Data-Driven Decision Support for Optimizing Cyber Forensic Investigations

Antonia Nisioti, *Member, IEEE,* George Loukas, Aron Laszka, and Emmanouil Panaousis, *Member, IEEE,*

***Abstract*—Cyber attacks consisting of several attack actions can present considerable challenge to forensic investigations. Consider the case where a cybersecurity breach is suspected following the discovery of one attack action, for example by observing the modification of sensitive registry keys, suspicious network traffic patterns, or the abuse of legitimate credentials. At this point, the investigator can have multiple options as to what to check next to discover the rest, and will likely pick one based on experience and training. This will be the case at each new step. We argue that the efficiency of this aspect of the job, which is the selection of what next step to take, can have significant impact on its overall cost (e.g., the duration) of the investigation and can be improved through the application of constrained optimization techniques.**

Here, we present DISCLOSE, the first data-driven decision support framework for optimizing forensic investigations of cybersecurity breaches. DISCLOSE benefits from a repository of known adversarial tactics, techniques, and procedures (TTPs), for each of which it harvests threat intelligence information to calculate its probabilistic relations with the rest. These relations, as well as a proximity parameter derived from the projection of quantitative data regarding the adversarial TTPs on an attack life cycle model, are both used as input to our optimization framework. We show the feasibility of this approach in a case study that consists of 31 adversarial TTPs, data collected from 6 interviews with experienced cybersecurity professionals and data extracted from the MITRE ATT&CK STIX repository and the Common Vulnerability Scoring System (CVSS).

***Index Terms*—Cyber forensics, digital forensics, multi-stage attacks, decision support, optimization**

I. Introduction

As adversaries advance their techniques, the impact of cyber attacks on businesses grows [1]. The longer it takes to discover and analyze an attack, the greater the impact on the affected organization. Indicatively, the mean time to identify a security breach in 2019 has been estimated to be 206 days [2]. Naturally, this time also depends on the efficiency of the cyber forensic investigation [3].

Consider a complex attack that consists of multiple attack actions. When one of these is discovered, for example by observing the modification of sensitive registry keys, suspicious network traffic patterns or the abuse of legitimate credentials, a forensic investigation can be triggered to uncover the rest of the attack. The sooner these remaining attack actions are uncovered, the lower the impact in terms of financial or reputation loss and the better the technical and business response of the targeted organization, especially if the attack is ongoing [4].

Cyber forensic analysts face a number of challenges that stem from the extensive amount of data that is being produced, the new technologies that arise constantly and the increasing complexity of attacks [5]. Even though there are guidelines and general methodologies, such as [6] and [7], they focus on the forensic soundness and general guidance on the methodology, not on the efficiency of the investigation. The actual practical methods followed are based on their own experience and eagerness to update their knowledge on new technologies and attack trends.

In practice, time is a limiting factor as to what an investigator is able to achieve given the large number and variety of Tactics, Techniques, and Procedures (TTPs) used by threat actors. Efficiency is an issue widely recognized by practitioners in cyber forensics [8]. We argue that a decision support framework based on constrained optimization can help to increase the efficiency of the investigation process by decreasing the duration of the investigation and taking in consideration the likely impact of each attack action.

In this paper, we propose DISCLOSE, which is the first data-driven decision support framework that offers these capabilities to a forensic investigator. DISCLOSE allows the investigator to navigate through the attack action space based on evidence discovered up to that point, an available budget and the expected benefit acquired by each available investigation action. More specifically, DISCLOSE is the first data-driven decision support framework for the optimization of multi-stage cyber forensic investigations that utilizes:

- public knowledge bases of TTPs and all information discovered in previous steps of the investigation,
- the probabilistic relations between TTPs, currently used by threat actors, which are calculated via a threat information exchange platform,
- a proximity parameter, which stems from the projection of quantitative data regarding adversarial TTPs on an attack life cycle model,

before calculating the next optimal inspection for the Defender to accomplish.

We evaluate DISCLOSE using a case study of 9 attack scenarios (3 incidents and for each incident 3 different triggering actions) that are based on real-world data gathered through:

- 6 interviews with experienced cybersecurity professionals,

A. Nisioti, G.Loukas and E.Panaousis are with the Department of Computing and Mathematical Sciences, University of Greenwich, London SE10 9BD, U.K. (email: a.nisioti@greenwich.ac.uk; g.loukas@greenwich.ac.uk; e.panaousis@greenwich.ac.uk).

A. Laszka is with the Department of Computer Science, University of Houston, TX, USA (alaszka@uh.edu).

Manuscript accepted January 16th, 2021.

- the MITRE ATT&CK Structured Threat Information Expression (STIX) [9] repository, which is one of the most popular threat intelligence knowledge bases. It utilizes the widely known STIX language and format to make publicly available adversarial TTPs based on real-world observations,
- the Common Vulnerability Scoring System (CVSS).

The rest of the paper is organized as follows. Section II presents the related work in the fields of correlation and attribution of cybersecurity incidents, forensic soundness, and reasoning support and investigation efficiency. Section III presents the system model, while Section IV presents the core methodology to generate optimal suggestions for the investigator using the system model. Section V presents a case study where DISCLOSE and three more policies are applied on a set of real world TTPs and their performance is compared and discussed. Section VI discusses the limitations of the proposed method as well as future work, and Section VII concludes the paper. Finally, Appendix A contains Table IX with all the acronyms and abbreviations used in the rest of this paper as well as their corresponding expansion.

II. Related Work

Existing frameworks for assisting digital forensics can be classified into those related to: (i) correlation and attribution, (ii) the soundness of the forensic evaluation, and (iii) reasoning support and efficiency. Below, we describe the most important previous work in each category and highlight how DISCLOSE differs or improves upon it. This section presents only those works that are most relevant to our approach. The reader may refer to [10], [11], [12], and [13] for further reading.

A. Correlation and attribution

Event correlation refers to the process of aggregating and analyzing heterogeneous data from a number of different sources with the aim of reconstructing malicious behaviors. Event correlation differs from the classic intrusion detection as it does not aim to simply produce potential malicious indicators.

Milajerdi et al. have proposed HOLMES [14], which utilizes the typical APT life-cycle and MITRE's ATT&CK framework to generate high-level graphs of potentially malicious events for the investigator. The authors have created a TTP specification to map audit log flows to TTPs and a number of noise reduction methods to decrease the levels of false positives. For the detection and construction of the audit log flows, they have combined tag propagation and policies for backward and forward analysis as per [15], [16]. However, the assumption that all of the APT stages will be retrieved successfully, which is what mainly drives the creation of the graph, is not well-founded. Moreover, by considering only the TTPs in ATT&CK, it misses the opportunity to utilize any other aspect of the MITRE ATT&CK knowledge base. On the contrary, while DISCLOSE uses the TTPs provided by ATT&CK, it also utilizes the MITRE ATT&CK STIX repository in order to extract relevant data and calculate the probabilistic relations between those TTPs, as will be thoroughly discussed later.

Hossain et al. [17] have presented a tag-propagation graph generation that aims to support the analyst in dealing with the large amount of data. The authors have proposed two new tag propagation techniques, namely tag attenuation and tag decay, which are based on common behaviors of benign Unix processes in order to generate a compact scenario graph of the malicious behavior and decrease the number of false positives. A different use of graphs in digital forensics for attribution has been presented in [18] and [19]. The proposed network forensics tool prototype uses a fuzzy reasoning module based on evidence graphs in order to correlate hosts that belong in the same attack. The prototype uses IDS alerts as primary evidence and host logs as secondary evidence. Graph based clustering has been used by Studiawan et al. [20] to detect anomalies in access control logs. The proposed solution uses an improved MajorClust algorithm on graph structures produced by access logs and a dynamic threshold to identify potential violations, which can then be inspected by an analyst. The aforementioned approaches aim to detect or automatically correlate parts of the attacks rather than help and guide the investigator through the technical part of the investigation, i.e. the analysis of the incident.

B. Forensic soundness

The soundness of a forensic investigation can affect both the success of the investigation itself, i.e. mishandling evidence may lead to their damage, as well as the ability to use the findings of the investigation, i.e. for legal prosecutions in a court of law. The investigation methodology has to be reliable, repeatable and well documented [21] and follow certain guidelines. The following works aim to assist the investigator on performing forensic sound methodologies on a range of different cases.

Horsman [22] has proposed the Digital Evidence Reporting and Decision Support (DERDS) framework, which helps the investigator assess the soundness of her inferences, assumptions or conclusions on each step of the investigation. A similar approach has been presented by Beebe et al. in [23] as a multi-tier objective framework for digital investigations. The first tier provides more generic guidance while the rest are more complex and specific to the investigator choices. Both frameworks are based on a collection of guidelines that are presented to the investigator while our approach is a combination of data-driven methodology and cybersecurity categorization. One commonality of our approach with the one in [23] is that we are tailoring future suggestions on inspection results so far.

Finally, a more technical approach that aims to ensure the soundness of an investigation has been proposed in [24]. The authors have presented an inference system that detects the use of anti-forensic attacks on the case data and reverts their effects. All aforementioned frameworks of this subsection focus on assisting or ensuring the forensic soundness of the investigation, while DISCLOSE aims to assist the technical part of the forensic analysis and help the investigator overcome the technological and adversarial challenge.

C. Reasoning and efficiency

The following works focus on either enhancing the reasoning process or the efficiency of an investigation. Increasing the efficiency of an investigation refers to decreasing the time or resources required for the investigation without negatively affecting the objectivity and the quality.

Similarly, *reasoning* is the process that the forensic investigator follows to create logical links between the uncovered evidence in an objective and logical manner. Having a robust and sound reasoning process is a crucial part of any forensic investigation as the opposite may lead to misinterpretations and ultimately false conclusions. However, not all frameworks that assist the reasoning process have the same focus or aim to support the same types of investigation. Some may be focused on the technical analysis part of an investigation while others may support a higher abstraction level. A popular category of reasoning frameworks is case-based reasoning, which includes frameworks that aim to assist utilizing the solutions of previous similar problems [25]. This can be achieved with the use of predefined rules, models, previous data or a combination of the above. Frameworks that use predefined rules are ruled-based, while the ones that utilize past data are called data-driven.

Turnbull et al. [26] have presented a multi-ontology framework for digital forensics as well as SPARQLer, a forward-chain rule-based reasoning system that aims to assist the investigator to move from computer-specific information to a higher abstraction. Similarly, the authors of [27] have presented a framework that models the initial and current state of a system as a transition system in order to allow the investigator to verify if a specific reasoning hypothesis is possible on this system. Both works aim to assist a higher abstraction level of the investigation than DISCLOSE, namely the general reasoning process of the investigator and not the technical analysis reasoning.

Saad and Traore [28] have proposed an ontology-based knowledge representation framework for network forensic investigations. The framework consists of an ontology, which has different classes to describe network forensics domain objects and their relations, and a reasoning system. The ontology can be used manually by the investigator as a model-based reasoning framework or as input to a system, which reconstructs an attack. Nieto has proposed JUDAS, a tool for extracting unique objects, i.e. users and devices, from case data and correlate them using rules, additional data and OSINT [29]. Both works propose reasoning frameworks that are based on pre-defined rules and objects that need to be manually populated. On the contrary, DISCLOSE provides data-driven case-based reasoning support and combines both data from a well-known knowledge base (or any other base that the investigator might choose to use) and two established cybersecurity attack life cycle models.

Horsman et al. [30] have presented CBR-FT, a reasoning-based technique for more efficient device triage. Similarly to our approach, CBR-FT uses a knowledge base of past cases to calculate a similarity rating for file system paths, which is then used to offer predictions for the triage process of the current case. The knowledge base has to be populated by the investigators with previous cases and some of the input variables have to be defined by the current investigator. CBR-FT provides a general probability for a path of a file system to contain evidence. On the contrary, DISCLOSE uses Tactics, Techniques, and Procedures (TTPs) instead of paths, which allows for more flexibility and the analysis of different data sources and mediums. Furthermore, DISCLOSE is an interactive data-driven case-based reasoning framework, which instead of simply providing a probabilistic ranking at the start of an investigation irrespective of the investigation itself as CBR-FT, it takes in consideration the progress of the investigation and re-calculates the probabilistic relations between TTPs at each step.

Likewise, Nassif et al. in [31] have proposed the use of clustering techniques to tackle the problem of processing large amount of unstructured documents during a forensic investigation. The authors have evaluated a variety of clustering algorithms and identified hierarchical algorithms as the best in terms of time and effectiveness. Although this work aims to decrease the investigation time and increase the efficiency, similarly to DISCLOSE, it does it in a different way by proposing a tool that decreases the required time of crucial task of the technical analysis.

De Braekt et al. [32] have presented a framework that—similarly to our work—aims to increase the efficiency of a forensic investigation. While [32] aims to achieve this by providing guidelines to the investigator on how to streamline the workflow and make more efficient use of the available resources, DISCLOSE uses known TTPs and previous data to guide the investigator on each step. An approach much more similar to ours has been presented in [33]. While our aim is to prioritize the inspections of the attack actions to increase the efficiency of the investigation, the authors focus on database auditing prioritization which is modelled as a Stackelberg game between the adversary and the auditor. Evaluated in two real medical alert datasets, the proposed approach outperformed traditional non-game-theoretic approaches. While the authors optimize the auditing policy of the auditor as a single process, DISCLOSE works at a step by step basis utilizing the previous inspections performed and their results to offer optimized suggestions to the investigator.

Finally, evidence graphs have been proposed in the past in order to support an investigation and increase its efficiency. Liu et al. [34], [35], [36] have proposed the use of evidence graphs for broader context to the investigator, while Barrère et al. [37] have used core graphs, ‘condensed’ or ‘induced’ versions of attack graphs for increased efficiency. Contrary to the aforementioned approaches, which all use vulnerabilities to model the investigation, DISCLOSE uses TTPs from MITRE ATT&CK. This offers a more pragmatic and representative approach, as incidents under investigation are modelled against the current real adversarial techniques, which is the state-of-the-art in the cybersecurity industry.

In conclusion, although a number of frameworks for digital forensics have been proposed, none had the same aim or followed the same method as DISCLOSE. More specifically:

- All the reviewed methods aimed to support the investigation and increase its efficiency in a different way, either

by presenting a set of guidelines, a reasoning system or an automated correlation tool.
Contribution 1: No related work has presented a framework like DISCLOSE that follows and supports the investigation process in a stepwise manner by suggesting possible attack actions to be analyzed.

- Previous works present general guidelines/suggestions to the user and do not tailor it either to the investigation or its progress. **Contribution 2**: DISCLOSE ingests the feedback of the investigator at the beginning and at each step of the investigation and uses it to tailor future suggestions based on the ongoing results.
- All previous reasoning frameworks model a single attack instance as a vulnerability which is not representative of the current threat landscape, as (a) many attack actions do not include the exploitation of a vulnerability, (b) many attack actions include more than one vulnerability, and (c) a vulnerability is not able to capture the adversarial behavior and contextual information but rather just a weakness on the system. **Contribution 3**: DISCLOSE improves upon this by using adversarial TTPs that align with the current threat landscape allowing for a representative modelling of the investigation and increasing the applicability of the framework to a wider range of cases. A TTP is more abstract and has the ability to describe the adversarial behavior, its objective and execution. For this reason, the use of TTPs allows DISCLOSE to overcome shortcomings such as zero-day vulnerabilities.
- Even though some works utilized the MITRE ATT&CK TTPs, similarly to DISCLOSE, they do not utilize any other aspect of the knowledge base, such as the attributes and relationships contained in it.
Contribution 4: DISCLOSE improves on this by using the ATT&CK STIX repository to extract a range of fields, which are used to calculate the probabilistic relations between TTPs providing data-driven decision support to the investigator.
- **Contribution 5**: Finally, contrary to previous works DISCLOSE does not simply use a data-driven approach but combines some of the fields extracted from the ATT&CK STIX repository with a well-known attack life cycle model. Thus our framework is not simply data-driven but also aware of the internal structure of an incident, which allows it to optimize its suggestions even further as will be shown in Section V.

III. System Model

We assume that a digital forensic investigator, henceforth called the Defender, has been assigned by an organization to analyze a cyber incident after one or more signs of an intrusion have been detected. Her goal is to reconstruct the steps performed by an adversarial entity referred to as the Attacker. To achieve this, the Defender must collect and analyze data in order to uncover evidence that will lead to the disclosure of as many attack actions performed by the Attacker as possible. This may lead to the potential attribution of the Attacker and enable the organization to strengthen its defenses preventing any further damage and obtaining evidence to take legal actions against the Attacker.

A. Attack actions and inspections

The organization has detected an incident, but it is not aware of which attack actions have been executed. The Defender's goal is to uncover these steps. The Defender knows that the attack actions used by the Attacker must be a subset of all available attack actions $\mathcal{A}$. Let $\mathcal{A}_P \subset \mathcal{A}$ denote the set of attack actions performed by the Attacker. With the exception of at least one attack action, which as mentioned earlier has triggered the investigation, these are unknown to the Defender at the beginning of the investigation, but they are gradually discovered as the Defender takes successful inspection actions. We assume that the Defender has a set of available inspection actions (henceforth called inspections) to choose from and that there is one-to-one correspondence between attack actions and inspections, i.e., each of the inspections is capable of revealing one and only one attack action and this attack action can be revealed only by that particular inspection. Thus, the Defender's problem can be formulated using only attack actions, as each attack action that has potentially be performed by the attacker equals to an inspection action of this attack action for the Defender.

We also assume that the efficacy of any inspection in disclosing whether the associated attack action was part of the incident equals one, i.e. the probability of successfully discovering if the attack action was performed or not equals one, as considering partially efficient inspections is out of scope of this paper. In other words, whenever the Defender inspects an attack action, she always successfully discover if the attack action is part of the incident or not. The investigation process takes place in a stepwise manner. In each step of the investigation process, the Defender chooses an attack action and performs investigation to discover whether this attack action was executed or not as part of the incident. We denote by $\mathcal{Y}^{(k)} \subset \mathcal{A}$ the set of attack actions that the Defender has discovered until step k and by $\mathcal{N}^{(k)} \subset \mathcal{A}$ the ones that were found as non-executed. Note that $\mathcal{Y}^{(k)} \subseteq \mathcal{A}_P$ and $\mathcal{N}^{(k)} \subseteq \mathcal{A} \setminus \mathcal{A}_P$.

1) Inspection actions analysis: For every attack action $i \in \mathcal{A}$, there is a cost C_i of undertaking the inspection that discovers this step. This cost expresses the resources that the Defender has to spend to undertake it. For example, this could represent the time required for the inspection or hardware and software resources, or a combination of them. The inspection cost increases with the complexity of the corresponding attack action and the heterogeneity of the data sources required to perform the inspection, while it is lower when there are tools available to automate aspects of the inspection.

Similarly, for every attack action $i \in \mathcal{A}$, we define B_i as the benefit of the Defender when it is discovered that i was executed during the incident. We consider B_i to be gained only when attack action i was actually executed during the investigated incident. Our model can also capture loss that derives from not discovering an executed attack, which is reflected on the collected benefit; but we choose not model this

explicitly for the sake of simplicity. Moreover, the loss from inspecting a non-executed attack action is indirectly captured in our model, as spending budget on non-executed attack actions may lead to exhausting the budget before uncovering all the performed attack actions.

This benefit expresses the value to the organization from: (i) *recovering after the incident*, and (ii) *increasing their defenses* to prevent future damages caused by similar incidents. In the first case, uncovering the exploitation of a vulnerability might cause the organization to patch it or a spear-phishing email with a malicious attachment might motivate better staff awareness training. In the second case, uncovering the data exfiltration of confidential client data might allow the organization to do better damage control and therefore minimize both financial and reputational losses.

TABLE I: List of symbols.

Symbol	Description
Constants	
$\mathcal{A}$	set of all possible attack actions
$\mathcal{D}$	set of all previous incidents
C_i	inspection cost of attack action i
B_i	benefit from inspection of attack action i
R	investigation budget
$\mathbf{T}$	attack actions dependency table
n_i	number of previous incidents with attack action i
n_{ij}	number of previous incidents with both attack actions i, j
d_{ij}	distance between attack actions i and j
Choice Variables	
$\mathcal{Y}^{(k)}$	set of executed attack actions inspected until step k
$\mathcal{N}^{(k)}$	set of attack actions inspected until step k but not executed
Variables	
$\mathcal{A}_P$	set of attack actions taken by the Attacker
$C(k)$	overall investigation cost until step k
$B(k)$	overall investigation benefit until step k
$P_i^{(k+1)}$	payoff of inspection step i at step $k+1$
$\mathbf{\Gamma}^{(k)}$	investigation's results vector at step k
$\boldsymbol{w}^{(k)}$	proximity vector at step k
$C(k)$	overall investigation cost until step k

B. Attack actions dependencies

We assume that the Defender has access to a representative number of past incidents (i.e., public dataset of real-world cyber incidents). For each past incident, we know exactly which attack actions were executed and which were not. We let $\mathcal{D}$ denote the set of incidents (i.e., our dataset). For each attack action $i \in \mathcal{A}$, we let n_i denote the number of incidents that attack step i appears in: $n_i = |\{I \in \mathcal{D} \mid i \in I\}|$.

Further, given $i, j \in \mathcal{A}$, we let n_{ij} denote the number of incidents in which both i and j appear: $n_{ij} = |\{I \in \mathcal{D} \mid i \in I \wedge j \in I\}|$. We then define and approximate the conditional probability for $j \in \mathcal{A}_P$ given that $i \in \mathcal{A}_P$ as

$$t_{ij} = \Pr[j \in \mathcal{A}_P \mid i \in \mathcal{A}_P] = \tag{1}$$

$$= \frac{\Pr[j \in \mathcal{A}_P \wedge i \in \mathcal{A}_P]}{\Pr[i \in \mathcal{A}_P]} \approx \frac{n_{ij}}{n_i}. \tag{2}$$

By analyzing previous incidents, we can calculate all the conditional probabilities between each pair of attack actions. We assume that all incidents follow a probability distribution, which can be obtained from historical data using past incident reports to calculate their empirical probabilities. To aggregate these probabilistic connections of attack actions, we define the *dependency table* $\mathbf{T} = \begin{bmatrix}\mathbf{t}_{*1} & \mathbf{t}_{*2} & \dots & \mathbf{t}_{*|\mathcal{A}|}\end{bmatrix} = [t_{ij}] \in [0,1]^{|\mathcal{A}|\times|\mathcal{A}|}$. The dependency table is created and constantly populated from *forensic reports* and *knowledge bases*, such as MITRE [38] and SANS[1], formed by previous incidents.

IV. The Framework

In this section, we present the core methodology of DISCLOSE, which allows it to generate optimal attack action suggestions for the investigator.

A. Problem Formulation

The Defender's goal is to maximize the benefit collected during the investigation, that is, maximizing $\sum_{i\in\mathcal{Y}^{(k)}} B_i$, without exceeding a given budget R, that is, ensuring that $\sum_{i\in\mathcal{Y}^{(k)}\cup\mathcal{N}^{(k)}} C_i \leq R$. Note that while the Defender knows in advance which sets of investigation actions are feasible with respect to the budget, she does not know which investigations will be productive since it does not know which attack actions were executed. Thus, the Defender has to maximize its expected benefit from the investigation.

In each investigation step, the Defender must select an inspection that she has not selected before. After each step k of the investigation, the Defender derives $\mathcal{Y}^{(k)}$ and $\mathcal{N}^{(k)}$. For each $i \in \mathcal{Y}^{(k)}$, the Defender can use the information provided by $\mathbf{t}_{i*}$ to estimate the probabilities of all other non-inspected attack actions. These probabilities are leveraged by the Defender in the algorithm used to select the next inspection.

B. Investigation

To represent these probabilities, we define the *investigation results* vector $\mathbf{\Gamma}$, which holds the probabilities of inspections to discover an attack action in the next investigation step. We denote by $\mathbf{\Gamma}^{(k)} = [\Gamma_j^{(k)}]$ the state of the vector $\mathbf{\Gamma}$ at the investigation step k, where $\Gamma_j^{(k)}$ is the probability of inspection j to discover an attack action given all attack actions discovered until step k. If $j \in \mathcal{A} \setminus \{\mathcal{Y}^{(k)} \cup \mathcal{N}^{(k)}\}$, then $\Gamma_j^{(k)} \in [0,1]$ else $\Gamma_j^{(k)} = 0$ as j has already been selected by the Defender during the first k investigation steps.

The Defender can leverage the information stored in $\mathbf{\Gamma}^{(k)}$; but in order for the information to be meaningful, the Defender must update $\mathbf{\Gamma}^{(k)}$ after each inspection. In essence,

[1]https://www.sans.org/

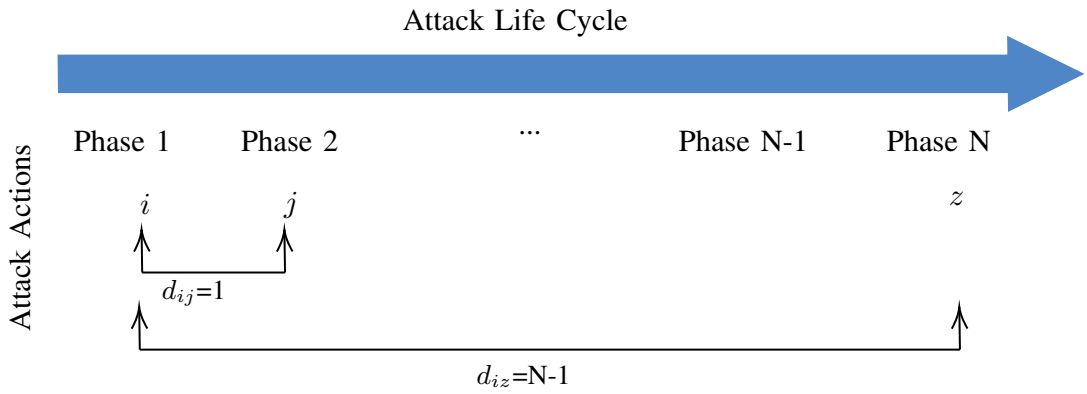

Fig. 1: Distance of attack actions when positioned on attack life cycle. d_{ij} refers to the distance between attack action i and j, while d_{iz} refers to the distance between attack action i and z. The smaller the distance, the more credible the probabilistic information that i provides for j.

$\mathbf{\Gamma}^{(k)} \neq \mathbf{\Gamma}^{(k+1)}$ if and only if the inspection taken in step $k+1$ discovers an attack action. A straightforward approach to update $\mathbf{\Gamma}^{(k)}$ is to assume that the Defender compares the value that is already stored in the vector with the corresponding value in the dependency matrix and keeps the maximum, as follows:

$$\Gamma_j^{(k)} = \begin{cases} 0, & j \in \{\mathcal{Y}^{(k)} \cup \mathcal{N}^{(k)}\} \\ \max\{\Gamma_j^{(k-1)}, t_{ij}\}, & \text{otherwise} \end{cases} \quad (3)$$

where i is the last successfully discovered attack action.

The aforementioned approach has a shortcoming that stems from the fact that the criteria for updating vector $\mathbf{\Gamma}^{(k)}$ is not based on any digital forensic property but rather simply on keeping the maximum value. This is true as table $\mathbf{T}$ holds the probabilistic relations between attack actions that stem from the observation of past incidents, but does not contain their conceptual relationships with regards to the structure of an attack. The structure of an attack refers to the logical organization of the attack actions within an incident. As it is widely accepted, attack actions within an incident do not happen in a random fashion, but instead follow a specific attack life cycle [39]. For instance, the *Spear-phishing Attachment through email* and the *Replication Through Removable Media* techniques are both used at the early infection stage of an incident thus being very close to each other on the attack life cycle [38]. Similarly, the *Pass-The-Hash* lateral movement and the *Credential Access* are both positioned on the later stages of the attack life cycle and are close to each other [38]. On the other hand, the *Spear-phishing Attachment through email* and the *Pass-The-Hash* lateral movement are positioned far away from each other on the attack life cycle.

Therefore, through the course of an incident, some attack actions are closer than others and thus more relevant. In other words, the closer two attack actions are the more credible is the probabilistic information they can provide for each other. For instance, in the aforementioned example *Spear-phishing Attachment through email* is closer to the *Replication Through Removable Media* than *Pass-The-Hash*. Let us now consider that we have discovered the *Spear-phishing Attachment through email* and the *Pass-The-Hash*, and the probabilistic value for *Replication Through Removable Media* that the *Spear-phishing Attachment through email* provides is smaller than the one the *Pass-The-Hash* provides. Based on the attack life cycle models, it is clear that both the *Spear-phishing Attachment through email* and the *Replication Through Removable Media* are TTPs that are used for the same purpose, i.e. the initially infection. Moreover, they are least likely to be used together in the same incident, which will be reflected on table $\mathbf{T}$. Based on the simple approach previously presented in (3), which does not take in consideration any attack life cycle model, the Defender would mistakenly keep the larger value for the *Replication Through Removable Media*. This could potentially lead to the inspection of this action and unnecessary spending of the budget on an action that is less likely to have happened.

For this reason, we introduce a *proximity parameter* that utilizes the distance between two attack actions in the attack life cycle model. This parameter will be used on each step k when updating vector $\mathbf{\Gamma}^{(k)}$. We define the distance d_{ij} between two attack actions i and j as their distance if we positioned them on an abstract attack life cycle, as presented on Figure 1. If attack action i is positioned on the first phase of the attack life cycle and attack action j is positioned on the second phase, their distance would be $d_{ij} = 1$. Similarly, if attack action z is positioned on the N-th phase of the attack life cycle, the distance from i would be $d_{iz} = N-1$. The smaller the d_{ij} the more credible is the probabilistic information that i provides for j, i.e. t_{ij}. Therefore, the probabilistic information that j provides for i is more credible than the one provided by z, because $d_{ij} < d_{iz}$. It is worth noting that this does not affect in any way the contents of the dependency table $\mathbf{T}$, it just affects the way they are prioritized during the update of $\mathbf{\Gamma}^{(k)}$.

We now revise the previous update process based on the above, so that instead of prioritizing larger values, it prioritizes ones that are closer. Similar to vector $\mathbf{\Gamma}^{(k)}$, we define the *proximity vector* $\boldsymbol{w}^{(k)}$ that at any step k holds for each j the distance from the attack action that provided the value $\Gamma_j^{(k)}$. Essentially $w_j^{(k)}$ shows how trusted is the value $\Gamma_j^{(k)}$. After

a successful inspection of an attack action i, we choose the probability that is most trusted based on the corresponding distance value. If for an attack action j the current value $w_j^{(k-1)}$ in the proximity vector is greater than the d_{ij} we set $\Gamma_j^{(k)} = t_{ij}$. On the other hand, if $w_j^{(k-1)}$ is smaller than d_{ij} we keep $\Gamma_j^{(k)} = \Gamma_j^{(k-1)}$. Finally, if the two values are equal, we choose to keep the maximum of the two as they are both of the same proximity. Therefore each vector $\mathbf{\Gamma}^{(k)}$ can be updated as:

$$\Gamma_j^{(k)} = \begin{cases} 0, & j \in \{\mathcal{Y}^{(k)} \vee \mathcal{N}^{(k)}\} \\ t_{ij}, & d_{ij} < w_j^{(k-1)} \\ \Gamma_j^{(k-1)}, & d_{ij} > w_j^{(k-1)} \\ \max\{\Gamma_j^{(k-1)}, t_{ij}\}, & \text{otherwise} \end{cases} \tag{4}$$

After updating vector $\mathbf{\Gamma}^{(k)}$ we update also the distance vector $\boldsymbol{w}^{(k)}$ in a similar way. If $\Gamma_j^{(k)} = \Gamma_j^{(k-1)}$ then $w_j^{(k)} = w_j^{(k-1)}$. Likewise, if $\Gamma_j^{(k)}$ was updated through the dependency matrix $\mathbf{T}$ then $w_j^{(k)}$ would be updated with the relevant distance value d_{ij}.

C. Optimal inspection actions

We define the aggregated cost of the entire investigation until step k as:

$$C(k) = \sum_{h \in \{\mathcal{Y}^{(k)} \vee \mathcal{N}^{(k)}\}} C_h. \tag{5}$$

Likewise, the *overall benefit* of the investigation until step k is $B(k) = \sum_{h \in \mathcal{Y}^{(k)}} B_h$, as the Defender gains the benefit value only from inspecting attack actions that have been performed by the Attacker. In each step k, in order to choose the next inspection step of the investigation, we calculate the *expected payoff* of each available inspection and choose the action that maximizes it. We denote by $P_j^{(k+1)}$, the expected payoff of inspection j at step $k+1$ and we define this as $P_j^{(k+1)} = \Gamma_j^{(k)} \frac{B_j}{C_j},\ \forall j \notin \{\mathcal{Y}^{(k)} \vee \mathcal{N}^{(k)}\}$.

Algorithm 1 Action Selection Function

Input: $G, \mathbf{\Gamma}^{(k)}$
function INSPECTIONSELECTION($R, \mathbf{\Gamma}^{(k)}$)
  $selected \leftarrow \arg\max_{\forall x \in \mathbf{\Gamma}^{(k)}} P_x$
  while $(C(k) + C_{selected}) > R$ **do**
    $\Gamma_{selected}^{(k-1)} \leftarrow 0$
    if $(\mathbf{\Gamma}^{(k)} = \mathbf{0})$ **then**
      $selected \leftarrow -1$
      break
    end if
    $selected \leftarrow \arg\max_{\forall x \in \mathbf{\Gamma}^{(k)}} P_x$
  end while
  return $selected$
end function

Algorithm 2 DISCLOSE

Input: attack action z that triggered the investigation, $R, \boldsymbol{C}, \boldsymbol{B}, \boldsymbol{d}, \boldsymbol{c}$
Initialization: $\mathbf{\Gamma}^{(0)} \leftarrow \boldsymbol{t_z}$; $\boldsymbol{w}^{(0)} \leftarrow [d_{c_z c_j}]$; $P(0) = 0$
while ($\mathcal{A}_P \subseteq \mathcal{Y}^{(k)}$ AND $\mathbf{\Gamma}^{(k)} \neq \mathbf{0}$) **do**
  $selected \leftarrow$ INSPECTIONSELECTION($R, \mathbf{\Gamma}^{(k)}$)
  if $selected$=-1 **then**
    break
  end if
  $C(k) \leftarrow C(k) + C_{selected}$
  $\Gamma_{selected}^{k} \leftarrow 0$
  if $selected \in \mathcal{A}$ **then**
    Append $selected$ to $\mathcal{Y}^{(k)}$
    $B(k+1) \leftarrow B(k) + B_{selected}$
    $P(k+1) \leftarrow P(k) + B_{selected}$
    for $j \leftarrow 1, |\mathcal{A}|$ **do**
      if $j \in \mathcal{Y}^{(k)}$ **then**
        $\Gamma_j^{(k)} \leftarrow 0$
      else if $d_{c_{selected} c_j} < w_j^{(k-1)}$ **then**
        $\Gamma_j^{(k)} \leftarrow t_{selected j}$
        $w_j^{(k)} \leftarrow d_{c_{selected} c_j}$
      else if $d_{c_{selected} c_j} > w_j^{(k-1)}$ **then**
        $\Gamma_j^{(k)} \leftarrow \Gamma_j^{(k-1)}$
        $w_j^{(k)} \leftarrow w_j^{(k-1)}$
      else
        $\Gamma_j^{(k)} \leftarrow \max\{\Gamma_j^{(k-1)}, t_{selected j}\}$
        $w_j^{(k)} \leftarrow w_j^{(k-1)}$
      end if
    end for
  else
    Append $selected$ to $\mathcal{N}^{(k)}$
  end if
end while

However, as stated in the formulation of the problem, there is a constraint that affects the choice of the Defender in each step. During any investigation, the Defender has a fixed budget of resources R (e.g., time) that can be utilized. Thus, the cost of the chosen inspection cannot exceed the remaining budget of the investigation. From the beginning until the end of the investigation, the Defender chooses inspection actions as described above with the aim of maximizing the overall benefit given the budget constraint. The investigation is terminated when: (i) budget is depleted, (ii) or there are none available inspection actions. The DISCLOSE framework, which has been presented in this section, can be found in Algorithm 2, while the attack action selection process is depicted as a separate function in Algorithm 1 for readability reasons.

V. CASE STUDY

This section presents a case study where DISCLOSE is assessed using three attack scenarios, which are based on the TTPs of the MITRE ATT&CK knowledge base and corresponding probabilistic relations. These relations originate from publicly available reports [38], which have been processed and

TABLE II: Subset of TTPs used in the case study labeled with the original IDs from the MITRE ATT&CK framework.

#	Attack action	#	Attack action
0	Credential Dumping (T1003)	16	System Service Discovery (T1007)
1	Custom C&C Protocol (T1094)	17	User Execution (T1204)
2	Data from Local System (T1005)	18	Replication Through Removable Media (T1091)
3	Data from Network Shared Drive file (T1039)	19	Data Encrypted for Impact (T1486)
4	Drive-by Compromise (T1189)	20	Virtualization/Sandbox Evasion (T1497)
5	Exfiltration Over Alternative Protocol (T1048)	21	DLL Search Order Hijacking (T1038)
6	Exfiltration Over C&C Channel (T1041)	22	Credentials in files (T1081)
7	Network Share Discovery (T1135)	23	Remote System Discovery (T1018)
8	New Service (T1050)	24	Windows Remote Management (T1028)
9	Pass the Hash (T1075)	25	Brute Force (T1110)
10	PowerShell (T1086)	26	External Remote Services (T1133)
11	Registry Run Keys / Startup Folder (T1060)	27	Network Service Scanning (T1046)
12	Scripting (T1064)	28	Create Account (T1136)
13	Spearphishing Attachment (T1193)	29	Web Shell (T1100)
14	Spearphishing Link (T1192)	30	Exploit Public-Facing Application (T1190)
15	Standard Application Layer Protocol (T1071)		

made available through the MITRE ATT&CK STIX repository. First, we present the attack action space $\mathcal{A}$ and data for each attack action i that includes: (i) its probabilistic relation t_{ij} with the rest of the attack actions, (ii) its corresponding cost C_i and benefit B_i, (iii) as well as the proximity parameter d_{ij}. Then, we define the attack actions in $\mathcal{A}_P$, which were performed by the Attacker against the organization, during the three different attack campaigns.

The scope of this is to apply DISCLOSE and measure its performance as well as compare it with CBR-FT [30], which was firstly mentioned in Section II, and a baseline approach.

The baseline approach is a version of DISCLOSE that does not take into consideration any attack life cycle model and thus does not use the proximity parameter, which was introduced in Section IV. In order to compare DISCLOSE with a past related framework from the literature we have chosen to implement two different version of CBR-FT, mentioned in Section II. Both are reasoning frameworks, that aim to focus the technical part of the analysis of an investigation and utilize past incidents.

However, the original CBR-FT presented in [30] differs in some ways from DISCLOSE thus we chose to create two versions of CBR-FT that are inspired but not identical to the original. These follow the same procedures and formulas and they are also comparable to DISCLOSE. More specifically, we have identified the following differences between DISCLOSE and CBR-FT and coped with them in the following ways:

- CBR-FT aims to help the investigator by providing a list of popular filesystem paths that might contain evidence. Similarly, DISCLOSE aims to suggest to the investigator attack actions to be inspected that might contain and reveal evidence. Therefore the original CBR-FT is suitable for filesystem investigations while DISCLOSE does not utilize only a specific type of evidence (filesystem, memory dumps, network traffic etc). To bridge this difference, we simply treated the attack actions as paths but kept the logic and formulas used by CBR-FT.
- CBR-FT does not formalize a multistage investigation, i.e. does not use a stepwise approach to follow the investigation, but can still be used for one. Therefore, we utilized the CBR-FT formula to calculate the probabilities at the start of the investigation and at each step we allowed the Defender to select the attack action with the highest probability until the end of the budget.
- Finally, we decided to implement two versions of CBR-FT, where the first (CBR-FT v1) utilizes the trigger action that DISCLOSE uses, while the second (CBR-FT v2) one does not, in order to allow more comparison. Therefore, we respected this characteristic and implemented CBR-FT without a cost benefit element as well, to stay as close as possible to the original framework. framework.

The main purpose of this case study is to demonstrate the applicability of DISCLOSE on an investigation, to advise the Defender on optimal selection actions to take during the case investigation.

A. Attack action space

To evaluate DISCLOSE, we use a subset of TTPs from the MITRE ATT&CK Enterprise matrix as the attack space $\mathcal{A}$, presented in Table II. This is a representative manageable in size sample of the available TTPs that is used for demonstration and evaluation purposes. The usage of the whole TTP space was avoided for simplicity and readability reasons. Despite this being a small scale case study choosing an optimal inspection action during the dedicated time of the investigation is not trivial. The evaluation of the objective function that solves the investigator's optimization problem, i.e. the maximization of their potential payoff, makes this task significantly complex. This is because a number of variables,

such as the number of available reports, TTPs used by threat actors, relationships between those TTPs etc., change daily.

For the Defenders to be able to make informative and optimal choices, they would need to be able to keep up with these changes. However, the large amount of data that can must be considered during the investigation's decision making process, the complexity of their processing and the fast pace of the aforementioned changes make the task unachievable. For instance, it is impossible for a human to ingest all the new data that are being made available daily or to be aware and research every technique used by threat actors. Even by taking in consideration the specialization that a Defender may have, i.e. only work with Unix systems, this is an extremely hard task. This is simply due to the fact that a human cannot ingest and process a large amount of data and variables as efficiently as a computer. In order to help Defenders overcome these challenges, DISCLOSE automated the aforementioned tasks uses a data-driven approach to ingest and process all the necessary data in order to offer guidance and allow for faster and more efficient investigation.

B. Dependency matrix

As presented in Section III, one of the main components of DISCLOSE is the dependency matrix $\mathbf{T}$, which contains the probabilistic relationships between the available attack actions.

In order to calculate the probabilities of the the dependency matrix $\mathbf{T}$, we have utilized the STIX 2.0 GitHub repository[2] of the ATT&CK knowledge base to collect the relationship objects for each TTP and correlate them. More specifically, MITRE has made available through their TAXII 2.0 server and STIX 2.0 GitHub repository all the information that they collect and process on adversarial TTPs and APT groups. Using the STIX 2.0 standardized language they have created STIX Domain Objects (SDOs) for APT group, TTPs, software etc. Each object contains information such as name, reference IDs and a number of object specific fields. For example, a technique SDO has fields for the platform that this TTP can be used on, required permissions etc, while a group SDO has a field for known aliases. SDOs are "connected" with other SDOs using STIX Relationship Objects (SROs) that consist of at least three mandatory fields: relationship_type, source_ref and target_ref. For instance, if a TTP has been observed being used by a group, there will be a SRO connecting the TTP SDO to the group SDO, which, except from the three main fields, will also contain a list of publicly available reports that support this relationship.

To access this information, we have developed a number of Python scripts making use of the python-stix2[3] library, which allows users to query either a TAXII 2.0 server or a STIX 2.0 GitHub repository. We firstly, constructed queries to retrieve all the available TTP SDOs and for each of them all the corresponding SROs. We then extracted from each SRO the available report names and URLs and created a dictionary of TTPs and the reports they are referenced in. Finally, using this, we correlate the TTPs and create our dependency table using the corresponding formula for t_{ij} from Section III. For readability reasons, the contents of the dependency table, along with the rest of the evaluation data, can be found in the relevant repository[4].

We note that each inspection of an attack action i does not equal to a single action for the Defender, as this would be unrealistic. On the contrary it often includes the inspection of one or more data sources while searching for evidence of the same malicious action. This does not affect DISCLOSE, as we model each attack action as a TTP and thus the inspection of this attack action includes all activities related its analysis. For instance, to uncover evidence for the attack action 18, "Replication Through Removable Media", the Defender may inspect three different data sources namely: (i) events from process monitoring, (ii) a copy of the registry of the victim's computer, (iii) and events from monitoring file access within removable media devices, such as USBs. Ideally, the Defender should inspect all three sources in order to properly validate her findings.

This case study assumes that the victim organization collects the following data on an every day basis: (i) events by monitoring processes with command line (cmd) logging enabled, (ii) network connections, (iii) registry changes, (iv) PowerShell usage, (v) the access of crucial utilities in each computer in the infrastructure, (vi) network flow information as well as vii) a number of security event IDs from the Active Directory. Moreover, we assume that the organization has given to the Defender access to the victim computer, for the purpose of this investigation.

C. Proximity parameter

In this case study, we have chosen to calculate the proximity parameter d_{ij}, which was presented in Section IV, by combining the Cyber Kill Chain [4], which is one of the most well-known and accepted attack life cycle models, and the categories provided for each TTP by MITRE ATT&CK. MITRE splits their TTP categories and the TTP themselves under three different matrices: PRE-ATT&CK, Enterprise, Mobile and ICS [38]. The PRE-ATT&CK matrix contains adversarial activities that take place outside the infrastructure of the organization before the attack and are related to its preparation. This may include activities (TTPs) such as information gathering and preparation and testing of the adversarial infrastructure. The Enterprise matrix contains activities (TTPs) used by adversaries against enterprise networks during the attack campaign and covers most popular operating systems and platforms. Finally, the Mobile matrix contains TTPs against mobile devices while the ICS matrix is focused on Industrial Control Systems. The matrix that is relevant and used in the rest of this paper is the Enterprise, while the other three are considered out of scope.

The TTP categories in the Enterprise matrix are[5]: Initial Access, Execution, Persistence, Privilege Escalation, Defense Evasion, Credential Access, Discovery, Lateral Movement,

[2]https://github.com/mitre/cti

[3]https://github.com/oasis-open/cti-python-stix2

[4]https://github.com/isec-greenwich/disclose

[5]https://attack.mitre.org/techniques/enterprise/

TABLE III: MITRE ATT&CK categories mapped to kill chain phases with greyed out boxes corresponding to phases and categories that are out of DISCLOSE's scope.

	Cyber Kill Chain Phases					
	Reconnaissance & Weaponization	Delivery	Exploitation	Installation	Command & Control	Actions on Objectives
ATT&CK	PRE-ATT&CK	Initial Access	Execution Defense Evasion	Persistence Execution	Command and Control	Credential Access Discovery Collection Exfiltration Lateral Movement Privilege Escalation Impact Execution

Collection, Command and Control, Ex-filtration and Impact. Similarly, the Kill Chain phases are: Reconnaissance, Weaponization, Delivery, Exploitation, Installation, Command and Control and Actions on Objectives. Table III presents the mapping of the MITRE TTP categories on the Kill Chain phases. Essentially, each MITRE TTP Category, and thus the TTPs that fall under this, is mapped to the Kill Chain phase that is used for. For instance, the Initial Access category is mapped to the Delivery phase as it contains TTPs that enable the Attacker to deliver malicious software to the victim infrastructure and penetrate it. The first two kill chain phases, namely Reconnaissance and Weaponization, which are represented by the MITRE PRE-ATT&CK matrix, are greyed out because as explained are out of the scope of this paper.

D. Cost-benefit parameters

Table IV presents the benefit B_i and cost C_i values for each attack action i. Here we consider that the benefit depends on the severity of the attack action that is uncovered. As such, the values of the benefit vector have been calculated using the Common Vulnerability Scoring System (CVSS) [6].

CVSS is an publicly available framework that enables experts to access the characteristics and severity of software vulnerabilities and it consists of three metrics: Base, Temporal, and Environmental. In this paper we use the Base Score for computing the benefit value B_i of each attack action, but the rest of the metrics could be potentially used by investigators to personalize the values to a specific victim organization, as discussed later in Section VII.

In more detail, for each attack action we input its characteristics, such as the required privileges or user interaction, the vector of the action or its effect on each of the confidentiality, integrity and availability (CIA) triad properties, to the CVSS Base Score Calculator to obtain the relevant value. This value then reflects the severity of a TTP or a vulnerability if realized by the Attacker. For instance, attack action 6 (from Table II) has a CVSS Base Score $B_6 = 71$ as it uses the network as the attack vector to ex-filtrate the data, does not necessarily require high privileges or user interaction and affects both the confidentiality and the integrity of the data. Here it should be noted that although the Attacker aims to compromise the confidentiality of the data, they also unintentionally affect their integrity because just accessing any file they change the relevant timestamps.

Furthermore, in the absence of a standardized metric that can be used to valuate the cost C_i of an attack action, the cost values have been assigned using a two-phase interviews process of practitioners, similarly to [22] and [30]. During the first phase, six structured interviews were conducted with experienced cybersecurity professionals to identify the criteria to be used for the cost valuation as well as the rating scale. A range of different cybersecurity expertise were chosen for the interviews in order to acquire indicative values for the cost C_i. More specifically, the professionals chosen were: (i) one digital forensic expert, (ii) one penetration tester, (iii) one security consultant, (iv) two Security Information and Event Management (SIEM) experts and (v) one academic in incident response and digital forensics.

Further information regarding the background and experience of each interviewee can be found in Table V.

The criteria that were identified from the interviews were (a) the analysis type, i.e. the level of automation that the available tools provide for the analysis of a specific attack action, and (b) the analysis complexity, which is affected by the technical difficulty level of the analysis, the amount of data and the number of different data sources to be examined. Moreover, a rating scale from one to ten was chosen as a smaller scale was not able to represent properly the cost differences between the available attack actions.

At the second phase, cybersecurity practitioners were asked to rate the analysis cost for each attack action in a scale of one to ten the identified criteria were used in a second round of interviews based on the previously identified criteria. The participants' answers were in all cases either identical or no more than two values apart, which showcase that the cost values collected are indicative as the experts opinions converge. The results of this round can be found in Table IV.

[6]https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

TABLE IV: Calculated cost C_i and benefit B_i for each attack action i in the case study of this paper.

i	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
B_i	47	63	71	71	75	71	71	33	61	78	88	55	88	88	68	63	33	47	36	57	36	53	47	33	78	72	74	33	63	64	65
C_i	8	8	6	5	6	8	7	5	3	7	7	3	7	5	5	6	5	3	5	6	7	9	5	5	7	4	7	6	6	5	7

TABLE V: Background information of experts interviewed for gathering the cost values C_i used by DISCLOSE.

#	**Job Title**	**Main Duties**	**Experience (Years)**
1	Security Operations Advisor	Strategic advisory on SecOps	12
2	Digital Forensics Expert	DFI chapter leader for Advisory firm	20
3	Senior Lecturer in Incident Response & Digital forensics	Research, Consulting & Teaching in Incident Reponse / OWASP Dorset Chapter Leader	13
4	Enterprise Security Architect	Review and deploy of Security Architectures for new and modified solutions, Proof of concept or value for identified solutions, Mapping security controls to technical, administrative and physical controls	14
5	Senior Red Team Analyst	Scoping, leading, and executing intelligence-led and objectives-focused adversarial threat emulations. Current focus is: OSINT, social engineering, C&C channels, post-exploitation, and physical attacks	10
6	Associate security consultant	Penetration testing, reporting and client scoping	2

E. Incidents overview

For the purposes of this case study we have chosen to evaluate all the selected methods against three incidents. Each incident consists of attack actions from Table II. We have chosen incidents that are representative of the current threat landscape by utilizing the data we have acquired from the MITRE ATT&CK Repository in order to understand the popularity of each TTP and its usage. However, we did not want to evaluate the performance of the selected methods simply against the most common incidents and TTPs, but also against less common but impactful ones. We would like to see how the methods would perform against an incident that includes TTPs that may not be amongst the most popular but still are highly correlated with the rest of the incident's attack actions. Essentially this means that maybe a TTP is not very popular in general, but it is performed with very high probability after or before another TTP that has already been uncovered.

The first cyber incident in this case study consists of a subset of the attack actions $\mathcal{A}_P^{(1)}$ that were presented in V-A. More specifically, we assume that $\mathcal{A}_P^{(1)}$=$\{13,\ 17,\ 12,\ 10,\ 15,\ 8,\ 2,\ 6\}$.

Initially the Attacker crafts and sends a spearphising email ($A_{P1}^{(1)} = 13$) to one of the employees of the target organization. The spearphising email contains a malicious Word (docx) file with a PowerShell payload that will be executed through the exploitation of a Microsoft Office vulnerability when the file is opened. When the employee receives the email, downloads the malicious attachment and opens it, the payload is executed ($A_{P2}^{(1)} = 17$). This allows a PowerShell script to run, which initiates a connection from the victim to the Attacker and therefore creates a channel of communication, i.e. a command and control (C&C) channel ($A_{P3}^{(1)} = 12$ and $A_{P4}^{(1)} = 10$).

The Attacker then uses the C&C channel ($A_{P5}^{(1)} = 15$) to execute a number of malicious PowerShell commands and scripts. Next, a new service is created through the established C&C channel as a persistence mechanism for the Attacker ($A_{P6}^{(1)} = 8$). Finally, the Attacker uses the C&C channel to locate ($A_{P7}^{(1)} = 2$) and ex-filtrate the discovered data from the victim computer ($A_{P8}^{(1)} = 6$).

For space reasons, even though DISCLOSE and the rest of the selected frameworks will be used against three attack scenarios in this section, we will only present here in detail the description for one of them. Similarly, the other two attack scenarios are $\mathcal{A}_P^{(2)}$=$\{14,\ 17,\ 10,\ 12,\ 15,\ 11,\ 23,\ 0,\ 24\}$ and $\mathcal{A}_P^{(3)}$=$\{30,\ 29,\ 15,\ 10,\ 12,\ 0\}$.

F. Evaluation

In this subsection, we evaluate DISCLOSE and compare its performance to the rest of the selected policies, in two ways: (i) using a range of budget values and (ii) using a fixed budget. This allow us to compare the four methods in a higher level and a more detailed way.

In both cases, all policies are evaluated against all three incidents, i.e. $\mathcal{A}_P^{(1)}$, $\mathcal{A}_P^{(2)}$ and $\mathcal{A}_P^{(3)}$. For each incident, we run every policy three times, each time using a different trigger action $\mathcal{Y}^{(1)}$.

1) Evaluation against different budget values: Figure 2 presents the performance of all four policies, DISCLOSE, baseline, CBR-FT v1 and CBR-FT v2, against different budget values. Essentially, we have allowed each policy to run without a budget limit until all performed attack actions are revealed. Each of the four subfigures corresponds to one of the four policies and depicts its performance in all nine performance instances, i.e. three incidents with three different trigger actions. The group legend under the figure depicts which line corresponds to each pair of incident and trigger action. Each incident is depicted in a different color and each trigger action uses a different line. $\mathcal{A}_P^{(1)}$ is depicted in blue, $\mathcal{A}_P^{(2)}$ in green and $\mathcal{A}_P^{(3)}$ in red. The first trigger of each scenario is depicted with a dashed line, the second with a sold line and the third with a dotted line. For instance, for all policies the green solid line represents DISCLOSE's performance in the second incident when the trigger action is attack action 15, which is noted on the legend as $\mathcal{A}_P^{(2)}, \mathcal{Y}^{(1)} = \{15\}$.

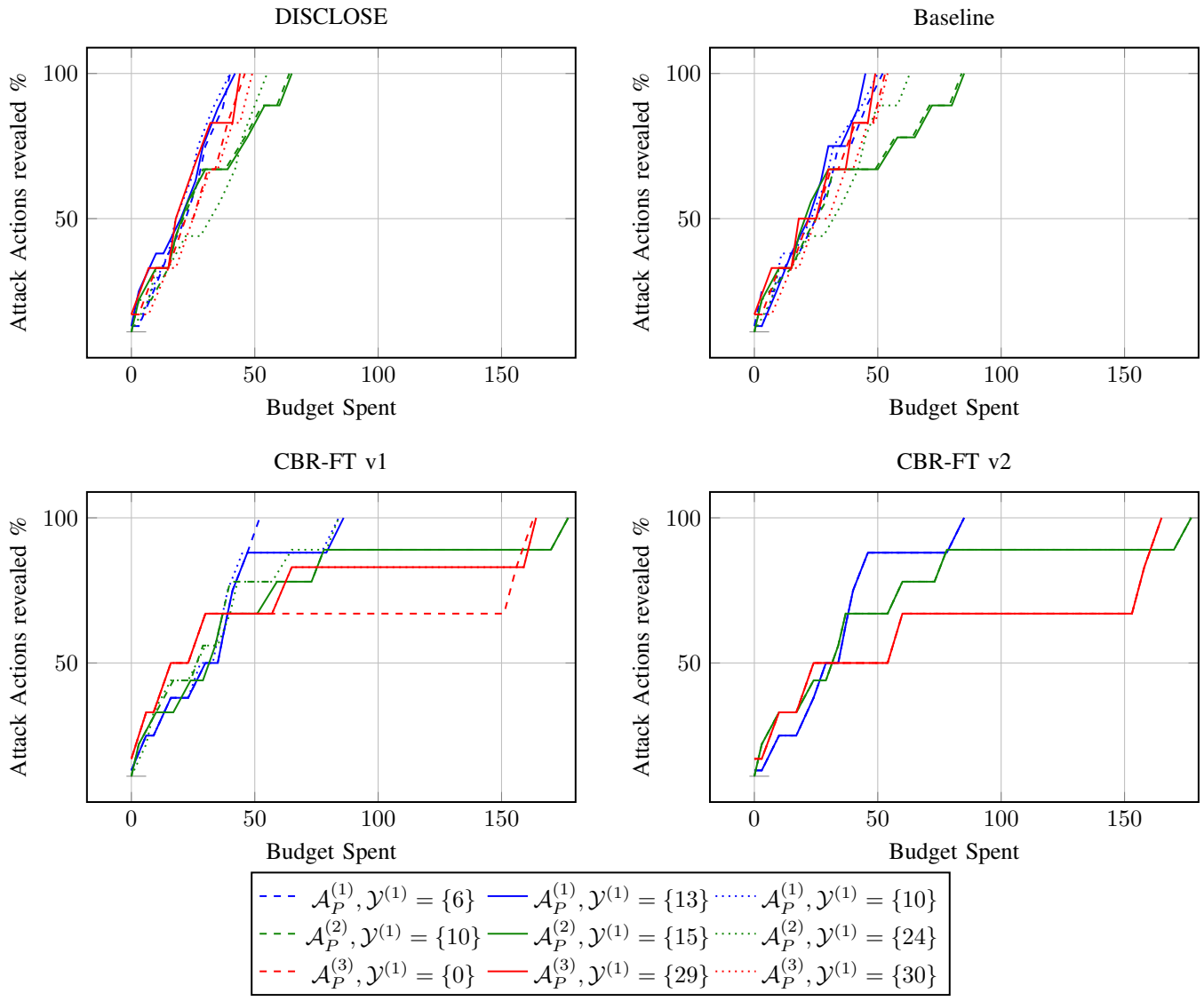

Fig. 2: Performance of the four evaluated policies (one graph per policy) in terms of percentages of attack actions revealed, towards the overall incident disclosure, given different budget values. For a given budget, the best policy is always the one that unveils the highest, among all competing policies, percent of actions. $\mathcal{Y}^{(1)}$ is the trigger attack action, while $\mathcal{A}_P^{(x)}$ represents incident x. In all graphs, DISCLOSE outperforms all other three policies.

Remark 1: As can be seen in all cases, both versions of CBR-FT and the baseline required a larger budget for the inspection of all the attack actions due to a high number of attack actions that were not part of the attack incident.

More specifically, the largest budget spent by DISCLOSE to fully investigate any scenario was 65, for the baseline 85, for CBR-FT v1 and CBR-FT v2 177.

However, this does not mean that the suggestion of extra attack actions is always ineffectual. It is expected that a number of extra attack actions will be suggested to the defender, as DISCLOSE does not detect the incident but rather supports the decisions based on the dependency table $\mathbf{T}$ the proximity parameter and the results of the previous inspections. Thus the distinction between an effective and an ineffectual extra suggestion depends on if the suggested attack action could potentially fit in the incidents based on the previously discovered actions. This will be further presented and discussed in the next subsection, Section V-F2.

2) Evaluation with fixed budget: Tables VII and VIII present the results of the four policies for all the three attack scenarios and different trigger attack actions. Specifically, for each policy the results depicted are the percentage of uncovered attack actions, the extra suggested actions and the collected payoff in each case. Similarly Table VI holds all the attack actions that each of the four policies suggested in all nine cases to the investigator, i.e. $\mathcal{Y}^{(k)} \cup \mathcal{N}^{(k)}$, in the order that they were selected starting with the trigger action. In each case we note in bold the extra attack actions that are not part of the incident under investigation but were suggested.

In all three, a fixed budget between 45 and 65 has been used based on the average values of the overall cost and the length of each scenario. This comparison aims to compare the

suggestions of each approach in greater detail than Figure 2, so as to evaluate the amount of budget spent and assess why the different policies suggest more extra attack actions that are not part of the incident and if these are effective or ineffectual.

Remark 2: The Defender using the DISCLOSE policy inspected all the performed attack actions in all cases before running out of budget, while the Defender using the other policies missed some of them. For instance the DISCLOSE policy during the investigation of attack scenario 1 suggested the inspection of attack action 11, which represents the Registry Run Keys / Startup Folder TTP. This is an acceptable outcome for a number of reasons. Firstly, in every investigation it is expected that the actions performed will not always reveal evidence. For instance, the Defender can always reject one of the proposed attack actions and the algorithm should provide a new one. Secondly and most importantly, in the current threat landscape, this attack action could potentially be part of the attack scenario based on the previously discovered attack actions.

In the same case the baseline and both versions of CBR-FT suggested the inspection of attack action 14. However, attack action 14 represents the Spearphishing Link TTP, which does not fit with the rest of the performed attack actions in scenario 1. This is another initial access TTP, attack action 13, has already been uncovered as can be observed in Table VI. Therefore, attack action 14 should have not been suggested and would most probably be wasting the Defender's budget. Similarly, in scenario 2 attack action 29 (Web Shell) is suggested by the baseline and attack action 13 (Spearphishing Attachment) is suggested by CBR-FT v1 and v2. In the first case (i) a persistence TTP has already been uncovered (attack action 11) and (ii) it does not fit with the rest of the incident. In the second case attack action 14 was already uncovered which is also an initial access TTP and thus attack action 13 was not probable.

In all cases, baseline, CBR-FT v1 and v2 suggested a higher number of extra attack actions to be inspected than the DISCLOSE policy (see Table VII and VIII).

More importantly, many of the suggested attack actions cannot conceptually fit in the incident under investigation. **Remark 3**: In most cases, this results in the exhaustion of the available budget before the revelation of all the performed attack actions. It is worth noting that the number of attack actions that will be suggested and inspected but not reveal evidence will vary based on the TTPs performed in the incident and the current attack trends which are represented by the dependency matrix $\mathbf{T}$.

Remark 4: Another important observation from Table VIII is that both versions of CBR-FT produce the same ordering every time irrespective of the scenario under investigation. This is being caused by the fact that the probabilities used by CBR-FT for suggestions are calculated once at the start of the investigation and represent the probability of each attack action being taken by the attacker, not for this specific investigation, but generally. More specifically, in the calculation the number of reports that mention a specific TTP as part of an attack is divided with the number of reports of all TTP, which is a very large number. Thus, the produced information is very generic and not able to capture the relations between TTPs. On the contrary DISCLOSE uses the progression of the investigation so far and the probabilistic relations between actions to calculate the probabilities for each attack action at each step, which is how it is able to achieve a better performance.

Remark 5: Moreover, for the same reasons both versions of CBR-FT are able to suggest only the parts of the incident that

TABLE VI: Comparison of inspected actions by each policy for all investigated scenarios. $\mathcal{Y}^{(1)}$ notes the trigger action, $\mathcal{A}_P^{(x)}$ represents incident x and $\mathcal{Y}^{(k)} \cup \mathcal{N}^{(k)}$ notes all selected actions. Extra attack actions are noted in **bold**.

	$\mathcal{Y}^{(1)}$	Policy	$\mathcal{Y}^{(k)} \cup \mathcal{N}^{(k)}$
$\mathcal{A}_P^{(1)}$	6	Disclose	6, **11**, 15, 2, 12, 13, 17, 10, 8
		Baseline	6, **11**, 15, 2, 12, 13, 17, **14**, 10
		CBR-FT1	**6**, 15, **11**, 12, **9**, 10, 13, **14**, 17
		CBR-FT2	15, **11**, 12, **9**, 10, 13, **14**, 17, 8, **16**
	13	Disclose	13, 12, 10, 17, 8, 2, 15, 6
		Baseline	13, 17, **14**, 12, **11**, 10, 15, 8, **29**, 2
		CBR-FT1	13, 15, **11**, 12, **9**, 10, **14**, 17, 8
		CBR-FT2	15, **11**, 12, **9**, 10, 13, **14**, 17, 8, **16**
	10	Disclose	10, **11**, 12, 13, 17, 15, 8, 2, 6
		Baseline	10, **11**, 12, 13, 17, **14**, 15, 8, **29**, 2
		CBR-FT1	10, 15, **11**, 12, **9**, 13, **14**, 17, 8, 2
		CBR-FT2	15, **11**, 12, **9**, 10, 13, **14**, 17, 8, **16**
$\mathcal{A}_P^{(2)}$	10	Disclose	10, 11, **13**, 12, 15, 17, 14, **8**, **4**, 0, 24, **2**, 23
		Baseline	10, 11, **13**, 12, 15, 17, 14, **8**, **29**, **4**, **2**, 0, **30**
		CBR-FT1	10, 15, 11, 12, **9**, **13**, 14, 17, **8, 2**, **16**, 0
		CBR-FT2	15, 11, 12, **9**, 10, **13**, 14, 17, **8**, **2**, **16**, 0, **25**
	15	Disclose	15, 11, 12, **13**, 17, 14, 10, **8**, **4**, 0, 24, **2**, 23
		Baseline	15, 11, 12, **13**, 17, 14, 10, **8**, **29**, **4**, **2**, 0, **30**
		CBR-FT1	15, 11, 12, **9**, 10, **13**, 14, 17, **8**, **2**, **16**, 0
		CBR-FT2	15, 11, 12, **9**, 10, **13**, 14, 17, **8**, **2**, **16**, 0, **25**
	24	Disclose	24, 10, 11, **13**, 12, **2**, 0, 15, 17, 14, 23
		Baseline	24, 10, 11, **13**, 12, **2**, 0, 15, 17, 14, **1**, 23
		CBR-FT1	24, 15, 11, 12, **9**, 10, **13**, 14, 17, **8**, **2**, **16**
		CBR-FT2	15, 11, 12, **9**, 10, **13**, 14, 17, **8**, **2**, **16**, 0, **25**
$\mathcal{A}_P^{(3)}$	0	Disclose	0, **11**, 10, 12, **13**, **17**, 15, **8**, 29, 30
		Baseline	0, **11**, 10, **13**, 12, **17**, 15, **8**, 30, **26**
		CBR-FT1	0, 15, **11**, 12, **9**, 10, **13**, **14**, **17**, **8**, **2**
		CBR-FT2	15, **11**, 12, **9**, 10, **13**, **14**, **17**, **8**, **2**, **16**, **25**
	29	Disclose	29, 30, **26**, 15, 10, 12, **17**, **28**, 0
		Baseline	29, 30, **26**, 15, **11**, 10, 12, **13**, **17**, **8**
		CBR-FT1	29, 15, **11**, 12, **9**, 10, **13**, **14**, **17**, **8**, **2**
		CBR-FT2	15, 11, 12, 9, 10, 13, 14, 17, 8, 2, 16, 25
	30	Disclose	30, 26, 0, 11, 10, 12, 17, 15, 8, 29
		Baseline	30, **26**, 0, **11**, 10, 12, **13**, **17**, 15, **8**
		CBR-FT1	30, 15, **11**, 12, **9**, 10, **13**, **14**, **17**, **8**, **2**
		CBR-FT2	15, **11**, 12, **9**, 10, **13**, **14**, **17**, **8**, **2**, **16**, **25**

TABLE VII: Collected Payoff, Revealed and Extra Actions for DISCLOSE and Baseline using fixed budgets on different incidents and trigger actions. $\mathcal{Y}^{(1)}$ notes the trigger action, while $\mathcal{A}_P^{(x)}$ represents incident x.

	$\mathcal{Y}^{(1)}$	DISCLOSE approach			Baseline		
		Revealed	Extra Inspections	Payoff	Revealed	Extra Inspections	Payoff
$\mathcal{A}_P^{(1)}$	6	100%	11	577	88%	11, 14	516
	13	100%		577	88%	14, 11, 29	506
	10	100%	11	577	88%	11, 14, 29	506
$\mathcal{A}_P^{(2)}$	10	100%	13, 8, 4, 2	567	78%	13, 8, 29, 4, 2, 30	456
	15	100%	13, 8, 4, 2	567	78%	13, 8, 29, 4, 2, 30	456
	24	100%	13, 2	567	100%	13, 2, 1	567
$\mathcal{A}_P^{(3)}$	0	100%	11, 13, 17, 8	415	86%	11, 13, 17, 8, 26	351
	29	100%	26, 17, 28	415	86%	26, 11, 13, 17, 8	368
	30	100%	26, 11, 17, 8	415	86%	26, 11, 13, 17, 8	351

TABLE VIII: Collected Payoff, Revealed and Extra Actions for CBR-FT v1 and v2 using fixed budgets on different incidents and trigger actions. $\mathcal{Y}^{(1)}$ notes the trigger action, while $\mathcal{A}_P^{(x)}$ represents incident x.

	$\mathcal{Y}^{(1)}$	CBR-FT v1			CBR-FT v2		
		Revealed	Extra Inspections	Payoff	Revealed	Extra Inspections	Payoff
$\mathcal{A}_P^{(1)}$	6	75%	11, 9, 14	445	75%	11, 9, 14, 16	435
	13	75%	11, 9, 14	435	75%	11, 9, 14, 16	435
	10	88%	11, 9, 14	506	75%	11, 9, 14, 16	435
$\mathcal{A}_P^{(2)}$	10	78%	9, 13, 8, 2, 16	456	78%	9, 13, 8, 2, 16, 25	456
	15	78%	9, 13, 8, 2, 16	456	78%	9, 13, 8, 2, 16, 25	456
	24	78%	9, 13, 8, 2, 16	487	78%	9, 13, 8, 2, 16, 25	456
$\mathcal{A}_P^{(3)}$	0	66%	11, 9, 13, 14, 17, 8, 2	286	50%	11, 9, 13, 14, 17, 8, 2, 16, 25	239
	29	66%	11, 9, 13, 14, 17, 8, 2	303	50%	11, 9, 13, 14, 17, 8, 2, 16, 25	239
	30	66%	11, 9, 13, 14, 17, 8, 2	304	50%	11, 9, 13, 14, 17, 8, 2, 16, 25	239

correspond to popular TTPs. Thus less popular attack actions will be missed. For instance, if an attack action is generally not very popular but it is highly correlated with a previously revealed action, it will be missed by CBR-FT. On the contrary, DISCLOSE will be most probably able to suggest it for the reasons explained in **Remark 4**.

Remark 6: Finally, in some instances ($\mathcal{A}_P^{(1)}$, Trigger action 10 or $\mathcal{A}_P^{(3)}$) CBR-FT v2 achieves a slightly better performance than CBR-FT v1, but still worse than both DISCLOSE and the baseline. The reason for this increase is that in many cases the initial trigger action, which is give to CBR-FT v2, would be missed if it was not provided.

VI. Discussion and future work

In a real-world case, Defenders of any kind (security analysts, forensic investigators, SIEM experts etc.) may use DISCLOSE to assist their multi-stage investigations in order to increase their efficiency and be able to cope with a larger range of investigations. An investigator is expected to input to DISCLOSE all the information known at the start of the investigation, by selecting one or more trigger actions, as well as an estimate of the available budget (time). DISCLOSE uses this data, updates the relevant matrices, calculates the payoff values for each attack action, suggests the next attack action for inspection and prints all relevant data, such as data sources, event ID, patterns, as it is pulled from the MITRE ATT&CK repository. At the moment DISCLOSE only suggests one attack action at each step but this can tailored to offer more than one suggestions for inspection allowing the investigator to choose from them. Further, DISCLOSE assumes that the investigator takes up every recommended inspection. However, in reality the investigator might choose not to follow a suggestion for a number of reasons and would like to receive another one. In the future, we plan to expand DISCLOSE to ingest the inspector's choice to reject the proposed inspection and use this knowledge into the calculations of the succeeding proposed inspections. Yet, upon completion of the inspection of an attack, the inspector could feedback information to DISCLOSE about the success of the taken inspection improving the effectiveness of the framework in each investigation step.

In its current version, DISCLOSE considers the efficacy of the investigator as known. For ease of presentation, we have assumed that the efficacy of the investigator equals 1, namely she always discloses the attack action mapped to the selected inspection as long as this actions was taken during the cyber incident. Making a more realistic assumption, this efficacy depends on the experience of the investigator. Nevertheless,

this is a straightforward addition to DISCLOSE and will be presented in our future work.

Furthermore, in the presented case study the dependency table $\mathbf{T}$ was calculated using the MITRE ATT&CK STIX repository. However, it is expected that for a real world application a number of different sources will be aggregated for the calculation of $\mathbf{T}$ in order to align with prominent cyber threats increasing the performance of DISCLOSE. For instance, the values of the dependency table $\mathbf{T}$ could be enhanced if combined with data about attack trends extracted by social media, such as twitter. In many cases, information regarding attack patterns and TTPs shared via social media either does not appear in cybersecurity reports. Similarly, DISCLOSE currently uses only the probabilistic relations between the attack actions and the proximity parameter used to guide the suggested inspections. In future work DISCLOSE can be improved by using Indicators of Compromise (IoCs) by being entered during the inspection and matched by DISCLOSE against threat intelligence databases.

As a result of the dependency of table $\mathbf{T}$ on past incident data, one could argue that DISCLOSE is limited to the investigation of only previously published vulnerabilities. However, the usage of TTPs makes DISCLOSE applicable across an ample range of incidents, as TTPs are abstracted from specific attack events or vulnerabilities and are focused on describing the adversarial behavior itself. When the investigator inspects a TTP, any relevant exploited vulnerabilities (including zero-day ones) could be identified, depending on the investigator's analytic capabilities and complementary, to DISCLOSE, software tools. Thus, DISCLOSE supports investigators with choosing TTPs optimally while the disclosure of any zero-day exploitation is subject to the above factors.

One of the assumptions that was made in our case study was that the incident involves a single host, nonetheless in most real life scenarios this is not true. In the case of a real world investigation, table $\mathbf{T}$ would act as a look up template table which would automatically be "duplicated" as many times as needed during the investigation, based on the involved assets. This allows the proposed method to be extendable and easily applied to any investigation, regardless of the number of assets involved.

Regarding the data sources to be inspected by the investigator, this depends on the availability of data at the time of the investigation and it is linked to the forensic readiness of each organization. Every organization collects data from different sources and uses them for online detection through a number of mechanisms, such as firewalls, IDS, SIEM, and Endpoint Detection and Response. Ideally, the collection of more data should equal greater visibility within the organization and better forensic readiness. However, this suffers from the excessive data volume that infrastructures produce everyday. It is also challenging for organizations to proactively collect data, due to privacy legislations, while they have to justify the intended purpose of this collection through its relevance to cybersecurity decisions and systems. Data such as memory dumps, copies of the registry of a computer and network traffic could only be collected on request.

In our case study, the values of the benefit vector were calculated using the CVSS Base Score for each attack action. However, in many cases the benefit of some attack actions may vary based on the organization's operations, the nature, or the importance of the inspected asset. For instance, an organization may prioritize the confidentiality of its stored data over its integrity determining uniquely the benefit values of available inspections. Undoubtedly, the Defender should be able to easily modify the benefit vector according to their requirements. This can be achieved by combining the CVSS Environmental Score with the Base Score to calculate the overall CVSS Score. We expect that in the beginning of the investigation the analyst has access to information regarding the significance of assets. This can be made available either from an inventory of assets, via interviews with asset stakeholders who may have perform asset pricing, and the Defender's experience with identifying impact based on previous incidents. Upon capturing this data, the Defender can automatically revise the benefit vector, based on the Environment Score of CVSS. In the current model an inspection is beneficial to the Defender only when discloses an attack action. In practice even when the inspection of i is unsuccessful it will lead to some knowledge acquisition enabling the Defender to realize what has not happened narrowing down the number of inspection choices for the rest of the investigation. Future work of ours will use this knowledge acquisition as part of determining the DISCLOSE's recommended inspections. Finally, regarding the cost vector we assigned values using structured interviews with six experienced cybersecurity professionals. However, a higher number of interviews would result in more generalized cost values, which would make DISCLOSE more robust and applicable in an ample variety of cases.

VII. Conclusion

A forensic analysis team will often have limited resources to investigate incidents while trying to minimize the impact to the organization's assets in a timely and effective manner. Analysts are required to constantly develop their skills and expand their knowledge and methodology in order to address the complexity that is stemming from the emergence of new adversarial techniques and technologies.

To this end, this paper proposed a novel model for decision support which aims to help the analyst overcome these challenges. We based our model on the creation of an attack action space created via public knowledge bases of MITRE ATT&CK TTPs, as well as forensic methodologies and resources. In this way we empower the analyst to progress through the investigation in an efficient way based on the recommended inspection actions of our model. We take into consideration the probabilistic relation and proximity values between the available attack actions, the ongoing findings of an investigation, the benefit and cost of each inspection, and the available budget of the investigator.

We demonstrate and evaluate DISCLOSE using three realistic incidents and data collected from the MITRE ATT&CK STIX repository, six structured interviews with experienced cybersecurity professionals and the Common Vulnerability Scoring System (CVSS). Our evaluation suggests that: (a)

DISCLOSE is capable of supporting investigators, increasing the efficiency of the investigation process while minimizing the required budget and (b) the proximity parameter used by DISCLOSE outperforms the approach that uses only extracted probabilistic relations as well as both versions the CBR-FT inspired policy.

APPENDIX A
TABLE IX: List of acronyms.

Acronym	Description
APT	Advanced Persistent Threats
C&C	Command & Control
CIA	Confidentiality, Integrity & Availability
CVSS	Common Vulnerability Scoring System
IoC	Indicators of Compromise
SDO	STIX Domain Objects
SIEM	Security Information and Event Management
SRO	STIX Relationship Objects
STIX	Structured Threat Information eXpression
TAXII	Trusted Automated eXchange of Indicator Information
TTP	Tactics, Techniques, and Procedures

REFERENCES

[1] K. Finnerty, S. Fullick, H. Motha, J. N. Shah, M. Button, and V. Wang, "Cyber security breaches survey 2019," 2019.

[2] I. Security, "Cost of a data breach report 2019," 2019.

[3] A. Brinson, A. Robinson, and M. Rogers, "A cyber forensics ontology: Creating a new approach to studying cyber forensics," *Digital Investigation*, vol. 3, pp. 37–43, 2006.

[4] L. Martin, "Cyber kill chain®," *URL: http://cyber. lockheedmartin. com/hubfs/Gaining the Advantage Cyber Kill Chain. pdf*, 2014.

[5] V. Diaz, D. Emm, and C. Raiu, "Kaspersky security bulletin 2019: Advanced threat predictions for 2020," 2019.

[6] K. Kent, S. Chevalier, T. Grance, and H. Dang, "Guide to integrating forensic techniques into incident response," *NIST Special Publication*, vol. 10, no. 14, pp. 800–86, 2006.

[7] J. Williams, "Acpo good practice guide for digital evidence," *Metropolitan Police Service, Association of chief police officers, GB*, 2012.

[8] V. S. Harichandran, F. Breitinger, I. Baggili, and A. Marrington, "A cyber forensics needs analysis survey: Revisiting the domain's needs a decade later," *Computers & Security*, vol. 57, pp. 1–13, 2016.

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

[10] J. Navarro, A. Deruyver, and P. Parrend, "A systematic survey on multi-step attack detection," *Computers & Security*, vol. 76, pp. 214–249, 2018.

[11] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications surveys & tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[12] S. Alharbi, J. Weber-Jahnke, and I. Traore, "The proactive and reactive digital forensics investigation process: A systematic literature review," in *International Conference on Information Security and Assurance*. Springer, 2011, pp. 87–100.

[13] A. Nisioti, A. Mylonas, P. D. Yoo, and V. Katos, "From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3369–3388, 2018.

[14] S. M. Milajerdi, R. Gjomemo, B. Eshete, R. Sekar, and V. Venkatakrishnan, "Holmes: real-time APT detection through correlation of suspicious information flows," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 1137–1152.

[15] M. N. Hossain, J. Wang, O. Weisse, R. Sekar, D. Genkin, B. He, S. D. Stoller, G. Fang, F. Piessens, E. Downing *et al.*, "Dependence-preserving data compaction for scalable forensic analysis," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1723–1740.

[16] M. N. Hossain, S. M. Milajerdi, J. Wang, B. Eshete, R. Gjomemo, R. Sekar, S. Stoller, and V. Venkatakrishnan, "SLEUTH: Real-time attack scenario reconstruction from COTS audit data," in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 487–504.

[17] M. N. Hossain, S. Sheikhi, and R. Sekar, "Combating dependence explosion in forensic analysis using alternative tag propagation semantics."

[18] W. Wang and T. E. Daniels, "Network forensics analysis with evidence graphs (demo proposal)," in *Proceedings of the digital forensic research workshop*, 2005.

[19] ——, "Building evidence graphs for network forensics analysis," in *21st Annual Computer Security Applications Conference (ACSAC'05)*. IEEE, 2005, pp. 11–pp.

[20] H. Studiawan, C. Payne, and F. Sohel, "Graph clustering and anomaly detection of access control log for forensic purposes," *Digital Investigation*, vol. 21, pp. 76–87, 2017.

[21] A. of Chief Police Officers, "Acpo good practice guide for digital evidence for digital evidence," *URL:https://www.digital-detective.net/digital-forensics-documents/ACPO_Good_Practice_Guide_for_Digital_Evidence_v5.pdf*, 2012.

[22] G. Horsman, "Formalising investigative decision making in digital forensics: Proposing the digital evidence reporting and decision support (DERDS) framework," *Digital Investigation*, vol. 28, pp. 146–151, 2019.

[23] N. L. Beebe and J. G. Clark, "A hierarchical, objectives-based framework for the digital investigations process," *Digital Investigation*, vol. 2, no. 2, pp. 147–167, 2005.

[24] S. Rekhis and N. Boudriga, "A system for formal digital forensic investigation aware of anti-forensic attacks," *IEEE transactions on information forensics and security*, vol. 7, no. 2, pp. 635–650, 2011.

[25] J. Kolodner, *Case-based reasoning*. Morgan Kaufmann, 2014.

[26] B. Turnbull and S. Randhawa, "Automated event and social network extraction from digital evidence sources with ontological mapping," *Digital Investigation*, vol. 13, pp. 94–106, 2015.

[27] S. Soltani and S. A. H. Seno, "A formal model for event reconstruction in digital forensic investigation," *Digital Investigation*, vol. 30, pp. 148–160, 2019.

[28] S. Saad and I. Traore, "Method ontology for intelligent network forensics analysis," in *2010 Eighth International Conference on Privacy, Security and Trust*. IEEE, 2010, pp. 7–14.

[29] A. Nieto, "Becoming judas: Correlating users and devices during a digital investigation," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3325–3334, 2020.

[30] G. Horsman, C. Laing, and P. Vickers, "A case-based reasoning method for locating evidence during digital forensic device triage," *Decision Support Systems*, vol. 61, pp. 69–78, 2014.

[31] L. F. da Cruz Nassif and E. R. Hruschka, "Document clustering for forensic analysis: An approach for improving computer inspection," *IEEE transactions on information forensics and security*, vol. 8, no. 1, pp. 46–54, 2012.

[32] R. I. de Braekt, N.-A. Le-Khac, J. Farina, M. Scanlon, and T. Kechadi, "Increasing digital investigator availability through efficient workflow management and automation," in *2016 4th International Symposium on Digital Forensic and Security (ISDFS)*. IEEE, 2016, pp. 68–73.

[33] C. Yan, B. Li, Y. Vorobeychik, A. Laszka, D. Fabbri, and B. Malin, "Get your workload in order: Game theoretic prioritization of database auditing," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 2018, pp. 1304–1307.

[34] C. Liu, A. Singhal, and D. Wijesekera, "Using attack graphs in forensic examinations," in *2012 Seventh International Conference on Availability, Reliability and Security*. IEEE, 2012, pp. 596–603.

[35] ——, "Mapping evidence graphs to attack graphs," in *2012 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2012, pp. 121–126.

[36] ——, "Creating integrated evidence graphs for network forensics," in *IFIP International Conference on Digital Forensics*. Springer, 2013, pp. 227–241.

[37] M. Barrère, R. V. Steiner, R. Mohsen, and E. C. Lupu, "Tracking the bad guys: An efficient forensic methodology to trace multi-step attacks using core attack graphs," in *2017 13th International Conference on Network and Service Management (CNSM)*. IEEE, 2017, pp. 1–7.

[38] MITRE ATT&CK. Accessed: Jun. 22, 2020. [Online]. Available: https://attack.mitre.org

[39] B. Stojanović, K. Hofer-Schmitz, and U. Kleb, "APT datasets and attack modeling for automated detection methods: A review," *Computers & Security*, p. 101734, 2020.