Learning about the adversary

Azqa Nadeem (Delft University of Technology, azqa.nadeem@tudelft.nl), Shanchieh Jay Yang (Rochester Institute of Technology, Jay.Yang@rit.edu), Sicco Vewer (Delft University of Technology)

Abstract

The evolving nature of the tactics, techniques, and procedures used by cyber adversaries have made signature and template based methods of modeling adversary behavior almost infeasible. We are moving into an era of data-driven autonomous cyber defense agents that learn contextually meaningful adversary behaviors from observables. In this chapter, we explore what can be learnt about cyber adversaries from observable data, such as intrusion alerts, network traffic, and threat intelligence feeds. We describe the challenges of building autonomous cyber defense agents, such as learning from noisy observables with no ground truth, and the brittle nature of deep learning based agents that can be easily evaded by adversaries. We illustrate three state-of-the-art autonomous cyber defense agents that model adversary behavior from traffic induced observables without a priori expert knowledge or ground truth labels. We close with recommendations and directions for future work.

1. Introduction: "Know thy enemy"

Understanding the capabilities of an adversary is one of the first principles of warfare (McFate 2005). It allows to categorize adversaries based on their capabilities, and thus help with designing effective and targeted countermeasures. Attacker modeling aims to quantify the risks associated with an adversary. Specific abuse cases can be designed using these models and security guarantees can be provided. To this end, several threat assessment models have been proposed. For example, the Capability, Opportunity, Intent (COI) model, also referred to as the Capability, Opportunity, Motivation, Behavior (COM-B) model is one of the most widely used threat assessment models in psychology, business management, and military defense (Steinberg 2005, 2007; Michie et al. 2011). "Capability" is defined as an attacker's capacity to undertake the task at hand. "Opportunity" refers to the presence of an operating environment that enables the attacker to perform the task, and "Intent" refers to the brain processes that make the attacker act upon the task. A "behavior" is an act of performing the task itself, and is directly influenced by the attacker's capability, opportunity and intent (Michie et al. 2011). Risk can be measured as a product of the attacker's intent and capability.

Tactics, Techniques and Procedures (TTP) describe the abilities and behavior of a cyber adversary. TTPs are usually an expression of an attacker's training, and thus are extremely difficult to alter, once detected. TTPs relate to an adversary's capability of employing a strategy to obtain their objectives. These strategies are often visible in observable data generated by a targeted system, e.g., network traffic and intrusion alerts. A vulnerability in the target system presents as an opportunity for the adversary,

while the adversary's intent is often implicitly inferred through their actions via observables.

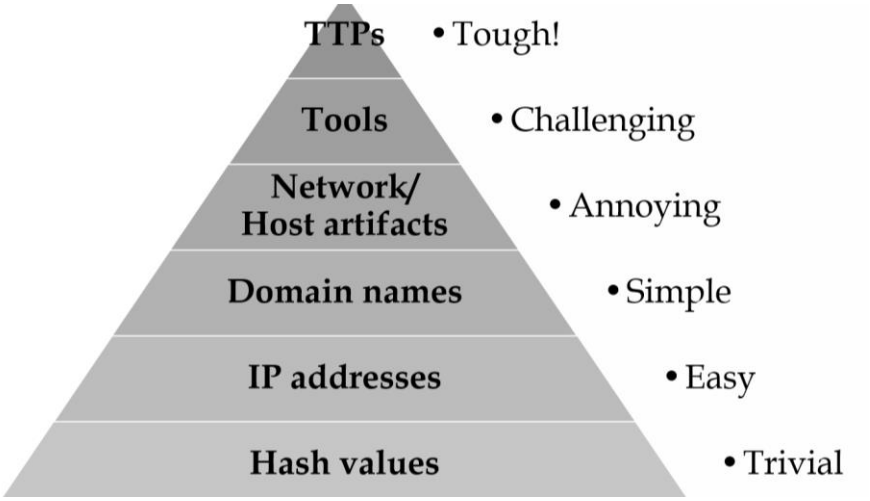

Figure 1: The Pyramid of Pain (Bianco 2013) shows the difficulty of obtaining various levels of Indicators of Compromise (IOCs).

1.1. Learning from observable data

Obtaining threat intelligence regarding TTPs from observables is extremely difficult, as indicated by the Pyramid of Pain (Bianco 2013). The Pyramid of Pain describes the difficulty of obtaining various kinds of Indicators of Compromise (IoCs), where the difficulty increases as one goes up the pyramid, see Figure 1. Lockheed Martin's Cyber Kill Chain (Hutchins et al. 2011) and MITRE's ATT&CK (Strom et al. 2018) are two of the most popular frameworks to study the structure of a cyber-attack in terms of tactics and techniques. The Cyber Kill Chain, shown in Figure 2, models the attack process as a sequential chain of seven steps that an attacker must complete in order to obtain their objective, and thus implementing countermeasures to break the chain may be a useful defense strategy. ATT&CK is a popular behavioral model for the TTPs used by cyber adversaries. Though extremely comprehensive, the attack types in ATT&CK cannot be easily linked to observable signatures. Recently, Moskal et al. (Moskal and Yang 2020) have developed an Action-Intent framework (AIF) based on the ATT&CK framework that links attacker intent with intrusion alert signatures.

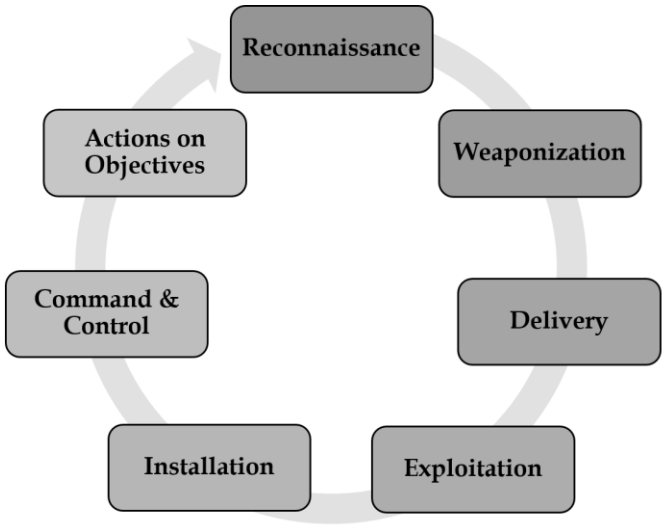

Figure 2: The Cyber Kill Chain categorizes a cyber-intrusion in seven phases, starting with scanning for reconnaissance and ending with actions on objectives.

There are two main approaches to building adversary behavioral models: expert-knowledge-driven and data-driven approaches. Expert-knowledge based approaches rely on human expertise curated over several decades' worth of experiences, which makes them largely manual and time-consuming in nature. Many existing techniques are expert-knowledge driven. Consequently, these models must be updated periodically to accurately reflect the evolving threat landscape. For example, malware detectors often use handcrafted template-based signatures, which are straightforward to evade, as indicated by several studies (Marpaung et al. 2012; Afianian et al. 2020). Attack graphs are another example of adversary behavioral models. Traditional attack graphs are based on Topological Vulnerability Assessment (TVA), which correlates extensive expert input and system vulnerabilities (Noel et al. 2009). As such, the process of constructing such attack graphs is labor-intensive, and it is ineffective to constantly rely on vulnerability scanning to accurately capture the threat landscape, since not all vulnerabilities are known in advance (Jha et al. 2002).

Data-driven approaches can be automated since they utilize observable artifacts collected from intrusion alerts, network traffic, software code, or shared threat intelligence feeds. In recent years, there has been an explosion of data-driven approaches for malware detection (Souri and Hosseini 2018; Nadeem et al. 2022a), malware analysis (Ucci et al. 2019; Piplai et al. 2020; Nadeem et al. 2021a), attacker strategy extraction (Alsaheel et al. 2021; Moskal and Yang 2021a; Nadeem et al. 2022b), and intrusion detection (Buczak and Guven 2016; Rimmer et al. 2022), among other tasks. Although quite promising, building effective and reliable data-driven agents is difficult: The challenges are related to the quality and availability of the observable data, the assumptions on the used models, and the interpretability and robustness of these models. For example, reconstructing adversary behavior from intrusion alerts would only be successful for actions that generated alerts: the actions for which the attackers

managed to evade detection cannot be observed in the data, and consequently cannot be modeled.

1.2. Definitions

In this chapter, we primarily focus on data-driven autonomous cyber defense agents because of their life-long ability to learn without too much human involvement.

For clarity, we define the following terms:

- Cyber adversary: *A cyber adversary is a single or group of human actors or automated agents that intend to perform malicious actions that harm other cyber resources. The actions can also have physical aspects, e.g., as in the case of social engineering attacks. The risk associated with a cyber adversary is related to their perceived capabilities and intent.*
- Adversary behavior: *A behavior is a learnt abstraction (model or pattern) from observable data that can be interpreted and transferred to other systems.*
- Adversary intent: *Intent is defined as the relationship between an adversary and their action, which lends insights into the motivations that lead to an attack. Intent is inferred through observed actions. A framework such as ATT&CK or Action-Intent framework aim to connect the intended attack stage with the corresponding tactics, techniques and procedures (TTPs).*
- Observables: *Autonomous cyber defense agents extract observables from data sources (via sensing) to learn about adversary behavior. These include, but are not limited to, software logs (network traffic, intrusion alerts, system logs), software code (malware binaries decompiled or otherwise), threat intelligence (feeds shared among organizations, collected through open source threat intelligence (OSINT)). "Features" are attributes derived from observables that model the adversary behavior. Note that obtaining real-world and usable observables is one of the biggest challenges in constructing autonomous cyber defense agents, as described later in the chapter.*

In addition, we define a data-driven autonomous intelligent cyber defense agent (D-AICA) as being an autonomous, data-driven software agent that learns contextually meaningful cyber adversary behaviors.

- Autonomous: *A white box machine learning based semi-supervised or unsupervised agent (model) that does not require frequent human intervention for (re-)learning. The white box model enables a human-in-the-loop setting where a security analyst can debug and understand the inner-workings of the agent.*
- Data-driven: *An agent (model) that learns from observable data artifacts.*
- Contextually meaningful: *An agent (model) that produces contextually meaningful output by correlating several temporally-linked observables from different modalities in order to provide a holistic view of the threat landscape, instead of viewing a single observable in vacuum. The output must also be multi-faceted and adjusted according to the operator's level of understanding.*

In the rest of the chapter, we describe the challenges of designing effective data-driven autonomous agents, followed by detailed illustrations of three state-of-the-art data-driven autonomous cyber defense agents. We specifically focus on agents that learn from traffic induced data. We close with recommendations and future research directions for R&D practitioners who are looking to get into developing and utilizing data-driven autonomous cyber defense agents.

2. Challenges for data-driven autonomous cyber agents

Data-driven autonomous agents utilize some form of machine learning to extract patterns from observables in order to learn cyber adversary behaviors, and to detect and analyze nefarious activities. Several challenges need to be addressed in order to design an effective agent. These challenges are related to the learning environment captured by the threat landscape, the availability of observable data, the assumptions that go into modeling cyber adversary behavior, and the open-world evaluation of said agent. We briefly describe these challenges below:

2.1. Evolving and adversarial threat landscape

In cybersecurity, there is a continual arms-race between attackers and defenders, which causes the threat landscape to evolve rapidly, making it near-impossible for autonomous agents to rely on supervised learning, or much of a priori expert knowledge. An autonomous agent is expected to perceive the changing adversary behaviors and relearn along with them. This can be done by detecting the changing data distributions (also known as Concept Drift). Change detection is commonly utilized in anomaly detection agents to keep up with the evolving systems. An anomaly detector typically models the normal state of a system in order to detect deviations from it. Over time, the system behavior may evolve, either due to system upgrades or new features, which may trigger the anomaly detector to raise false alarms for normal behavior that no longer fits its criteria of normality (Hammerschmidt et al. 2016; Jordaney et al. 2017). Hence, the agent must detect when the data distribution has changed sufficiently and relearn what the 'new' normal behavior looks like.

Cyber adversaries also actively try to evade detection by obscuring their activities. An autonomous agent must expect such evasion attempts and proactively defend against them. For instance, malware authors often evade detection by obfuscating malware (e.g., by encoding or encrypting it). Malware detector agents can fail if the attributes (features) used to model the malware are based on its appearance rather than its behavior. Behavioral features are shown to be more resilient to obfuscation attempts (Cai et al. 2019).

With the recent rise of adversarial machine learning for offensive security, a new breed of evasive attacks has surfaced that challenge the fundamental laws of machine learning. Deep learning models have shown to be particularly brittle to this kind of evasion attempts. Firstly, evasive malware samples can be crafted by either perturbing an existing malware sample to look like goodware, or by perturbing it to the extent that it no longer belongs to the training data distribution (a phenomenon known as "Out Of Distribution" sample detection). In a study, Kolosnjaji et al. were successful in fooling

several state of the art classifiers by altering less than 1% of the malware code (Kolosnjaji et al. 2018). Secondly, the data on which an agent learns can be poisoned so it no longer performs as expected (Chen et al. 2018). Thirdly, backdoors may be planted in a trained model so it becomes blind to adversary-chosen targets (Severi et al. 2021). These troubling results were followed by the proposal of several robust/hardened classifiers that include adversarial examples in the training process in order to identify and eliminate the so-called "blind spots" of the malicious domain. Despite recent advances, many classifiers are unable to provide robustness guarantees, making their use and deployment in the real-world tricky.

2.2. Data availability and quality

The concerns regarding the quality and availability of observable data to train autonomous agents is a known problem with multiple facets (Du et al. 2018; Nadeem et al. 2022a). Industry practitioners are hesitant, and are often contractually not allowed to share observables since they may contain sensitive information about their clients. It takes significant efforts and resources to make datasets publicly available. However, open source datasets quickly become obsolete due to the rapidly evolving threat landscape. Besides, open source data is often collected in isolated lab environments that do not accurately capture reality. For example, the well-known CTU-13 dataset (García et al. 2014) contains network traffic collected by running botnet-infected virtual machines, where the source IP and source port number are highly indicative of malicious hosts. Thus, using these features alone results in almost perfect classification. Such a situation almost never occurs in reality. Moreover, seamlessly incorporating real (benign) traffic into synthetic malicious traffic is non-trivial, since the differences in underlying network properties makes certain features unusable. For example, timestamps are generally unusable for such synthetically generated datasets, while in reality, these are among the most critical factors for threat intelligence and are indicative of attacker behavior.

Furthermore, open source datasets often have noisy ground truth labels. For example, the malware family names linked to open source malware datasets have repeatedly been shown to be noisy and unreliable. To partially resolve this issue, VirusTotal (VT) executes multiple Anti-Virus vendors and aggregates their results to determine whether a binary is indeed malicious. AVClass (Sebastián et al. 2016) was developed specifically to determine the true malware family label among the (many) VT labels. This unreliable nature of ground truth makes supervised learning very challenging, since the model is learning from faulty labels. Recent works have proposed learning from noisy labels as a potential countermeasure (Croft et al. 2022), though it is unclear how such methods fare when deployed in the real-world. Semi-supervised and unsupervised learning techniques appear to be more suitable paradigms for training autonomous cyber defense agents. For instance, Nadeem et al. have utilized unsupervised learning to construct network behavioral profiles of malware samples to characterize them, instead of relying on noisy family labels (Nadeem et al. 2021a).

Finally, open source datasets are often not at the desired granularity required for a specific task. For example, threat intelligence feeds are often too generic to be useful

(Sauerwein et al. 2017; Schaberreiter et al. 2019). In addition, the majority of the traces in network induced observables, such as intrusion alerts and network traffic, often reflect benign behavior, while those related to malicious activities are rare. Thus, it is often easier to perform anomaly detection on network traffic, rather than multi-class classification of the anomalies themselves. Learning with infrequent data remains an open problem in the machine learning community (Lu et al. 2020). A solution for obtaining good quality experimental data is to use honeypots. Honeypots are bogus, deceptive systems that lure adversaries into attacking it, and enable security practitioners to investigate how a particular threat actor operates. Honeypots have been used occasionally to collect granular observables (and threat intelligence) in order to understand an adversary's TTPs (Alata et al. 2006). Though, the design of realistic and robust honeypots is an open area of research (Surber and Zantua 2022).

2.3. Modeling adversary behavior

The Pyramid of Pain (Figure 1) shows the various levels of Indicators of Compromise (IOCs) that can be extracted from observables. As one moves up the pyramid, the IOCs get harder to extract. The IOCs regarding adversary behavior and strategy (the TTPs) are at the very top of the pyramid. Extracting insights regarding adversary strategies is difficult due to a multitude of reasons, one of them being the noisy nature of the observables, and another being the arbitrary nature of human actions.

When designing an autonomous agent, one must determine whether to model an attack or the attacker, since the former is relatively simpler to model. For example, a botnet detector that models the periodicity between network requests is more likely to succeed in its objective (Eslahi et al. 2015), compared to an autonomous agent that models the time it takes a human adversary to complete a task, which can be arbitrarily difficult. As such, autonomous threat actors, e.g., malware are much more deterministic than human actors. Thus, the observable features that characterize malware and human actors must be different. For example, in order to characterize a malware, observable features regarding its functionality are chosen, while to characterize a malware's author, features related to the code writing style are chosen, such as a function's name, or length of added comments (Nadeem et al. 2022a). Ultimately, it is important to borrow insights from clinical psychology and criminology to understand the role certain features play in order to effectively model adversary behavior.

2.4. Modeling context

The context (or semantics) of observables is crucial to accurately model adversary behavior. In order to comprehend an attacker's intent, it is paramount to view the observable data in the context of different perspectives. For example, one can assess a) temporal anomalies such as a host becoming excessively active during early morning or weekend hours, when less traffic is observed typically, b) irrational access attempts with mismatched port numbers, c) correlated increase in activity from multiple sources, and d) unexpected outbound traffic from specific internal hosts. An effective autonomous cyber defense agent is expected to learn such contextually meaningful adversary behaviors.

Additionally, investigating multiple data sources is important to establish confidence in an agent's judgment. For example, if an agent detects a data exfiltration attempt, only investigating an intrusion alert with data exfiltration signature does not provide sufficient evidence. Instead, the corresponding host's system logs should be cross correlated to see if a sensitive file was accessed and transmitted over the network. This is known as multimodal learning, i.e., learning from different data sources. Multimodal learning is a highly coveted property of autonomous agents, but presents a few challenges: 1) Data from different modalities often exist in different dimensionalities that must be brought to a common representation before learning, e.g., consider diverse data sources like threat intelligence feeds and network traffic. 2) Aligning and reasoning over semantically-linked events from different modalities is often not straightforward, e.g., a network packet that causes a denial of service attack may not have a corresponding entry in the system logs at the same timestamp since the system was unresponsive for the duration of the attack. Nevertheless, some of the challenges also present as opportunities: 1) Co-learning, or transferring what is learnt from one modality across different modalities is a promising avenue to handle the unlabeled nature of some modalities, e.g., Knowledge graphs can be used as a domain knowledge-rich modality to model adversary behavior from intrusion alerts. 2) Translating one modality to another could prove useful for creating synthetic anonymized observables that can be easily shared with the research community.

2.5. Interpretable approaches

Although repeated human intervention is not required to train an autonomous cyber defense agent, removing a human analyst entirely from the loop makes it difficult to understand what the agent is learning (Sejnowski 2020). This is more so the case for the recently proposed deep learning models and complex ensembles of machine learning pipelines that turn the whole agent into a black box. Metrics alone cannot adequately capture the performance of such a black box. Without a qualitative analysis, the metrics may give a false sense of how well an agent is actually working. For example, in a recent study, it was observed that a highly performant "Wolf vs. Husky" image classifier in fact did not learn the distinguishing features of the two animals, but rather looked at the image background to make its decision (Ribeiro et al. 2016). As it turned out, the training data contained all wolf images in snowy backgrounds, while husky images in non-snowy backgrounds. This is known as the "Clever Hans phenomenon" (Samhita and Gross 2013). Machine learning models can easily learn such biases if the training data contains them, and it becomes extremely difficult to debug them if they are black boxes (Rudin 2019). Aside from ethical and moral repercussions, these biases may be exploited by adversaries in their favor.

Recent studies show that there does not necessarily have to be a trade-off between explainability and performance, i.e., interpretable models can sometimes even achieve better performance (Letham et al. 2015). In fact, when a model is interpretable, it allows humans to learn from it, which ultimately also elevates human performance. The cybersecurity field has placed a renewed focus on designing interpretable and explainable autonomous agents in recent years. There are several approaches that attempt to explain the inner workings of a black box model, e.g., by learning a simpler

surrogate model (Szczepański et al. 2020), or by providing feature importance (Apruzzese et al. 2020), to name a few. While promising for verifying the correctness of black box models, the fidelity and trustworthiness of these explanations themselves can be subjected to attacks (Slack et al. 2020). Therefore, there is an increasing emphasis on interpretable by-design models for decision support, where human analysts are kept in the loop. These approaches enable the debugging and auditing of an agent to ensure that it learns exactly what it is intended to learn. For example, a recent study proposed a multi-step explanation system to make network intrusion detection systems more interpretable (Liu et al. 2021). Their system explains the model internals, its decisions, and also provides explanations based on the level of expertise of the operator. These types of autonomous agents are more likely to be deployed and used by security operators since they keep the human in the loop and allow them to intervene when necessary.

2.6. Open-world evaluation

The evaluation of an autonomous cyber defense agent must be designed with care. For instance, when dealing with datasets that have noisy ground truth, matching predicted labels with true labels is an unreliable and dangerous evaluation technique. Similarly, for unsupervised tasks such as clustering, the traditional definition of a true positive does not hold since data may be assigned to arbitrary clusters in different executions. Thus, a pair-wise co-occurrence method that looks at whether data items from one class are placed in the same cluster is a more suitable choice (Manning et al. 2010).

Furthermore, the choice of certain metrics may lead security practitioners to misleading conclusions (Jordaney et al. 2016). For example, an anomaly detector trained on a highly imbalanced dataset may not detect even a single anomaly, while still achieving impressive accuracy. This is why it is imperative that practitioners do not rely entirely on metrics, and attempt to understand the inner workings of the autonomous agent. One way to achieve this is by explaining a detected anomaly. Li et al. (Li et al. 2019) explain network anomalies using a local explanation method that ranks the most important features that led to a network flow being classified as an anomaly. These insights not only help security operators generate effective preventative policies, but also enable them to debug the anomaly detector, if necessary.

Finally, open-world studies investigating the generalizability of autonomous agents are imperative to get a glimpse of whether such an agent behaves as expected when deployed. Any number of reasons may cause the agent's performance to decline, including incorrect assumptions, mishandled edge cases, data quality, concept drift, and evasion attempts. Recognizing the unique challenges that emerge when machine learning meets cybersecurity is the first step towards the solution. Sethi et al. (Sethi and Kantardzic 2018) coin this crossroads between cybersecurity, machine learning, and streaming data mining as "Dynamic Adversarial Mining", which considers the combined problems of streaming data mining and adversarial learning in order to build generalizable autonomous cyber defense agents.

3. Approaches and Advancements

Below, we describe three cutting-edge cyber defense agents that model contextually meaningful adversary behavior with little to no ground truth labels, namely ASSERT (use case 1), SAGE (use case 2), and HeAT (use case 3). Although they are not fully autonomous, these approaches are certainly pushing the boundary of what can be learnt from observables. Note that these use cases specifically design agents that learn from traffic induced data, such as intrusion alerts.

In typical enterprise networks, intrusion detection and prevention systems (IDS) act as gatekeepers for adversaries, and raise alerts if any malicious activity is detected. Intuitively, intrusion alerts can provide insights into the attacker intent and it should be possible to reverse engineer attacker behavior from them. However, it is a challenging task for supervised learning, since intrusion alerts are rarely accompanied by ground truth labels. In this section, we describe three autonomous cyber-defense agents that extract insights about adversary behavior and attack campaigns from intrusion alerts.

3.1. Use case 1: Attack Model Synthesis (ASSERT)

In this example, Yang et al. (Okutan and Yang 2019; Yang et al. 2021) aim to monitor the evolving threat landscape by continuously synthesizing and updating emerging attack behavior models. To this end, they develop ASSERT – an unsupervised information theoretic learning framework that analyzes an endless stream of intrusion alerts to either dynamically generate a new model when an emerging attack is identified, or to update existing models if attack behaviors change. These models aggregate numerous related alerts to describe the "type of host" an adversary is targeting, "how" they are doing it, and the "intended outcome" they are trying to accomplish. Analysts can utilize their time more efficiently by focusing on critical attack models instead of the overwhelming streaming alerts. The overall component diagram of ASSERT is shown in Figure 3.

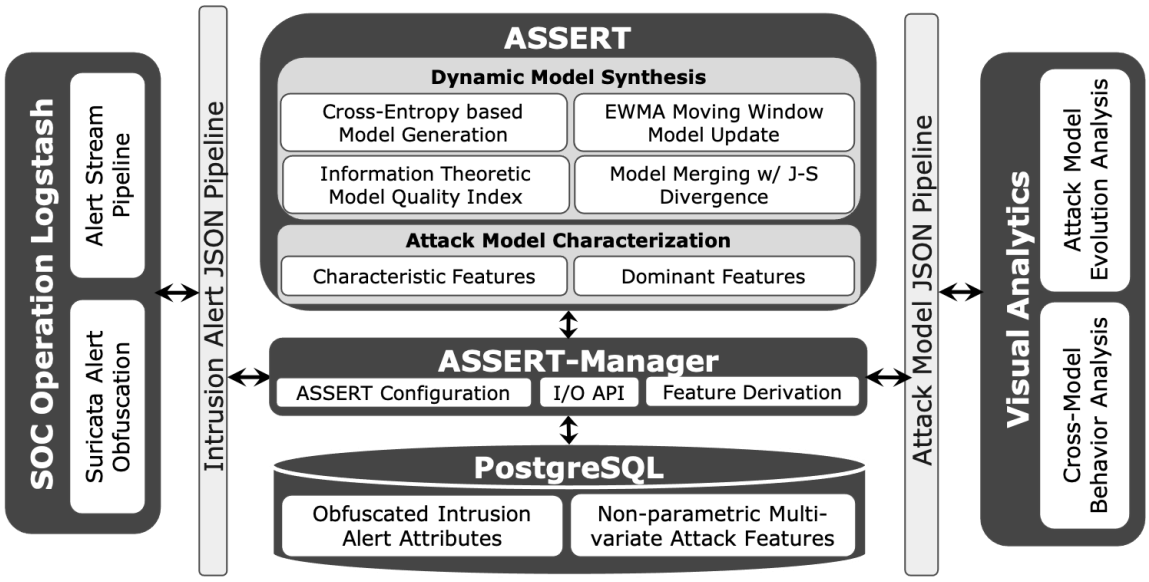

Figure 3: The system architecture of ASSERT. It takes intrusion alerts as input and produces visual representations of attack behavior models. (Adapted from Yang et al. 2021)

ASSERT is an information theoretic, unsupervised, continual learning system that consumes streaming alerts to synthesize statistical attack models in near real-time without requiring expert knowledge. It takes intrusion alerts generated by Suricata (https://suricata.io/) and produces attack models, with both the I/O pipelines in JSON format. ASSERT analyzes aggregated intrusion alerts as non-parameterized data distributions in order to identify and separate emerging attack models. There are three main components in the core engine of ASSERT: a) alert transformation into attack action aggregates, b) unsupervised information theoretic synthesis of attack models, and c) interpretation of attack models.

The first component aims to transform heterogeneous alert attributes into a set of contextually meaningful attack features. Specifically, ASSERT focuses on the following attack action dimensions:

- Attack Intent Stages (AIS) (Moskal and Yang 2020): a condensed version of MITRE's ATT&CK categories to imply the intended consequences of an observed attack action. A Pseudo Active Transfer Learning (PATRL) (Moskal and Yang 2021a) approach has been developed to automatically transform alert signatures into AIS.
- Targeted Services: Contextually, the targeted services, as implied by the port numbers, are one of the most indicative characteristics for an attacker's behavior. This is done through a regularly updated mapping of port numbers to known services or labels indicating reserved or other uses of TCP and UDP ports.
- Attack Maneuver: This is a mapping of IP addresses to categorical maneuvers, reflecting both, the direction of the observed action (i.e., inbound, outbound, internal) and the change(s) in source and destination IPs between consecutive alerts (e.g., the src IP of the new alert is the dst IP of the last alert, or the new alerts has the same src IP but a different dst IP) in the same alert stream.
- Attack Speed: This feature is derived based on the time elapsed between consecutive alerts in the same alert stream. The time is discretized in a logarithmic manner to reflect the significant variation (from nano-secs to mins or hours) in attack speed.
- Attack Source: This feature reflects the "region" and the "blacklistness" of the source of the attack based on the IP address(es). Note that the source can be the source or destination IP, depending on which one is external to the targeted network and the type of attack actions. For example, the external destination IP of a data exfiltration attack would be the attack source, instead of blindly treating all the IPs in the src-IP field as the attack source.

The second component is the main algorithm that enables the unsupervised attack model synthesis process. Figure 4 shows the high-level process, where X is an aggregate of attack actions transformed from intrusion alerts by processing small batches of alerts, and Q^* is the best attack model that matches the characteristics exhibited by X. The $H()$ function represents the cross-entropy between the two

distributions, P_X and P_Q, and serves as a proxy of Kullback-Leibler divergence (KLD) since H(P_X) is the same in the argmin process. The use of H_{max} threshold provides a computationally efficient and effective heuristic to determine whether the new aggregate sufficiently resembles the best-matched model or should be used to create a new model. The Model Quality Index (MQI) measures the overall quality of model separation in the joint attack feature space, by integrating Jensen-Shannon divergence (JSD) and the notion of Wemmert-Gancarski Index (WGI)[1].

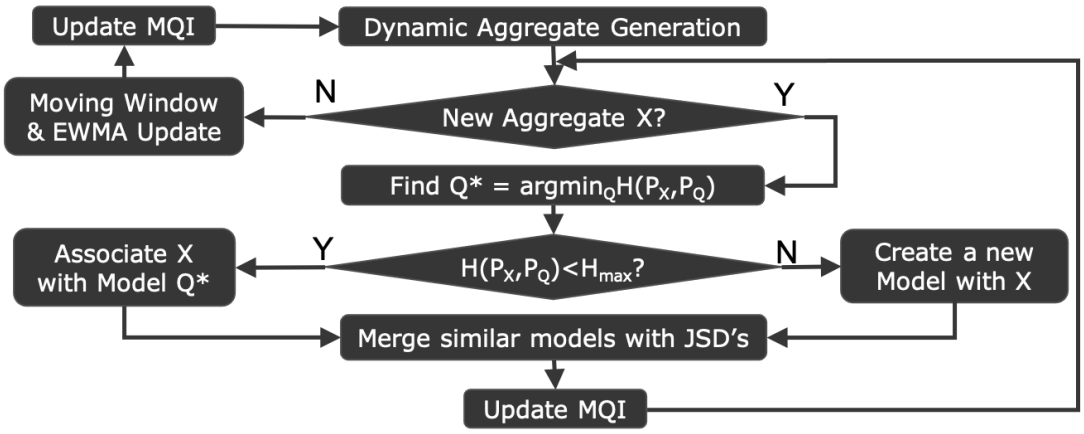

Figure 4: The unsupervised attack model synthesis process of ASSERT. (Adapted from Yang et al. 2021)

The final component of ASSERT aims to provide an interpretable set of characteristics for each attack model. Continuing the information theoretic framework, the characteristic feature for each attack action dimension of model Q is as follows:

$$x_Q^* = \arg max_{x|\bar{Q}\neq Q} \left(p_{Q(x)} \log p_{\bar{Q}(x)}\right)$$

where $p_{Q(x)}$ and $p_{\bar{Q}(x)}$ are the probabilities of the feature x in Q and $\bar{Q}$ (all other models that are not Q), respectively. Intuitively, this finds the feature that is prominent (not necessarily the most) in Q but very rare or non-existent in any other model. The characteristic features provide an intuitive way for the analysts to comprehend and differentiate the attack models.

The authors worked with a real-world SOC operation to process streaming Suricata alerts and to synthesize attack behavior models. Over a month of continuously running ASSERT, the system maintains approximately 20 to 25 attack models even with tens of millions of intrusion alerts. Figure 5 shows a cropped screenshot from ASSERT output. In this case, there were 21 unique attack models. One of the attack models drew the analyst's attention: It shows a potential critical persistent code execution attack through Kerboros authentication. This is a persistent attack because most of the observed actions are inbound traffic with no change on the source or target IPs. There are other malicious activities observed in this persistent attack, including privilege escalation and data exfiltration. Recognizing this critical attack model by using the autonomous ASSERT helps the analysts to focus on the relevant intrusion alerts and

[1] https://cran.r-project.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf

system logs, and determine effective remediation to treat the vulnerabilities, and to defend against the adversary.

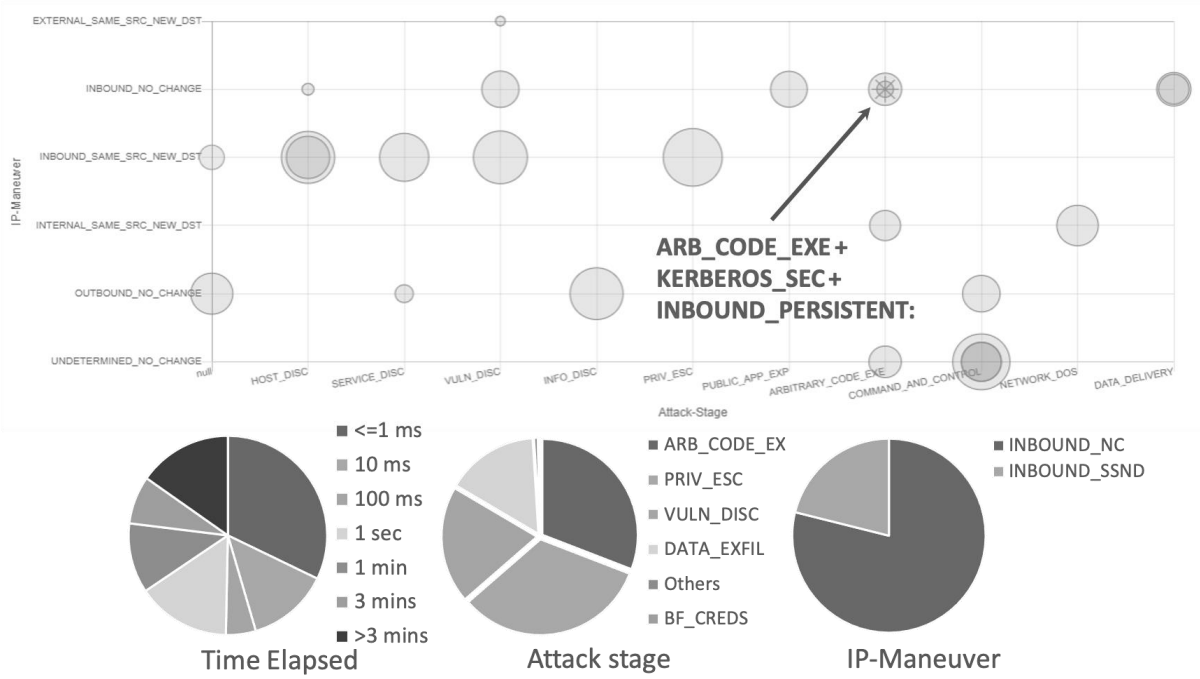

Figure 5: (Top) A screenshot of attack models produced by ASSERT with a persistent arbitrary code execution through Kerberos. (Bottom) The pie charts show the model's attack features in Attack Speed, Attack Intent Stage, and Attack Maneuver. (Adapted from Yang et al. 2021)

3.2. Use case 2: Attacker strategy extraction (SAGE)

In this example, Nadeem et al. (Nadeem et al. 2021b, 2022b) propose a data-driven attack graph approach to extract attacker strategies without a priori expert input. Attack graphs (AG) are well-known models of attacker strategies that assess pathways utilized by a cyber adversary to penetrate a network. Existing techniques to construct attack graphs are based on Topological Vulnerability Assessment (TVA), which requires significant expert input to correlate system vulnerabilities. However, it is expensive and ineffective to constantly rely on expert input and vulnerability scanning in real-world operations. Meanwhile, Security Operations Centers (SOC) often investigate millions of intrusion alerts on a daily basis. Alert correlation techniques help to reduce the overall load of intrusion alerts by aggregating alerts that originate from the same attacker action. While useful in its own right, alert correlation does not show attack progression and attacker strategies. Instead, in this study, the authors define a novel adversary behavioral model, i.e., an "Alert-driven Attack Graph", that learns attacker strategies directly from intrusion alerts without a priori expert input.

The authors develop SAGE – an interpretable sequence learning pipeline that constructs attack graphs from the actions observed through intrusion alerts, without a priori expert knowledge. SAGE utilizes an unsupervised statistical model, known as a

suffix-based probabilistic deterministic finite automaton (S-PDFA) to learn attacker strategies from intrusion alerts, and display them in the form of attack graphs. The authors discuss two application scenarios for alert-driven attack graphs: 1) SAGE is designed to augment existing intrusion detection systems for triaging critical attack scenarios that might require urgent attention. Thus, instead of investigating thousands of tabular alerts, SOC analysts can visualize attacker strategies, and investigate only a selection of intrusion alerts relevant to a critical attack path. They can use these AGs to understand how an attack transpired, and to extract threat intelligence about adversaries based on historically observed malicious activities. 2) Alert-driven attack graphs can monitor and rank red-teaming exercises. For instance, AGs can be reviewed after a training exercise to determine which team member(s) managed to find the shortest path to an objective, or to find redundant paths indicative of communication problems between the team members.

The overall component diagram of SAGE is given in Figure 6. SAGE consumes Suricata alerts in JSON format as input, and generates images of the resulting attack graphs as output. The steps for learning alert-driven attack graphs are given as follows:

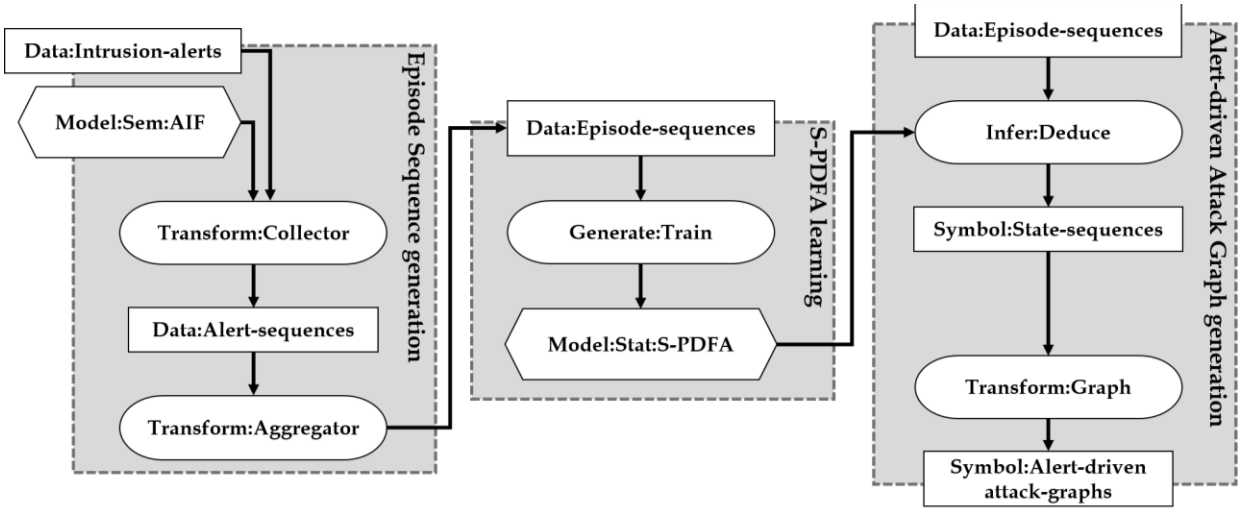

Figure 6: An overview of SAGE. It aggregates intrusion alerts into episode sequences, which are used by the S-PDFA to learn temporal and probabilistic relationships between them. Alert-driven attack graphs are extracted from the S-PDFA for each victim host and exploited objective. (Adapted from Nadeem et al. 2022b)

- Intrusion alerts are pre-processed and augmented with the Attack Intent Stages (AIS) from the Action-Intent Framework (Moskal and Yang 2020). Similar to the previous use case, the attack stages indicate the intended consequence of an attacker action.
- The alerts are aggregated into “Episodes” (Moskal et al. 2018) such that an episode characterizes an attacker action.
- The episodes are arranged in sequences for each attacker-victim IP pair.
- The episode sequences are partitioned for each attack attempt. The start of a new attack attempt is indicated by observing a low-severity episode followed by a high-severity episode. These episode subsequences form the training data for the S-PDFA.

- An S-PDFA is learnt using the FlexFringe automaton learning framework (Verwer and Hammerschmidt 2017). An example of the S-PDFA learnt from over a million alerts is given in Figure 7.
- The episode subsequences are replayed through the S-PDFA. This step augments the episodes with their respective contextual information (i.e., state identifiers).
- Finally, the augmented subsequences are transformed into attack graphs, where one attack graph is generated for every objective exploited on each of the victim host(s).

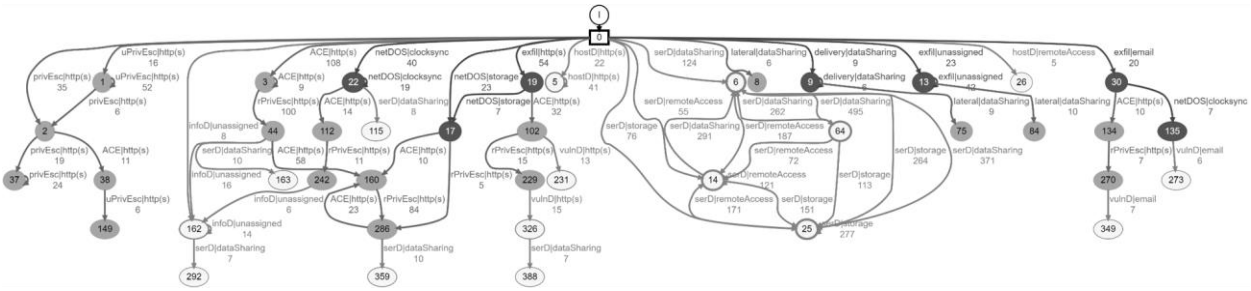

Figure 7: The S-PDFA learnt from over a million intrusion alerts collected through the Collegiate Cyber Defense Competition. The states (vertices) are colored according to severity: the darker the color, the more critical the attacker action is. The edges describe an episode as a combination of the attack stage and the targeted service. (Adapted from Nadeem et al. 2022b)

Nadeem et al. identify three design challenges for learning alert-driven attack graphs: 1) Alert-type imbalance: Severe or critical alerts are infrequent, while non-severe alerts (e.g., related to network scans) are common. Frequency-analysis methods discard infrequent events as noise, thus making most machine learning methods unsuitable for this application. 2) Modeling context: The same alert signature may be involved in different strategies. This is indicated by the neighboring alerts, which can be used to model an alert's context so as to distinguish between similar attacker strategies. 3) Interpretable model: SOC analysts are often contractually obligated to investigate all alerts, making black box models inherently unsuitable since it is often not possible to reverse engineer the alerts behind a classifier decision.

The S-PDFA is responsible for addressing the design challenges: 1) A suffix-based model is specifically chosen to highlight infrequent episodes, without discarding any low-severity episodes. Since severe episodes always appear at the end of the episode subsequences, a suffix-based model is a natural choice. 2) The state identifiers of the S-PDFA model capture an episode's context. Using the Alergia heuristic (Carrasco and Oncina 1994) for state merging, states having similar futures and pasts are merged, while those leading to significantly different outcomes are not. 3) The Markovian property of the S-PDFA, together with Sink states make the model components interpretable. Sinks are states that occur too infrequently to learn from. The authors remove low-severity sinks from the S-PDFA, making the model cleaner and easier to follow. Additionally, the Markovian property ensures that the input transition symbols of a state are unique, making it easier to interpret the meaning of a state. In this case,

the states represent milestones achieved by an attacker. Overall, the S-PDFA shows a bird's eye view of all the attacker strategies that can be observed in an alert dataset. The deterministic nature of the S-PDFA makes it algorithmically-transparent. The parameters of the model are selected through trial-and-error of visualizing the S-PDFA until it matches the authors' intuition about the data, making it design-transparent (Roscher et al. 2020).

A notional alert-driven attack graph is shown in Figure 8. The root node of an alert-driven attack graph shows the IP address of a victim host and the objective exploited on that host. The graph shows all the attempts made by the attackers to reach the objective. The vertices reflect actions (characterized by episodes) taken by the attackers to obtain the objective, while the edges are annotated with timestamps. All adversaries that achieve the objective are shown in the same graph to aid strategy comparison. The authors compare the complexity of the attack graphs (in terms of vertices and edges) against several baselines, and find that SAGE generates the most succinct graphs.

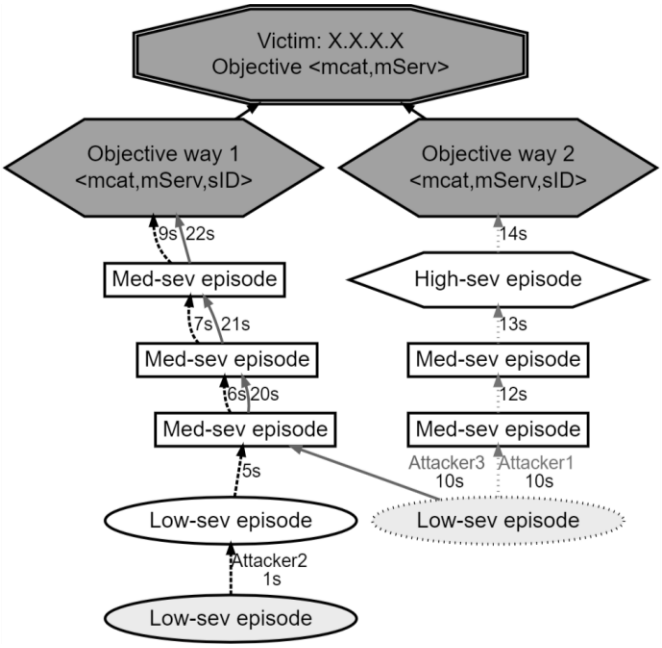

Figure 8: A notional alert-driven attack graph showing paths towards an objective. Each vertex represents an episode. The vertex labels are <mcat, mServ, sID>, reflecting the AIF attack stage, targeted service, and state identifier from S-PDFA. The vertex shape indicates the episode severity. The root nodes reflect the attacker's objective and the victim host. The dotted vertices reflect the sink states. The edges show attack progression and are annotated with the time since the first alert was captured. The attacker's IP address is stated next to the first action in an attack path. (Adapted from Nadeem et al. 2021c)

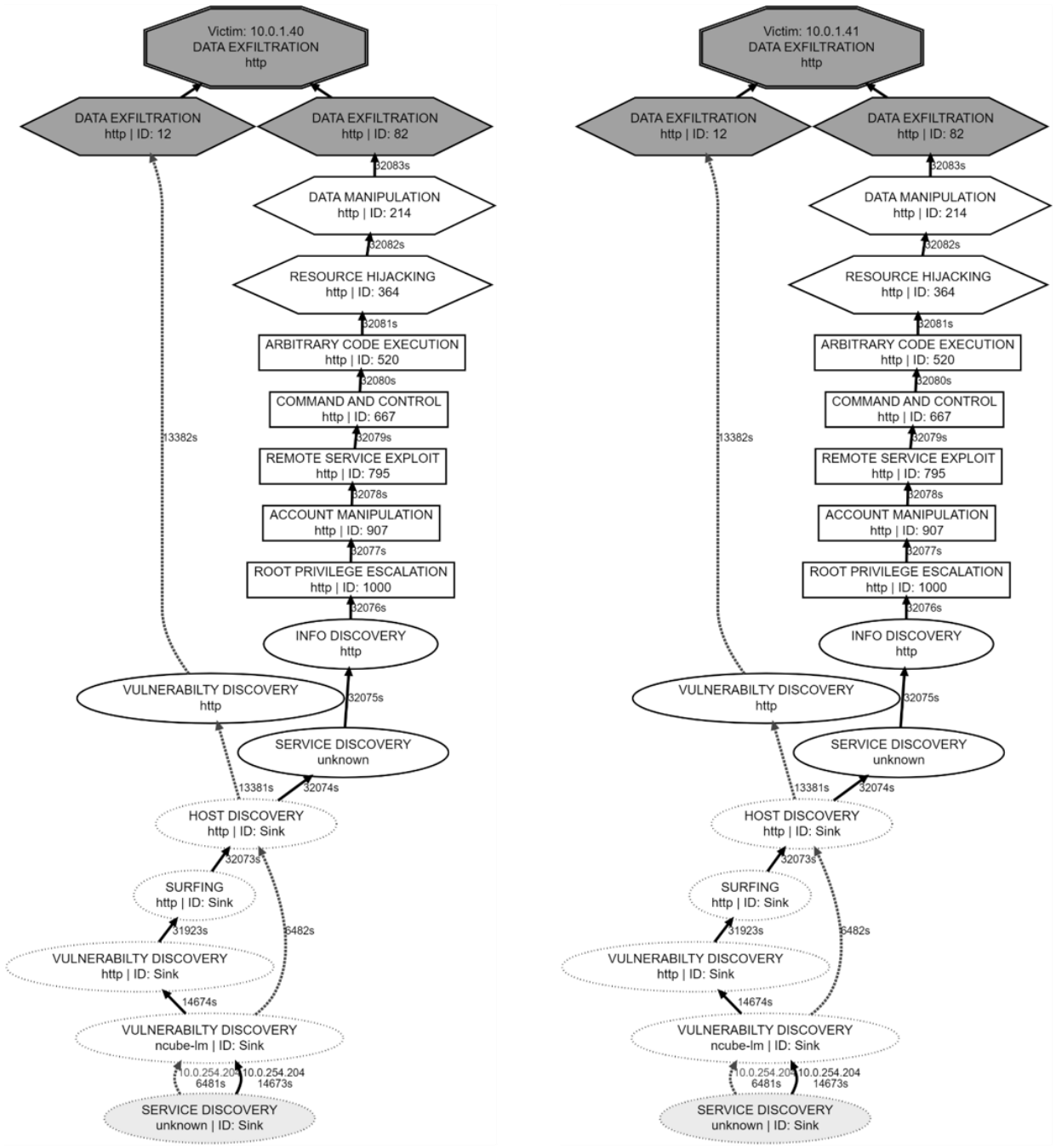

Figure 9: The alert-driven attack graphs from two different victim hosts. They show that two attackers attempt to exfiltrate data over the HTTP service on both hosts, where one team takes significantly more actions to obtain the objective than the other. Both the attack graphs are identical in terms of the attacker actions and their timestamps, indicating a potentially scripted attack attempt. (Adapted from Nadeem et al. 2022b)

The authors learn attacker strategies used by participating teams in the Collegiate Penetration Testing Competition (CPTC) (Munaiah et al. 2019), and the Collegiate Cyber Defense Competition (CCDC) from 2018. The CPTC alert dataset is

composed of 330,270 Suricata-based alerts generated by 6 participating teams, while the CCDC dataset is composed of over 1 million alerts. SAGE compresses all the CPTC alerts into 93 attack graphs, and all the CCDC alerts into 139 attack graphs. These graphs show the strategies employed by the various attackers to obtain their objectives. By visualizing the alert-driven attack graphs, the authors observe several insights regarding attacker strategies: First, the attackers seem to follow shorter paths to re-exploit objectives in 84.5% of the cases, which also appeals to common sense, since an attacker does not necessarily need to go through reconnaissance and scanning when they already know how to exploit a vulnerability. Second, they discover potentially scripted attacks by observing identical attack graphs for several hosts. The intuition is that the automated nature of a scripted attack targets several hosts simultaneously, which results in identical attack graphs for these hosts. An example of a potentially scripted data exfiltration attempt is given in Figure 9 that shows identical attack graphs for two different victim hosts. In addition, the authors posit that the rarity of certain attack paths can serve as fingerprints for attacker re-identification. They also propose a metric based on weighted average percentage to rank the participating teams based on the fraction of observed critical actions. This metric can provide a faster and cheaper alternative to manually ranking teams. Finally, by comparing the S-PDFA models of different alert captures, the authors conclude that it might be easier to break the cyber kill chain, and to place countermeasures in some network infrastructures, depending on the reachability of critical milestones.

3.3. Use case 3: Attack campaign discovery (HeAT)

In this example, Moskal et al. (Moskal and Yang 2021b) aim to reverse engineer an attack campaign, given a critical intrusion event. They develop HeAT – a semi-supervised learning system that incorporates analyst domain knowledge regarding the contribution of intrusion alerts to an attack campaign in a so-called “Alert Episode HeAT” value. For a given critical event, prior alerts are grouped into alert episodes and each alert episode is given a HeAT value estimating its contribution towards the critical event. A HeAT value of 0 indicates no contribution, and a higher value indicates increasing contribution towards the critical event. This way, HeAT sorts out the relevant alert episodes and gives an estimated progression of how past malicious activities have led to the critical event. HeAT tries to mimic and automate the triaging process of human analysts by learning their emphasis on particular features (such as port-service, attack intent stages, and IP addresses) to reconstruct an attack campaign. The system overview of HeAT is shown in Figure 10.

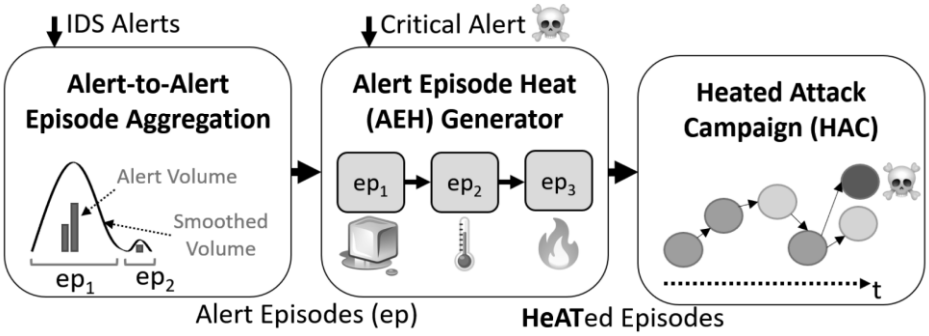

Figure 10: The system overview of HeAT. It aggregates intrusion alerts into episodes, and extracts features from them. These features are used in a tabular model to predict their corresponding episode's HeAT value. Given a critical event, the episodes with high HeAT values are used to reconstruct an attack campaign. (Adapted from Moskal and Yang 2021b)

Moskal et al. define an "Attack Campaign" as a sequence of attacker actions showing how an attacker(s) gained initial access to the network and eventually obtained their objectives. In other words, an attack campaign enumerates multi-stage actions leading up to a critical event, by taking into consideration the network's context (e.g., services running, IP subnets) and the relationship between prior alerts and the critical event, similar to that used for ASSERT (Use case 1). An "Alert Episode HeAT" is a numeric value between 0 and 3 that captures the analyst opinion about the contribution of a given alert episode to a critical event. To this end, the authors curate a small labeled dataset composed of intrusion alerts and their corresponding HeAT values that are manually assigned by security analysts. They train a tabular model on this labeled dataset to predict HeAT values of other unseen alerts.

The process of attack campaign extraction is as follows:

- Intrusion alerts are aggregated into episodes using a Gaussian smoothing approach on a per-attack stage basis. Note that the concept of an episode is similar to that of SAGE (Use case 2) with a different implementation.
- Several network agnostic features are used to characterize the timing-related, IP-related, and action-related differences between two episodes. These network agnostic features allow to uncover similar attack campaigns across different network infrastructures.
- The features are given to the tabular model to predict the HeAT values. Naturally, the HeAT values are predicted for alerts that appeared before a selected critical alert.
- Finally, the alerts with non-zero HeAT values are used to reconstruct the attack campaign.

The authors also propose an entropy-based "HeAT-gain" metric to evaluate the quality of extracted attack campaigns. This metric is based on the diversity and completeness of attack stages in the attack campaign, the reduction of irrelevant alerts in the attack campaign, and the overall coherence between analyst opinions and the

predicted HeAT values for the alerts in the attack campaign. The authors envision that HeAT-gain can also be used to prioritize attack campaigns.

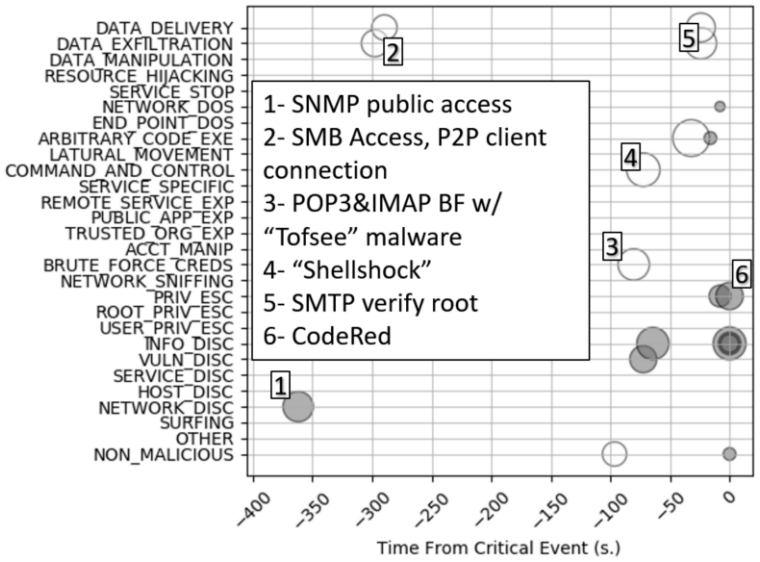

Figure 11: HeATed attack campaign for the CodeRed exploit from the CCDC dataset. (Adapted from Moskal and Yang 2021b)

The authors extract attack campaigns from the Collegiate Penetration Testing Competition (CPTC) (Munaiah et al. 2019), and the Collegiate Cyber Defense Competition (CCDC) from 2018. They observe that the CodeRed exploit appears several times in both datasets. Specifically, in the CPTC dataset, they present an example of two different types of adversaries that exploit CodeRed in significantly different ways, i.e., a script kiddie and a calculated adversary. The authors find 144 episodes showing that the script kiddie targets multiple different hosts with approximately uniform time between attempts, indicating a scripted attempt. On the contrary, there are 19 episodes related to the calculated adversary, showing that they consistently target a single host with significant time between attempts. HeAT succeeds in discarding a remarkable amount of irrelevant alerts for this exploit in the given time-frame, i.e., a reduction of 71% and 92% episodes for the script kiddie and the calculated adversary, respectively. They also demonstrate that the HeAT values learnt from the CPTC dataset can be used to identify similar exploits in the CCDC dataset, which was collected in a significantly different environment. Figure 11 shows an attack campaign extracted from the CCDC dataset for the CodeRed exploit. The authors find striking similarities between the campaigns from both datasets. Namely, the adversary found a vulnerable SMB share that enabled them to deliver malware to the victim host. POP and IMAP are targeted due to a vulnerability in the mail server, which is used to gain initial access to the victim host. They also use the Shellshock vulnerability and SMTP verification of root access to gain further access to the victim host. Even though this attack campaign is not exactly identical to the ones found in the CPTC dataset, the authors demonstrate that HeAT generalizes to other network infrastructures and enables analysts to find similar attack campaigns.

4. Thoughts and Future Opportunities

The three cyber defense agents described in Section 3, namely ASSERT (use case 1), SAGE (use case 2), and HeAT (use case 3) actively address the challenges listed in Section 2:

- The three agents are designed specifically to help security analysts manage the rapidly evolving threat landscape. Particularly, ASSERT continuously creates and updates attack behavior models as new intrusion alerts come in. All three agents use several open source unlabeled experimental datasets that resemble real-world operations.
- They also conduct extensive qualitative analysis since they operate in either semi-supervised or unsupervised settings. The three agents model adversary behavior differently, i.e., ASSERT builds attack models with an emphasis on continual learning, HeAT reconstructs attack campaigns with an emphasis on forensic analysis, and SAGE models attacker strategies with an emphasis on threat intelligence.
- In addition, they all model contextual information using some combination of temporal and statistical features. SAGE particularly also discovers probabilistic patterns in intrusion alerts to model their semantics.
- Finally, the three agents utilize visual analytics to communicate their findings with security analysts. SAGE specifically uses an interpretable model, and has a detailed explainability analysis of all its components.

Needless to say, developing a data-driven autonomous cyber defense agent is a challenging task. When designed carefully, such an agent can increase the productivity of security analysts by tenfold. However, it is not always straightforward to realize when an agent works as expected. Below, we list a few recommendations for R&D practitioners and researchers who are getting started in this field:

- While observable data reveals a lot about an adversary, it does not show the full picture: It requires carefully selected data sources to form a reliable picture of an adversary's behavior, and even then, putting those data sources together is not so straightforward (recall the challenges from, e.g., Multimodal learning). Oftentimes, the ideal dataset is not even available, and one must make do with noisy and unlabeled datasets.
- Beware of spurious and undesired correlations learnt by autonomous agents: First, if there is bias in an observable dataset, it will likely be learnt by the autonomous agent, making it behave unexpectedly, especially for minority classes. Second, an autonomous agent will find patterns even when there are none in the training dataset. For example, a clustering algorithm will always find clusters, even when the dataset does not have any. Therefore, it is important to run an autonomous agent on drastically different use cases and verify their output, whenever possible.
- Be prepared to handle a lot of false positives, but also be on the lookout for false negatives: Autonomous agents will likely never achieve a 100% accuracy in modeling adversary behavior, owing to the complexity of the real world and the noise in observables. These errors can either appear as false positives (false

alarms) or false negatives (missed opportunities). False alarms are often investigated manually. In real-world SOC operations, even 0.1% of false alarms can be too much to handle on a daily basis (Axelsson 2000). False negatives can be even more dangerous because they give a false sense of security. These could either refer to adversary behaviors not picked up by the autonomous agent, or not yet executed by the adversary in the first place. While there is little that can be done about the latter, the former is a big problem since there are typically insufficient traces in the observables to learn about them. Infrequent pattern mining is currently an open area of research.

- Beware of misleading metrics: It is convenient to choose from several autonomous agents when their performance can be quantified in terms of metrics. Metrics like accuracy, F1 score, Area Under the Curve (AUC) are widely used in the machine learning world. However, metrics can mislead if not correctly chosen: It is easy to obtain impressive accuracy on highly imbalanced datasets when the autonomous agent does not even work. Qualitative studies, although time-consuming and frequently subjective, provide much deeper insights into how an agent actually works, and whether it works as expected.
- Choose interpretable models: It is understandable to get swept away by the giant state of the art models. However, understanding how they actually reach their decisions is extremely difficult. When practitioners are able to understand the model internals, they can debug and fine-tune those models for further performance optimizations. Spotting bugs in interpretable models is much easier than debugging a black box model. The recent emphasis on explainable AI and interpretable models by-design has given evidence that white box models achieve competitive performance compared to their deep learning counterparts. It is also easier to trust the decisions of an interpretable model over a black box model, even when post-hoc explanations are provided, which themselves can be manipulated. Besides, adversaries are known to exploit weaknesses in cyber defenses. Recent studies have also shown that robust models are more interpretable, since a robust model appears to provide more human intelligible explanations (Ross and Doshi-Velez 2018).
- Multi-faceted explanations: An intuitive explanation that does not accurately explain an agent's behavior is more dangerous than a poor explanation. The faithfulness of an agent and its explanations are both equally important. Multi-faceted explanations from different perspectives can help identify any discrepancies between an agent's decision and an analyst's intuition. For example, finding indicators of compromise from multiple data sources can ensure that an intrusion alert is not a false alarm. To understand why an agent assigns a particular label to an observable, it can be helpful to investigate the labels of similar observables that receive a different label. This is known as contrastive explanations, and it can help answer "what-if" questions about the agent's reasoning.
- Realistic assumptions: Having unrealistic expectations of what an autonomous agent may be able to do can set one up for failure. Despite recent advances, it is important to understand that machine learning can only do so much, especially with noisy observables. Instead, it is better to identify the key strengths of

autonomous agents. For example, we know that autonomous agents are much better at monotonous tasks, and can browse through large volumes of observables much faster than human analysts. Realizing this strength, Holder et al. (Holder and Wang 2021) design an autonomous cyber defense agent to serve as a junior 'support' analyst for human security analysts. The autonomous agent takes over the repetitive and time-consuming jobs, such as scouring the Internet for resources, correlating patterns in large volumes of intrusion alerts, and presenting them to their human counterparts. The human analysts are then free to spend their time doing more complex tasks, such as investigating critical attacks and threat hunting.

Given the increasing number of cyber-attacks in recent times, autonomous cyber defense agents will become a necessity in dealing with large volumes of observables. However, there is still much to be done regarding the adversarial robustness and effectiveness of these agents. We discuss a few prominent research directions below:

- Learning from temporal features is challenging since features are often out-of-sync. Although temporal features provide a lot of contextual cues, it is not always apparent how to represent and learn from them effectively. There are two interesting problem classes: Picking an event at present and looking backwards in time can provide forensic and provenance analysis capabilities, while looking forward can provide threat intelligence and predictive capabilities.
- Learning from infrequent observables that reflect the rare adversary behaviors is an important open problem.
- We expect that multimodal learning and multi-faceted explanations will play a key role in designing trustworthy and robust cyber defense agents that learn contextually meaningful adversary behaviors.
- Incorporating concise reporting and visualization tools that reduce the cognitive load on security analysts is important for effective communication between analysts and autonomous cyber defense agents.
- Cyber defense agents with threat prioritization capabilities that can present their findings at the right level of abstraction (based on an analyst's expertise) are important for usability and deployability.

5. Summary and Conclusions

This chapter discusses how an autonomous cyber defense agent can gain insights into the behaviors and intents of cyber adversaries. The popular Capability, Opportunity, and Intent (COI) model helps categorize various types of adversaries, while the Cyber Kill Chain and ATT&CK frameworks describe the tactics, techniques and procedures (TTPs) of cyber adversaries. Specifically, the Action Intent Framework (AIF) is derived from MITRE ATT&CK that infers the intent of cyber adversaries from traffic induced observable data.

The rapidly evolving threat landscape and adversary TTPs have made it near-impossible for autonomous agents to rely on a priori expert knowledge. Instead, the emphasis should be on the design and deployment of data-driven autonomous agents

that learn contextual meaningful adversary behaviors from observable data. Although enticing, designing effective and reliable data-driven agents is difficult due to:

- The evolving and adversarial threat landscape that requires proactive and robust machine learning models,
- The unavailability of good quality observable data with corresponding ground truth that makes supervised learning paradigm unsuitable,
- The unpredictable nature of cyber adversaries and their tactics that require context modeling to adequately capture their behavior,
- The challenges of open world evaluation of black box machine learning models to ensure that the agent behaves as expected.

This chapter illustrates three state-of-the-art use cases for autonomous cyber defense agents that learn adversary behavior from traffic induced observables in order to determine an effective remediation and to assist security analysts in defending against the adversary in a timely manner.

- ASSERT is an unsupervised continual learning system that synthesizes and updates emerging attack behavior models from intrusion alerts in a streaming setting without expert input. The behavior models provide a statistical summary of the various attacks conducted by cyber adversaries.
- SAGE follows a two-step approach to reconstruct attacker strategies from intrusion alerts by first learning an unsupervised interpretable model that discovers temporal and probabilistic patterns in intrusion alerts, and then representing the discovered attacker strategies in the form of targeted attack graphs, also without any expert input. The attack graphs provide a dynamic view of the network by showing how specific attacks transpired, and enable visual analytics regarding attacker behavior dynamics.
- HeAT is a semi-supervised learning system that integrates analyst domain knowledge and reverse engineers multi-stage attack campaigns given a critical intrusion alert. This system helps analysts in uncovering the chain of actions that led to a critical alert while discarding thousands of irrelevant alerts.

How the use cases address the aforementioned challenges is described, together with some words of caution for practitioners getting started in the field, pertaining to observable data, spurious correlations, misleading metrics, and interpretable approaches. Finally, a few prominent future research directions are provided with respect to learning paradigms and reporting techniques.

References

Afianian A, Niksefat S, Sadeghiyan B, Baptiste D (2020) Malware Dynamic Analysis Evasion Techniques. ACM Computing Surveys 52:1–28

Alata E, Dacier M, Deswarte Y, et al (2006) Collection and analysis of attack data based on honeypots deployed on the Internet. In: Quality of Protection. Springer US, pp 79–91

Alsaheel A, Nan Y, Ma S, et al (2021) ${ATLAS}$: A Sequence-based Learning Approach for Attack Investigation. In: 30th USENIX Security Symposium (USENIX Security 21). pp 3005–3022

Apruzzese G, Andreolini M, Marchetti M, et al (2020) Deep Reinforcement Adversarial Learning Against Botnet Evasion Attacks. IEEE Trans Netw Serv Manage 17:1975–1987. https://doi.org/10.1109/TNSM.2020.3031843

Axelsson S (2000) The base-rate fallacy and the difficulty of intrusion detection. ACM Trans Inf Syst Secur 3:186–205. https://doi.org/10.1145/357830.357849

Bianco D (2013) The pyramid of pain. Enterprise Detection & Response

Buczak AL, Guven E (2016) A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. IEEE Communications Surveys Tutorials 18:1153–1176. https://doi.org/10.1109/COMST.2015.2494502

Cai H, Meng N, Ryder B, Yao D (2019) DroidCat: Effective Android Malware Detection and Categorization via App-Level Profiling. IEEE Trans Inf Forensics Secur 14:1455–1470. https://doi.org/10.1109/TIFS.2018.2879302

Carrasco RC, Oncina J (1994) Learning stochastic regular grammars by means of a state merging method. In: Grammatical Inference and Applications. Springer Berlin Heidelberg, pp 139–152

Chen S, Xue M, Fan L, et al (2018) Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach. Comput Secur 73:326–344. https://doi.org/10.1016/j.cose.2017.11.007

Croft R, Ali Babar M, Chen H (2022) Noisy Label Learning for Security Defects. arXiv [cs.SE]

Du P, Sun Z, Chen H, et al (2018) Statistical Estimation of Malware Detection Metrics in the Absence of Ground Truth. IEEE Trans Inf Forensics Secur 13:2965–2980. https://doi.org/10.1109/TIFS.2018.2833292

Eslahi M, Rohmad MS, Nilsaz H, et al (2015) Periodicity classification of HTTP traffic to detect HTTP Botnets. In: 2015 IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE). pp 119–123

García S, Grill M, Stiborek J, Zunino A (2014) An empirical comparison of botnet detection methods. Comput Secur 45:100–123. https://doi.org/10.1016/j.cose.2014.05.011

Hammerschmidt C, Marchal S, State R, Verwer S (2016) Behavioral clustering of non-stationary IP flow record data. In: 2016 12th International Conference on Network and Service Management (CNSM). pp 297–301

Holder E, Wang N (2021) Explainable artificial intelligence (XAI) interactively working with humans as a junior cyber analyst. Human-Intelligent Systems Integration 3:139–153. https://doi.org/10.1007/s42454-020-00021-z

Hutchins EM, Cloppert MJ, Amin RM, Others (2011) Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Leading Issues in Information Warfare & Security Research 1:80

Jha S, Sheyner O, Wing J (2002) Two formal analyses of attack graphs. Proceedings 15th IEEE Computer Security Foundations Workshop. CSFW-15

Jordaney R, Sharad K, Dash SK, et al (2017) Transcend: Detecting concept drift in malware classification models. In: 26th USENIX Security Symposium (USENIX Security 17). pp 625–642

Jordaney R, Wang Z, Papini D, et al (2016) Misleading metrics: On evaluating machine learning for malware with confidence. Tech Rep

Kolosnjaji B, Demontis A, Biggio B, et al (2018) Adversarial Malware Binaries: Evading Deep Learning for Malware Detection in Executables. In: 2018 26th European Signal Processing Conference (EUSIPCO). pp 533–537

Letham B, Rudin C, McCormick TH, Madigan D (2015) Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model. aoas 9:1350–1371. https://doi.org/10.1214/15-AOAS848

Li H, Wei F, Hu H (2019) Enabling Dynamic Network Access Control with Anomaly-based IDS and SDN. In: Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization. Association for Computing Machinery, New York, NY, USA, pp 13–16

Liu H, Zhong C, Alnusair A, Islam SR (2021) FAIXID: A Framework for Enhancing AI Explainability of Intrusion Detection Results Using Data Cleaning Techniques. Journal of Network and Systems Management 29:40. https://doi.org/10.1007/s10922-021-09606-8

Lu Y, Richter F, Seidl T (2020) Efficient Infrequent Pattern Mining Using Negative Itemset Tree. In: Appice A, Ceci M, Loglisci C, et al. (eds) Complex Pattern Mining: New Challenges, Methods and Applications. Springer International Publishing, Cham, pp 1–16

Manning C, Raghavan P, Schütze H (2010) Introduction to information retrieval. Natural Language Engineering 16:100–103

Marpaung JAP, Sain M, Lee H-J (2012) Survey on malware evasion techniques: State of the art and challenges. In: 2012 14th International Conference on Advanced Communication Technology (ICACT). pp 744–749

McFate M (2005) The military utility of understanding adversary culture. OFFICE OF NAVAL RESEARCH ARLINGTON VA

Michie S, van Stralen MM, West R (2011) The behaviour change wheel: a new method for characterising and designing behaviour change interventions. Implement Sci 6:42. https://doi.org/10.1186/1748-5908-6-42

Moskal S, Yang SJ (2020) Cyberattack Action-Intent-Framework for Mapping Intrusion

Observables. arXiv [cs.CR]

Moskal S, Yang SJ (2021a) Translating Intrusion Alerts to Cyberattack Stages using Pseudo-Active Transfer Learning (PATRL). In: 2021 IEEE Conference on Communications and Network Security (CNS). pp 110–118

Moskal S, Yang SJ (2021b) Heated Alert Triage (HeAT): Network-agnostic extraction of cyber attack campaigns. In: Proceedings of the Conference on Applied Machine Learning for Information Security

Moskal S, Yang SJ, Kuhl ME (2018) Extracting and Evaluating Similar and Unique Cyber Attack Strategies from Intrusion Alerts. In: 2018 IEEE International Conference on Intelligence and Security Informatics (ISI). pp 49–54

Munaiah N, Pelletier J, Su S-H, et al (2019) A Cybersecurity Dataset Derived from the National Collegiate Penetration Testing Competition. In: HICSS Symposium on Cybersecurity Big Data Analytics

Nadeem A, Hammerschmidt C, Gañán CH, Verwer S (2021a) Beyond Labeling: Using Clustering to Build Network Behavioral Profiles of Malware Families. In: Stamp M, Alazab M, Shalaginov A (eds) Malware Analysis Using Artificial Intelligence and Deep Learning. Springer International Publishing, Cham, pp 381–409

Nadeem A, Rimmer V, Joosen W, Verwer S (2022a) Intelligent Malware Defenses. In: Batina L, Bäck T, Buhan I, Picek S (eds) Security and Artificial Intelligence: A Crossdisciplinary Approach. Springer International Publishing, Cham, pp 217–253

Nadeem A, Verwer S, Moskal S, Yang SJ (2022b) Alert-Driven Attack Graph Generation Using S-PDFA. IEEE Trans Dependable Secure Comput 19:731–746. https://doi.org/10.1109/TDSC.2021.3117348

Nadeem A, Verwer S, Moskal S, Yang SJ (2021b) Enabling Visual Analytics via Alert-driven Attack Graphs. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. Association for Computing Machinery, New York, NY, USA, pp 2420–2422

Nadeem A, Verwer S, Yang SJ (2021c) SAGE: Intrusion Alert-driven Attack Graph Extractor. In: 2021 IEEE Symposium on Visualization for Cyber Security (VizSec). pp 36–41

Noel S, Elder M, Jajodia S, et al (2009) Advances in Topological Vulnerability Analysis. In: 2009 Cybersecurity Applications Technology Conference for Homeland Security. pp 124–129

Okutan A, Yang SJ (2019) ASSERT: attack synthesis and separation with entropy redistribution towards predictive cyber defense. Cybersecurity

Piplai A, Mittal S, Joshi A, et al (2020) Creating Cybersecurity Knowledge Graphs From Malware After Action Reports. IEEE Access 8:211691–211703

Ribeiro MT, Singh S, Guestrin C (2016) "why should I trust you?": Explaining the predictions of any classifier. arXiv [cs.LG]

Rimmer V, Nadeem A, Verwer S, et al (2022) Open-World Network Intrusion Detection. In: Batina L, Bäck T, Buhan I, Picek S (eds) Security and Artificial Intelligence: A Crossdisciplinary Approach. Springer International Publishing, Cham, pp 254–283

Roscher R, Bohn B, Duarte MF, Garcke J (2020) Explainable Machine Learning for Scientific Insights and Discoveries. IEEE Access 8:42200–42216. https://doi.org/10.1109/ACCESS.2020.2976199

Ross A, Doshi-Velez F (2018) Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing Their Input Gradients. AAAI 32:

Rudin C (2019) Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence 1:206–215

Samhita L, Gross HJ (2013) The “Clever Hans Phenomenon” revisited. Communicative & Integrative Biology 6:e27122

Sauerwein C, Sillaber C, Mussmann A, Breu R (2017) Threat Intelligence Sharing Platforms: An Exploratory Study of Software Vendors and Research Perspectives. In: Wirtschaftsinformatik 2017 Proceedings

Schaberreiter T, Kupfersberger V, Rantos K, et al (2019) A Quantitative Evaluation of Trust in the Quality of Cyber Threat Intelligence Sources. In: Proceedings of the 14th International Conference on Availability, Reliability and Security. Association for Computing Machinery, New York, NY, USA, pp 1–10

Sebastián M, Rivera R, Kotzias P, Caballero J (2016) AVclass: A Tool for Massive Malware Labeling. In: Research in Attacks, Intrusions, and Defenses. Springer International Publishing, pp 230–253

Sejnowski TJ (2020) The unreasonable effectiveness of deep learning in artificial intelligence. Proc Natl Acad Sci U S A 117:30033–30038. https://doi.org/10.1073/pnas.1907373117

Sethi TS, Kantardzic M (2018) When Good Machine Learning Leads to Bad Security. Ubiquity 2018:1–14

Severi G, Meyer J, Coull S, Oprea A (2021) Explanation-Guided Backdoor Poisoning Attacks Against Malware Classifiers. In: 30th USENIX Security Symposium (USENIX Security 21). pp 1487–1504

Slack D, Hilgard S, Jia E, et al (2020) Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society. Association for Computing Machinery, New York, NY, USA, pp 180–186

Souri A, Hosseini R (2018) A state-of-the-art survey of malware detection approaches using data mining techniques. Human-centric Computing and Information Sciences 8:1–22. https://doi.org/10.1186/s13673-018-0125-x

Steinberg A (2007) Predictive modeling of interacting agents. In: 2007 10th International Conference on Information Fusion. pp 1–6

Steinberg AN (2005) An approach to threat assessment. In: 2005 7th International Conference on Information Fusion. p 8 pp.–

Strom BE, Applebaum A, Miller DP, et al (2018) Mitre att&ck: Design and philosophy. Tech Rep NAVTRADEVCEN

Surber JG, Zantua M (2022) Intelligent Interaction Honeypots for Threat Hunting within the Internet of Things. CISSE 9:5–5. https://doi.org/10.53735/cisse.v9i1.147

Szczepański M, Choraś M, Pawlicki M, Kozik R (2020) Achieving Explainability of Intrusion Detection System by Hybrid Oracle-Explainer Approach. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp 1–8

Ucci D, Aniello L, Baldoni R (2019) Survey of machine learning techniques for malware analysis. Comput Secur 81:123–147. https://doi.org/10.1016/j.cose.2018.11.001

Verwer S, Hammerschmidt CA (2017) flexfringe: A Passive Automaton Learning Package. In: 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp 638–642

Yang SJ, Okutan A, Werner G, et al (2021) Near Real-time Learning and Extraction of Attack Models from Intrusion Alerts. arXiv [cs.CR]

Author Biographies

Azqa Nadeem is a Ph.D. candidate and junior lecturer in the Intelligent Systems department (Cyber Analytics Lab) at Delft University of Technology in the Netherlands. She has a master of computer science with an emphasis on data science and cyber security from Delft University of Technology, and a B.S. in computer science from the National University of Science and Technology, Pakistan. Her research interests include cyber-attack modeling, data analytics, network security, explainable machine learning, and sequence models. Applications include intrusion alert driven attack graph generation, and the behavioral profiling of malware samples.

Sicco Verwer is an Associate professor at Delft University of Technology in machine learning for cybersecurity. He is the head of the TU Delft Cyber Analytics Lab where he works on understandable AI for intrusion detection and software understanding. His team won several AI challenges including ones on learning software models, automated reverse engineering, and adversarial machine learning. He received many grants and awards for his research including prestigious VENI and VIDI grants from NWO, and a test-of-time award from ECMLPKDD for his pioneering work on discrimination-free classification.

Shanchieh (Jay) Yang is a Professor in the Department of Computer Engineering and Director of Global Outreach for Global Cybersecurity Institute at Rochester Institute of Technology. He received his M.S. and Ph.D. degrees in Electrical and Computer

Engineering from the University of Texas at Austin in 1998 and 2001, respectively. His research focuses on cyber-attack modeling, machine learning, and simulation to enhance cyber situational awareness and anticipatory cyber defense. He was a NSF Trusted CI Fellow in 2019 and a NSF Trusted CI TTP Fellow in 2020. He was recognized in 2019 with IEEE Region 1 Outstanding Teaching in an IEEE Area of Interest Award – for outstanding leadership and contributions to cybersecurity and computer engineering education.