TRAIL: A Knowledge Graph-based Approach for Attributing Advanced Persistent Threats

Isaiah J. King*†, Ramiro Ramirez*, Benjamin Bowman*, H. Howie Huang*†
*Cybermonic LLC, McLean, VA †The George Washington University, Washington, DC
iking5@gwu.edu, {ramiro, benjamin}@cybermonic.com, howie@gwu.edu

***Abstract*—Open-source intelligence exchanges provide a rich repository of indicators of compromise (IOCs). These IOCs are used to build detection signatures and blocklists in production cybersecurity environments as well as prior works. In this work, we investigate their utility for cyberattack attribution. To do this, we create a novel system called TRAIL that builds a knowledge graph of network-based IOC co-occurrences in cyberattacks, and their relations to other IOCs. After analyzing 4,500 cybersecurity events attributed to 22 different advanced persistent threats (APTs), the knowledge graph holds over 2.1 million nodes with 7.9 million edges. We analyze the knowledge graph this system produces using conventional machine learning, graph analytics, and a graph neural network to quantify the degree to which APTs leave identifiable clues in their IOCs. Using the TRAIL method to enrich the IOC feature space, IOCs can individually be attributed to the APT that generated them with 45% accuracy. When attributing groups of IOCs that made up cyberattacks, indirect resource reuse alone accurately attributed 82% of samples. When we used both graph topology and feature analysis and analyzed events with a graph neural network, attribution accuracy increased to 84%. Finally, we conducted a 6-month study of new cyber events our models had never seen. We found that our models continue to achieve similar accuracy on real-world data to what was observed experimentally, so long as the database is no more than 1 month out of date.**

***Index Terms*—Big Data applications, Cyber threat intelligence, Graph theory**

I. INTRODUCTION

In 2020, more than 30,000 public and private organizations found out they had been compromised. An unprecedented number of private, local, state, and federal agencies had all fallen victim to one of the most notorious cyberattacks of the 21st century: the SolarWinds incident [1]. Advanced attackers put a backdoor into the SolarWinds Orion product, which the company itself shipped to its many clients. The victims of this attack wanted to know: who did this? Immediately, identifying the group responsible for this attack was a top priority for the US government–so important that a cybersecurity advisor was added to the National Security Council for this task [2]. After a six-month investigation, the US government [3] as well as private cybersecurity firms [4] attributed the attack with high confidence to the Russian threat actor, APT29.

Confidently identifying the threat actor responsible for a cyberattack (or *attribution*) is vital for both the government and the private sector [5]. Unfortunately, this process is both challenging and time-consuming. Singer and Friedman state that "perhaps the most difficult problem [in cyber security] is that of attribution" [6, p. 73]. This is due to the ease of anonymity skilled adversaries enjoy over the internet [7], and attackers' motivations to disguise their true affiliations [8], or even to emulate and frame other groups [9]. For example, attackers obfuscate themselves by placing foreign language strings in their malware binaries [10] or changing their machines' internal clocks to disguise their time zone [9]. There are an uncountable number of tricks that attackers could employ to disguise who they are in a constant game of cat and mouse.

On the other hand, there are some artifacts that are difficult to fake: e.g., logistical resources such as command and control servers, which are expensive to buy or rent, and time-consuming to set up. Seeing these resources reused can be strong support for attribution [9]. Recent approaches that take a relational view of advanced persistent threat (APT) campaigns have found that when viewed as a graph, specific shared resources form dense cliques representing code or resource reuse [11]. Beyond reusing individual resources, many prior works have shown that the features of indicators of compromise (IOCs), especially the features of malicious binary files such as strings in the code, or registry keys the program edits can be used as evidence to attribute them to their authors [12], [10], [13], [14], [15], [16].

Unfortunately, network-based IOCs (URLs, domains, and IPs), the focus of this work, are largely unstudied for APT attribution. Prior works to classify domain names [17], [18], [19] and URLs [20], [21], [22] only focus on labelling them as malicious or benign. Not only is this a gap in the body of research but ignoring network-based IOCs neglects intelligence from the largest source of attacker information. Open-source intelligence (OSINT) exchanges have far more network-based IOCs than binaries. At the time of writing, VirusTotal processed 3.21 million new and distinct URLs in the past 24 hours, compared to just 1.1 million files [23]; AlienVault OTX tracks 26 million domains, compared to only 2 million SHA-256 hashes [24]. In addition to their sheer volume, network-based IOCs are among the most transparent IOCs to analyze. File hashes are useful for detecting resource reuse, but without deeper binary analysis, they only provide information about exact file co-occurrences in different attacks. In contrast, network-based IOCs have a rich feature set, if one takes the time to analyze them. The features of IOCs provide insight into attackers' behavioral patterns: where they register domains, which servers they use, and how many domains link to the same IP. This information is available through

OSINT tools, whereas features about files are less commonly shared. Additionally, many network-based IOCs have direct relationships to other, potentially novel IOCs. URLs are always served on domains; domains are always hosted on IPs; IPs are always part of ASN groups. We will show that network-based IOCs[1], due to the ease of extracting additional features and additional relationships, can begin to characterize the *behavior* of adversaries.

In this paper, we address the APT attribution challenge by developing a novel system, TRAIL, which leverages OSINT data to build a rich knowledge graph of attacker information, which we call the TRAIL knowledge graph (TKG). The TKG is composed of feature-enriched IOCs that have been observed in attributed cyber incidents, built entirely using open-source intelligence[2]. While there are a plethora of malware datasets labeled with APTs [12], [25], [26], to our knowledge, the TKG is the first large dataset that links network-based IOCs to the threat actors that produced them. This open-source dataset establishes a baseline benchmark and will enable researchers to explore new research directions in this area, for example, APT attribution via feature analysis, graph analytics, or natural language processing on the raw text of threat reports. This dataset focuses on attributed *events*: user reports of cyber incidents listing observed IOCs and the associated threat actors. Each event is viewed as a node, labeled with the APT the report was attributed to, and with edges to the IOCs included in the incident report. As part of the TKG construction process, the IOCs are passed through an additional feature-engineering phase to capture more than just resource reuse, to try to relate the circumstances of their creation to adversary behaviors. This feature enrichment process reveals edges between IOCs when they are related in some way. In total, the dataset consists of 4,512 events, which are attributed to 22 APTs, and a total of 2.121 million IOCs (119K IPs, 354K URLs, 1.641M domains, and 6K ASNs).

After we have constructed the full knowledge graph, we can analyze it to study APT behaviors. Specifically, we aim to answer two important research questions:

- **RQ1**: How similar are network-based IOCs left by the same attacker during different events?
- **RQ2**: How often are resources reused across different attacks?

To answer the first question, we analyze the IOCs using traditional machine learning techniques. We find that URLs can be attributed to the APT that generated them with 46% accuracy, IPs can be attributed with 38% accuracy and domains with 29%. Each model performed better than random, which indicates that individual IOCs do not carry much information that can be used for attribution, but they do hold some information. These results support our claim that while APTs take care to disguise themselves, they sometimes reveal subtle patterns that manifest in the underlying IOC features.

[1]For simplicity, we refer to these as "IOCs" for the remainder of the paper.
[2]Source code and data available at https://github.com/cybermonic/trail

To answer the second question, we view IOCs not on their own, but in the context of the complete TKG. We use label propagation through the network to attribute unlabeled events, given their partially labeled neighbors. This approach classifies unlabeled events with 82% accuracy. These results show that though direct resource reuse is rare, because paths exist through the TKG linking unrelated events attributed to the same APT, indirect infrastructure reuse does occur. We can further improve attribution accuracy by taking advantage of the tabular features of IOCs in conjunction with the topological features of the knowledge graph. As our initial experiment showed, the features of individual IOCs do contain valuable information. To incorporate these features with the relational data, we train a graph neural network (GNN) [27] to classify events by their threat actor. This further improved the event attribution accuracy to 85%.

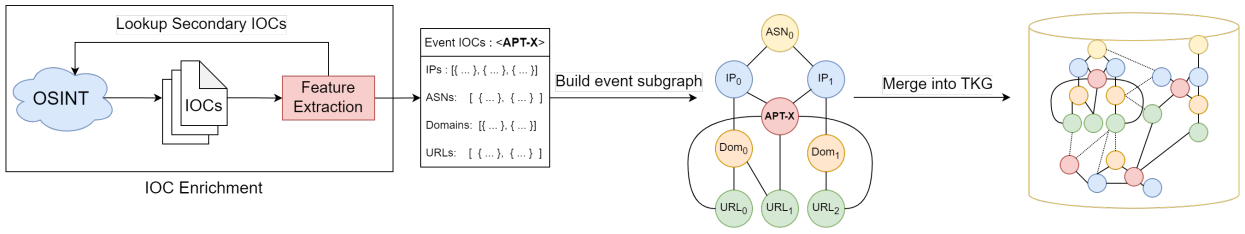
Finally, we conducted a long-term case study. We evaluated the ability of our model to attribute new cyber events over an 8-month period. In these experiments, we found that by retraining the GNN model monthly, and continuously updating the TKG, the model continues to achieve similar accuracy on data at least as recent as one month. We conclude that while statistical methods should not be used for definitive attribution, the TRAIL graph and the models we train upon it could be advantageous tools in cyber incident attribution.

In summary, this work makes the following contributions: we present the first dataset of IOCs observed in incident reports attributed to threat actors. To the best of our knowledge, we conduct the first study on attributing network IOCs to APTs, and the first study on attributing cyber events to APTs using only network IOCs.

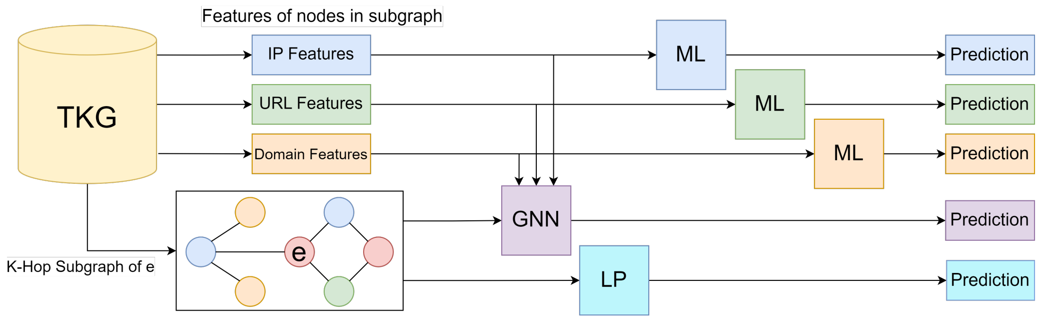
II. BACKGROUND

Indicators of compromise (IOCs) are any information that can be used to identify a potentially compromised system [28]. Often, IOCs are broken into three types: atomic, computed, and behavioral [29]. Atomic indicators are things like IP addresses or domain names, while computed indicators are data derived from individual items in an incident, such as file hashes. Behavioral indicators are collections of atomic and computed indicators grouped in meaningful ways that describe attackers' behaviors. Villalón-Huerta et al. [30] argue that atomic and computed IOCs on their own are of little use in the long term. A truly advanced persistent threat can avoid reusing the same IOCs if that is their desire. It is only by grouping them together and using them to describe attackers' behaviors that analysts can produce IOCs with long-term value. For this reason, we are motivated to store not only the individual IOCs of events, but their relations to each other, and their various co-occurrences in a graph structure.

Open Source Intelligence (OSINT) platforms are databases of publicly available IOCs, descriptions of attacks, and sometimes analyses of IOCs. Generally, sharing cyber threat intelligence (CTI) is beneficial for everyone involved, as attackers often reuse infrastructure during the same campaign [31]. However, this process has drawbacks: sometimes the data



(a) TRAIL System Architecture for building the cyber-events knowledge graph



(b) Ways of analyzing the TKG

Fig. 1: Full system overview. The TRAIL system parses and enriches open source threat intelligence to create the TRAIL Knowledge Graph (TKG). We then then analyze the TKG with three different approaches: classical ML, label propagation, and graph neural networks.

shared is not reliable [32] owing to the anonymous nature of CTI sharing platforms. Even worse, there is often a focus on quantity over quality with CTI sources [33] so it is important to analyze IOCs shared rather than blindly add them to blocklists.

III. Overview

In this section, we provide an overview of the TRAIL system, and how we analyze the knowledge graph it produces. Figure 1a illustrates the full system architecture to build the TRAIL knowledge graph. Using public threat intelligence repositories, we collect data in the form of raw JSON files that contain IOCs associated with incident reports, and the threat actor to whom the report is attributed. Next, we use open-source tools to analyze each IOC reported in the event and generate its features. Often, this leads to the discovery of additional IOCs related to those reported in some way (e.g., IPs discovered while analyzing a domain). As part of the enrichment process we request additional analysis on those IOCs to track as many features as possible. The original event, the reported IOCs, and any secondary IOCs we discovered during enrichment are then converted into an attributed graph according to the schema shown in Figure 2. Each event is represented as a central node labeled with a threat actor. This event node is connected to all IPs, URLs, and domains that were listed in the threat intelligence report. We also create edges between the reported IOCs and their secondary IOCs discovered in the enrichment process. Finally, the subgraph represented by the event node and its primary and secondary IOCs is merged into the TRAIL Knowledge Graph (TKG).

As illustrated in Figure 1b, in this work, we investigate three approaches to analyzing the TKG: traditional machine learning, traditional graph analysis, and graph neural networks. Traditional machine learning (ML) only considers the features of individual IOCs and ignores their relationships. These models are optimized to predict the threat group that produced any single IOC. Traditional graph analysis ignores nodes' features, and only considers their relationships. We use a process called label propagation (LP), where event nodes in the TKG send a vector representing their label to each of their neighbors. This process repeats iteratively; every labeled node sends its label to its neighbors. This results in a vector on each node that corresponds to its distance from every labeled event node. This vector represents the probability distribution function for an event's attribution. Lastly, graph neural networks (GNNs) take node features and relationships into consideration when making predictions. Using the k-hop neighborhood of the event, and the features of nodes in that subgraph, the GNN will produce a probability distribution function of which threat actors are most likely associated with a given event. In the following sections, we will discuss the knowledge graph construction and analysis in greater detail.

IV. TRAIL Knowledge Graph Construction

The TRAIL knowledge graph (TKG) is a graph of IOCs involved in various OSINT incident reports related by their co-occurrence in cyber events, and their operational relationships. For simplicity, we refer to the attributed cybersecurity incidents shared on these platforms as *events*. Each event is associated with several IOCs that the analysts who reported it wished to share and is associated with a single threat actor. Each IOC may have multiple features, depending on what kind of IOC it is. All nodes and relations are stored in a neo4j graph database [34]. In total we collected 4,512 events, each of which had, on average, 190 associated IOCs. The events collected were created between February of 2015 and May of 2023. In the rest of this section, we will detail our methodology for building this knowledge graph and generating features for each IOC.

A. Data Collection

We use the AlienVault OTX API to search for events tagged with APT names and their aliases. We selected OTX because it was the largest database with a public API we could find, and it already aggregates many existing MISP feeds [35]. While in this work, we only check AlienVault OTX for events, TRAIL could easily be extended to parse the responses from other data providers. As a precaution, if an event had multiple valid APT aliases as tags, it is ignored unless the tags all map to the same APT. This is to avoid downloading IOC dumps that are unrelated or relate to multiple incidents. At the end of this process, we have a list of event IDs and the APTs they are associated with. If an APT had at least 25 events attributed to it, we included the events in the TKG.

The next stage of the process is to get features for each IOC and to link the IOCs into an attributed graph. Each event contains a list of IOCs and their types, but to extract further information about each IOC, we request an analysis of it, either from OTX or another open-source tool (such as dig or passive DNS). This returns a great deal of information about each IOC, both in terms of its relations to other IOCs and in terms of its individual features; we will discuss the specifics of these features and relations in the following subsections. The output of the IOC analysis will often contain more IOCs. For example, using dig on a domain returns a list of IP addresses (A records); likewise, using dig on IPs yields records linking IPs to domains and ASNs. These are what we refer to in Figure 1a as “Secondary IOCs”. We collect and analyze these as well, though we do not directly associate them with events. This process of IOC discovery can be repeated indefinitely, but due to time and space constraints, we limit it to two hops from the initial event. Even with this constraint, in the resulting graph, 75% (1.581 out of 2.125 million) of nodes are secondary IOCs. This enrichment process is part of what makes the TKG so useful. Many tools already exist generate features for IOCs, but TRAIL goes beyond this and finds relationships too. While IOCs may not be directly reused, we will show that their related infrastructure sometimes is.

B. Feature Extraction

While it is useful to know which IOCs have been used by which APTs in the past, adversaries will endeavor to obfuscate their activities if they wish to remain obscured. To this end, a sneaky APT will try to use unique resources for attacks that cannot be traced back to them, should this be their desire. However, it is our hypothesis that either because details are overlooked, resources are being recycled, or for any other number of reasons, features more subtle than exact IOCs may get reused. In this work, we wish to quantify how often the *traits*, e.g., where domains are registered, the configuration of web servers, etc., of IOCs are reused by the same APTs.

While the feature set for domains and URLs overlaps with the prior works on lexical and DNS analysis, we add additional data about the servers that host URLs, and the files hosted at each address. We also track the country code, top-level domain (TLD), and whether the domains have an `NXDOMAIN` record–meaning it has been deactivated since being reported.

In total, we extract 507 features for IPs, 249 categorical features for country codes, and a one-hot vector for which if any of the top 265 IP issuers granted the IP address. URLs have the most features: 1,517 in total. A one-hot vector with 106 categories for the file type hosted at the URL address, 21 categories for the file class, 68 for the HTTP response code, 12 for the encoding method, 944 for the type of server used, 50 for the operating system run on that server, 183 possible services that could be running on the server, the top 100 TLDs, and 10 lexical features. Lastly, domains have 115 features total: the first 100 dimensions are a one-hot vector of the TLD, the next 9 represent the count of unique DNS records of each type captured in passive DNS, then, a single feature for if an `NXDOMAIN` record was found, and 4 lexical features.

Prior works such as Thin et al. [18] and Lui et al. [17] have shown that lexical features of domain names and statistical features in DNS records can be used to classify domains as malicious or benign. Likewise, there are numerous prior works that aim to classify URLs as malicious or benign using purely lexical features [20], [21], [22]. IP addresses have a dearth of features on their own, which may explain the lack of prior work in classifying them. Nonetheless, we track as many features as we can.

The most likely reason prior works tend to focus on lexical features is that they are inexpensive to generate. They are immediately extractable from the text of a simple list of IOCs. In addition to this ease of access, they can hint at adversaries’ tactics, techniques, and procedures (TTPs). For example, domains’ length and number of digits can be used to identify which domain generation algorithm was used, if relevant, linking domains to malware families. Other features like period count, can indicate TTPs such as subdomain hijacking. However, other categories of features such as geographic data, information about the adversary’s servers, and as Thin et al. [18] identify, DNS records, can also hold vital information. To address this problem, after building lists of IOCs associated with cyber-attacks, we use additional OSINT sources to analyze them.

Both IP addresses and domains can be queried in passive DNS databases that provide historic DNS records for them. In the case of IP addresses, this yields domains that have historically been linked to them. In the case of domains, passive DNS can provide IP addresses in the form of A records, and other domains if CNAME redirect records exist. We collect all passive DNS records for both kinds of IOCs

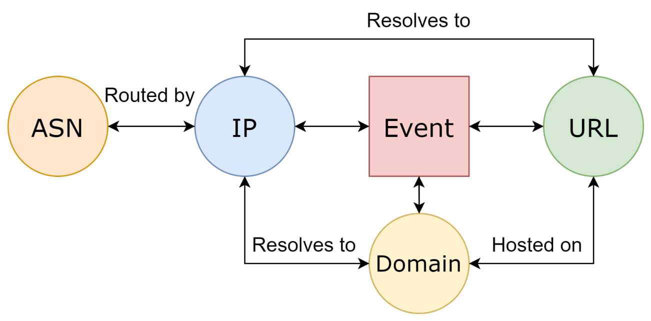


Fig. 2: IOC Graph Schema

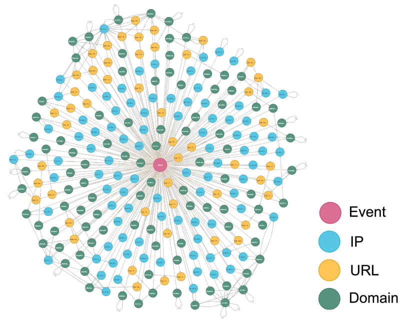


Fig. 3: Ego-net around an APT28 event in the TKG.

to use as which we use both as relational data during the graph construction phase (domain to IP edges) as well as node features (counts of unique record types). For IP addresses, we also use open-source IP lookup services to get the country code associated with the address, as well the ASN group it belongs to, and its issuer. Finally, for URLs, simply cURLing them, and analyzing the server header provides a great deal of data if the URL is still alive. In practice, the output of all these tools is archived by the OTX database, so we simply query their APIs to find these features and then convert them to vectors.

C. Graph Construction

Figure 2 shows the complete schema of possible relationships in the TKG. The most important relation is the one between the event node and IOCs. The event node represents the threat report itself. Its only feature is the APT with which it is associated. The event node has an edge from itself to any URL, IP, or domain that appears in the report it represents.

The IP analysis reports yield an ASN and a list of DNS A records. ASNs are added as additional nodes in the graph and are connected by an edge to the IP address. The DNS lists function as the second-order edges between IPs and domains. Domains and URLs have an obvious relation observable without analysis: a particular URL is hosted at a domain address. For example, if the URL `hxxp://threebody[.]cn/trisolaris.php` was provided as an IOC, the TKG would automatically create a node and request analysis for the domain `threebody[.]cn` and create a relation between them. The analysis reports for URLs also include the IP address it resolves to. Likewise, the DNS records returned from domain analysis contain `A` and `AAAA` records which also resolve to IP addresses. This relation is represented by an edge between the IP and URL or domain in the TKG. Table I contains a complete list of every edge type, and if they are derived, how they are obtained.

TABLE I: Relations captured during dataset construction

Source	Edge type	Destination	Description
Event	InReport	IP Domain URL	If an IOC appears in an incident report in an open source intelligence sharing platform
IP	`A` Record	Domain	Passive DNS captured a resolution from this IP to a domain name at some point in the past
	InGroup	ASN	The ASN containing the IP address. This is contained both in the passive DNS output, and can be found using tools such as `dig` or `wholis`
URL	ResolvesTo	IP	The IP address associated with the URL obtained via `nslookup` and passive DNS records of the domain hosting the URL
	HostedOn	Domain	The domain the URL is hosted on. Obtained via lexical analysis of the URL
Domain	ResolvesTo	IP	A resolution from this domain to an IP address obtained either by `nslookup` or `A` records in passive DNS

An ego network, or ego-net, is a graph focused on a single node (called the “ego”), such that every other node in the graph (called the “alters”) has a link to it [36]. A collection of cyber events can be thought of as a series of ego-nets where each event is an ego, and the IOCs reported therein are the alters. By applying further IOC analysis, we expand these ego-nets into richer graphs with additional edges that can connect alters, or discover new IOCs, potentially creating new paths between events. As an example, consider Figure 3, which shows the ego-net formed by a single event analysts attribute to APT28. Using this event as an example, the IOCs `1.0.36[.]127`, `v5y7s3[.]l2twn2[.]club`, and `hxxp://sfj54f7[.]17ti3sk[.]club/?H3%2540ba&d` are all present in this report. After analyzing the IP address, we find it is associated with China, and its estimated latitude and longitude, but there is no information about its ASN or issuer. Using passive DNS, we find that the domain has a single `A` record from 2021. This gives us a relation from the domain to the IP address it resolved to, a new IOC to look up, and features for first and last seen dates by the passive DNS service. Finally, checking the cached response headers of the URL reveals that it was running on an nginx server at the time it was first reported and that the content type was text/html encoded using gzip. In total, this subgraph has 239 related IOCs: 94 IPs, 95 domains, and 50 URLs.

We find that 85% of event nodes are two hops away from another event node, meaning 85% of OSINT reports share at least one IOC with at least one other report. In total, the final graph contains 4,512 events attributed to 22 APTs, 2.1 million IOC nodes, and 7.9 million edges in total.

V. The TKG Dataset Report

In this section we will describe the dataset we generated according to the process described in Section III. Beyond simply collecting IOCs and generating features for them, we link them together to form a large knowledge graph. In this section, we study the properties of the knowledge graph that is formed by this data and the relations that IOCs have with one another.

Table II shows more detailed information about the knowledge graph. Even though the number of events per APT can be small, we have a detailed picture of how these events overlap, and when they share resources. Also included in the table is the column "1st Order", meaning the number of IOCs that were directly reported in the events. Here, we see that almost 75% of the nodes in the graph were found during the enrichment process described in Section III.

Of the first-order IOCs, there is an additional concept of "Average Reuse", which measures how many distinct events contain a given IOC. Most IOCs only appear in one or two reports, as Figure 4 illustrates; however, a few appear in many. The most frequently repeated IP addresses and domains were those of Cobalt Strike-related C2 servers, supporting our claim that expensive infrastructure is likely to be recycled during the same campaigns. The most frequently repeated URLs were links to Chinese SSL certificates. Further analysis shows these certificates were related to WannaCry ransomware.

The TKG is strongly connected, with the largest connected component containing 2,123,737 (99.94%) of the nodes in the graph. We also note that when we analyze the subgraph of only first-order IOCs, the number of connected components increases from 161 to 477. The diameter of the largest component also decreases from 23 to 20 hops. This shows that the additional enrichment step reveals previously unknown links between IOCs, providing value to the analysis process. It is also interesting to note that the second and third largest connected components are both single FIN11 events, and the fourth and fifth largest connected components were both pairs of overlapping APT38 events. This suggests that nodes in the graph are well-clustered and that the shorter a path that exists between two events, the more likely they are to be attributed to the same APT.

VI. Data Analysis Methodology

In **RQ1** we ask to what extent IOCs in different cyber incidents are similar if the same APT generated them. IOC similarity is measurable through statistical analysis of their

TABLE II: Node and edge counts in the TKG

Type	Nodes	Edges	Avg. Degree	1st Order	Avg. Reuse
Events	4,512	857,174	190.0	N/a	N/a
IPs	119,194	2,936,002	24.63	51.85%	2.944
URLs	354,138	996,685	2.814	93.21%	1.253
Domains	1,641,194	3,026,628	1.844	10.65%	1.497
ASNs	6,028	99,873	16.57	N/a	N/a
Total	2,125,066	7,916,362	3.725	26.66%	1.513

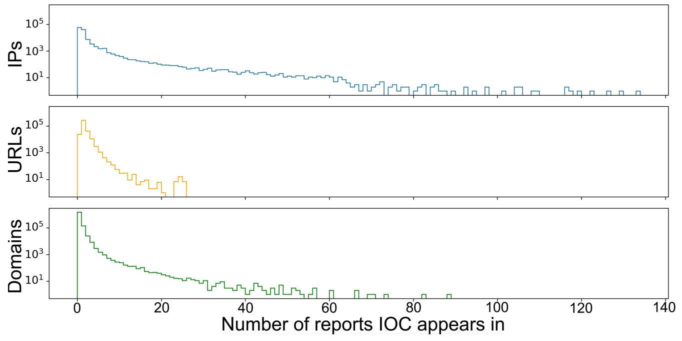


Fig. 4: IOC Reuse by IOC type.

features. Thus, to answer the research question, we apply traditional ML techniques to IOC features. In **RQ2** we ask to what extent resource reuse can be used to attribute attacks to APTs. To measure this, we look to the graph structure of the TKG. We will show that resource reuse manifests in the graph as paths between events, where shorter paths entail more direct resource reuse. Graph neural networks incorporate both statistical analysis of IOC features and graph structural analysis of the TKG. So, a GNN's effectiveness at positively attributing events will help answer both research questions. If the GNN is effective, it demonstrates both IOC similarities *and* resource reuse between attacks. In this section, we will describe the techniques used to analyze the TRAIL dataset: traditional ML, graph structural analysis, and graph neural network analysis. Each approach takes a list of IOCs, or a subgraph of the TKG as input and aims to predict the most likely APT associated with the attack artifacts.

A. Traditional ML

We explored three traditional classifiers: Neural Networks (NN), XGBoost (XGB) [37], and Random Forest classifiers (RF) [38]. We selected these three classifiers because they are adaptable to many classification problems. To strengthen our model's robustness and curtail overfitting, we employed a 5-fold cross-validation strategy. This accentuates the model's robustness and generalizability.

Preprocessing: For preprocessing, we first addressed the issue of imbalanced data. Specifically, for domains, URLs, and IPs, we applied the SMOTE [39] technique for over-sampling. Additionally, for domains, we engineered a new feature named '$active\_period$', which was the difference between the '$last\_seen$' and '$first\_seen$' timestamps. The final preprocessing step was to ensure all features underwent standard scaling. Using the training set as a basis, we find the mean and standard deviation, and rescale all of the data such that the mean is 0 and the variance is 1.

Neural Network: The neural network architecture uses an input layer of 2048 neurons, and four hidden layers with 1024, 512, 128, and 64 neurons, respectively. Between each layer, there is an ReLU activation and a batch normalization layer. Additionally, there is a dropout layer with a rate of 50% in the first three hidden layers for regularization. The output layer has a neuron count equivalent to the number of classes, 22, and uses softmax activation. The model is optimized using cross-entropy loss.

XGBoost and Random Forest For the XGBoost classifier, we selected the multiclass soft-probability learning objective. For both XGBoost and Random Forest, the hyperparameters were optimized using the Tree of Parzen Estimators (TPE) method provided by Hyperopt [40].

B. Label Propagation

As has been shown by [11], APTs will often reuse difficult-to-recreate pieces of infrastructure such as Command and Control (C2) servers. Graph traversal analyzes the way IOCs overlap between cyberattacks as APTs reuse resources and attempt to use these overlaps to attribute events.

Formally, a graph, $\mathcal{G}$, is an ordered pair of sets, $\{\mathcal{V}, \mathcal{E}\}$, such that the set of edges, $\mathcal{E}$ is a subset of $\mathcal{V}^{(2)}$, pairs of nodes in $\mathcal{V}$. An attributed graph is a graph with a function $f : \mathcal{V} \rightarrow \mathbb{R}^d$, so each node is associated with a vector of attributes, or node features. We represent the IOCs as a graph of related nodes, where some subset of those nodes are attributed events.

For a given event node e if a path exists, $\mathcal{P} = \{e, n_1, ..., n_k\}$ where $(e, n_1) \in \mathcal{E}$ and $\forall\ 0 < i < k$, $(n_i, n_{i+1}) \in \mathcal{E}$, we assume that each node in the path is related to the APT associated with e. Nodes one-hop away from e are suspicious by definition, whereas those more distant from e are less suspicious. Our approach aims to capture each node's level of association with each APT by capturing its average distance from each event node. We accomplish this using label propagation [41].

Formally, this method can be described as follows: let $\ell : E \rightarrow \mathcal{A}$ be a function that maps some subset $E \subseteq \mathcal{N}$ to a set of labels $\mathcal{A} = \{a_1, ..., a_m\}$. In this case, E is the set of event nodes, and $\mathcal{A}$ is the set of APTs they are associated with. Let $\mathbf{F}_0 = |\mathcal{N}| \times |\mathcal{A}|$, be a one-hot encoded matrix of labels, where rows corresponding with events in E are labeled, and others are zero. Let $\mathbf{A}$ be the adjacency matrix of the graph. Then the n^{th} iteration of label propagation can be described recursively as

$$\mathbf{F}_n = \mathbf{D}^{\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}\mathbf{F}_{n-1} \tag{1}$$

where $\mathbf{D}$ is the diagonal matrix of the degree of each node. In practice, this is accomplished using message passing over an edge index [42]. Let $\mathbf{Z} = \mathbf{F}_N$, the output of the final iteration of Equation 1. To convert this into a probability distribution, we calculate the softmax of each nonzero row. This produces node representations that contain information about each node's distance to a given event node, as well as how many paths exist to each event node.

A significant limitation of label propagation for attribution is that if an event contains IOCs that have never been observed before, the labeled events in the TKG are unreachable. Consequently, these nodes remain unattributed. Label propagation provides several advantages over traditional ML approaches, however. It is inductive, meaning it can propagate labels that it has never seen before, unlike ML approaches which can only predict classes they have been trained to predict. Additionally, label propagation automatically ignores low-information IOCs. For example, if an APT uses a common public IP address, this may have paths to several unrelated events. This means the node has very little information to propagate; the more unrelated events connected to this node, the closer the label it propagates is to a uniform distribution, which in turn lowers the signal for noisy, unrelated labels.

C. Graph Neural Networks

To incorporate the benefits of both aforementioned methods, we use GNNs [27]. GNNs are a natural extension of the label propagation method described in the previous subsection. Rather than using just relational information about nodes, we also provide the model with feature information, $\mathbf{X}$. This information is passed through a learnable non-linear function. Mathematically, this is expressed by changing Equation 1 to

$$\mathbf{H}^{(\ell)} = \sigma\left(\mathbf{D}^{\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}\mathbf{H}^{(\ell-1)}\mathbf{W}^{(\ell)} + \mathbf{b}^{(\ell)}\right) \tag{2}$$

where $\mathbf{H}^0 = \mathbf{X}$, $\sigma(\cdot)$ is a nonlinearity function such as ReLU, and the variables $\mathbf{W}^\ell$ and $\mathbf{b}^\ell$ are trainable parameters. This is the characteristic equation for graph convolutional networks [43], which require the entire graph to be held in memory, and to calculate representations for each node. However, due to the size of the TKG, it is more efficient to use GraphSAGE [44]. Rather than computing Equation 2 for every node, GraphSAGE only computes it for nodes of interest. Formally, the new equation is defined as

$$\mathbf{h}_v^{(\ell)} = \sigma\left(\frac{1}{|\mathcal{N}(v)|}\sum_{n\in\mathcal{N}(v)} \mathbf{h}_n^{(\ell-1)}\mathbf{W}^{(\ell)} + \mathbf{b}^{(\ell)}\right) \tag{3}$$

where $\mathcal{N}(v)$ represents the one-hop neighbors of node v. To improve stability, after the aggregation step in Equation 3, node representations are also L_2 normalized,

$$\frac{\mathbf{h}_v^{(\ell)}}{\|\mathbf{h}_v^{(\ell)}\|_2}. \tag{4}$$

In this work, we implement a GraphSAGE model with output dimension $|\mathcal{A}|$. We train it to classify a subset of unlabeled event nodes E using their k-hop neighborhoods as input. The output forms a probability distribution function of the APT that each event is likely associated with. To this end, we optimize the model using cross-entropy loss on the softmax of its final layer.

From the success of previous GNN-based IOC analytic techniques such as [45], we expect this to capture useful information about IOCs, but there is an issue: each kind of IOC has a different dimensionality. URLs, IPs, and domains are represented using 1,516, 506, and 115 features respectively. One solution to this is to simply pass the features through linear layers, and project them all into the same dimensions, but this is suboptimal. To address this problem, we use autoencoders

(AEs) [46], which project IOC features into low-dimensional space for easier processing, then attempt to reconstruct the original inputs, ensuring information is preserved during the encoding process. This requires training two separate models: an encoder, $f(\cdot)$ and a decoder $g(\cdot)$, both implemented with a two-layer feed-forward neural network. Both have hidden layers with 512 dimensions, and the encoding dimension is 64.

To train these additional networks, we keep track of an additional reconstruction loss function:

$$\mathcal{L}_R = \|\mathbf{X} - g(f(\mathbf{X};\theta_f);\theta_g)\|^2. \tag{5}$$

We create three separate encoder-decoder modules to independently project URLs, IPs, and domains to equal-dimension vectors that are fed into the GraphSAGE module.

VII. APT Attribution Results

A. Individual IOC Attribution

We have endeavored to capture more than simplistic lexical features in our dataset creation. Our choice of features aims to capture adversary behavior–what kind of servers they use, their IP address providers, domain registration patterns, and more. Although IOCs are transient, APT behaviors remain more consistent, offering a robust basis for attribution [30]. To answer **RQ1**, the efficacy of our models will measure how well the features we selected reflect APT behaviors, and how consistent APT behaviors are, if at all.

Experimental setup. In our experiments, we measured the performance of XGB, RF, and NNs for IOC attribution. Given the distinct feature set for each IOC type—domains, IPs, and URLs—we trained each model individually. We limited training to first-order IOCs linked solely to one threat actor, ensuring clear labels, and excluding multi-labeled IOCs. This also ensured only event-related IOCs were included, avoiding benign secondary IOCs (e.g., nameservers).

Table III presents the average results from five-fold cross-validation for each model across each type of IOC. We evaluate the models using the metrics accuracy (Acc.) and balanced accuracy (B-acc.), the latter being especially relevant given the imbalanced nature of our dataset.

Observation #1: ***While individual IOCs can be attributed at a rate better than random, they yield limited information on their own.***

We observe that the NN displayed the best performance in the attribution of IPs, with an accuracy of 37.96%. This suggests that the inherent non-linearity of NNs might be more adept at capturing patterns within the sparse IP features.

However, when attributing URLs and domains, the NN's performance diminishes compared to the other methods. XGB and RF, while trailing the NN in IP attribution, performed competitively in URL and domain attribution. Specifically, XGB achieved a 45.90% accuracy for URLs, outperforming both the NN and RF. Interestingly, URLs emerged as the most accurately classified IOC. This could be attributed to their high degree of features, making them the most descriptive IOC type. Additionally, as suggested by Villalón-Huerta et al. [30], APT behavior is a better indicator than atomic IOCs alone. URLs carry a high degree of information about attacker tools, and reveal information about their server, its operating system, and the file encoding methods–essentially capturing APT behavior.

We must concede that for each IOC, the balanced accuracy was low. Even for URLs, it was just over 25%–not high enough for reliable use. However, while the attribution of individual IOCs offers some insights into attackers' patterns, attacks are represented by large groups of IOCs, not individual ones. Considering IOCs in isolation overlooks important correlations between co-occurring IOCs, or important links to other observed attacks.

TABLE III: Individual IOC attribution for different models (Average over 5-folds)

Model	IP		URL		Domain	
	Acc.	B-acc.	Acc.	B-acc.	Acc.	B-acc.
XGB	0.3174	0.1975	**0.4590**	**0.2531**	**0.2894**	**0.1609**
NN	**0.3796**	**0.2260**	0.3395	0.1742	0.1087	0.1004
RF	0.2431	0.1708	0.3419	0.2193	0.1297	0.1248

B. Event Attribution

In the second research question, **RQ2**, we proposed concerned using the groups of IOCs reported in events to attribute the overall incident. In this section, we will show how we do this by formulating event attribution as a node classification problem within a graph. Here, we will describe how we used traditional machine learning, graph traversal, and graph neural networks to predict which APT was attributed to a cyberattack given the IOCs observed in that campaign, and the edges in the TKG.

Experimental Setup. We train each model using stratified k-fold validation on disjoint subgraphs of the TKG. To evaluate traditional ML models, we classify each IOC in an event individually. Then, the mode of their predictions is used as the final output. For graph traversal, we masked out the labels for $\frac{1}{k}$ of the events in the graph. Then we iteratively perform label propagation as described in Equation 1. We report the results of 2, 3, and 4 layers of label propagation, which are represented as LP 2L/ 3L/ 4L respectively in Table IV. We tested up to 10 layers, but the results did not improve after 4 propagation iterations.

For GNNs, the procedure is similar. The training set is split into a training and validation set, and the model is optimized on how well it can predict the labels of events in the training set with the knowledge of the labels in the validation set. During validation, the event nodes in the training set are given labels, and the validation nodes labels are masked. Finally, during testing, only the event nodes in the test set have their features masked. The hyperparameters selected are those that had the best results on the validation sets before considering the test sets. Each GNN is trained with a learning rate of

TABLE IV: Event attribution accuracy of various approaches (Average over 5-folds)

Model	Acc	B-Acc.
XGB	0.4663 $\pm$ 0.0055	0.2911 $\pm$ 0.0087
NN	0.2622 $\pm$ 0.0095	0.1617 $\pm$ 0.0097
RF	0.6878 $\pm$ 0.0068	0.5491 $\pm$ 0.0061
LP 2L	0.7589 $\pm$ 0.0059	0.7434 $\pm$ 0.0061
LP 3L	0.7934 $\pm$ 0.0053	0.7660 $\pm$ 0.0054
LP 4L	0.8236 $\pm$ 0.0061	0.7734 $\pm$ 0.0057
GNN 2L	0.8338 $\pm$ 0.0079	0.7793 $\pm$ 0.0086
GNN 3L	0.8396 $\pm$ 0.0101	0.7860 $\pm$ 0.0131
GNN 4L	**0.8405** $\pm$ 0.0113	**0.7922** $\pm$ 0.0098

0.0001, has 512 hidden dimensions, and uses 64-dimensional encodings for the IOC input features. All models are trained and evaluated on the same data split.

Observation #2: ***The graph structure is crucial for attributing attack campaigns, and IOC features further improve attribution accuracy.***

Label propagation achieved high metrics; however, it required traversing deeper into the graph than did GNNs to achieve the same level of accuracy. Note that LP 2L is a measure of direct resource reuse. The only two-hop paths that propagate labels are of the form $e_i \rightarrow$ IOC $\rightarrow e_j$. Results from any 2L model are equivalent to the results if we did not apply the extra enrichment process while collecting the data. As additional layers are added, the importance of secondary IOCs becomes more apparent. Paths between events with length greater than 2 necessitate the use of secondary IOCs. For example, paths like $e_i \rightarrow IP \rightarrow domain \rightarrow e_j$ are common, and only available because of secondary IOCs. The 2L LP model can also be used as a baseline to determine the difficulty of the dataset. Though it is not the only step a human analyst would take to attribute an event, direct resource reuse can be a high-confidence indicator used during the attribution process [31]. LP 4L is notably the only model that takes advantage of the ASN nodes in the graph. Any path through an ASN node that starts and ends with an event must be of the form, $e_i \rightarrow IP_n \rightarrow ASN \rightarrow IP_m \rightarrow e_j$, meaning it would take 4 propagation steps for the information to transmit. This extra possible path between events improves LP significantly, though it still is not as effective as even shallow GNNs.

GNN 2L, 3L, and 4L are analyzing the same subgraphs as LP 2L, 3L, and 4L respectively. The large increase in accuracy means the additional neural network parameters are learning from the features. We observe that even the 2L GNN strongly outperforms the deepest LP models. This indicates that the linear combination of resource reuse achievable by humans is not sufficient to classify more difficult events via their IOCs. These results show that nonlinear relationships between IOC features and the specific paths by which information flows between events are necessary information to consider for accurate attribution. These results also suggest that IOC features allow models to be more discerning of APT classes with less support in the dataset compared to label propagation, which is better suited to classify nodes that are part of larger structures within the graph.

C. Case Study

To further illustrate the utility of the TRAIL knowledge graph and the various ways it can be analyzed, we consider a case study: attempting to attribute a cyberattack from the wild. Using an attributed AlienVault report that was created on Sept. 28th, 2023, four months after the creation of the TKG, we investigate the performance of our graph-based attribution methods. The report describes an APT38 phishing campaign with 10 associated IPs and 10 domains.[3]. After enriching the report, we found 19 additional IP addresses and 2,629 domains from passive DNS. This brings the total count of IOCs from 20 to 2,668. After the new report is merged into the TKG, its 2-hop neighborhood contains 9,405 total IOCs.

Model Performance. After the event is merged into the TKG, there are 14 attributed events 2-hops away and 24 attributed events 3-hops away. Using label propagation, we can easily attribute this event to APT38, as all its neighbors are also APT38 events. However, in a more realistic setting, we may not know its neighbors' labels either; we would simply understand that these events are part of a related, but as of yet unattributed campaign. So instead, we assume that the labels of these events are unknown, and we only have the features and the enriched subgraph. Without any knowledge of the neighbors' labels, a GNN trained on the full TKG correctly predicted the event should be attributed to APT38 with 48% confidence. Allowing the model to see neighbors' labels increases this to 88% confidence. These results further confirm the utility of the TKG.

Observation #3: ***IOCs and their features, especially when viewed as a group in the knowledge graph, describe APT behavior well enough to be used for attribution.***

Figure 5 shows the 2-hop subgraph of where this new event intersects with the existing TKG. 40% of the domains and

[3]https://otx.alienvault.com/pulse/651a6bf5c2ef0eb8b7bda145

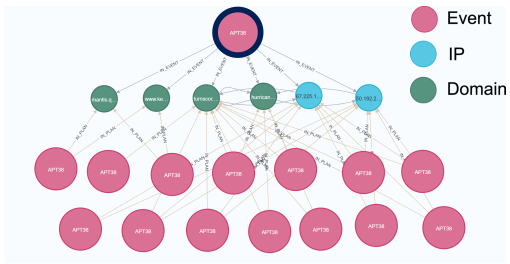


Fig. 5: All events 2-hops from the new node. Note: every event pictured is attributed to APT38 (new event outlined in black)

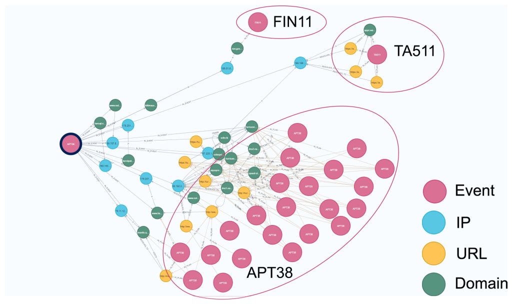


Fig. 6: All events 3-hops away from the new node (new event outlined in black)

20% of the IPs were reused in other APT38 events. Zooming out to 3 hops away in Figure 6, we observe 22 APT38 events compared to just a single FIN11 event, and a single TA511 event. These other events are both linked by a single IP address that resolved at one point to URLs and domains involved with their respective attacks. The APT38 events, on the other hand, have far more direct links, either because URLs hosted on the same domains seen in this new attack were previously used, or because they too shared an IP address. We are confident this resource reuse points to a likely APT38 attribution.

In the threat report, the analysts cite the initial method of compromise–phishing over LinkedIn (which was also the method of compromise in many of the 2-hop events in the TKG)–and specifics about the binary files used by the adversary as their method of attribution. They mention that the network IOCs are compromised, legitimate servers, and refer to this as a "typical, yet weak-confidence behavior, of Lazarus [(APT38)]" [47]. They do not mention, however, that these IOCs were reused in other attacks as part of a larger campaign called "Operation DreamJob" [48]. With our method, if any one event in the operation were attributed, either by malware analysis as was done by this report, or by our methods, we can immediately spread this attribution through the entire TKG, attributing the cluster of events that represented the campaign.

Months-long Investigation In the entire month of June, 2023, there were 22 unique reports attributed to an APT we tracked in the TKG. Without retraining, or updating the TKG, we evaluate the GNN model on these unseen reports. We show the confusion matrix of the results in Figure 7. There were 10 reports attributed to APT38, 6 attributed to APT37, 5 attributed to KIMSUKY, and 1 attributed to APT27. The GNN model accurately attributed 80% of the APT38 events, and 80% of the KIMSUKY events, however it struggled to correctly attribute the other groups. The only APT27 event was confused with another Chinese APT group, though their behaviors and targets are dissimilar. All 6 APT37 events were misclassified as other North Korean groups. However, this may be because North Korean groups have such overlap that they often all reported as Lazarus (APT38) [49], which accounts for 4/6 of the misclassifications. For this reason, we are confident in our model's ability to correctly attribute novel events with reasonable accuracy. We also note that the confidence level

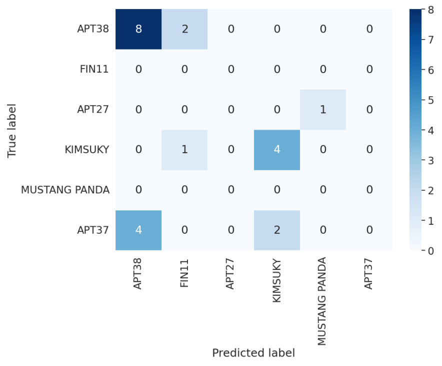


Fig. 7: Confusion matrix for the GNN models attributing new events for the month of June 2023.

for the false positives was always less than 0.8, while the true positives were always > 0.99. Not attributing events unless the model's confidence surpasses some threshold would improve the models' rate of misclassification, and make it more robust to new APTs it was never trained on. We leave this as a topic for future work.

We further investigate the performance of our model for the remainder of the year 2023. We first measure it by independently evaluating all reports from each month using the same pre-trained model from the previous section. As is evident from the growing gap in evaluation metrics present in Figure 8, the more out-of-date the model is, the more it benefits from retraining. At the same time, we also evaluate a GNN that was iteratively fine-tuned using data from the previous month to retrain before evaluating on the next month. The gulf between the performance of an up-to-date model and the old model increased by 3.5% for each additional month that passed. Clearly in a realistic setting, the GNN should be retrained frequently to avoid this drift in quality. In our experiments, training the GNN from scratch on only CPUs took an hour. Fine-tuning it with only the previous month's events took less than five minutes (<10 epochs before convergence), so the cost is very low.

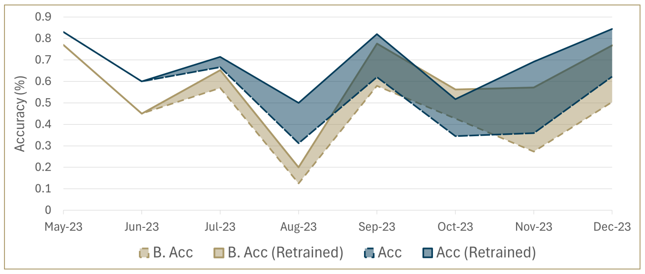


Fig. 8: Performance degradation as TKG and models become more out-of-date. Lower, dotted lines represent (balanced) accuracy without retraining or updating TKG; upper, solid lines represent (balanced) accuracy after retraining and updating the TKG on the previous month's data.

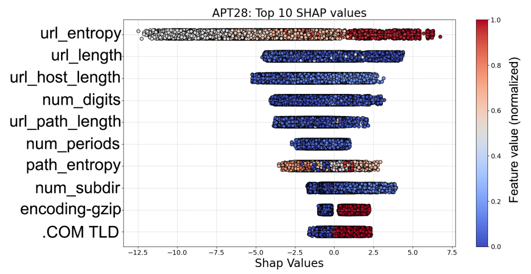


Fig. 9: Beeswarm plot of the top 10 most impactful features in the XGB URL classifier for APT28. Each point represents the SHAP values for a specific sample's feature. The further a datapoint is to the right, the more it is positively associated with that APT. The color intensity reflects the value of the normalized feature.

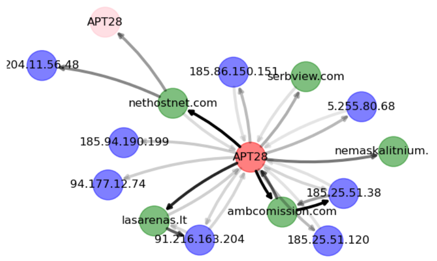


Fig. 10: Subgraph identified by GNNExplainer of the top 15 most important nodes used to correctly classify an APT28 event (the red node in the middle). The darkness of edge weights indicates each edge's importance in the classification.

D. Model Explainability

The SHapley Additive exPlanations (SHAP) framework [50] is a method to explain which features have the strongest effect on a machine learning model's prediction. The traditional machine learning models that we used for individual IOC attribution can use this technique to generate precise signatures that uniquely characterize individual IOCs that specific threat groups might deploy. As an example, Figure 9 depicts the top 10 most important features for the threat actor class APT28, derived from our XGB URL classifier.

We observe that APT28 activity has higher levels of URL entropy, and those URLs host gzip-encoded files. High entropy in URLs might indicate obfuscation techniques or attempts to hide malicious intent, making automated detection more challenging. It could also imply that the reported URLs were used to host files, whose names were similarly obfuscated, or randomized. These observations agree with the known techniques, tactics, and procedures of APT28 [51], [52], [53]. The gzip-encoded files hosted at these URLs could be an attempt to evade standard detection mechanisms that monitor file content. The use of gzip specifically hints that they prefer GNU-based operating systems for their infrastructure.

For the graph-based models, we can use explanation models to identify the most important IOCs involved with a given event. We use GNNExplainer [54] on a pretrained 3-layer GNN to find the subgraph that contributed most to the model's prediction. Figure 10 illustrates one such subgraph for an APT28 event. We observe several IP addresses (blue) surrounding the event being classified (red). Upon further examination, these IP addresses are mostly Latvian, and host links to phishing websites and malware droppers. The important domains (green) are similarly flagged as malicious. One domain, `nethostnet[.]com`, was reused by a different campaign, as it creates a path to another APT28 event (pink).

Though there does exist an important path between two APT28 events, it is significant to note that the vast majority of important edges are between the event and individual IOCs, meaning resource reuse is not the primary method by which the GNN is making its predictions. Rather, the features of IOC nodes, such as their country of origin, or DNS registry history seem to make a larger impact on how the model makes its predictions. Algorithms like GNNExplainer allow analysts to quickly identify the most significant IOCs used in the model's prediction. Even if the prediction is wrong, analysts may still use the IOCs identified as important to continue their search. This aspect of the TRAIL system could allow for faster, and more robust attribution by humans, as it identifies important IOCs that as a group, categorize the behavior of an adversary.

VIII. Related Work

Prior, non-technical works have said, "attribution is an art: no purely technical routine, simple or complex, can formalize, calculate, quantify, or fully automate attribution." [55, p. 30]. Such works generally dissuade analysts from relying on statistical means for attribution [8]. Nonetheless, data scientists keep inventing them. In this section, we will discuss some of these related works and compare them to our approach.

Binary file attribution is a well-studied field wherein analysts attempt to attribute malware to the threat group that created it. Perhaps because there are so many available datasets [12], [25], [26] there are many approaches. Interestingly, older approaches attribute binaries to specific countries rather than APTs [14], but more recent approaches recognize that individual countries can contain multiple threat actors [10], [15], [16]. We were particularly influenced by [13]: their model, a fusion of SMOTE and RF, counteracts the challenges of data imbalance and multiclassification. We built upon this approach for IOC attribution using not just RF, but XGB, and NNs, which introduced an additional layer of depth, potentially capturing more intricate patterns in the data.

Non-binary IOC Analysis has had considerably less attention than malware attribution. Most related works analyzing this class of IOCs predict if individual domains or URLs were malicious or benign, rather than which APT produced them [17], [19], [20], [21], [22]. Notably, [56] and [57] utilize label propagation through IP and domain graphs, but their approaches classify individual IOCs as either malicious or benign. This makes it difficult to compare our approach to prior works as our dataset is comprised only of IOCs that have appeared in cyber-incidents, meaning we assume that they are all malicious. We were particularly influenced by the method used by Thin [18] to generate features for domains. They also supplement lexical features with counts of DNS records, though they classify the domains as malicious or benign. The most similar approach we found to our own was that of Kazato et al. [45]. They also built a knowledge graph from OSINT sources; however, they lack the notion of an event node. They also analyzed their graph with a GNN; however, it was to predict if IOCs are malicious or not. It would be difficult to directly compare our approach to theirs, not just because they use an entirely different graph schema, but because they label IOCs as malicious or benign rather than the events that used them. We feel our work builds on theirs, as our graph has 100x more nodes, and our analysis is more granular.

IOC Graphs have been gaining traction lately as a tool to understand attacker behavior. Pelofske et al. [11] have the most expansive one, with node types for hashes, CVEs, usernames, and more in addition to IPs and domains, all collected from OSINT sources. Notably, they also use the concept of event nodes, though they do not attribute them to APTs. They analyze resource reuse and the cliques that form around attack infrastructure, however, they ignore the relations between IOCs themselves, focusing only on first-order IOCs and edges from them to events. Similarly, Ren et al. [58] crawled OSINT sources to construct a graph of higher-level descriptors, relating attacks to victims and the infrastructure used. They did not use this for automated attribution; rather they used their knowledge graph to generate human-readable descriptors of attacker behavior and to summarize their activities. Sebastián & Caballero utilize a similar IOC graph to detect malicious developers attempting ban evasion in mobile app markets [59]. Using a pruned IOC graph, their attribution is based on path detection between developers in the IOC graph. Other graph-based approaches, such as [60] and [61] use more behavioral descriptors for their relations and relate APTs to each other rather than to individual events, though neither approach attempts to use the knowledge graphs they generate for automated attribution.

IX. Discussion

Limitations: Though we have demonstrated that analysis of the TKG can aid in attributing cyberattacks, our method does have several limitations. For example, the GNN and traditional ML models can only attribute events to one of the 22 APTs they were trained to classify. If they observe data from unknown APTs or non-APT malware, it would incorrectly attribute the event to one of the APTs it was trained to classify. It would require retraining for these models to classify new APTs. While this is a fundamental flaw in supervised learning approaches, there exist potential solutions in confidence-based thresholding or unsupervised approaches. For example, label propagation does not have this issue, as it is non-parametric. This means if labeled data from an unseen APT is added to the TKG, no retraining is required for label propagation to classify a future, unlabeled event with paths to this new training data. Additionally, our experiments and prior works [11] showed that IOCs from the same APTs tend to form dense cliques in IOC graphs. This means it is unlikely that a short path would exist from a novel APT event to any labeled event in the TKG. The label propagation model would output a low confidence score, or no score at all for the new data. Implementing a thresholding system, where low-confidence predictions are classified as "unknown" or "out of distribution," could account for this issue. Supervised and unsupervised models could be used. We leave this as a subject for future work

Another limitation on this study is data quality. For example, there were many erroneous "URLs" in our dataset that were actually javascript snippets that likely matched a regex pattern and made their way into automated report generators. We have endeavored to remove these. Still, incorrect associations may be present in the dataset. It is also important to note that attribution is rarely done with complete confidence [7]; however, for the purposes of our study, user-assigned labels for an event's threat actor are assumed to be ground truth.

Future Work: Including other node types, like hashes, in the TKG could capture many more important relationships between CTI reports. Investigating alternative labels is another promising research direction. This work focused on classifying APTs, but other categorization methods, such as classifying by malicious versus benign, by country, or by motivation may offer valuable insights. Finally, zero-shot inference to detect novel APTs or benign indicators, either using the model's confidence score or through other means, would increase robustness and usability in more realistic scenarios.

X. Conclusion

In this work, we created and present for others to analyze the TRAIL Knowledge Graph. We collected co-occurring IOCs from OSINT sources to act as ego-nets centered around a node labeled with a threat actor that represents a cybersecurity event. To find higher-order connections in the knowledge graph we further enrich these IOCs, and through this process connected 99.94% of the IOCs in one connected component. Our analysis showed that using IOC features alone, individual artifacts of cyberattacks could be attributed to their associated threat actor with reasonable accuracy. When analyzed as groups, using just the topology of the TKG, we could attribute unlabeled cyber-attacks with 82% accuracy. Incorporating both features and topology improved attribution accuracy to 85%. Our results would suggest that though they try to hide their tracks, when APTs attack, they often leave a TRAIL.

Acknowledgements

This work was supported in part by DARPA under agreement number HR001122C0153. The views, opinions, and/or findings expressed in this material are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Distribution Statement A (Approved for Public Release, Distribution Unlimited).

References

[1] S. Oladimeji and S. M. Kerner, "Solarwinds hack explained: Everything you need to know," *TechTarget*, June 2023. [Online]. Available: https://www.techtarget.com/whatis/feature/SolarWinds-hack-explained-Everything-you-need-to-know

[2] N. Bertrand, "Biden taps intelligence veteran for new white house cybersecurity role," *Politico*, Jan 2021. [Online]. Available: https://www.politico.com/news/2021/01/06/biden-white-house-cybersecurity-neuberger-455508

[3] United States Whitehouse, "Fact sheet: Imposing costs for harmful foreign activities by the russian government," *White House Statements and Releases*, April 2021. [Online]. Available: https://www.whitehouse.gov/briefing-room/statements-releases/2021/04/15/fact-sheet-imposing-costs-for-harmful-foreign-activities-by-the-russian-government/

[4] Mandiant, "Unc2452 merged into apt29: Russia-based espionage group." [Online]. Available: https://www.mandiant.com/resources/blog/unc2452-merged-into-apt29

[5] United States Whitehouse, "National security strategy of the united states of america," *The National Archives*, Dec 2017. [Online]. Available: https://trumpwhitehouse.archives.gov/wp-content/uploads/2017/12/NSS-Final-12-18-2017-0905.pdf

[6] P. W. Singer and A. Friedman, *Cybersecurity: What everyone needs to know*. oup usa, 2014, ch. Whodunit? The Problem of Attribution.

[7] D. Tran, "The law of attribution: Rules for attribution the source of a cyber-attack," *Yale JL & Tech.*, vol. 20, p. 376, 2018.

[8] N. Tsagourias, "Cyber attacks, self-defence and the problem of attribution," *Journal of conflict and security law*, vol. 17, no. 2, pp. 229–244, 2012.

[9] F. Skopik and T. Pahi, "Under false flag: Using technical artifacts for cyber attack attribution," *Cybersecurity*, vol. 3, pp. 1–20, 2020.

[10] M. Kida and O. Olukoya, "Nation-state threat actor attribution using fuzzy hashing," *IEEE Access*, 2022.

[11] E. Pelofske, L. M. Liebrock, and V. Urias, "Cybersecurity threat hunting and vulnerability analysis using a neo4j graph database of open source intelligence," *arXiv preprint arXiv:2301.12013*, 2023.

[12] H. Haddadpajouh, A. Azmoodeh, A. Dehghantanha, and R. M. Parizi, "Mvfcc: A multi-view fuzzy consensus clustering model for malware threat attribution," *IEEE Access*, vol. 8, pp. 139 188–139 198, 2020.

[13] S. Li, Q. Zhang, X. Wu, W. Han, and Z. Tian, "Attribution classification method of apt malware in iot using machine learning techniques," *Security and Communication Networks*, vol. 2021, pp. 1–12, 2021.

[14] I. Rosenberg, G. Sicard, and E. David, "Deepapt: nation-state apt attribution using end-to-end deep neural networks," in *Artificial Neural Networks and Machine Learning–ICANN 2017: 26th International Conference on Artificial Neural Networks, Alghero, Italy, September 11-14, 2017, Proceedings, Part II 26*. Springer, 2017, pp. 91–99.

[15] Y. Shin, K. Kim, J. J. Lee, and K. Lee, "Art: automated reclassification for threat actors based on att&ck matrix similarity," in *2021 world automation congress (WAC)*. IEEE, 2021, pp. 15–20.

[16] Q. Wang, H. Yan, and Z. Han, "Explainable apt attribution for malware using nlp techniques," in *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, 2021, pp. 70–80.

[17] Z. Liu, Y. Zeng, P. Zhang, J. Xue, J. Zhang, and J. Liu, "An imbalanced malicious domains detection method based on passive dns traffic analysis," *Security and Communication Networks*, vol. 2018, 2018.

[18] T. T. Thin, "Malicious domain detection based on decision tree," *Proceedings of the Computer Security Symposium*, pp. 17–20, 2020.

[19] S. Wang, B. Lang, N. Xiao, and Y. Chen, "Aspioc: Aspect-enhanced deep neural network for actionable indicator of compromise recognition," in *Information Security: 25th International Conference, ISC 2022, Bali, Indonesia, December 18–22, 2022, Proceedings*. Springer, 2022, pp. 411–421.

[20] A. Das, A. Das, A. Datta, S. Si, and S. Barman, "Deep approaches on malicious url classification," in *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2020, pp. 1–6.

[21] B. Janet, R. J. A. Kumar *et al.*, "Malicious url detection: a comparative study," in *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*. IEEE, 2021, pp. 1147–1151.

[22] R. Verma and A. Das, "What's in a url: Fast feature extraction and malicious url detection," in *Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics*, 2017, pp. 55–63.

[23] "VirusTotal - stats," https://www.virustotal.com/gui/stats, accessed on 07/03/2023.

[24] "Alienvault - open threat exchange," https://otx.alienvault.com/browse/global/indicators, 2023, (Accessed on 07/03/2023).

[25] G. Laurenza, R. Lazzeretti, and L. Mazzotti, "Malware triage for early identification of advanced persistent threat activities," *Digital Threats: Research and Practice*, vol. 1, no. 3, pp. 1–17, 2020.

[26] L. F. M. Liras, A. R. de Soto, and M. A. Prada, "Feature analysis for data-driven apt-related malware discrimination," *Computers & Security*, vol. 104, p. 102202, 2021.

[27] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International conference on machine learning*. PMLR, 2017, pp. 1263–1272.

[28] C. Harrington, "Sharing indicators of compromise: An overview of standards and formats," *EMC Critical Incident Response Center*, 2013.

[29] E. M. Hutchins, M. J. Cloppert, R. M. Amin *et al.*, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Leading Issues in Information Warfare & Security Research*, vol. 1, no. 1, p. 80, 2011.

[30] A. Villalón-Huerta, I. Ripoll-Ripoll, and H. Marco-Gisbert, "Key requirements for the detection and sharing of behavioral indicators of compromise," *Electronics*, vol. 11, no. 3, p. 416, 2022.

[31] W. Tounsi and H. Rais, "A survey on technical threat intelligence in the age of sophisticated cyber attacks," *Computers & security*, vol. 72, pp. 212–233, 2018.

[32] S. Murdoch and N. Leaver, "Anonymity vs. trust in cyber-security collaboration," in *Proceedings of the 2nd ACM Workshop on Information Sharing and Collaborative Security*, 2015, pp. 27–29.

[33] D. Chismon and M. Ruks, "Threat intelligence: Collecting, analysing, evaluating," *MWR InfoSecurity Ltd*, pp. 1–36, 2015.

[34] "Neo4j graph database & analytics — graph database management system," https://neo4j.com/, 2023, (Accessed on 07/03/2023).

[35] M. Project, "MISP galaxy," https://misp-galaxy.org/, 2024, (Accessed on 03/21/2024).

[36] R. A. Hanneman and M. Riddle, *Introduction to social network methods*. University of California, Riverside: Riverside, CA, USA, 2005, ch. 9. Ego networks.

[37] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[38] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[39] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[40] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *International conference on machine learning*. PMLR, 2013, pp. 115–123.

[41] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," *Advances in neural information processing systems*, vol. 16, 2003.

[42] M. Fey and J. E. Lenssen, "Fast graph representation learning with pytorch geometric," *arXiv preprint arXiv:1903.02428*, 2019.

[43] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[44] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[45] Y. Kazato, Y. Nakagawa, and Y. Nakatani, "Improving maliciousness estimation of indicator of compromise using graph convolutional networks," in *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–7.

[46] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[47] P. Kálnai, "Lazarus luring employees with trojanized coding challenges: The case of a spanish aerospace company," WeLive Security, Threat Intelligence Report, Sept 2023, (Accessed on 10/09/2023). [Online]. Available: https://www.welivesecurity.com/en/eset-research/lazarus-luring-employees-trojanized-coding-challenges-case-spanish-aerospace-company/

[48] M. INTELLIGENCE and CONSULTING, "Stealing the lightshow (part one) — north korea's unc2970," Mandiant, Threat Intelligence Report,

Mar 2022, (Accessed on 10/09/2023). [Online]. Available: https://www.mandiant.com/resources/blog/lightshow-north-korea-unc2970

[49] "Lazarus Group," https://attack.mitre.org/versions/v14/groups/G0032/, accessed: 2024-03-20.

[50] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[51] M. ATT&CK, "Apt28: Techniques, tactics, and procedures," Available at https://attack.mitre.org/groups/G0007/ (2024/10/02).

[52] F. iSight Intelligence, "APT28: At the Center of the Storm," FireEye, Threat Intelligence Report, Jan. 2017. [Online]. Available: https://www.mandiant.com/sites/default/files/2021-09/APT28-Center-of-Storm-2017.pdf

[53] F. Hacquebord, "Two Years of Pawn Storm: Examining an Increasingly Relevant Threat," TrendMicro, Threat Intelligence Report, Apr. 2017.

[54] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," *Advances in neural information processing systems*, vol. 32, 2019.

[55] T. Rid and B. Buchanan, "Attributing cyber attacks," *Journal of strategic studies*, vol. 38, no. 1-2, pp. 4–37, 2015.

[56] I. M. Khalil, B. Guan, M. Nabeel, and T. Yu, "A domain is only as good as its buddies: Detecting stealthy malicious domains via graph inference," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, 2018, pp. 330–341.

[57] P. Manadhata, S. Yadav, P. Rao, and W. Horne, "Detecting malicious domains via graph inference," in *Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop*, 2014, pp. 59–60.

[58] Y. Ren, Y. Xiao, Y. Zhou, Z. Zhang, and Z. Tian, "Cskg4apt: A cybersecurity knowledge graph for advanced persistent threat organization attribution," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[59] S. Sebastián and J. Caballero, "Towards attribution in mobile markets: Identifying developer account polymorphism," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 771–785.

[60] E. K. J. Hooi, A. Zainal, M. A. Maarof, and M. N. Kassim, "Tagraph: Knowledge graph of threat actor," in *2019 International Conference on Cybersecurity (ICoCSec)*. IEEE, 2019, pp. 76–80.

[61] V. Mavroeidis, R. Hohimer, T. Casey, and A. Jesang, "Threat actor type inference and characterization within cyber threat intelligence," in *2021 13th International Conference on Cyber Conflict (CyCon)*. IEEE, 2021, pp. 327–352.