.$$

Here, $P(A)$ is the prior probability of an APT being active (reflecting our baseline belief or threat level), $P(E \mid A)$ is the likelihood of seeing the collected evidence if an APT is indeed active (which would be high if the evidence aligns with known APT behavior patterns), and $P(E)$ is the overall probability of that evidence under all scenarios. In reality, one may not do this using a formula but instead utilize a Bayesian network — a directed acyclic graph whose nodes are variables like certain observed events or system states, and whose edges denote conditional dependencies. For instance, a simplified insider threat detection Bayesian network would comprise a node named "user is compromised" which controls the probability of other nodes such as "user accesses sensitive files at abnormal time" or "user downloads new software." Each piece of evidence assists in incrementally upgrading the probability of the hidden node ("user is compromised"). By setting a threshold on the posterior probability $P$(compromised | evidence), an alert can be generated when evidence collectively suggests a high likelihood of malicious activity.

This Bayesian approach is well-suited for behavioral profiling and for combining multiple weak indicators in a principled way. Unlike a monolithic classifier, a Bayesian model can be built in a modular fashion, incorporating expert insight about conditional relations (for example, if an account shows logins from two countries within an hour, the probability of compromise increases significantly). Over time, as more evidence is gathered, the model's belief in the presence of an APT can either grow stronger or diminish, providing a form of cumulative reasoning. The use of Bayesian inference in threat hunting underscores the "structured" paradigm: we start with an initial model structure and prior probabilities that encapsulate what we believe about attacks and normal behavior, and then we update that model with data. This yields a transparent inference trail that analysts can examine (which evidence contributed most to raising the alarm?), aiding trust and interpretability in AI-driven threat hunting.

Structured methods, like rule-based systems and supervised (including probabilistic) models of learning, play an important role in the identification of APTs as they use patterns of known ill behavior. Structured methods excel at identifying very accurately instances of known risk and at giving sound explanation for their warnings. However, as the APT campaigns keep evolving and introduce new patterns, structured methods must be supplemented by less structured analytics [12]. We now move to unstructured methods, which seek to find the surprise and the new aspects of an attack that structured techniques cannot detect.

## 4. Unstructured Approaches: Unsupervised Learning and NLP

**Algorithm 3:** Unsupervised Anomaly Detection

**Input:** Raw data $D$, Threshold $\tau$
**Output:** Anomaly alerts
Preprocess $D \rightarrow \mathbf{X}$;
Cluster $\mathbf{X}$ (DBSCAN/k-means);
Train autoencoder $AE$ on $\mathbf{X}$;
Build authentication graph $\mathcal{G}$;
Compute node embeddings $\mathbf{E}$;
Fuse scores: $s(x) = \alpha\|x - AE(x)\| + \beta\|\mathbf{E}_v^{\Delta}\|$;
**for** $x \mid s(x) > \tau$ **do**
  Flag anomaly
**end**
;
**return** *Top-K anomalies*;

Unstructured threat hunting methodologies accept open-ended exploration and data-led inquiry, functioning without pre-existing initial hypotheses about how badness appears. Rather than looking for preconceived patterns, such methods scan through vast amounts of data for anything that "appears different" from anomalous or interesting. This philosophy has a parallel with unsupervised machine learning approaches, which aim to discover structure in data when labeled examples do not exist. For APT hunting, unstructured approaches are precious because advanced attackers typically use new or extremely customized techniques that do not match up with any recognized signatures or templates. Casting a broad net and allowing the data itself to infer what is most likely to be anomalous best provides defenders with the chance of catching the subtle hints of an APT's existence that structured methods may overlook. But this is accompanied by the challenge that not every anomaly is malicious—separating true threats from harmless anomalies becomes a highest priority, and often this means additional analysis or context correlation (which situational methods are able to provide).

**Algorithm 4:** NLP Threat Analysis

**Input:** Logs $L$, Emails $M$, Reports $T$
**Output:** IoCs, Sequence anomalies
Embed text with BERT/LLM;
Cluster logs → rare patterns;
Train LSTM on process chains;
**for** *low-probability sequences* **do**
  Flag
**end**
;
Extract IoCs from $T$ (NER/regex);
Analyze $M$ with phishing classifier;
**return** *Correlated threats*;

Most unstructured threat hunting tools rely on unsupervised learning [13]. Essentially, unsupervised learning has algorithms that can consume a dataset (e.g., a collection of system log messages, network flows, or user activity traces) and discover intrinsic patterns or clustering in the data. One prevalent unsupervised task is clustering: partitioning similar data points into groups based on some idea of similarity. In a hunting situation, an analyst can group user accounts according to their daily usage patterns, forming groups of users with similar behaviors. When a low-privilege user account bunches together with accounts usually belonging to system administrators (quite possibly as a result of anomalous spiking on high actions or sensitivity on accessed resources), there is likely takeover of the account and utilization in a system-admin role—a distinctive indicator of attacker lateral movement. Likewise, network connections can be grouped by normal hour-of-day usage or by amount and type of data transferred. An outlier cluster of data with a normally idle server that makes large outbound transfers to an unknown external host might be a sign of a possible data exfiltration by an APT.

In addition to clustering, anomaly detection algorithms specifically seek to find data points that are significantly different from the norm. These algorithms create a model of "normal" from previous experience and then validate new observations against it. In practice, the challenge is to define "normal" within a dynamic IT environment itself—normal is a moving target due to changing user behavior and systems. Nevertheless, numerous statistical and machine learning approaches have been successfully utilized. Basic statistical thresholds (for example, flag anyone who downloads more than 1 GB in a day if ever they downloaded no more than 100 MB before) are one simple kind of anomaly detection. More complex algorithms build up a multi-dimensional profile of typical behaviour. For instance, an algorithm can learn the normal range of values for hundreds of aspects of a database access event (time, type of query, amount of data, source of event, etc.) and classify an event outside of such learned range on several dimensions as anomalous. Distance-based methods such as $k$-Nearest Neighbors (kNN) based anomaly detection calculate the average distance from a point to its $k$ nearest neighbors in the past data; if this distance is more than a certain threshold, the point is

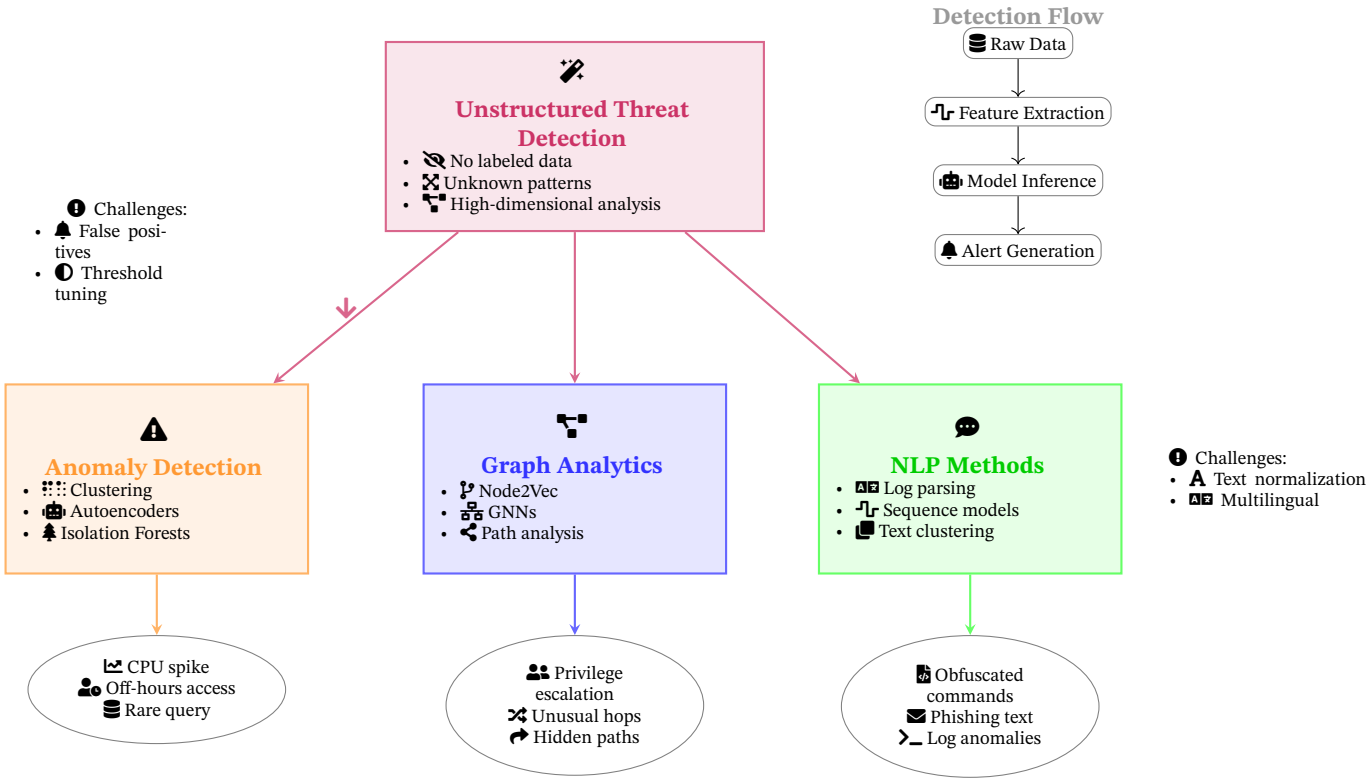


**Figure 5.** Unstructured Detection Approaches: Anomaly Detection, Graph Analytics, and NLP Techniques for Unknown Threats

classified as an outlier. Analogously, density-based methods such as DBSCAN can detect points that occur in the areas of low density in feature space and mark them as potential anomalies suggesting atypical behavior. [14]

Machine learning further offers even stronger anomaly detection capabilities like one-class classifiers and autoencoders. One-class Support Vector Machines, for example, try to learn a boundary encompassing most of the normal data points in the feature space such that any new point falling outside this boundary is classified as an anomaly. Isolation Forest is another method, which creates a forest of decision trees that split the data; outliers are the objects that are separated by the trees with fewer splits (i.e., it was easier to separate them from the other data, suggesting they are outliers). Autoencoders, neural networks that learn to compress and subsequently re-create data, have recently become popular for cybersecurity anomaly detection. By learning an autoencoder on a big dataset of assumed normal data (such as benign network traffic or system logs), the network learns to encode the data into a low-dimensional latent space and decode back. It gets optimized to reduce reconstruction error (the difference between original input and its reconstructed output). If the input is a novel input (e.g., a sequence of system calls from a running process) and has a high reconstruction error, this indicates that the input is not similar to patterns the autoencoder has learned—i.e., abnormal. Formally, if $x$ represents the input feature vector and $g(f(x))$ is the autoencoder's reconstruction of $x$ (with $f$ encoding and $g$ decoding), we measure an anomaly score as the reconstruction loss $L(x) = \|x - g(f(x))\|^2$. When $L(x)$ exceeds a chosen threshold, the event corresponding to $x$ is flagged for investigation. This technique can catch very subtle deviations across multiple correlated features, which is often the signature of an APT's handiwork: not a single blatant anomaly, but a constellation of slight abnormalities that collectively indicate something is wrong.

An illustrative example of unsupervised anomaly detection in APT hunting is user behavior analytics for insider threats. Without labeling any user as malicious beforehand, one can model each user's typical daily routine (logon times, accessed systems, data transfer amounts, etc.) using a mixture of statistical distributions. If an insider begins acting out of character—say, accessing a repository of trade secrets at 2 AM and copying large files—the anomaly detection system would compare these actions to the user's historical profile and flag them as irregular. The advantage here is that it can detect a potential insider-assisted APT exfiltration even if that specific scenario was never seen before, because it relies only on the assumption that individuals usually have consistent behavior patterns. In general, unsupervised methods shine in identifying "never seen before" incidents, which is precisely the realm where APTs operate during their most innovative attacks. [15]

Graph analytics also are a foundation for unstructured threat hunting, particularly for lateral movement and multi-stage attack campaigns that are characteristic of APTs. In most APT intrusions, the complete story of malicious activity only can be deduced by aggregating numerous distinct events on various systems — essentially building a graph of entities (users, machines, files, processes) and relationships among them (login activity, network connections, file accesses). Graphical representations enable hunters to visualize and algorithmically examine the relationships and paths an attacker may follow. For instance, one can construct an "authentication graph" where nodes are user accounts and computer systems, and an edge from user $u$ to host $h$ indicates that $u$ has logged into $h$ at some point. Normally, this graph might decompose into clear clusters (users accessing their department's systems, administrators accessing servers in their purview, etc.). If an attacker compromises a user account and uses it to move laterally, the graph might show a new pattern: the account node connecting to a cluster of machines it never accessed before. An unsupervised graph algorithm could detect this novelty — for example, by noticing that the edge $u \to h$ is anomalous given the historical connectivity of $u$ and the typical roles of $u$ and $h$. Community detection algorithms might flag that $u$ has jumped into a different community of hosts, or that $h$ now has an unexpected connection from outside its usual user community. Such findings strongly suggest lateral movement or privilege abuse.

For systematic graph anomaly detection, graph embedding and

graph neural network methods can be utilized. Analysts can use techniques such as Node2Vec or DeepWalk to perform random walks in the graph for learning vector representations (embeddings) of nodes such that similar network neighborhoods get similar vectors in the embedding space. When a node abruptly begins to interact with a hitherto unseen region of the network, its embedding will change or it will acquire new neighbors that change the similarity measures. It is possible to measure how "out-of-place" a node's activity is by comparing the current embedding of the node with its past embedding (or with peer nodes' embeddings that serve similar purposes). Analogously, one can insert full subgraphs or sequences of actions in order to identify unusual attack behavior. More sophisticated are Graph Neural Networks (GNNs), which may learn (in a semi-supervised or self-supervised manner based on anomalies as negatives) to predict an anomaly score for nodes or edges by propagating information from their neighbors across several layers [16]. For instance, a GNN can learn to assign a high anomaly score when otherwise isolated node suddenly has an edge to a very sensitive node it never had before, based on the properties of both nodes and the rest of the graph. These graph-based methods are especially effective against APTs, which have the habit of arriving as a stepping-stone series of compromise within a network; if the chain as a whole graph shape is recognized, then the attack is caught even if each step individually was evasive.

The other significant field of unstructured threat hunting is Natural Language Processing (NLP) analysis of unstructured data. The majority of the security-relevant information arrives in text or semi-structured format: system event log message logs, free-form audit records, email headers and body (within phishing analysis situations), or even threat intelligence feed reports that summarize adversary tactics. NLP approaches allow such text to be interpreted and analyzed in order to elicit signals within that may foretell malicious action. For example, look at messages in an operating system's event log—some of the events have encoded codes, but many events consist of a text description. NLP can be used to label these descriptions or group similar log events. When a service is crashed by an attacker with an out-of-the-ordinary error message, or generates an abnormal sequence of log messages (e.g., by crashing a service, or by engaging in an activity which does not happen very often), NLP-based clustering will isolate such log lines as indicating a new unique cluster and deserving of closer inspection. Similarly, language models can be used with streams of firewall deny logs or command-line input, reading them as sentences of unorthodox syntax or word pairs (such as an unusual set of command-line flags or an strangely named domain in a web request) to equate syntactic irregularities.

A new area of use for NLP in threat hunting is interpreting chains of events as language through the use of sequence modeling algorithms. This can be accomplished by annotating various kinds of events or activities (e.g., token "LOG1" might denote successful login activity, "PROC_START" a process start activity, etc.), and then inputting streams of these tokens into a model such as an LSTM (Long Short-Term Memory network) or Transformer to determine what normal sequences look like. The model essentially acquires a "grammar" of normal behavior. If an APT executes a new series of operations (for instance, a login followed by a lightning-fast privilege escalation exploit and an unexplained network connect), the sequence model would label it as one with low probability and mark it as abnormal. Such was proven through proof-of-concept research wherein the models with millions of log lines training data are able to catch injected unknown malicious sequences by the training data. It's akin to a language model marking an improperly spelled or nonsensical sentence in English; in this case, the "language" is the sequence of system activity, and the behavior of an APT will be a nonsensical sentence in the grammar of normal system activity. [17]

NLP is also employed when analyzing data from external to the system that can be useful in informing threat hunting. Threat intelligence reports, for instance, are authored in natural language, detailing new attack methods or malware indicators. NLP can be employed by AI systems to extract the aforementioned indicators (i.e., IP addresses, file hashes, or particular technique names) automatically and compare them with internal data, effectively bridging external unstructured information with internal telemetry. This gets closer to situation and intelligence-led hunting (also dealt with in the next section) but shows the way unstructured data such as text can be leveraged through AI to facilitate threat hunting. Another trend making its presence felt is the implementation of LLMs that aid hunters in parsing and making sense of large bodies of textual information (such as incident tickets, alert summaries, or system documents) to derive context within a few seconds.

In unstructured methods, the analyst will frequently have to interpret and triage unsupervised algorithm output. Not every anomaly is an attack; many are merely peculiar but harmless system quirks or user activity. Output from clustering or anomaly detectors therefore usually goes into a human analysis process or into additional correlation steps (perhaps with context, as the next section will discuss). The advantage of unstructured methods is to throw a broad net and to reveal those unexpected events no one knew should be openly pursued. When combating APTs, in which creativity and novelty are the tools of choice for the attacker, having AI that can create the unexpected is an asset. Through the use of unsupervised learning and NLP on the large logs and streams of data within an organization, the defenders will be able to construct a sort of radar for the unknown that will complement the rule-based "known bad" sensors. The second is to analyze and contextualize these signals with situational awareness and make raw anomalies actionable intelligence on real-time threats.

## 5. Situational Approaches: Context-Aware AI

Situational threat hunting techniques emphasize the integration of context and in-the-moment situational awareness into the detection process. Situational techniques are different from strictly structured techniques that look for known patterns, or unstructured techniques that identify anomalies as isolated occurrences, in that situational techniques consider events in the context of what is happening in and around the firm at the moment. Context can encompass a wide range of things: the level of criticality of the assets involved, users' role and normal behavior and how they affect events, active campaigns and campaigns that are known through threat intelligence, known system vulnerabilities, and even business context like current projects or events that may provide the reason behind observed unusual activity. The aim of context-aware AI in threat hunting is to be able to make wiser choices regarding what is suspicious activity by knowing the environment. For example, a failure in authentication at a database server would be typically an anomaly of low level, but if such a server holds financially sensitive data (high asset value), and if the attempt has been made from a machine that was never before tried to access it (low-probability relationship) during the morning time while there have been rising reports of database-targeted APT threats lately (outside context), then this event becomes much more significant. Context makes the plain observations richer so that the AI can rank and understand anomalies as not isolated occurrences but as aspects of a evolving scenario.

Creating a knowledge base that the AI can query in processing is one of the critical components of context-aware AI. This has generally been in the guise of a security knowledge graph or an aggregate data model that gathers user, device, application, vulnerability, network topology, and threat intelligence data. Nodes can be objects in such a graph (e.g., a host, a user account, an IP address, a file, a vulnerability ID) and edges are interactions or relationships (e.g., "host Y has vulnerability Z," "user X logged into host Y," or "IP W linked to malware command-and-control by threat intel"). If the anomaly is picked up by another means, context-aware reasoning may walk this graph to incorporate some relevant information into the anomaly. For example, if a process run is flagged as anomalous on a host, the

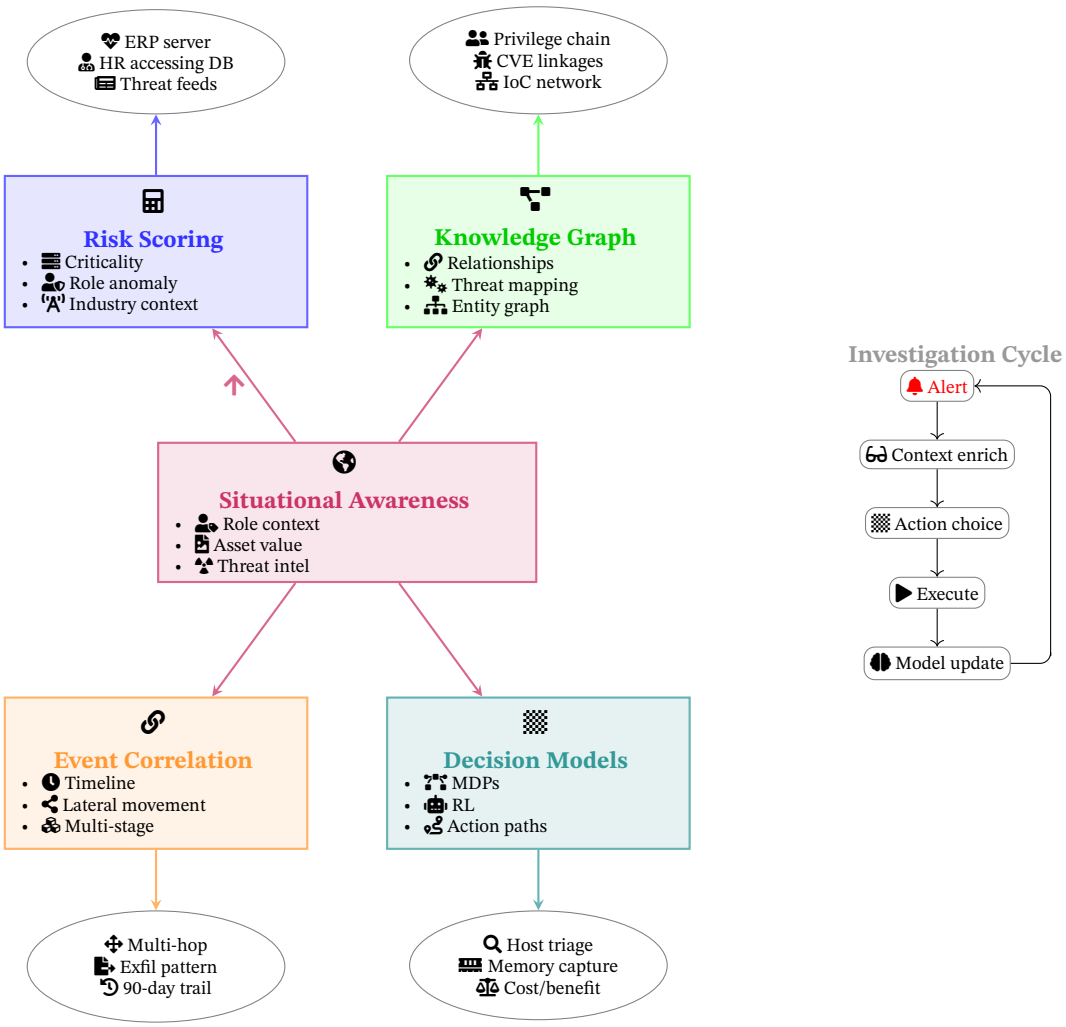


**Figure 6.** Situational Threat Hunting: Context-Aware AI Components and Investigation Lifecycle

knowledge graph might indicate that this host is two hops from an internet-facing server recently hacked and that it has an unpatched critical vulnerability which is widely employed by malware [19]. This situational context entirely flips the evaluation of the anomaly — what would have been a low-confidence indicator now is associated with a known activity and a known vulnerability, signifying active lateral movement. Such processing can be engineered into AI processes to do so automatically: asking the graph for each alert in order to collect context (e.g., "has the source IP of this occurred within other suspicious activity?" or "does this contain any high-value asset involved?"), and adjusting the risk score of the alert depending on the result.

In practice, context-aware threat hunting can be done by incorporating rule-based logic in combination with AI or by applying machine learning models accepting context features as inputs. One simple one is risk score systems: a machine learning model (a decision tree or logistic regression, say) would be trained on past incidents to estimate the chance that an alert is a true positive, not just the attributes of the alert itself, but also contextual attributes. Those are the asset's pertinence, whether the asset is known to have exploitable vulnerabilities, whether the user account involved has anomalous behavior in other directions, and whether similar alerts have been seen in nearby time windows. This kind of model is taught to understand that, for example, an admin account logging in from a foreign nation at 3 AM during there being higher threat intelligence alert levels is high likelihood of compromise, while a developer account logging in from the same foreign nation at the same time during a scheduled maintenance window (with a change ticket on the system, another bit of context) can be harmless. Through learning from situational information, the AI can minimize false positives and maximize confident true positives and, in essence, make its sensitivity situational requirement-dependent.

Beyond studying individual incidents independently, contextually aware AI can correlate several isolated events to define a grander scale malicious campaign. The majority of APT intrusions are merely a chain of minor temporal deviations and on systems that remain of interest only when grouped as a whole. A context system can link these pieces with commonality and temporal proximity. For example, if during a sixty-minute window a low-level malware alert shows on the workstation of an employee, the same account has a failed access on a server that he does not use, and then a file integrity monitoring system detects a new tool install on the server. Individualy, any of these would probably rate as no better than moderate risk. Linked together, the AI sees a pattern: workstation alert as first compromise, the unsuccessful login attempt as stolen creds being attempted against a server, and the new server tool as the attacker having gained persistence. The incident here—same user account affected, infrequent access pattern, and timing coincidence—reminds us that these incidents likely constitute part of a single attack chain. Through the combination of multi-source signals like this, context-aware AI inadvertently does incident correlation, fusing weak signals into strong indications of an ongoing APT [20]. Not only does this better detect (not missing the attack that would be undetected were all of the signals kept in silos), but also helps responders deliver an integrated incident context. The AI can then return, for example, a timeline or networked graph of proximate events, with analysts

**Algorithm 5:** Context-Aware Threat Hunting

**Input:** Raw alerts $A$, Knowledge graph $\mathcal{G}$, Threat intel $T$
**Output:** Prioritized incidents, Response recommendations
Construct dynamic knowledge graph:

$$\mathcal{G} \leftarrow (V, E) \text{ with } V = \{\text{assets, users, vulns}\}, E = \{\text{relation}\}$$

;
**foreach** *alert* $a \in A$ **do**
  Enrich $a$ with $\mathcal{G}$ context:

$$s_c(a) = f(\text{asset\_value}, \text{user\_role}, T_{\text{relevance}})$$

  ;
  **if** *a has linked entities in* $\mathcal{G}$ **then**
    $s(a) \leftarrow s(a) \times \text{graph\_connection\_weight}$;
  **end**
**end**
Correlate alerts via temporal-spatial patterns: [18]

$$C_i = \{a_j | \text{shared\_entities}(a_j) \cap \text{time\_proximity}(a_j)\}$$

;
Train context-aware risk model:

$$P(\text{malicious}) = \sigma(\mathbf{w}^T[\mathbf{x}_a; \mathbf{x}_c])$$

;
Form MDP for hunt guidance:

$$\langle S, A, P, R, \gamma \rangle \leftarrow \text{investigation\_state}, \text{transition\_model}$$

;
**while** ∃ *unresolved high-risk alerts* **do**
  Choose action $a^* = \arg\max_a Q(s, a)$ ; `// Optimal policy`
  Update $\mathcal{G}$ with new evidence;
  **if** *contains APT pattern* **then**
    Simulate response outcomes → recommend containment;
  **else**
    Mark as false positive;
  **end**
**end**
**return** *Prioritized incident chain*;

now being provided with situational awareness of the extent and the unfolding of the attack. Such automated correlation is invaluable in modern SOCs, wherein volume of alerts becomes insurmountable; context-aware algorithms eliminate noise and construct the clues that actually count.

Context-awareness also applies to the method in which the threat hunting process is executed. Situational awareness typically implies hunts are initiated or tuned on external events or changing knowledge while investigating. For example, if there is some recently revealed critical vulnerability (e.g., a zero-day in an email server) and there is rumor that some APT actor is already exploiting it out in the field, then a situational-aware system can alert the threat hunters (or trigger automated scans) to go search for signs of that exploit being present in the company's email logs and server telemetry. AI can help by translating the description of the vulnerability (via NLP) and figuring out what type of log patterns or system activity would indicate exploitation, then scanning for that very same pattern. In this manner, the search is not static but dynamically informed by today's threat landscape. A second example would be during the midst of an active incident: while hunters are revealing one compromised host, a context-aware AI can forecast which other systems are most likely to be vulnerable (perhaps by measuring the blast radius in a network graph or by recognizing attacker patterns), and recommend allocating hunting resources there. This real-time prioritization can be thought of as decision support driven by AI, where the AI is constantly examining the status of the investigation and the environment to suggest what to do next or where to concentrate, similar to a GPS directing a driver on an evolving path.

The concept of using AI to guide a sequence of investigative actions leads to the domain of sequential decision-making models, notably Markov Decision Processes (MDPs) and their extensions. An MDP provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision-maker. Formally, an MDP is defined by a tuple $(S, A, P, R, \gamma)$ where $S$ is a set of states describing the system, $A$ is a set of possible actions, $P(s'|s, a)$ gives the probability that action $a$ in state $s$ will lead to state $s'$, $R(s, a, s')$ is a reward function for making a transition from $s$ to $s'$ via action $a$, and $\gamma$ is a discount factor (with $0 \leq \gamma < 1$) that downweights future rewards. In a threat hunting context, one could conceptualize the state $s \in S$ as representing the current knowledge about the system's security (e.g., which hosts are suspected compromised, what evidence has been found, the status of investigations), and actions $a \in A$ as steps an analyst or AI agent could take (such as collecting more data from a host, analyzing a particular log, quarantining a device, or concluding the hunt). The transition probabilities $P$ would reflect the uncertain nature of investigations: choosing an action (e.g., running a scan on a host) will yield new evidence that updates the state of knowledge, but there is uncertainty in what that evidence will be (the scan might find malware or find nothing) [21]. The reward function $R$ can be designed to encode the goals of threat hunting: positive reward for confirming a threat or conclusively clearing a suspicion, negative reward (penalty) for false alarms, missed detections, or the time/effort consumed by certain actions.

Applying an MDP or its partially observable counterpart (POMDP, which takes into account the fact that the actual state — whether there is an APT infection on a host, for instance — can be unobservable) enables threat hunting to be formally defined as an optimization problem. The aim of the AI is to find an optimal policy

$$pi : S$$

$toA$ maximizing the cumulative expected reward, meaning to find best any possible attacks with minimal effort waste and disturbance. It is very hard to solve such an MDP analytically in a large enterprise environment due to the state and action space explosion. But there

is an ongoing research effort that has started exploring diminished or approximated models through which reinforcement learning algorithms can acquire effective hunting policies. For example, an AI agent will learn that whenever a specific sequence of low-level alarms and context is encountered (state), it is best to do a more detailed forensic analysis on a specific server (action) before searching other machines. With time, the agent should learn investigation techniques that lead to early capture of APTs (high reward for compromise detection early) and steer away from paths that had previously ended in dead ends (which are time penalized). While this is a very experimental field currently, it is an interface to context-aware AI: not so much enabling data analysis, but actually guiding the hunting process in real time according to circumstances as they arise, learning through experience how to best tackle advanced investigations.

Context-aware AI is also a critical consideration for incident response decision-making, often spilling over into the latter stages of threat hunting. As more proof emerges that an APT does exist, the question then becomes one of how to contain and destroy it. AI can aid in situational decision-making by representing or applying game-theoretic models to predict attacker response. For instance, when a business is choosing whether to take down a compromised server immediately (which immobilizes that branch of the attack but alerts the intruder), or keep it running with careful surveillance (to observe what the intruder is doing at the risk of ongoing data loss), an AI model could compare the result of both actions. By modeling the attacker and defender as players in a partially observable stochastic game, the AI is able to extrapolate probable attacker action for various defender action, effectively suggesting which action offers the best risk vs. intelligence gain trade-off. This is a very sophisticated example of context-aware analysis — weighing situational factors and probable attacker reaction to inform strategic action [22]. While venturing somewhat beyond strict "threat hunting" into response, it indicates the situational awareness continuum: the same context that facilitates detection can facilitate response.

In general terms, situational methods add a layer of contextual intelligence to threat hunting that allows AI systems to determine what anomalies are actually important in the bigger picture of an organization's security at a given time. Regardless of whether through knowledge graphs that consolidate heterogeneous data sources, context-aware risk models that contextually scale detection, or context-aware sequential decision models that adaptively guide the investigation process, context-aware AI is an extrapolation towards cybersecurity systems that react not just to data but are also adaptive to the environment and adversary that they defend against. This greatly enhances signal-to-noise ratio, drawing attention to where it will do the most good and shifting in response as conditions change — an ability that is required when facing adaptive, insidious APT opponents. The advantages of such advanced AI come with their limitations, though; these are described in the subsection below describing the limitations and hazards of using AI for threat hunting.

## 6. Challenges and Limitations

While AI-fueled threat hunting provides capabilities against APTs, it also presents various challenges and restrictions that need to be realized. Implementing AI in a security environment is not a matter of plug-and-play; instead, it adds a new set of complexities. Among the basic issues, data quality and availability play a significant part. Proper machine learning, particularly supervised learning, needs substantial amounts of normal and malicious behaviors' datasets to create models. APT intrusions, by nature, are unusual and subtle events — companies have limited (if any) verifiable APT incidents in their history to learn from. Such lack of ground truth samples creates a problem of training good generalizing supervised models. For most cases, security companies opt for fabricating synthetic data or consolidating information from various firms (accompanied by the complication of sharing and privacy in the data) in order to replace a dearth of suitable true attack examples. Unsupervised techniques do eschew labeled attack data requirements but consequently have to suffer with deciding what constitutes "normal" in an ongoingly shifting backdrop. User activity and system setup change over time, i.e., models constructed using logs from previous month may get outdated if the company experiences a sudden change in operations or employees. There exists a need for continuous learning and adaptation processes, adding extra maintenance overhead and model drift danger unless handled properly. [23]

The second significant challenge is so-called base rate problem, inherent to all detection systems but seriously acute in the cybersecurity field. The base rate of actual intrusions (the proportion of occurrences that are in fact malicious) is very low in most networks versus a gigantic volume of benign events. This imbalance implies even an ideal classifier produces massive false alarms. For instance, suppose only 0.1% of events in a network are malicious (a generous estimate in many enterprises) and an AI system can detect 99% of malicious events (true positive rate) while only mistaking 1% of benign events as malicious (false positive rate). Even with these impressive-sounding metrics, if the AI flags an event as suspicious, the probability that the event is truly an attack would be on the order of only a few percent. Quantitatively, one could apply Bayes' theorem:

$$P(\text{attack} \mid \text{alert}) = \frac{0.001 \times 0.99}{0.001 \times 0.99 + 0.999 \times 0.01} \approx 9.8\%.$$

That implies that more than 90% of the alarms would be false positives, even assuming the detector is extremely accurate, just because genuine attacks are that infrequent out of all the events. In reality, most AI-based systems are not calibrated as this example; false positive rates may be worse or true positive rates worse, and that would make alert accuracy even more biased. The result is that security teams are flooded with AI-driven alerts that need to be triaged. Analysts have limited attention and time, and if most of the alerts are benign anomalies or noise, it can result in alert fatigue, where actual threats are ignored or the team becomes desensitized to alerts. Therefore, to have AI systems be calibrated to optimize between false negatives (missed attacks) and false positives is a sensitive ratio. Frequently thresholds are increased in an attempt to curb noise, but this can come at the expense of the AI missing the exact attacks it was designed to identify. The base rate problem highlights a limitation: AI can help to filter information, but without other context or constraints (such as those supplied by situational awareness or by human inspection), it has difficulty with both high sensitivity and high precision in the naturally skewed environment of APT detection.

Data quality makes matters worse. APT attackers are adept at minimizing their footprints, and occasionally they can manage or disable logging on hacked systems [24]. If the AI data sources are unaware of certain attacker activities, then there is no amount of high-powered analysis that will be able to compensate for it. And then there is the issue of "dirty" training data: if historical logs used to train an anomaly detector inadvertently include undiscovered attack activity, the model will mistakenly include those bad patterns in its idea of normal. Essentially, the existence of an APT can pollute the baseline. This is an actual threat in unsupervised learning; e.g., an APT secretly exfiltrating small amounts of data every day for several months will probably have its activity marked as normal by a model trained on data from those months. Maintaining training data purity is either through meticulous retrospective labeling (difficult if the attacks were never found) or repeatedly retraining and validating models as fresh data regarding incidents emerges. Furthermore, enterprise data is noisy and heterogeneous—logs can have missing records, inconsistent structure, or can shift after software updates—so AI models struggle to stay robust without ongoing maintenance by engineers.

Adversaries are not passive observers of the game; they proactively react to and avoid defensive controls, even AI-based. Adversarial

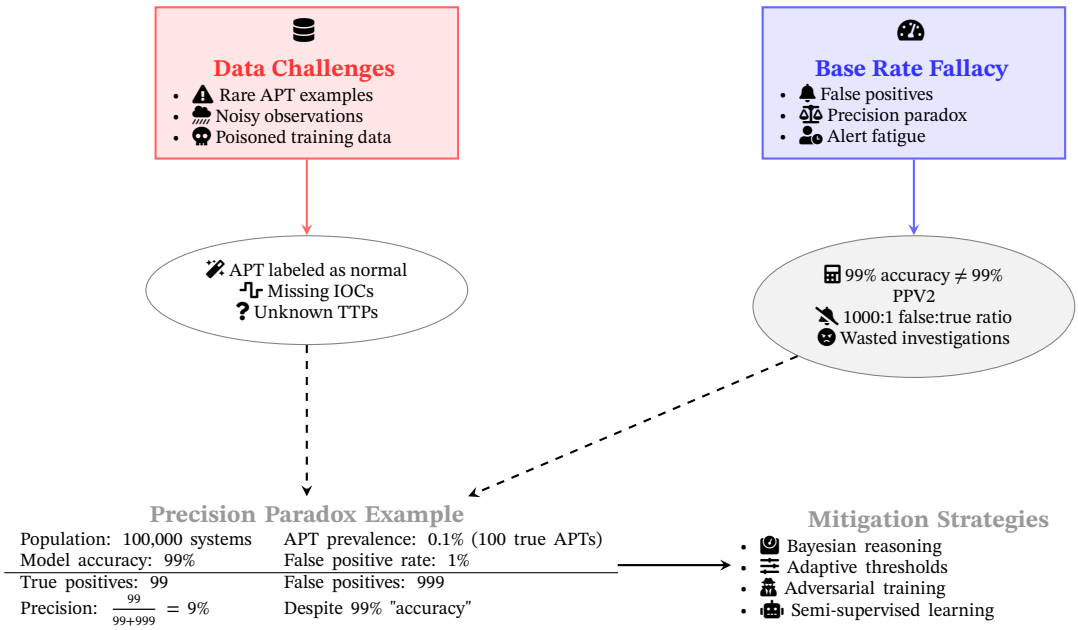


**Figure 7.** APT Detection Challenges: Data Scarcity and Base Rate Fallacy in Operational Environments


**Figure 8.** AI Cybersecurity Trilemma: Adversarial Threats, Interpretability Demands, and Operational Constraints

machine learning is one of the most powerful challenges that AI faces in the threat hunting aspect. Attackers can intentionally craft inputs to mislead machine learning. This has been shown to occur elsewhere (e.g., computer vision, where classifiers are fooled by un-

perceivable image perturbations), and the same principles extend to cybersecurity. E.g., an attacker may adjust their malware's features or their command-and-control traffic marginally in order to circumvent detection by an ML model, a technique not unlike when hackers alter malware in order to bypass signature-based antivirus. If they are aware of (or can guess) what the features an analyst will base a defensive model upon, they may attempt to remain just within what a model views as normal. A trivial example is an attacker that finds out that a behavior analytics system triggers based on five successive failed logons; they could script their tool to attempt only four before refraining, hoping to fly beneath the radar. More sophisticated examples involve creating network traffic patterns that resemble regular heartbeat traffic or combining malicious code with benign libraries such that file-based sensors see a familiar harmless pattern. AI-aware attackers might even use generative models to create "adversarial examples" of network traffic or log streams that are functionally malicious but look harmless to the AI [25]. A further attack technique is poisoning: if an attacker can inject false information into the training pipeline of an organization (for example, by generating large amounts of fake alarms that the system learns to disregard, or breaking into a sensor to generate spurious logs), they can lower the model's effectiveness or generate blind spots. Although such scenarios require advanced capabilities, APT teams are usually well-equipped with resources and research skills, and therefore the defenders need to assume that AI-powered defenses will be studied and exploited by persistent attackers. This makes deployment of AI an endless cat-and-mouse process, in which models have to be revised and tested for attacks repeatedly, and more robust methods such as adversarial training (training the model on instances of how it would be attacked) are added into the equation.

A second limitation is explainability of AI models. Security is an arena where knowing why an alert occurred is essential. Analysts have to trust that an AI's output is correct before they can take action, especially if that action could be disruptive (like quarantining a server or reporting an insider's misbehavior). Most effective ML models are black boxes, though: a deep neural network might flag a process as malicious 99% of the time, but can't explain it in a clear manner like a human-written rule can. That lack of transparency can hold things back. If the AI is unable to justify its reasoning, analysts may be hesitant to respond to its warnings, or, on the other hand, they may spend time attempting to decipher a warning that was merely a false positive because of some spurious correlation the model picked up. Transparency also makes it more difficult to debug the system — if it misses an APT, it is unclear whether it was because of a gap in training data, a model flaw, or something novel that it had not seen. There is active research in explainable AI (XAI) for security to facilitate model decision justification (e.g., where it identifies what features had the greatest contribution toward an anomaly score, or translating an alert to similar known malicious states). However, interpreting hard models in a manner that does not degrade accuracy is still a problem. As a reaction, some practitioners prefer simpler, easier-to-understand models (e.g., decision trees or rule-based) even though they are less precise, to make sure that it is possible for a human being to adhere to the rationale. It's always a trade-off between performance and interpretability when using AI in cybersecurity.

Working difficulties also put strong constraints [26]. Incorporating AI technology into the SOC process is no simple task. These tools can draw on dedicated infrastructure (for handling big data or executing AI models at pace) and skilled staff to set and maintain them. Not all organizations are able to enable a data science team in the SOC, and AI models will quickly degrade if not closely monitored. Moreover, resistance from analysts who feel threatened by job loss due to automation or are used to conventional tools can be expected. Time and good experience replace trust in AI suggestions, and initial failure taints a team's perception of the technology. Then there is the illusion of safety: deploying a system can make the senior leadership believe threat hunting is "solved," which can ensure less support for human talent. In fact, human analysts are still irreplaceable; AI can identify trends, but humans offer intuition, confirmation, and decision in uncertainty. Plugging in AI must be seen as augmenting the human hunter, not replacing it. That includes cultural training and adaptation to make sure analysts learn to work with the AI (acknowledging its output, aware of its limitations, and providing feedback to improve it).

Real-world constraints such as computational cost and latency are constraints as well. Powerful models (such as deep learning networks or sophisticated graph algorithms) can be costly to compute. Executing them in real time against streaming data can be prohibitively challenging without enormous investments in computing resources. This is why the majority of the detections by AI happen in batch mode or asynchronously, i.e., they may not detect an in-progress APT until many minutes or hours after the incidents have happened. During this period, the attacker could succeed in their goal. There is a compromise between speed and depth of analysis; it is difficult to achieve the correct balance [27]. Some companies respond to this by utilizing simpler real-time detection with more elaborate AI analysis offline but then the problem is to feedback results of the offline analysis into timely action.

Finally, there are ethics and privacy concerns. AI user behavior monitoring will breach employee privacy unless strictly controlled. For instance, an email content or keystroke monitoring AI for identifying insider threats is clearly privacy-sensitive in its need and may even breach legal or regulatory constraints. Even less sensitive information, such as log-in times or files accessed, if utilized in employee profiling, requires policy directives specifically articulating them. Surveillance and security are a thin line, and AI systems can quickly obfuscate that with the volume of data they can consume and correlate. A requirement to give transparency on what the AI is monitoring and having strict access controls on sensitive data is crucial in an effort to prevent abuse. Also, if an AI system does make a mistake (such as incorrectly labeling a vital system process as dangerous and disabling it), there can be major consequences, such as business continuity risks. Adding fail-safes and manual check steps into high-impact processes is therefore a significant business operational safeguard step.

Solutions to these are an ongoing and active area of research and development. Techniques like federated learning are being explored to facilitate collaborative model training on security data without the centralization of sensitive information (bypassing issues with data sharing). Adversarial training and stronger ML algorithms are being developed to make models more resistant to misleading. Man-machine hybrid workflows are being created in such a way that the most capable in both can complement each other — e.g., using AI to generate likely hypotheses and having humans confirm or follow up on them, with a feedback loop to iteratively improve the model. The above restrictions on AI place the hype over AI in threat hunting into context: APT protection is still a difficult problem, and AI is just one of several tools. Whether it works depends on prudent deployment, real-time analysis, and the understanding that technology is not a silver bullet against determined attackers. [28]

## 7. Emerging Directions

The use of AI for cyber threat hunting is set to change in various ways, fueled by both the constant innovation of attackers and developments in AI technologies themselves. One of the key trends is the quest for more robust and capable AI models. With one of the biggest challenges being evasion by adversaries, the future of research has centered on machine learning methods that continue to work even when adapting against attackers. This involves adversarially robust learning, in which models are trained on not only nominal instances but also on perturbed versions that mimic attacker tampering. Exposing models to prospective evasion attacks during training

**Table 1.** Emerging AI Techniques for Robust Cyber Threat Detection

| Technique | Objective | Key Features | Challenges | Example Use Case |
|---|---|---|---|---|
| Adversarially Robust Learning | Resist evasion attacks | Training with perturbed data | Model generalization | Training on spoofed logs |
| Neuro-Symbolic Systems | Combine learning with logic | Hybrid neural-rule systems | Rule-base scalability | Logic-guided anomaly detection |
| Online Learning | Continuous model updates | Real-time adaptability | Drift vs. deception balance | APT tactic adaptation |
| Contextual AI for OT/IoT | Domain-specific detection | Physics-integrated models | High signal variance | Power grid anomaly detection |
| Hierarchical AI Frameworks | Distributed threat analysis | Local-global fusion | System complexity | Endpoint-cloud coordination |

**Table 2.** Emerging in Explainable AI for Threat Hunting

| Method | Transparency Mechanism | Security Benefit | Implementation | AI Component Used |
|---|---|---|---|---|
| Attention Mechanisms | Input relevance weighting | Root cause indication | Neural layer introspection | Transformer encoders |
| Surrogate Models | Approximate black-boxes | Decision justification | Rule-based mapping | Decision trees, LIME |
| LLM Integration | Interactive explanations | Analyst assistance | Language model training on logs | Chat-based SOC assistant |
| Natural Language Justifications | Human-readable alerts | Analyst trust | Auto-generated rationales | Sequence-to-sequence models |
| Context-Aware Reporting | Situational explanations | Alarm triage | Threat intel fusion | Graph-based AI |

(a form of a digital "red team" training exercise) has the potential of the end detectors being built on less-spoofable features by the attacker. Neuro-symbolic approaches merging symbolic reasoning and neural networks are also of interest. For example, an emerging threat hunting system could employ a neural network to identify anomalous patterns, and subsequently apply rule-based verification (using known logical constraints or invariant system states) to filter out spurious results and validate real threats, Thus, an integration of data-driven and knowledge-driven AI can give rise to systems that are flexible and principled, capitalizing on the virtues of both structured and unstructured paradigms. We may also expect the emergence of online learning algorithms that dynamically update themselves as new data appear and therefore quickly adapt to newly emergent APT techniques. Such online or constant learning will have to be weighed against acumen (in order not to deceive the model with noise created by an attacker), but it provides a mechanism to stay ahead of the moving target that is attacker activity.

Another important future trend is enhancing explainability and transparency of AI threat hunting. Recognizing that the black-box nature of most models is one of the trust brakes, researchers have been working on explainable AI (XAI) methods specifically tailored to security. In coming years, security tools based on AI might become the norm to offer natural language explanations of the justification for their alarms, i.e., "This login was suspicious because it doesn't fit the user's 90-day behavior pattern and is accompanied by a known bad IP address." This can be achieved with attention mechanisms in neural networks that can identify what component of an input (which log records, which features) had the greatest impact on a decision, or by using surrogate models that map the advanced model with an explainable one for explanation. Also, the large language models themselves might be used in explainability; such a model might be trained on cybersecurity content to serve as an interactive analyst's tool that can summarize results from other models or respond to questions over why certain associations would represent danger. This is a crossover with the shift to use LLMs in security operations – i.e., to generalize events or recommend remediation actions – effectively driving AI not just into pattern recognition but into the space of reasoning and reporting. Next-generation SOC platforms should be able to deploy conversational AI agents that can process and interpret an onslaught of sensor inputs and give a unified account to humans, enabling the middleman translation from machine cognition to human comprehension. [29]

Human-AI fusion is bound to be more profound in the context of human-on-the-loop or human-in-the-loop, and not a substitution for human beings. There are a few ideas for how this could actually happen: One is that of a system that learns continuously with human judgment included in the loop. If an analyst investigates an AI alert and determines it was a false positive or a true positive, that can be fed back into the model (using something like reinforcement learning or active learning) to improve the next set of outputs. Thus, the AI is better attuned to the organization's specific environment and security team preferences. Another area is AI-powered incident triage and prioritization: as context awareness in AI systems evolves, they can potentially automate related alert clustering into incidents and determine a criticality level, steps that absorb analysts' time currently. One day, a researcher can begin by starting the day with an AI-briefing: a couple of priority incident abstracts abstracted from a thousands-strong feed of raw alerts supplemented with context sourced from threat intelligence. The AI could even present hypotheses on each incident (e.g., "these incidents show a multi-month credential-stalling campaign on finance department users") and provide suggested next steps to validate. Such alignment would allow human threat hunters to work off strategy and intuition with the thrust of AI-facilitated thorough data analysis.

**Table 3.** Human-AI Collaboration Models in Cyber Threat Hunting

| Model | Collaboration Mode | Feedback Mechanism | Use Case | AI Update Type |
|---|---|---|---|---|
| Reinforcement Learning Loop | Analyst-corrected learning | Reward-based signal | Alert labeling refinement | Policy update |
| Active Learning Loop | Human-in-the-loop sampling | Querying uncertain inputs | Ambiguous alert resolution | Training set update |
| AI-powered Triage | Priority classification | Context-based ranking | Incident prioritization | Heuristic refinement |
| Daily AI Briefings | Summarized incident feeds | Synthesized threat view | SOC daily starter | Natural language generation |
| Hypothesis Generation | Strategic recommendations | Scenario suggestion | Multi-step attack inference | Abductive reasoning |

**Table 4.** Federated and Industry-Specific AI Threat Hunting Approaches

| Approach | Data Handling Strategy | Security Focus | Collaboration Method | Example Sector |
|---|---|---|---|---|
| Federated Learning | Local training, shared models | Privacy-preserving sharing | Periodic model update exchange | Financial institutions |
| Secure Multi-party Computation | Encrypted joint computation | Confidential analytics | Distributed encrypted ops | Healthcare |
| Industry-specific AI Teams | Domain-tailored models | Sectoral threat patterns | Cross-org coordination | Critical infrastructure |
| Standardized Ontologies | Schema-based knowledge links | Interoperability | STIX/TAXII extensions | Threat intel sharing hubs |
| Anomaly Aggregation Models | Pattern diffusion across orgs | Preemptive warning | Multi-org training pipelines | Supply chain networks |

**Table 5.** Offensive Use of AI and Defensive Countermeasures

| Offensive AI Capability | Functionality | Defensive Counter-AI | Detection Mechanism | Risk Level |
|---|---|---|---|---|
| Automated Reconnaissance | Rapid network mapping | Behavior anomaly models | Lateral movement detection | High |
| Polymorphic Malware | Code mutation automation | Code pattern classifiers | Opcode entropy analysis | Critical |
| AI-Powered Phishing | Targeted content generation | LLM-based phishing detectors | Prompt structure markers | High |
| Deepfake Audio Attacks | Voice mimicry at scale | Audio provenance verification | ML signal trace analysis | High |
| Offensive AI Agents | Autonomous intrusions | Defensive AI swarms | Counter-agent modeling | Theoretical |

Boundary collaboration at the organizational level is a second potential path forward to optimizing AI-facilitated APT threat hunting. APT campaigns often focus on multiple organizations within a particular industry or supply chain, and a pattern that emerges in one network might be an omen for another. Raw data exchange among organizations, however, is plagued with confidentiality concerns. Technological developments in privacy-supporting machine learning, like federated learning and secure multi-party computation, promise to bring about a future in which the model is trained in a federated model: data inputs of each company go into a shared threat model without sensitive logs ever leaving their facilities. For instance, a federated learning environment might learn a global anomaly detection model that can see what APT activity is like in dozens of organizations and push each collaborator periodic model updates. Every organization gets to have a stronger model than it could have learned on its tiny attack data set, and every company gets to contribute a new attack pattern it discovers to the others if one of them discovers a new one [30]. We can also observe industry-specific AI threat hunting teams, in which information from critical infrastructure, financial, or healthcare sectors is utilized to train industry-specific models that are designed according to the specific APT patterns employed against the respective industries. Besides these initiatives, standardized ontologies and schemas of data (which can be developed atop existing platforms such as STIX/TAXII for threat intelligence exchange) will come into the picture, such that AI will be able to link isolated sets of data and collective knowledge in a seamless manner.

With evolving technological ecosystems, AI-based threat hunting will shift into new grounds that are in the focus of APT actors. The emergence of Internet of Things (IoT) and Operational Technology (OT) networks (i.e., industrial control systems) implies that future APTs will increasingly target such environments more frequently. AI methods will be created to manage the distinct data and threat models in these environments. For example, the identification of anomalies in power grid sensor values or factory control signals may deal with

quite distinct data (time-series sensor values) and would call for fusing cyber analytics with process physics models. Contextual AI in such a case could involve the physics of the process to prevent false alarms (being able to distinguish between an adversary-induced change and a normal variation in sensor readings). Likewise, when more resources get shifted into hybrid and cloud platforms, AI platforms will be compelled to scan across on-prem logs, cloud service telemetry, and even container orchestrations platforms. That in turn will compel innovation in AI that can run in distributed frameworks, perhaps using edge computing for local anomaly detection on endpoints and a central processor to correlate – essentially hierarchical AI.

On the defense side, the arms race is going to certainly involve APT groups using AI for their own ends, and threat hunting in the future will have to prepare for that. Attackers could use AI to automate discovery (rapidly mapping out a network and discovering vulnerabilities), create polymorphic malware constantly changing form, or drive spear-phishing campaigns with icy-cold tailored content at scale (using deepfake tech and smart language models). As a result, AI will be used by defenders to detect AI: for instance, creating models that are able to signal AI-generated phishing emails or deepfake voices used in social engineering phone calls, by detecting faint imprints of generative models. Such back-and-forth will give rise to an endless cycle of ingenuity. We can even imagine a situation where defensive AI agents and offensive AI agents are competing with each other temporarily within a network – the one struggling to ensnare and mislead the opponent, the other attempting to fly under the radar – borrowing themselves briefly aspects of autonomous cyber warfare for organizational networks. Although this is a fictional scenario, it highlights the need for research and AI adaptability to maintain a pace with cybersecurity; overconfidence would enable attackers to take the lead. [31]

## 8. Conclusion

The battle against Advanced Persistent Threats is generally perceived as an asymmetrical war, wherein defenders are expected to keep an eye on all the corners while attackers anticipate one fraction of a second of weakness to seize upon. AI has been here a key force multiplier for cyber threat hunting squads, arming them with solutions to scan large datasets and listen for the whispered signals of specially crafted intrusions. This paper examined how AI methods can be utilized via structured, unstructured, and situational avenues to improve APT detection. Structured methods introduce structure and wisdom into the equation, using rule-based decisioning and supervised learning to identify known malicious behavior with velocity and accuracy. Unstructured methods cast a broader net, employing unsupervised learning and NLP to reveal anomalies and relationships hidden in the data that may otherwise go undetected by biased judgment. Situational methods bring it all together by putting context into detection so that event interpretation is realized in terms of the environment and threat conditions of the time. Each of the classes addresses one of the problem's essential requirements: the known, unknown, and interpreted context. They collectively provide an overlapping defense in depth, and AI serves to play varied roles—ranging from a pattern-matching watchdog to an investigating sleuth to an examining counselor.

One of the findings in this study is that nothing can be properly done by a single technique or method. APTs are too polymorphic and dynamic; they will evade sheer signature-based protection, become lost in noise that frustrates anomaly sensors, and target loopholes in context-insensitive analysis. Systematic, non-systematic, and situational methods must be blended together, then [32]. AI-powered threat hunting is at its best when it reflects the multiform tactics of the enemy with a diversified defense. For instance, supervised models might be employed to actively seek out known bad behavior (snagging the low-hanging fruit of not-so-sophisticated intrusions), concurrent anomaly detection employed to monitor for suspicious system or user activity (catching precursory warning signs of an impending APT plan), and provide these systems with context (such that, e.g., an anomaly on a main server is automatically escalated in real-time, whereas the same anomaly on a pith-box terminal would be watched some more). This hybrid ensures that one method's weakness is counterbalanced by another's strength. Also, AI is not working in a vacuum; its success is complemented by the engagement of human analysts that steer it with hypotheses, test its conclusions, and infuse the intuition and strategic thinking algorithms lack. Dream-state is threat hunting maturity when AI and human combine systemically: the AI systematically searches within data and produces actionable leads, and the human exercises judgment and acumen over the leads and guides the investigation through to end.

In spite of improvements to date, the use of AI to hunt APT continues to be an iterative process as opposed to solved art. Challenges of data shortages, spurious positives, evasion attacks, and explainability require organizations to craft and iteratively refine their AI implementations with caution. Yet the future is a promising one. Trends in research and case studies suggest AI models identifying hitherto unknown attack patterns, and doing so sooner than was previously possible. They also propose AI highlighting connections between seemingly unrelated alarms, so responders can stitch together attack stories that would have required orders of magnitude longer to achieve by hand. The directions described indicate even more powerful and sophisticated AI tools coming down the pipeline—tools that will be stronger, clearer, and less noticeably integrated into security operations.

AI offers a way for defenders to counter all the more effectively. Through integration of knowledge of known threats in structured form, the power of discovery of unsupervised analytics, and the subtlety of contextual insight, organizations can fundamentally improve their capacity to detect and prevent even the most stealthy of attackers. Particularly, AI application in threat hunting is not a matter of replacing the human factor, but augmenting it—removing grunt work, streamlining analysis with machine accuracy and scale, and allowing security teams to respond more quickly and confidently. An active, AI-driven threat hunting approach is an anchor of cyber defense in this day and age that flips stealth and persistence upside down by allowing no evil endeavor, however secretive, to escape its net [33].

## References

[1] X. Zhou, R. Jiang, M. Tian, H. Qu, and H. Zhang, *ACM TURC - Temperature-sensitive Fingerprinting on ECU Clock Offset for CAN Intrusion Detection and Source Identification*. ACM, May 22, 2020. DOI: 10.1145/3393527.3393543.

[2] R. L. Neupane, T. Neely, N. Chettri, *et al.*, "Icdcn - dolus: Cyber defense using pretense against ddos attacks in cloud platforms," in *Proceedings of the 19th International Conference on Distributed Computing and Networking*, ACM, Jan. 4, 2018, pp. 30–10. DOI: 10.1145/3154273.3154346.

[3] S. S. Rizvi, T. Flock, T. Flock, and I. Williams, *ICSSA - Anomaly Detection to Protect Networks from Advanced Persistent Threats Using Adaptive Resonance AI Concepts*. IEEE, 2020. DOI: 10.1109/icssa51305.2020.00018.

[4] P. Karuna, H. Purohit, R. Ganesan, and S. Jajodia, "Generating hard to comprehend fake documents for defensive cyber deception," *IEEE Intelligent Systems*, vol. 33, no. 5, pp. 16–25, Sep. 1, 2018. DOI: 10.1109/mis.2018.2877277.

[5] J. Chacon, S. McKeown, and R. Macfarlane, *Cyber Security - Towards Identifying Human Actions, Intent, and Severity of APT Attacks Applying Deception Techniques - An Experiment*. IEEE, 2020. DOI: 10.1109/cybersecurity49315.2020.9138859.

[6] A. Khalid, A. Zainal, M. A. Maarof, and F. A. Ghaleb, "Crc - advanced persistent threat detection: A survey," in *2021 3rd International Cyber Resilience Conference (CRC)*, IEEE, Jan. 29, 2021, pp. 1–6. DOI: 10.1109/crc50527.2021.9392626.

[7] J. Wu, "New approaches to cyber defense," in Springer International Publishing, Dec. 3, 2019, pp. 113–157. DOI: 10.1007/978-3-030-29844-9_4.

[8] M. Fujimoto, W. Matsuda, and T. Mitsunaga, "Detecting attacks leveraging vulnerabilities fixed in ms17-010 from event log," in *2019 IEEE Conference on Application, Information and Network Security (AINS)*, IEEE, 2019, pp. 42–47. DOI: 10.1109/ains47559.2019.8968703.

[9] C. Krasznay and B. P. Hámornik, "Analysis of cyberattack patterns by user behavior analytics," *Academic and Applied Research in Military and Public Management Science*, vol. 17, no. 3, pp. 101–113, Dec. 31, 2018. DOI: 10.32565/aarms.2018.3.7.

[10] A. Alexopoulos, "Entropy-based evaluation of dns activity for threat hunting," *Journal of Applied Mathematics & Bioinformatics*, pp. 25–31, May 10, 2021. DOI: 10.47260/jamb/1112.

[11] Z. Liang, X. Zhang, D. Jiang, S. Wuqiang, and H. Li, *ATCI - Simulation of Operating System and Hardware Stripping Based on Sandbox Technology*. Springer International Publishing, Aug. 13, 2020. DOI: 10.1007/978-3-030-53980-1_93.

[12] A. K. Desta, S. Ohira, I. Arai, and K. Fujikawa, *ITNAC - MLIDS: Handling Raw High-Dimensional CAN Bus Data Using Long Short-Term Memory Networks for Intrusion Detection in In-Vehicle Networks*. IEEE, Nov. 25, 2020. DOI: 10.1109/itnac50341.2020.9315024.

[13] W. Wang, X. Zhang, L. Dong, Y. Fan, X. Diao, and T. Xu, "Cispbmei - network attack detection based on domain attack behavior analysis," in *2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, IEEE, Oct. 17, 2020, pp. 962–965. DOI: 10.1109/cisp-bmei51763.2020.9263663.

[14] W. Matsuda, M. Fujimoto, and T. Mitsunaga, "Real-time detection system against malicious tools by monitoring dll on client computers," in *2019 IEEE Conference on Application, Information and Network Security (AINS)*, IEEE, 2019. DOI: 10.1109/ains47559.2019.8968697.

[15] A. M. Lajevardi and M. Amini, "Big knowledge-based semantic correlation for detecting slow and low-level advanced persistent threats," *Journal of Big Data*, vol. 8, no. 1, pp. 1–40, Nov. 27, 2021. DOI: 10.1186/s40537-021-00532-9.

[16] A. Croix, T. Debatty, and W. Mees, "Training a multi-criteria decision system and application to the detection of php webshells," in *2019 International Conference on Military Communications and Information Systems (ICMCIS)*, IEEE, 2019, pp. 1–8. DOI: 10.1109/icmcis.2019.8842705.

[17] H. Kwon, Y. Kim, K.-W. Park, H. Yoon, and D. Choi, "Multi-targeted adversarial example in evasion attack on deep neural network," *IEEE Access*, vol. 6, pp. 46 084–46 096, 2018. DOI: 10.1109/access.2018.2866197.

[18] L. Huang and Q. Zhu, "Adaptive strategic cyber defense for advanced persistent threats in critical infrastructure networks," *ACM SIGMETRICS Performance Evaluation Review*, vol. 46, no. 2, pp. 52–56, Jan. 17, 2019. DOI: 10.1145/3305218.3305239.

[19] Z. Chen, J. Liu, Y. Shen, *et al.*, "Machine learning-enabled iot security: Open issues and challenges under advanced persistent threats," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–37, Dec. 3, 2022. DOI: 10.1145/3530812.

[20] K. Csaba and H. B. Péter, *Analysis of cyberattack patterns by user behavior analytics*, Dec. 31, 2018.

[21] M. Tian, R. Jiang, C. Xing, H. Qu, Q. Lu, and X. Zhou, "Ipccc - exploiting temperature-varied ecu fingerprints for source identification in in-vehicle network intrusion detection," in *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*, IEEE, 2019, pp. 1–8. DOI: 10.1109/ipccc47392.2019.8958766.

[22] S. Gong, "Intrusion detection model based on security knowledge in online network courses," in *2022 International Symposium on Advances in Informatics, Electronics and Education (ISAIEE)*, IEEE, 2022, pp. 325–328. DOI: 10.1109/isaiee57420.2022.00073.

[23] M. Miguez, *A hybrid intelligent intrusion detection system for advanced persistent threats*, 2020.

[24] Z. Hussain, A. Akhunzada, J. Iqbal, I. Bibi, and A. Gani, "Secure iiot-enabled industry 4.0," *Sustainability*, vol. 13, no. 22, pp. 12 384–, Nov. 10, 2021. DOI: 10.3390/su132212384.

[25] L. Huang and Q. Zhu, "Dynamic bayesian games for adversarial and defensive cyber deception," in Springer International Publishing, Oct. 26, 2019, pp. 75–97. DOI: 10.1007/978-3-030-02110-8_5.

[26] C. D. Xuan, D. Duong, and H. X. Dau, "A multi-layer approach for advanced persistent threat detection using machine learning based on network traffic," *Journal of Intelligent & Fuzzy Systems*, vol. 40, no. 6, pp. 11 311–11 329, Jun. 21, 2021. DOI: 10.3233/jifs-202465.

[27] M. Tian, R. Jiang, H. Qu, Q. Lu, and X. Zhou, "Advanced temperature-varied ecu fingerprints for source identification and intrusion detection in controller area networks," *Security and Communication Networks*, vol. 2020, pp. 1–17, Oct. 30, 2020. DOI: 10.1155/2020/8834845.

[28] J. Bieniasz and K. Szczypiorski, "Steganography techniques for command and control (c2) channels," in CRC Press, Sep. 26, 2019, pp. 189–216. DOI: 10.1201/9780429329913-5.

[29] D. Sobrín-Hidalgo, A. C. Vega, Á. M. G. Higueras, F. J. R. Lera, and C. Fernández-Llamas, "Systematic mapping of detection techniques for advanced persistent threats," in Springer International Publishing, Aug. 28, 2020, pp. 426–435. DOI: 10.1007/978-3-030-57805-3_40.

[30] T. Halabi, O. A. Wahab, R. A. Mallah, and M. Zulkernine, "Protecting the internet of vehicles against advanced persistent threats: A bayesian stackelberg game," *IEEE Transactions on Reliability*, vol. 70, no. 3, pp. 970–985, 2021. DOI: 10.1109/tr.2020.3046688.

[31] K. Li, H. Mu, C. Yang, J. Zhang, N. Jin, and J. Xu, "Abnormal message logic & sequence detection for process layer network in digital substation," in *2021 IEEE Sustainable Power and Energy Conference (iSPEC)*, IEEE, Dec. 23, 2021, pp. 4261–4266. DOI: 10.1109/ispec53008.2021.9735648.

[32] S. Wang, Z. Wang, T. Zhou, *et al.*, *Threatrace: Detecting and tracing host-based threats in node level through provenance graph learning*, Jan. 1, 2021. DOI: 10.48550/arxiv.2111.04333.

[33] S. Vivek, *Quantifying urban road network vulnerability and resilience to attacks*, Jan. 1, 2020. DOI: 10.48550/arxiv.2010.09887.