2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC) | 979-8-3503-3286-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/CCWC57344.2023.10099277

Activity-Attack Graphs for Intelligence-Informed Threat COA Development

Cole Mckee
United States Military Academy
West Point, NY, USA
cole.mckee@westpoint.edu

Kelsie Edie
United States Military Academy
West Point, NY, USA
kelsie.edie@westpoint.edu

Adam Duby
United States Military Academy
West Point, NY, USA
adam.duby@westpoint.edu

***Abstract*—A threat course of action (COA) describes the likely tactics, techniques, and procedures (TTPs) an adversary may deploy across the cyber kill-chain. Threat COA development and analysis informs hunt teams, incident responders, and threat emulation efforts on likely activities the adversary will conduct during an attack. In this paper, we propose a novel approach to generate and evaluate threat COAs through association rule mining. We identify frequent TTP itemsets to create a set of activity groups that describe associations between TTPs. We overlay activity groups to create a directed and edge-weighted activity-attack graph. The graphs hypothesize various adversary avenues of attack, and the weighted edges inform the analyst's trust of a hypothesized TTP in the COA. Our research identifies meaningful associations between TTPs and provides an analytical approach to generating threat COAs. Further, our implementation uses the STIX framework for extensibility and usability in a variety of threat intelligence environments.**

***Index Terms*—Cyber threat intelligence, attack graphs, threat analysis**

I. Introduction

As cyber attacks rapidly evolve and become increasingly complex, a high demand is placed on research to utilize data analytics that improve cyber threat intelligence (CTI). A critical component of CTI is developing and analyzing possible adversary courses of action (COA), which allows defenders to tailor defense designs and hunt for adversary activity.

This paper proposes a novel approach to develop, assess, and visualize threat COAs based on association rule mining on a dataset of attack techniques. Specifically, we identify sets of *activity groups*, which are weighted associations, or relations, between techniques used by APT groups to accomplish a tactical goal. An activity group between two sets of techniques, X and Y, is defined as an association rule of the form $X \rightarrow Y$. This informs the analyst that if X is realized in a threat COA, then Y can be expected based on the weight, or strength, of the association. Further, we identify activity groups at several layers of technique abstraction, depending on the analyst's requirements.

The views expressed in this paper are those of the authors and do not reflect the official policy or position of the United States Military Academy, the United States Army, the Department of Defense, or the United States Government.

We propose an algorithm that overlays the activity groups into an *activity-attack graph*, a directed edge-weighted graph that visualizes hypothesized threat COAs. Analysts seed the algorithm with known or interesting techniques, and it yields a graph that communicates hypothesized techniques.

Our approach adopts the data and conventions of MITRE's Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) framework [1]. This widely adopted repository contains real-world information on Advanced Persistent Threat (APT) groups, or adversarial threats, that carry out cyber attacks. ATT&CK defines common threat group names, adversary motivation, common software, and known techniques associated with each threat group. Techniques are methods (i.e. TTPs) deployed by an adversary to advance a tactical objective. They adopt a standard naming convention for consistency in intelligence reporting and analytics. For example, T1566 is the technique identifier for the use of phishing messages to gain access to a victim's machine.

Our implementation adopts the Structured Threat Information eXpression (STIX) framework [2] for communicating cyber threat intelligence. Standardization enhances the comprehensibility of the results and repeatability in threat analysis. Further, it enables analytical pivoting, which is an intelligence process that involves fusing data to gain new knowledge. STIX objects permit convenient consumption of our threat intelligence models into existing CTI processes.

In this paper, we make the following contributions:

- implement association rule mining to create activity groups at several layers of abstraction that guide COA development through learned relationships between ATT&CK techniques;
- propose an algorithm that overlays activity groups into an activity-attack graph (AAG) that visualizes threat COAs with their associated strengths;
- our data and implementation are made publicly available[1] for analyst use and future research.

The rest of this paper is organized as follows. Section II discusses the background and related work. Section III describes our approach. In Section IV, we present our results

[1]https://github.com/metalmulisha205/Intelligence-Informed-COA-Development

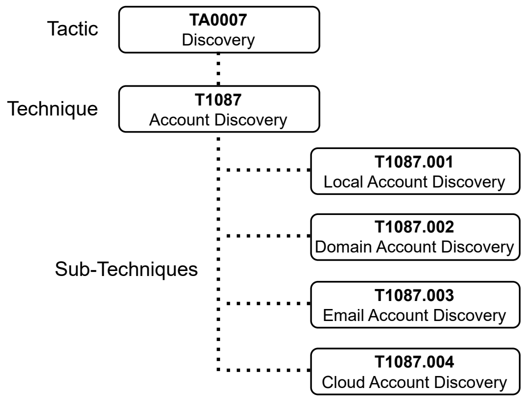

Fig. 1: Example of MITRE ATT&CK tactic, technique, and sub-technique relationship.

and discuss several case studies using our approach. Finally, we conclude and propose future research in section V.

II. Background and Related Work

A. Cyber Threat Intelligence

Cyber Threat Intelligence (CTI) is the process of obtaining and processing evidence-based knowledge about cyber threats [3]. A threat-informed defense has been shown effective at intercepting adversary campaigns [4], and provides situational awareness about threats to decision makers.

Most CTI models adopt the idea of an attack pattern, or a type of TTP that describes the resources an adversary uses to advance a tactical objective [3]. The most widely adopted representation of TTPs is from MITRE ATT&CK[2], a repository of cyber threat intelligence and a framework to describe and communicate cyber attacks across various stages of the cyber kill-chain [1]. For each known APT group, ATT&CK has a list of known techniques that are associated with the group. A technique describes how an adversary achieves a tactical goal. Within techniques, sub-techniques contain more specific information about how the technique is realized. For example, APT30 is known to use the sub-technique T1566.001 (Spearphishing Attachment), which is a sub-technique of T1566 (Spearphishing). An example of a tactic, technique, and sub-technique relationship is shown in Fig. 1.

The adoption of ATT&CK is prevalent in related CTI research. For example, Ahmed et al. showed how to assess risk and the probability of attack success based on known ATT&CK patterns [5]. It has also been shown that TTP-based hunting can be effective in threat detection [6]. Further, ATT&CK's ICS-focused CTI has enabled probabilistic attack pattern detection in industrial control systems [7].

Most related to our research, Al-Shaer et al. utilized hierarchical agglomerate clustering to identify relationships between ATT&CK techniques [8]. They clustered techniques together that had a tendency of co-occurring to assist analysts in detecting adversarial behavior using the learned relationships.

[2]https://attack.mitre.org/

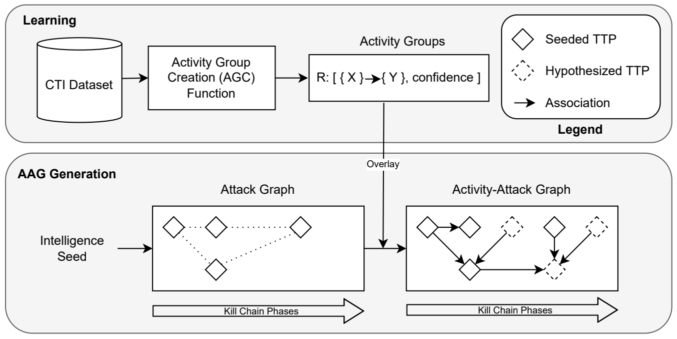

Fig. 2: Overview of approach to threat COA development.

B. The Diamond Model

The diamond model is a framework for describing adversary events as diamonds, which are 4-tuples that describe the adversary, victim, capability, and infrastructure [9]. The diamond model defines various means of grouping events using a graph-theoretic approach to communicate and visualize adversary activity. Specifically, it defines an attack graph as possible paths an adversary could take, and an activity thread as a graph representing events that have been known to happen. The overlay of activity threads with attack graphs is known as an *activity-attack graph*. Further, the diamond model defines an *activity group* as a set of associated events.

C. Motivation

A critical component to a robust CTI program is threat course of action (COA) development. A threat COA is an intelligence-informed enumeration of possible maneuvers an adversary can deploy against a defended asset. Threat COA development is used in incident response, threat emulation, and threat hunting.

If an incident responder observes evidence of a subset of techniques, then the analyst needs to have a general framework to search for additional activity to ensure the threat is contained and eradicated. Indeed, analysts can apply subject matter expertise and follow the evidence to continue the investigation, but some automation based on analytical intelligence can help reduce the search space in the absence of observed adversary behavior.

Our approach to automated threat COA development extends the concepts of diamond model activity groups and activity-attack graphs, using ATT&CK techniques as events, to develop intelligence-informed threat COAs. We use association rule mining and represent the learned rules as activity groups, and we overlay the groupings with known observed behavior and their potential mitigations to create activity-attack graphs that visualize a threat COA.

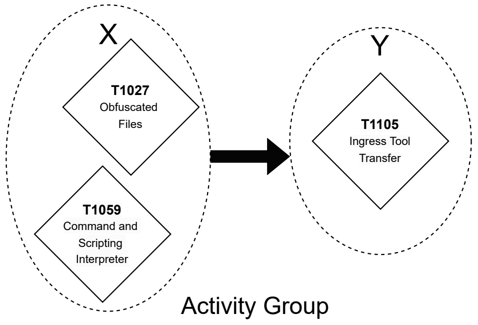

Fig. 3: Activity group sample. If T1027 and T1059 are observed, then T1105 is hypothesized.

III. Approach

In this section, we describe our approach to developing threat COAs. An overview is visualized in figure 2, and involves the following steps:

1) activity group creation (AGC) via association rule mining;
2) initialize the COA generation with an intelligence seed;
3) visualize the threat COA as an activity-attack graph.

A. Dataset

Our dataset contains 746 observations, each representing an attack campaign, threat group, or malware family. Each row is a list of ATT&CK techniques and sub-techniques associated with the observation. The average number of techniques in each observation is 14.7. Our dataset extends data from publicly available datasets,[3] including the ATT&CK framework. Table I contains a sample from the dataset used in this research.

B. Activity Group Creation (AGC) Function

We create activity groups by applying the apriori algorithm for association rule mining [10]. The approach finds combinations of frequent itemsets and associations using conditional probabilities. Formally, let T be the set of all known techniques in the ATT&CK framework. The activity group AG is an association between two sets of techniques, $AG : X \rightarrow Y$, where $x, y \in T$. The association is a representation of behavior Y, the consequent, occurring in a COA that exhibits behavior X, the antecedent. We refer to the consequent as a *hypothesized* technique. An example activity group is shown in Fig. 3.

We perform the AGC function on two variations of our dataset to create *specific* AGs and *abstracted* AGs. The specific AGs include all original sub-techniques, while the abstracted AGs were created after lifting the sub-techniques into their respective technique. As shown in Fig. 1, the sub-techniques of a given technique are semantically similar. As such, lifting the sub-techniques improves robustness in the AGs at the cost of specificity.

[3]https://github.com/tropChaud/Categorized-Adversary-TTPs

TABLE I: Dataset snippet.

ID	Techniques
1	T1204.002 T1021.004, T1102.002
2	T1484
3	T1068, T1553.005

C. Intelligence Seed

To generate an activity-attack graph, our algorithm requires the input of an initial intelligence seed. The seed S is a set containing at least one technique. Formally, $S = \{s : s \in T\}$, where $S \subseteq T$ and $S \neq \emptyset$. S initializes the generation of hypothesized techniques and relationships based on techniques of interest by the analyst. For example, if the analyst is developing COAs for a specific APT group, the system is seeded with the set of techniques known to be associated with the APT of interest. Our algorithm will generate an activity-attack graph that hypothesizes relationships between the seeded techniques and suggest additional techniques to consider in the threat COA. From an incident response perspective, the analyst can seed the algorithm using observed techniques to yield an activity-attack graph that hypothesizes additional techniques to investigate.

D. Activity-Attack Graph Generation

In this section, we present an algorithm to create an activity-attack graph that visually represents a threat COA. The graph distinguishes between seeded and hypothesized techniques, groups techniques based on their associated kill-chain tactic, and communicates relationships implied by the activity groups.

Algorithm 1 describes the graph generation routine. First, seed techniques are queued. Next, each technique in the queue is visited, dequeued, and added to a set of visited techniques. For each iteration of the queue, the set of visited techniques is checked against the set of activity groups to determine if all of the antecedents for any activity group are met. If the antecedent techniques for an activity group are present, the consequent techniques of the activity group are added to the queue and the implied relationship is recorded. This continues until all present relationships are exhausted and the result is a set of seed techniques, hypothesized techniques, and hypothesized relationships.

Finally, the results of our algorithm are used to produce an activity-attack graph such as the one in Fig. 2. Association arrows are drawn from the antecedent to the consequent and labeled with the confidence of the corresponding attack group that the relationship satisfies. It is important to note that the directionality of the edges in the graph do not imply a causal or temporal relationship. The categorization of techniques into their tactic (i.e., kill-chain phase) generally captures the casual and temporal progression of the attack. The directionality $\{x \rightarrow y\}$ in our graph implies that if x is accepted in the threat COA, then y should be considered based on the strength of the association. The metrics are for evaluating the strength and confidence of these associations is discussed in section IV-A.

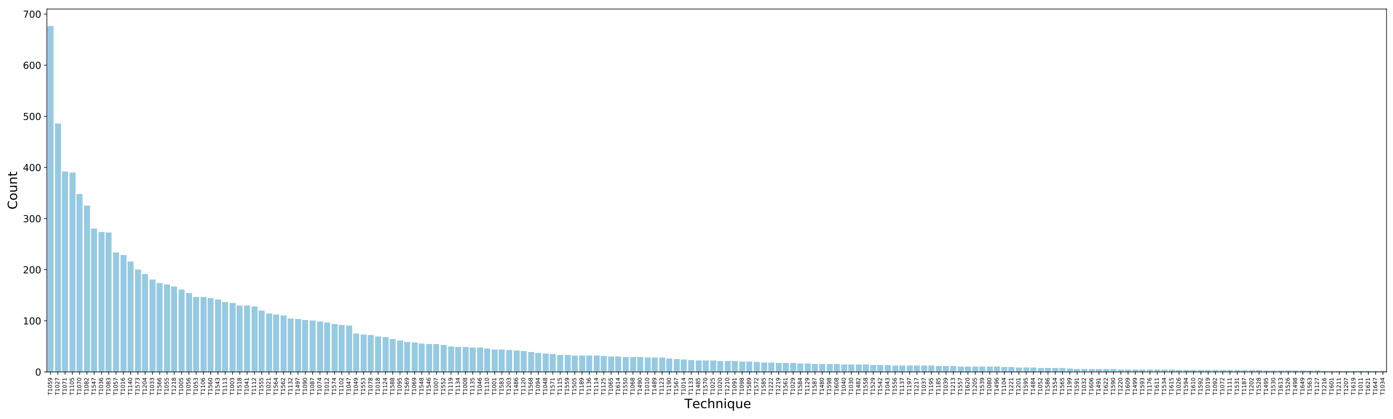

Fig. 4: Histogram of techniques observed in the dataset.

Algorithm 1: Activity-Attack Graph Creation.

```
$T$ : Set of all known ATT&CK TTPs, $t \in T$
$S$ : Set of seeded TTPs, $s \in T$, $S \subseteq T$
$AG$ : Set of activity groups, $R \in AG$, $R = (X \rightarrow Y, w)$,
  $X, Y \subset T$
$AAG$ : Activity-Attack Graph
$t$ = threshold
$q$ = queue of TTPs to visit
for $s$ in $S$ do
    $q.enqueue(s)$
end
while $q$ is not empty do
    $ttp = q.dequeue()$
    $AAG.addVertex(ttp)$
    if $(ttp \in R)$ and $w(R) \geq t$ then
        for $t,i$ in $R$ do
            $AAG.addVertex(t)$
            $AAG.addEdge(t_i, t)$
            $q.enqueue(t_i)$
        end
    end
end
```

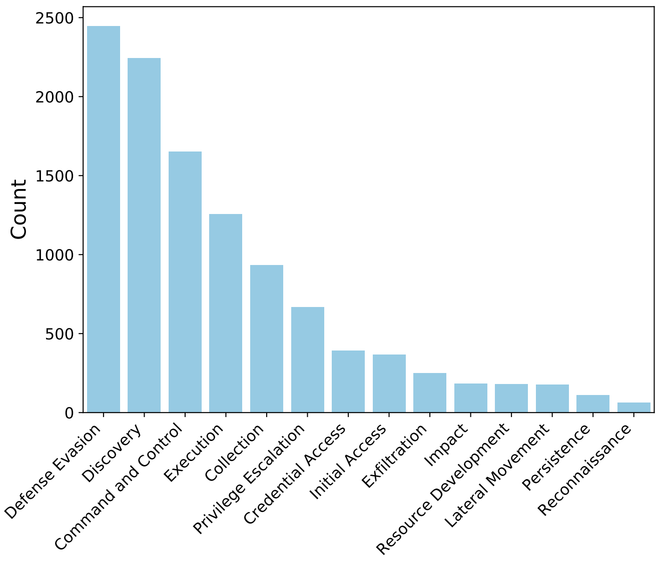

Fig. 5: Histogram of tactics represented in the dataset.

E. Standardization

Our intelligence-informed graphs are standardized in accordance with the STIX framework. We use the official MITRE ATT&CK objects to create STIX bundles of seed techniques, hypothesized techniques, and hypothesized relationships between techniques in the bundle. Techniques are represented using attack pattern objects and relationships are represented using relationship objects. The intended use of the STIX bundle is to generate an activity-attack graph to enable an analyst to visualize their seed techniques and how they relate to hypothesized techniques. Standardization enables analysts to easily extend our approach for their intelligence consumption demands. For example, analysts can use analytical pivoting to extend their contextual understanding of the attack, hunt for related adversary activity, and develop defensive courses of action.

IV. Results

In this section, we evaluate our learned activity groups, present our results, and discuss several COA case studies.

A. Evaluating Activity Groups

1) On the Distribution of Technique Observations: As shown in Fig. 4, the distribution of techniques observed in our dataset follows a power-law distribution. Few techniques are used most of the time (high-frequency techniques) and most techniques are rarely observed. In Fig. 5, we lifted the techniques into their respective tactic category and observe a similar distribution over tactics. Specifically, techniques related to defense evasion, discovery, command and control, and execution are observed more frequently. For example, T1027 (Defense Evasion::Obfuscated Files or Information) is the second most frequent technique. This is expected because many of our observations come from malware campaigns, and it is widely accepted that about 80 percent of malware

TABLE II: Summary of activity groups (AG).

AG Category	AG Count	Support(AG)				Confidence(AG)				Lift(AG)			
		Min	Max	Mean	Std Dev	Min	Max	Mean	Std Dev	Min	Max	Mean	Std Dev
Sub-techniques	2779	0.05	0.31	0.07	0.02	0.70	1	0.79	0.07	1.34	16.21	2.19	1.51
Abstracted	6461	0.05	0.38	0.07	0.03	0.70	1	0.80	0.07	1.24	9.19	1.88	0.74

TABLE III: Subset of activity groups.

Activity Group (TTP Associations)	Sup	Conf	Lift	Comments
$\{T1070, T1566\} \rightarrow \{T1071, T1105\}$	0.051	0.704	2.24	High lift; low support
$\{T1027, T1105\} \rightarrow \{T1059\}$	0.255	0.763	1.35	High lift; high support
$\{T1105, T1566.001\} \rightarrow \{T1204.002\}$	0.103	0.987	5.50	High lift; high confidence
$\{T1016, T1041, T1059.003\} \rightarrow \{T1082\}$	0.052	0.886	2.03	Low support; high confidence
$\{T1049\} \rightarrow \{T1016\}$	0.080	0.800	2.75	Low support
$\{T1017, T1033, T1547.001\} \rightarrow \{T1071.001\}$	0.063	0.839	2.08	High lift; low support

is packed [11]. T1059 (Execution::Command and Scripting Interpreter) is the most frequently observed technique (appears in 600 observations), which is intuitive because many attacks require the adversary to issue commands on the target's operating system.

Most of the techniques are rarely observed events. Due to challenges in capturing all adversary activity for intelligence reporting, the lack of a technique does not guarantee that the technique was not deployed. This is particularly true for resource development and reconnaissance, two of the rare event tactic categories observed in Fig. 5. Because these two phases of the kill-chain are typically deployed on the attacker's infrastructure or are conducted passively, analysts are constrained in their ability to observe such activities.

2) Activity Group Metrics: We evaluate the learned activity groups ($AG : X \rightarrow Y$) in terms of support, confidence, and lift. The support of an AG, defined in (1), is the frequency that the activity group was observed in the dataset. Support measure the importance, or pervasiveness, of an activity group and is best used to find frequent sets of techniques. Despite its utility, support suffers from the rare event problem. As discussed in section IV-A1, many of the techniques are rarely observed. As such, their learned associations will have low support.

$$Sup(AG : X \rightarrow Y) = P(X \cup Y) \tag{1}$$

The confidence of an AG, defined in 2, is the probability of the techniques in Y being observed under the condition that the techniques in X are observed. If $Sup(Y)$ is high (i.e., close to 1), then the confidence of $X \rightarrow Y$ will be high even if they are not associated. As such, confidence cannot be used exclusively to evaluate the quality of an activity group.

$$Conf(AG : X \rightarrow Y) = \frac{P(X \cup Y)}{Sup(X)} \tag{2}$$

The lift of an AG, defined in 3, is the ratio of the confidence of the AG and the support of the consequent. It is useful to measure the degree of interest in an association [12]. A lift value of 1 indicates X and Y are statistically independent. Such AGs represent weak associations. Larger lift values (> 1) suggest interesting and strong associations that are not the result of high-frequency antecedents or consequents. Lift is not susceptible to the rare event problem. However, noise in small datasets can create activity groups with misleadingly high lift values if the rare item occurs by chance. For our research, this risk is minimized because our dataset is manually curated by subject matter experts and is less vulnerable to noise. Further, the deployment of an adversarial technique in an attack is deliberate and not likely a random event observed by chance.

$$Lift(AG : X \rightarrow Y) = \frac{Conf(X \rightarrow Y)}{Sup(Y)} \tag{3}$$

B. Evaluating Activity Groups

We create two sets of activity groups: (1) learned AGs that include the original sub-techniques, and (2) learned AGs from the modified dataset that abstracted the sub-techniques into higher-level techniques. We select a minimum support threshold of 0.05, and a minimum confidence threshold of 0.70. These thresholds are configurable based on the needs and preferences of the analyst [13]. Further, only associations with a maximum of three antecedents and two consequents are included to reduce AG complexity and redundancy in the frequent itemsets.

Table II summarizes the statistical properties of both activity groups, and table III shows a small subset of learned AGs. We identified 2779 AGs that include both techniques and sub-techniques and 6461 AGs from the abstracted dataset. The abstraction into techniques increased the number of AGs, many of which are redundant if the directionality of the association is ignored. The abstracted AGs have a reduced average lift because the frequency of the most rare events increases due to the abstraction process. As discussed in section IV-A2, if the consequent is a rare event, then the lift value will be high.

Analysts must trust an AG's association rule to adopt it in their cyber threat intelligence processes. Support, confidence, and lift must all be considered when accepting or rejecting a hypothesized TTP (i.e., the consequent). AGs with high lift are generally considered interesting, but they tend to have lower support. This is may be due to limited intelligence reporting, or the attack pattern is indeed rarely executed in practice. As discussed in section III-C, an intelligence seed (set of known

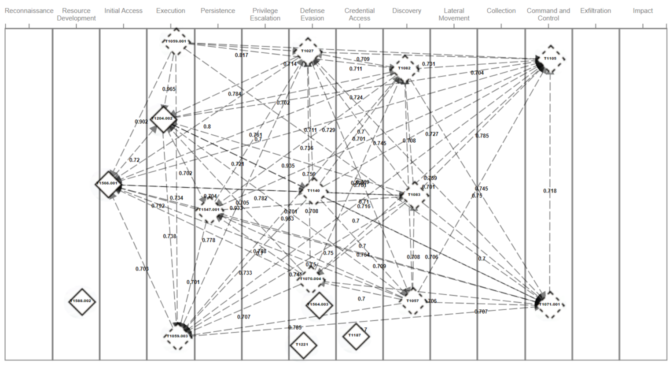

Fig. 6: Activity-attack graph displaying hypothesized and seed techniques and sub-techniques for APT DarkHydrus.

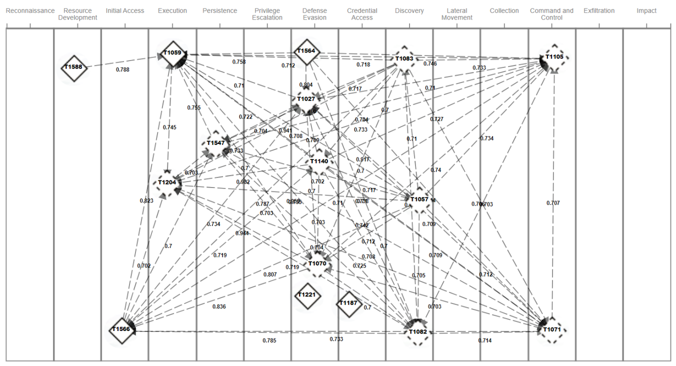

Fig. 7: Activity-attack graph displaying hypothesized and seed abstracted techniques for APT DarkHydrus.

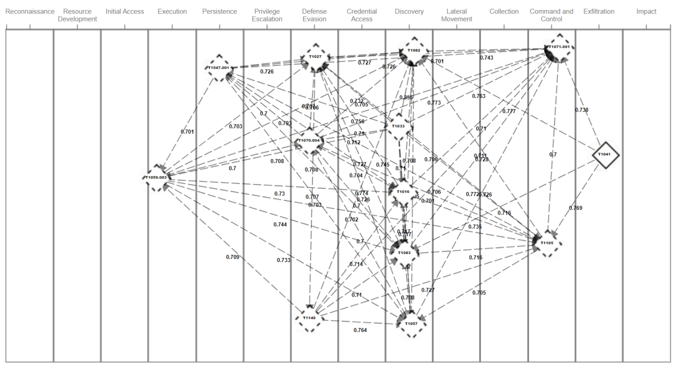

Fig. 8: Activity-attack graph displaying hypothesized techniques and sub-techniques seeded with T1041.

techniques to plan around) is required to inform which AGs to include in our threat course of action plan. The resulting activity-attack graph contains hypothesized TTPs that are the consequents of the AGs. The graph offers an additional metric we call the degree of a technique, $d(t)$. We define the degree as the cardinality of the neighborhood of the technique, $d(t) = |N(t)|$. A high degree indicates numerous threat avenues of approach may realize the hypothesized TTP, which informs the analyst to either accept or reject the hypothesized TTP.

C. Activity-Attack Graph Generation

1) Case Study: DarkHydrus: We generate activity-attack graphs using the approach described in section III-D for threat group DarkHydrus,[4] an APT known to target Middle Eastern governments and masquerades its command and control traffic as normal network and cloud infrastructure [14]. We seed the activity-attack graph generation routine with TTPs known to be deployed by DarkHydrus. The activity-attack graph in Fig. 6 was created using activity groups with specific sub-techniques. By contrast, the activity-attack graph in Fig. 7 was generated using abstracted AGs (no sub-techniques). Fig. 6 contains many sub-techniques of the hypothesized techniques found in Fig. 7; however, Fig. 7 contains more hypothesized relationships. This highlights a trade-off between using activity groups created using specific sub-techniques and activity groups using abstracted techniques. While both graphs provide realistic threat COAs, the abstracted approach may be less meaningful to analysts interested in specific sub-techniques.

2) Case Study: T1041: Fig. 8 displays the activity-attack graph generated by seeding our approach with technique T1041 (Exfiltration::Exfiltration over C2 Channel). This technique is observed when attackers use existing C2 channels and protocols as command and control communications to exfiltrate data. Our approach generated 12 hypothesized techniques and sub-techniques that are likely be observed prior to T1041. The algorithm also generated relationships (represented as weighted edges in the graph) between techniques. Adjacent to T1041 are two potential command and control techniques: T1071.001 (Application Layer Protocol::Web Protocols) and T1105 (Command and Control::Ingress Tool Transfer). The algorithm also proposes five hypothesized techniques used in the discovery phase of the kill-chain. This is reasonable because attackers will often enumerate their targets for information worth exfiltrating. The algorithm generates relationships tracing the attack back to the execution phase, where it hypothesizes that the technique used to execute this attack was T1059.003 (Command and Scripting Interpreter::Windows Command Shell) and persistence was gained using T1547.001 (Boot or Logon Autostart Execution::Registry Run Keys / Startup Folder). These are justified because a Windows command shell can be used to issue discovery commands throughout such an attack and edit the windows registry.

3) Case Study: Phishing Attack: We seed our algorithm with T1566 (Initial Access::Phishing) and T1024 (Execution::User Execution) to simulate a phishing attack. In this case study, we use the abstracted activity groups. Fig. 9 displays the activity-attack graph of the attack simulated. The algorithm generates T1547 (Persistence::Boot or Logon Autostart Execution), suggesting that this attack uses autostart events to run a malicious file or gain elevated privileges on the machine. The algorithm makes several suggestions for defense evasion techniques to avoid discovery. These include T1070 (Defense Evasion::Indicator Removal) and T1027 (Defense Evasion::Obfuscated Files or Information). Both are reasonable for a phishing attack as any malicious files downloaded

[4]https://attack.mitre.org/groups/G0079/

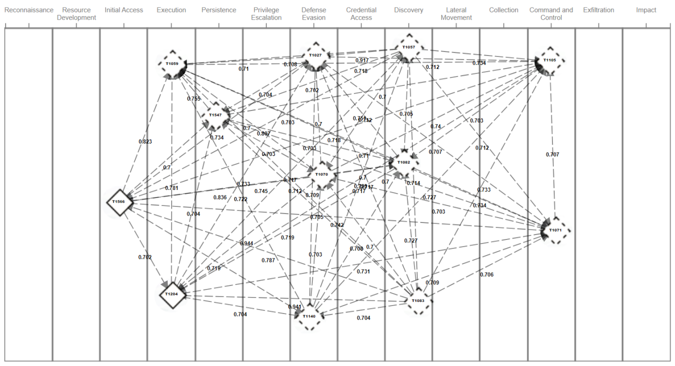

Fig. 9: Activity-attack graph displaying hypothesized abstracted techniques seeded with T1566 and T1024.

by the user are likely to attempt to use obfuscation and avoid detection by removing events and logs that might compromise the attack. The algorithm hypothesizes techniques used for discovery such as T1083 (File and Directory Discovery), and finally possible command and control techniques T1105 (Ingres Tool Transfer) and T1071 (Application Layer Protocol).

V. Conclusion

This paper proposes an activity group creation (AGC) function to generate activity groups that contain associations between MITRE ATT&CK techniques. Using the AGC function, we generate two sets of activity groups: one set containing activity groups with techniques and sub-techniques, and one set containing activity groups with the sub-techniques abstracted into their parent techniques. This enables an analyst to choose a set based on the techniques they have observed and the desired precision for threat course of action enumeration.

We introduce an algorithm to use our activity groups to generate an activity-attack graph seeded with a set of techniques. This graph suggests a list of hypothesized techniques for the analyst to investigate and hypothesized relationships between techniques in both the seed and suggestions. We showed how hypothesized events and relationships generated by our algorithm provide useful information to an analyst by providing recommendations of techniques to look for given observed techniques.

One limitation of ATT&CK technique analytics is the availability of data. Intelligence reports should continue to use standardized frameworks such as MITRE and STIX to further TTP data collection efforts. Larger datasets will enable our approach to find more interesting and important activity groups. In future work, we will extend our approach to produce a priority-based mitigation plan. Our STIX-based implementation enables analytical pivoting into other intelligence datasets, such as ATT&CK mitigation, for each seeded and hypothesized technique.

References

[1] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "Mitre att&ck: Design and philosophy," in *Technical report*. The MITRE Corporation, 2018.

[2] S. Barnum, "Standardizing cyber threat intelligence information with the structured threat information expression (stix)," *Mitre Corporation*, vol. 11, pp. 1–22, 2012.

[3] V. Mavroeidis and S. Bromander, "Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence," in *2017 European Intelligence and Security Informatics Conference (EISIC)*, 2017, pp. 91–98.

[4] 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*, 2011.

[5] M. Ahmed, S. Panda, C. Xenakis, and E. Panaousis, "Mitre att&ck-driven cyber risk assessment," in *Proceedings of the 17th International Conference on Availability, Reliability and Security*, 2022, pp. 1–10.

[6] P. Rajesh, M. Alam, M. Tahernezhadi, A. Monika, and G. Chanakya, "Analysis of cyber threat detection and emulation using mitre attack framework," in *2022 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*. IEEE, 2022, pp. 4–12.

[7] S. Choi, J.-H. Yun, and B.-G. Min, "Probabilistic attack sequence generation and execution based on mitre att&ck for ics datasets," in *Cyber Security Experimentation and Test Workshop*, 2021, pp. 41–48.

[8] R. Al-Shaer, J. M. Spring, and E. Christou, "Learning the associations of mitre att&ck adversarial techniques," in *2020 IEEE Conference on Communications and Network Security (CNS)*, 2020, pp. 1–9.

[9] S. Caltagirone, A. Pendergast, and C. Betz, "The diamond model of intrusion analysis," Center For Cyber Intelligence Analysis and Threat Research Hanover Md, Tech. Rep., 2013.

[10] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th international conference on very large data bases, VLDB*, vol. 1215. Santiago, Chile, 1994, pp. 487–499.

[11] L. H. Park, J. Yu, H.-K. Kang, T. Lee, and T. Kwon, "Birds of a feature: Intrafamily clustering for version identification of packed malware," *IEEE Systems Journal*, vol. 14, no. 3, pp. 4545–4556, 2020.

[12] P.-N. Tan, V. Kumar, and J. Srivastava, "Selecting the right interestingness measure for association patterns," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 32–41.

[13] E. Hikmawati, N. U. Maulidevi, and K. Surendro, "Minimum threshold determination method based on dataset characteristics in association rule mining," *Journal of Big Data*, vol. 8, no. 1, pp. 1–17, 2021.

[14] A. Alageel and S. Maffeis, "Hawk-eye: holistic detection of apt command and control domains," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, 2021, pp. 1664–1673.