IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 30 May 2022, accepted 26 July 2022, date of publication 4 August 2022, date of current version 11 August 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3196362

APPLIED RESEARCH

Discovering Coordinated Groups of IP Addresses Through Temporal Correlation of Alerts

MARTIN ZADNIK[1], JAN WRONA[1], KAREL HYNEK[1,2], TOMAS CEJKA[1], AND MARTIN HUSÁK[3]
[1]CESNET Association of Legal Entities, 160 00 Prague, Czech Republic
[2]Faculty of Information Technology, Czech Technical University in Prague, 160 00 Prague, Czech Republic
[3]Institute of Computer Science, Masaryk University, 602 00 Brno, Czech Republic

Corresponding author: Martin Husák (husakm@ics.muni.cz)

This work was supported in part by the Grant Agency of the Czech Technical University in Prague (CTU) through the MEYS of the Czech Republic under Grant SGS20/210/OHK3/3T/18, in part by the MEYS of the Czech Republic through the e-INFRA CZ Project under Grant LM2018140, and in part by ERDF ''CyberSecurity, CyberCrime and Critical Information Infrastructures Center of Excellence'' under Grant CZ.02.1.01/0.0/0.0/16_019/0000822.

ABSTRACT Network-based monitoring and intrusion detection systems generate a high number of alerts reporting the suspicious activity of IP addresses. The majority of alerts are dropped due to their low relevance, low priority, or due to high number of alerts themselves. We assume that these alerts still contain valuable information, namely, about the coordination of IP addresses. Knowledge of the coordinated IP addresses improves situational awareness and reflects the requirement of security analysts as well as automated reasoning tools to have as much contextual information as possible to make an informed decision. To validate our assumption, we introduce a novel method to discover the groups of coordinated IP addresses that exhibit a temporal correlation of their alerts. We evaluate our method on data from a real sharing platform reporting approximately 1.5 million alerts per day. The results show that our method can indeed discover groups of truly coordinated IP addresses.

INDEX TERMS Alerts, clustering, correlation, IP address, situational awareness.

I. INTRODUCTION

Nowadays, organizations are aware or become aware very fast that the Internet is not a safe place for their applications, services, infrastructure, or users. To face this challenge, the organizations deploy measures to reduce the threat landscape and, subsequently, the measures to recognize that a network threat materializes into an attack. The latter measures include various detection systems such as intrusion detection systems, network behavioral analysis systems, or honeypots. The survey of approaches to network-based intrusion and anomaly detection was elaborated by Drašar *et al.* in [1]. These systems are capable of monitoring the network activity and generating alerts reporting suspicious activity to Security Operation Center (SOC), Computer Security Incident Response Team (CSIRT), or Computer Emergency Response Team (CERT). Unfortunately, the suspicious activities are constant in the network, which renders it difficult to assess the high number of alerts generated by these detection systems manually [2]. Another issue of these detection systems is the high number of alerts that are not relevant to the particular assets, networks, services, or users [1], [3].

These issues are approached from various perspectives, which can be summarized into filtration [4], aggregation [5], prioritization [6], and correlation [7]. Since the correlation techniques are the most relevant for our research, we further elaborate on them in our related work.

While the previous research is focused on finding so-called meta-alerts, i.e., alerts composed of other alerts, for example, to express a multi-step attack, our aim is different. We propose to discover coordinated groups of IP addresses through the temporal correlation of their respective alerts. In comparison to the previous correlation techniques, we can benefit from the high number of alerts regardless of whether they are relevant or not. Therefore, we consider our research complementary to the previous research as it infers hidden

The associate editor coordinating the review of this manuscript and approving it for publication was Francisco J. Garcia-Penalvo.

information that would be lost by filtering, aggregating, prioritizing, and correlating the alerts into the meta-alerts. Inferred information does not contain any data about the particular attack scenario but rather improves situational awareness with knowledge of groups of coordinated IP addresses.

We argue that the knowledge of the groups of coordinated IP addresses (e.g., stored in a database) provides extra contextual information from the perspective of network digital forensics as well as from the perspective of automated reaction to the detected attacks. Prior to this work, such knowledge was inferred based on the same victim's IP address and port numbers or based on a sequence of alerts. We introduce an additional temporal perspective that complements previous approaches.

During an investigation of a cybersecurity incident, the analysts look for as much context as possible. Their goal is to obtain a global view of the incident to be able to answer questions such as what preceded the attack itself, are there other IP addresses involved? By consulting the database of the groups of IP addresses, the analyst finds out that the attacking IP address belongs to a larger group of IP addresses that regularly exhibit coordinated malicious behavior. Now the analyst can broaden the search in monitoring data not only to the reported IP address that overcame the defense of the victim but also to other potentially related IP addresses and looks for their prior activities and relationship. For example, the analyst discovers that there is the same successful attack performed by another IP address in the group to another victim IP address and because the attack does not negatively impact any provided service, the breach remains hidden and would go unnoticed otherwise.

Critical requirements of automated attack mitigation are the low number of false positives and timeliness. If an IP address is detected as offending and, at the same time, the IP address is also found to be a part of a group of misbehaving IP addresses, then it is an indicator that the detection is more likely to be a true positive. In some cases, it is acceptable to block an IP address upon its detection as suspicious automatically. In such cases, the predictive blocking of other IP addresses in its group improves timeliness significantly.

The knowledge of the groups of IP addresses also improves the capability of reputation databases, such as NERD [8], to predict future events. If the prediction is known for a subset of IP addresses in the group, then it can be extrapolated for the whole group.

At a glance, the contributions of this paper are as follows:

- We introduce a novel concept of grouping coordinated IP addresses based on their correlated temporal activity rather than by their properties, such as same port numbers or a sequence of actions.
- We propose a method to identify groups of the coordinated IP addresses.
- We evaluate our method utilizing raw network flow data and discuss the observed groups.

The evaluation is based on millions of alerts from a real alert-sharing system. The results show that the proposed method discovers coordinated groups of IP addresses and provides additional features to allow for group assessment and selection.

The remainder of this article is structured as follows. Section II provides an overview of state of the art in alert correlation and discusses their differences. Section III presents our method from a high-level view as well as elaborates on the details of our method. In section IV, we describe our dataset, the setup of parameters, the evaluation of clustering algorithms, and an assessment of the discovered clusters. Lastly, section V concludes this article and outlines our future work plans.

II. RELATED WORK

Related work is grouped into two main themes of this paper. First, we discuss foundations and approaches to alert correlation. Second, we discuss the aspects of collaboration in cybersecurity, including collaborated attacks and defenses.

A. ALERT CORRELATION

The topic of alert correlation has been investigated by cybersecurity researchers for almost two decades when the first research works in the field emerged [9], quickly followed by the correlation of alerts in a collaborative environment [10]. A comprehensive overview of alert correlation, including the detailed description of particular tasks and procedures, was proposed by Valeur *et al.* [11]. This work later inspired numerous alert correlation systems. The input data need to be normalized (usually using IDMEF format [12]), filtered, organized into smaller time windows, and aggregated into groups (sometimes called meta-alerts or correlated alert sets [13]). The common motivation for alert correlation in the related work is the reconstruction of an attack scenario, i.e., creating a sequence of events describing an attack [14]. Moreover, it is possible to use the attack scenario to perform impact analysis, project the continuation of an ongoing attack, estimate its focus, and prioritize the ongoing attacks by their severity [11].

An older survey [15] assessed techniques of alert correlation and stated that security systems (intrusion detection system, firewall, antivirus) differ in types of detection and that low-level alerts need to be analyzed with higher-level management systems, which is a common design of alert correlation tools up to now. However, the surveyed papers focus only on the relation between alerts in the context of attack scenarios and multi-stage attacks. The temporal criteria for alert correlation (mentioned in the survey) are rarely found in the literature. A more recent survey or alert correlation was proposed by Salah *et al.* [2] and focuses on model-based approaches.

There are a plethora of papers discussing particular aspects and approaches to alert correlation. Herein, we briefly introduce the most closely relevant ones. The fundamental goal of most of the related work is to identify related alerts in the context of multi-step attacks or attack scenarios, and they use alert fusion for the identification or prediction of the attack scenarios. The research revolves around correlation and fusion of alerts (also called events) using the following methods. *Bayesian networks* and *Bayesian Attack Graphs* are used in [13]; *Bayesian estimation* in combination with *Kalman Filter* are used in [16] to correlate events along observation time; *Bayesian networks* are also used in [17] to discover attack strategies; *Naive Bayes Classification* with characteristic vector representing attack samples is used in [18] (vectors consisted of seven attributes like protocol, time, IP addresses in attack sample); *Naive Bayesian networks* are also used in [19] for event correlation, especially to detect coordinated attacks, e.g., DDoS attacks; *Scenarios Graphs* are used in [14] to identify related alerts that could be part of a coordinated attack plan that could be missed by a human analyst; *Sequential pattern mining* algorithm was used in [20] to discover complicated multi-stage attack behavior patterns; *Data mining* techniques for fusing alerts into scenarios are used in [21] based on their similarities. *Hidden Colored Petri-Nets* are used in [22] for alert correlation and understanding, i.e., alerts are aggregated (fused) into sequences of scenarios (the paper does not cover detection of coordinated attacks which is left for future work). *Machine learning* approach, specifically *Multilayer Perceptron (MLP)* and *Support Vector Machine (SVM)*, in combination with *alert correlation matrix* (representing estimated probability of similarity between alert pairs) are used in [23] to determine which alerts in the past correlate with the current alerts. The goal of [23] is to group alerts and represent the correlated alerts as attack scenarios.

A recent paper by Kim *et al.* [24] claims that most studies were published a long time ago, i.e., before the concept of APT was formulated. Therefore, it is difficult to apply the methods to current multi-stage attacks. The authors introduce a similarity-based approach with a continuous score. Additionally, the proposed algorithm does not evaluate all the alerts but considers only relevant alerts based on time, so the algorithm is more efficient than previous existing works. Indeed, the other works [14], [19], [24] do not discuss different types of coordinated activities that repeat in time and also, compared to our work, do not cover random similarities in real traffic that necessarily lead to false-positive alerts. Recent work on alert correlation was proposed by Zhang *et al.* [25] and combines alert correlation with attack prediction by extrapolating attack scenarios by temporal correlation. In comparison to our work, the focus is aimed at the attack scenario construction, while we are more interested in the detection of coordinated malicious IP addresses. Still, it is one of the most closely related works.

B. COLLABORATIVE ATTACKS AND DEFENSES

Collaboration is a prominent aspect of this field. As the attackers started performing distributed or collaborative attacks, the defenders started building distributed and collaborative intrusion detection systems to combat these threats and improve intrusion detection capabilities. An older survey by Zhou *et al.* [26] covered both coordinated attack and their detection, while Elshoush *et al.* [27] surveyed alert correlation in collaborative intrusion detection systems. Earlier works provide surveys and taxonomies of collaborative and distributed intrusion detection [28], [29] or focus on particular issues of which formats and protocols to use for information sharing [30], how to defend against DDoS attacks collaboratively [31], or how to collaborate on a national level [32]. Various tools, protocols, and platforms are known to be used in practice for alert sharing and information exchange [33] to substitute complex collaborative intrusion detection systems that would often be difficult to deploy across multiple organizations, networks, or states. Recently, a collaborative intrusion detection system was proposed by Azad *et al.* [34]; the system fulfills current requirements on such systems, such as performing in a decentralized setting and preserving the privacy of the peers that share alerts.

The main goal of this work is the detection of coordinated IP addresses participating in an attack. Such attacks are referred to as coordinated or orchestrated in the literature and are typical for botnets. A command and control (C&C) of a botnet typically orchestrates the attacks launched by individual bots, e.g., to launch it simultaneously. To the best of our knowledge, this issue was not discussed specifically in terms of alert correlation in a collaborative environment that processes intrusion detection alerts. An exception is the older work by Benferhat *et al.* [19], who propose to detect collaborated attack plans such as DDoS, but their method does not identify the coordinated group itself.

Nevertheless, a different source of data (raw packet capture) allowed for an investigation into this topic. Network telescopes and darknets, i.e., unassigned blocks of IP addresses and tools to collect the network traffic incoming to them, are a valuable source of threat intelligence; they provide traces of network scanning, DDoS backscatter traffic, and other ‘‘background radiation’’ of the Internet [35]. A prominent source of such data is CAIDA network telescope.[1] Bou-Harb and Fachkha [36] discussed the inferring of large-scale scanning and DDoS attacks, which results in generating lists of IP addresses involved in the malicious activities. Bou-Harb *et al.* [37] then proposed CSC-Detector, a tool to infer large-scale orchestrated scanning campaigns from the network telescope. The CSC-Detector fingerprints scanning devices and correlates their behavioral profiles to infer the orchestrated scanning. The tool was shown to detect coordinated activities of major botnets at the time. Later, Bou-Harb *et al.* [38] improved their approach and discussed the issues related to processing big data coming from network telescopes, such as their sanitization, dimensionality reduction, and noise reduction. Similar goals and approaches could be found in works by Torabi *et al.* [39], who were able to infer IoT-infecting botnets specifically, or Richter and Berger [40],

[1] https://www.caida.org/projects/network_telescope/

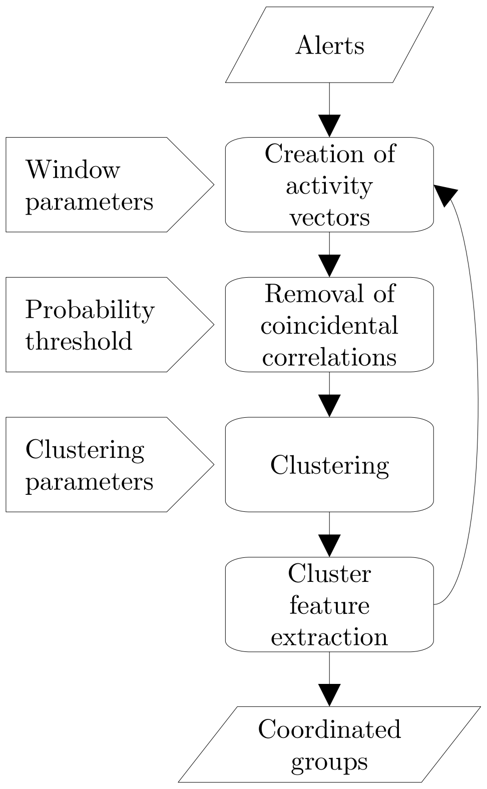

FIGURE 1. **The steps of the security event correlation method and its input parameters.**

who used packet capture from a large number of firewalls instead of a network telescope to characterize scanning activity.

III. OUR APPROACH: SECURITY EVENT CORRELATION IN TIME

Our goal is to define a method for grouping IP addresses that repetitively participate in coordinated security events (e.g., scanning or brute-force campaign, DDoS). Therefore, we propose the Security Event Correlation in Time (SECT) method, which is based on the temporal correlation of security alerts[2] using clustering algorithms. The proposed method is general enough to work with various identifiers such as domain names and URLs, not just IP addresses. Therefore, to describe the method itself, we use the general term *entity* instead of a more specific *IP address* which we use later on during evaluation.

A. METHOD OVERVIEW

This section provides a high-level overview that deliberately simplifies the method and omits details that are discussed in the respective sections. The SECT method can be divided into four consecutive steps. The individual steps with their input parameters are depicted in Fig. 1.

[2]In our case, the security alert is a report about the suspicious behavior of one or more IP addresses.

In the first step, the method analyzes incoming alerts and creates a binary vector of activity for each reported entity. The vector represents the behavior of its entity during a certain time window.

During the second step, the method filters out the entities with very low or very high activity in the time window. This filtration is necessary to remove the entities with a high probability of being correlated by coincidence. The details when two IP addresses are considered correlated are provided in section III-C, and details on coincidental correlation are discussed in section III-D.

Subsequently, cluster analysis is executed over the activity vectors to group the entities having similar vectors. Discussion about distance measures and clustering algorithms is provided in section III-E, and III-F respectively.

The groups are then assessed based on the number of members or a group survival index. We call this step cluster feature extraction. The extracted features serve for further analysis of the groups. Further details about these features are provided in section III-G. The result of the whole method is a list of groups of the entities which exhibit coordinated behavior with other entities in the same group.

B. CREATION OF ACTIVITY VECTORS

An activity vector v is created for each reported entity E. The definition of activity vector v is provided in section III-G.

Definition 1: Let $v = (v_1, v_2, \ldots, v_n)$ *be a vector where each* v_i *is an element of* $\{0, 1\}$*,* n *is the number of equal-sized slots* $s_1, \ldots, s_n$ *of time window* w*, so that* $\forall i \in \{1, \ldots, n\}$*:*

$$v_i = \begin{cases} 1 & \textit{iff } E \textit{ was reported in slot } s_i, \\ 0 & \textit{otherwise.} \end{cases} \tag{1}$$

The number n of equal-size slots is given by the duration t_s of each slot. Therefore, t_s is an input parameter of the method since it splits the time window w into n slots. A particular setup of t_s is discussed in section IV-B.

C. CORRELATION

The SECT method infers coordination from the correlation of the activity vectors. The correlation between two activity vectors is defined in definition 2.

Definition 2: Let a and b are the activity vectors, $T \in \mathbb{R}$*, and d is a distance measure (metric). The vectors a and b are correlated if and only if the* $d(a, b) \leq T$ *and we denote it as* $a \simeq b$

The distance measure d is the input parameter of our method, but for our evaluation, we select one particular distance measure which reflects our requirements, and we discuss this distance measure in section III-E. Definition 2 itself leads to cluster analysis of the activity vectors. In principle, our method works with any clustering algorithm, but of course, the results of various algorithms may differ. We discuss candidate algorithms in section III-F. The threshold value T is generally the clustering algorithm parameter, therefore an input parameter of our method, except for some clustering algorithms which calculate the threshold value themselves from the density function of input data.

Let A and B are distinct entities with their respective activity vectors a and b. In general, it is surely true, that $C(A, B) \Rightarrow a \simeq b$ where $C(A, B)$ means coordination between the entities. Unfortunately, we cannot straightly claim that the implication is true in the opposite direction as well. A coincidental correlation, input data noise, and other inaccuracies can produce illusory coordination. However, SECT minimizes these factors by aggressive filtering that is described in section III-D, and section III-G and therefore it is possible to build on the following presumption.

Presumption 1: For SECT filtered data applies:

$$C(A, B) \iff a \simeq b. \tag{2}$$

Finally, we define a coordinated group in definition 3.

Definition 3: Let E be an entity, and G is a group of entities:

$$E \in G \iff \exists x, \quad x \in G \wedge x \simeq E. \tag{3}$$

Definition 3 is formulated deliberately broad and flexible enough to ensure that almost any clustering algorithm can meet this definition.

D. COINCIDENTAL CORRELATION

The combination of definition 1 and definition 2 allows a certain possibility of correlation without coordination, i.e., two activity vectors being correlated purely coincidentally rather than by the underlying entities being anyhow coordinated. The probability of these coincidental correlations depends on the number of slots k when the entity is active, the total number of slots n in the time window, and the threshold value T. For the sake of simplicity, let us restrict ourselves to the threshold value $T = 0$. In this case, only equivalent activity vectors correlate with each other. Let $\mathcal{X}$ be a uniformly distributed discrete random variable representing the occurrence of the activity vector of length n with k active slots. Then the probability of occurrence of a particular vector x in $\mathcal{X}$ is an inverse of a combinatorial number:

$$P(\mathcal{X} = x) = \binom{n}{k}^{-1}. \tag{4}$$

We can create a set of independent and identically distributed (IID) random variables $\mathcal{X}$. Each random variable $\mathcal{X}$ in the set represents the observed entity's occurred activity vector.

The probability $P(r)$, that in $r \in \mathbb{N} \wedge r \geq 1$ IID random variables $\mathcal{X}$ are at least two occurrences of the same activity vector follows the equation used in the classical Birthday problem [41]. The classical birthday problem asks, what is the probability of finding at least one similar pair having the same birthday in a group of several individuals? When we map the number of individuals to r and the probability of birthday to $P(\mathcal{X} = x)$ we get:

$$\overline{P}(r) = \prod_{i=1}^{r} \Big(1 - (i-1) \cdot P(\mathcal{X} = x)\Big) \tag{5}$$

$$P(r) = 1 - \overline{P}(r) \tag{6}$$

To enumerate the probability $P(r)$ we can follow the approximation of birthday paradox using a Poisson distribution with parameter $\lambda = 0.5 \cdot \Big(r \cdot (r-1) \cdot P(\mathcal{X} = x)\Big)$ [41]:

$$P(r) \sim 1 - \frac{1}{e^{\lambda}} = 1 - \frac{1}{e^{0.5 \cdot \left(r \cdot (r-1) \cdot P(\mathcal{X}=x)\right)}} \tag{7}$$

In eq. (7), we can see that when the parameter λ is higher, the probability of coincidental correlation increases. The λ is defined by the number of random variables (in our case, determined by the number of observed entities) and the probability of a particular activity vector occurrence, which needs to be maintained in balance to keep the parameter low. However, these variables are gathered from input data, and we cannot control them. Thus, the SECT method calculates the probability of coincidental correlation for each activity vector and removes activity vectors (entities) with the probability $P(r)$ higher than a threshold p_{min}. The threshold is another parameter of our method, and its value is discussed in section IV-B.

E. DISTANCE MEASURE

In this section, we discuss the requirements imposed on the distance measure. There are many distance measures defined in the literature, but a majority of them does not satisfy all three metric axioms (identity, symmetry, and triangle inequality) necessary for the distance measure $d(u, v)$. The necessity of metric space for a clustering algorithm is described in the study [42], where the medoid-based algorithm used on non-metric space showed significant deterioration.

Granger Causality Test (GCT), a time series analysis test, was introduced in [43] for temporal comparison of related security alerts in DEF CON 9 dataset. However, our selection is limited to metric functions on the binary vectors. A typical example of these metrics is the family of Minkovski distances, which on binary space, is reduced to Hamming distance [44]. However, the Hamming distance values depend on the length of the activity vector; therefore, it has to be normalized, which is crucial for selecting threshold value T (see definition 1) independently of the size of the vector. Given binary vectors u and v, M_{00} is the total number of attributes where u and v both have a value of 0, M_{01} is the total number of attributes where u is 0 and v is 1, M_{10} is the total number of attributes where u is 1 and v is 0, M_{11} is the total number of attributes where u and v both have a value of 1, and $M_{00} + M_{01} + M_{10} + M_{11} = n$. Then Hamming distance d_H and normalized Hamming distance d_{Hn} are given as

$$d_H(u, v) = M_{01} + M_{10}$$

$$d_{Hn}(u, v) = \frac{M_{01} + M_{10}}{n}.$$

However, even after normalization, the Hamming distance favors vectors with lower activity (in the following examples we use a shortened notation for the binary vectors, e.g., (0, 0, 0, 0) = 0000, (0, 0, 1, 1) = 0011):

$$d_{Hn}(1111, 1100) = \frac{0+2}{4} = 0.5$$
$$d_{Hn}(1100, 1000) = \frac{0+1}{4} = 0.25$$

The vectors in the example above have different distances, even though the percentage difference of active time slots is the same (50 %), which does not meet our requirements. Therefore, we propose to use the Jaccard distance for binary attributes, which is in line with [44] and is defined as:

$$d_J(u, v) = \frac{M_{01} + M_{10}}{M_{01} + M_{10} + M_{11}}. \tag{8}$$

Note that the only difference between the normalized Hamming distance and the Jaccard distance is M_{00} (the passive slots) in the denominator of the latter. Consequently, the Jaccard metric approach deals with the problem of prioritizing lower activity vectors by taking into account only the active time slots. Calculating it for the vectors from the example above, we get the right results:

$$d_J(1111, 1100) = \frac{0+2}{0+2+2} = \frac{2}{4} = 0.5$$
$$d_J(1100, 1000) = \frac{0+1}{0+1+1} = \frac{1}{2} = 0.5$$

This property is also useful to mitigate the impact of T on the probability of coincidental correlation. The value of T affects it by allowing grouping also non-equal activity vectors. By using only active time slots, the Jaccard metric partially mitigates this effect because it linearly adapts the amount of allowed inequality to the number of activities and thereby also to the coincidental correlation probability. When we choose, e.g., $T = 0.25$, the vectors with activity lower than four must be the same to be correlated. This phenomenon can be clearly seen in this example:

$$d_J(1100, 1000) = \frac{1}{2} = 0.5$$
$$d_J(1110, 1100) = \frac{1}{3} = 0.\overline{3}$$
$$d_J(1111, 1110) = \frac{1}{4} = 0.25$$

The coincidental probability mitigation effect of the Jaccard metric applies to low-activity vectors while diminishing with the growing activity.

The coincidental probability increases with the amount of allowed difference and might even overgrow the p_{min}, which is calculated only for the exact vector match. This has to be taken into account during the selection of the time window, the time slot duration, and the filtering of high activity vectors. The higher the T, the more aggressive filtering should be applied to high-activity vectors.

F. CLUSTERING ALGORITHMS

This section serves as a brief introduction to clustering algorithms and their applicability in the SECT method. The SECT method works with any clustering algorithm, but each algorithm family is suitable for a different use case and produces considerably different clusters.

Clustering algorithms can be categorized based on their cluster model into connectivity-based (hierarchical clustering), centroid-based, density-based, distribution-based, but also other less known models exits.

Hierarchical clustering algorithms build a hierarchy of clusters. It can be either agglomerative (objects initially form individual clusters which are successively merged) or divisive (objects are initially part of a single cluster that is successively divided into partitions). The process can be represented using a dendrogram (a diagram depicting a tree), which offers two options for extracting clusters: cutting the tree at a given height (inter-cluster distance criterion) or a place with a specified number of clusters (number criterion). Other important parameters are the inter-cluster distance function (linkage) and distance computational function (metric). There are several types of linkage functions. However, the most applicable for the SECT method is the complete-linkage distance calculation because it makes the distance criterion parameter closely related to the T from the definition of correlation (see definition 2). The complete-linkage defines the distance between clusters as:

$$D(X, Y) = \max_{\forall x \in X, \forall y \in Y} d(x, y), \tag{9}$$

where D is the inter-cluster distance, X, Y are clusters, and d is the metric. When two clusters are closer than the minimal inter-cluster distance parameter, the algorithm merges them into one cluster. Thus, the parameter defines maximal distance within the cluster and can be regarded as a threshold value of T from definition 2.

A de facto synonym for centroid-based clustering is k-means. K-means [45] and all its variation are forming clusters based on the distance to the closest center point. The algorithm starts with a group of randomly placed centroids, which define the position of clusters and associate each object to the nearest centroid. Then it calculates the next centroid position by minimizing the distance to each cluster object. These operations are performed iteratively to optimize the placement of a cluster centroid.

Clusters in density-based algorithms are defined as areas of higher density than the rest. Objects with lower neighborhood density are considered to be noise or outliers. The typical example of a modern density-based algorithm is DBSCAN [46], HDBSCAN* [47], and OPTICS [48]. The HDBSCAN* is stated to have similar qualities as OPTICS, with a better algorithmic runtime complexity. The OPTICS algorithm can be regarded as a generalization of DBSCAN that addresses the problem of detecting clusters of varying density, and it also eliminates the need to set an exact value for ε (a maximal distance of two objects within one cluster). This parameter is closely related to T from definition 2.

There is no clustering algorithm universally recognized as the best or state-of-the-art. Each algorithm family varies significantly in its properties, and the appropriate clustering algorithm and parameter settings depend on the data set and the problem being solved. The most significant obstruction in the usage of the centroid-based algorithms is the need to set the number of clusters in advance. As a matter of fact, the number of clusters in the dataset is one of the desired outputs in our case. For this reason, centroid-based algorithms have been ruled out. Density-based clustering algorithms can form clusters of an arbitrary shape, in contrast to many other methods, which assume that clusters are convex shaped. This is generally considered an advantage, but it is not the case for SECT. An arbitrarily-shaped cluster might contain points fairly far apart from each other (low intra-cluster similarity). Thus, the notion of a cluster, as found by density-based algorithms, does not meet the intuitive notion of a coordinated group of entities (clusters with high intra-cluster similarity). Hierarchical clustering can form only convex-shaped clusters, does not require the number of expected clusters in advance, and can use an arbitrary distance function. Using the complete-linkage criterion, the linkage distance threshold parameter corresponds to threshold T from definition 2. The only drawback is the lack of concept of noise. For instance, DBSCAN has a *minPts* parameter determining the desired minimum cluster size; smaller clusters are considered to be noise. This rule can also be applied to hierarchical clustering. Through the process of elimination, agglomerative hierarchical clustering is the preferred choice.

G. CLUSTER FEATURE EXTRACTION

The cluster feature extraction step is included in the SECT method to describe each cluster with additional statistical, quantitative, and qualitative information. Such features can be further used to perform a "manual" evaluation by a human expert based on certain criteria, create a custom cluster quality scoring system, and so on.

A basic feature is the cluster size. The idea is that the small clusters with a low number of entities are often less relevant from the monitoring perspective as the attack is not focused on the monitored infrastructure; otherwise, more coordinated entities would likely appear. We can consider small clusters as background noise. For example, horizontal scanning of the whole IP address space by two coordinated machines can cause such noise. The larger the clusters, the more likely the attacks are relevant to the monitored infrastructure as more coordinated entities are involved.

Each cluster can be represented by an aggregate activity vector:

Definition 4: Let $v_a = \mathbb{R}^n$ be an aggregate activity vector of cluster G, such that $\forall i \in \{1, \ldots, n\}$:

$$v_{ai} = \frac{\sum_{v \in G} v_i}{|G|}. \tag{10}$$

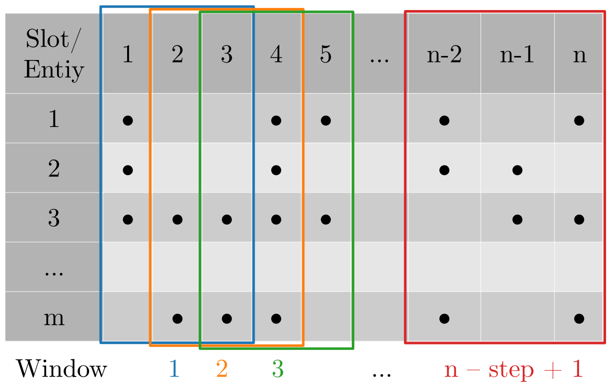

FIGURE 2. An example of a sliding window, each containing only three slots and a step of one slot.

In other words, the aggregate activity vector is of the same length as the entity activity vector with values corresponding to the mean activity of all members (a ratio of entities active in that particular time slot to the total number of member entities). The aggregate activity vector v_a is created for each discovered cluster. The activity vector v is defined in definition 1.

The aggregate activity vector is a time series of numbers between zero and one and is suitable not only as a cluster descriptor but also as an input to methods for analysing time-series data. One of the methods we find most useful is autocorrelation. Autocorrelation is a correlation of the series with a delayed copy of itself. It is used as a tool for the detection of repeating patterns, identifying the fundamental frequency, or lack of it. The evaluation section IV-C provides examples of how this information can be utilized in SECT.

As a next feature, we propose group[3] survival duration. The assumption is that the long-lasting groups are composed of stable, coordinated entities over time, and it is worth monitoring them, unlike the one-shot groups. We define survival duration as a time interval between the first and the last activity of a group.

H. ITERATIVE APPROACH

The SECT method is computationally intensive and memory demanding ($O(N^2)$), especially in cases when the number of unique entities grows with the window size.

Therefore, we introduce an iterative approach with a sliding window to deal with the analysis of longer observation intervals. Rather than analyze the observation interval at once in a single window, we split it into shorter consecutive and partially-overlapping windows of the same number of slots. An example of the sliding windows is depicted in Fig. 2. Each row represents the activity of an entity (a dot when the entity is active) while the window delimits its activity vector, which is to be analyzed. The window slides by a step to delimit the new interval.

[3]We use the term *group* to denote a group of entities (IPs) and *cluster* to denote the set of objects (activity vectors) in the same cluster.

The iterative approach fits well when the SECT method must work online, i.e., as the data arrive. The arriving data are accounted into the current window within the scope of the first SECT step (Creation of Activity Vectors). When the current window is at its end, the activity vectors of this window are processed by SECT. Within the scope of the Cluster Feature Extraction, the groups are subsequently paired with the groups of the previous window, and the survival duration is updated for the groups that survived from the previous window. We deliberately find pairs only with the groups appearing in the previous window to reduce the complexity of pairing. To find the most look-alike pairs (A, B) in two sets of groups, we use the Jaccard index:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{11}$$

We discuss the setup of the index threshold in section IV-B. Finally, as an aging strategy of the iterative approach, the groups that are not active longer than the observation interval are discarded.

IV. EVALUATION

In this section, we evaluate the SECT method by applying it to real data of network alerts. Our goal is to assess our method from various perspectives, such as how the method reacts to the specific setup of parameters, what kind of clusters the SECT method discovers and how the groups map on the raw network flow data (IPFIX [49]).

A. DATASET

The dataset[4] contains intrusion detection alerts obtained via the alert sharing platform [50] for 7 consecutive days from 2021-01-11 till 2021-01-17. A plethora of heterogeneous intrusion detection systems deployed across several organizations contributed to the sharing platform. The alerts are stored in Intrusion Detection Extensible Alert (IDEA) format [51], and the alerts are categorized using the taxonomy of security events included in the IDEA definition.

The dataset consists of almost 8 million alerts. The alerts are collected from nearly 30 detection systems, such as network behavioral analysis systems, honeypots, intrusion detection systems, and similar data sources deployed in 5 large distinct organizations: the national research and education network (NREN), two universities with large campus networks, a midsize cybersecurity vendor and an Internet service provider. Most of the alerts are raised by honeypots, such as LaBrea [52] or Cowrie [53], and network-based intrusion detection systems based on NetFlow [54], such as NEMEA [55]. The majority of alerts belongs to the category of network reconnaissance and vulnerability scanning. Another significant portion of alerts is composed of multiple login attempts, guessing/cracking of passwords, or brute force. The top three countries of the attack sources (according to IP addresses) are China, the US, and Russia, which altogether generate 30 % of all alerts.

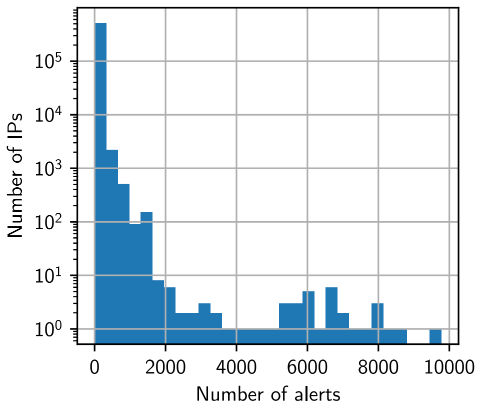

FIGURE 3. Histogram of the number of IP addresses based on the number of their respective alerts.

The alerts report suspicious behavior of about 520 thousand unique IP addresses. An IP address is an entity as considered in the SECT method, and, further on, we use IPs to refer to IP addresses. Fig. 3 displays a histogram of the number of IPs based on their number of occurrences in the alerts. The histogram shows that the vast majority of IPs cause an alert only a few times. We noticed that 50 % of IPs were reported only once or twice while few IPs are reported constantly.

B. PARAMETER SELECTION

Our dataset lasts for a week which renders it possible to find long-lasting groups of IPs, but processing such an amount of data in a single window would result in large memory requirements due to the distance matrix for all 520 thousand IPs. For this reason, we use the iterative approach introduced in section III-H. In our experiments, we set a duration t_w of a window to 24 hours. This value is inspired by blocklists, where every-day updates are a common best practice. Besides, 24 hours observation window leads to feasible resource consumption (approximately 170 thousand IPs). The longer the time window, the higher the probability of a change of the IP address affiliated with the host machine (e.g., an IP address may change due to the dynamic IP address allocation).

Parameter t_s directly affects the number of time slots n in a time window. Too short slot duration (e.g., 1 second) leads to an excessive resolution of activity vectors. The dataset, however, contains artifacts that need to be smoothed. We have inspected our dataset, and we have found out that there is a large, in order of minutes, imprecision of timestamps. The imprecision is caused by the way how the particular detectors work. For example, honeypots in our dataset report an alert either immediately upon every connection or periodically (every 5 minutes). Network behavioral systems collect flow data from probes that already introduce time differences [56]; then they analyze the collected data using either stream or

[4]The anonymized dataset is available at https://doi.org/10.5281/zenodo.4683701

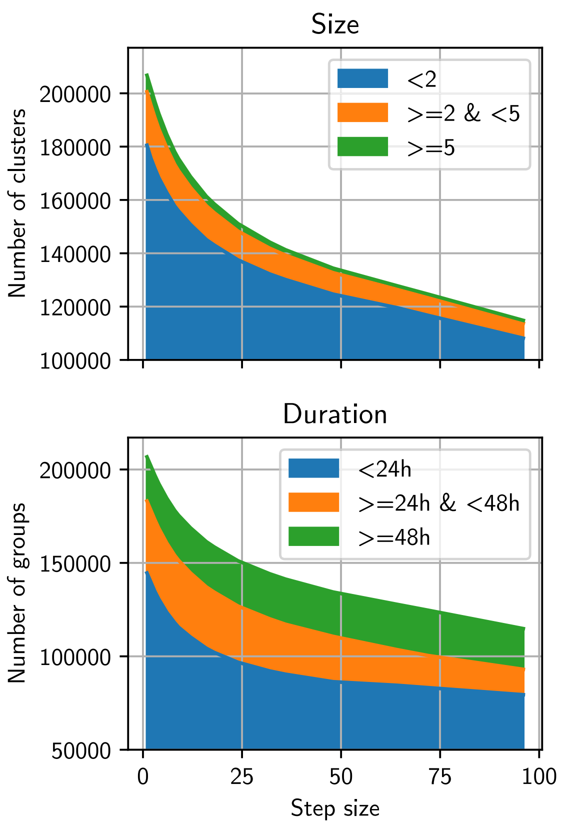

FIGURE 4. A relationship between the step size and the number of clusters/groups of different size and different duration. The colours represent three ranges of cluster sizes (number of IPs) and group duration.

TABLE 1. Probabilities of coincidental correlations for the first time window.

# of active slots k	# of combinations	# of IPs	$P(r)$
1	96	71984	1
2	4560	25293	1
3	142880	15940	1
4	3321960	10082	≈ 1
5	61124064	7179	0.343
6	927048304	5452	0.016
...	...	...	< 0.01
93	142880	84	0.024
94	4560	54	0.269
95	96	64	≈ 1
96	1	596	1

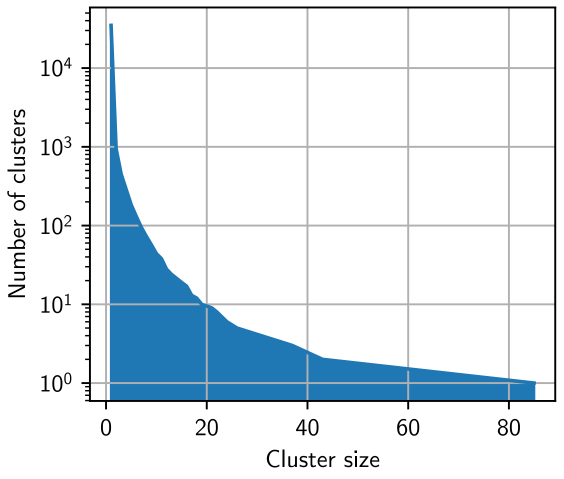

FIGURE 5. Distribution of clusters according to their size.

batch processing which results in immediate reporting or reporting at the end of the batch, respectively. On the other hand, too long slot duration causes a loss of information, i.e., aggregates more alerts into the same slot as well as increases the probability of coincidental correlation (section IIII-D) due to a lower number of slots. As a trade-off, we set the value of t_s to 15 minutes. This gives us $n = 96$ time slots for t_w of 24 hours.

We are aware of IPs that are reported during the entire week, and we assume the groups span more than one day. Therefore we employ the pairing of the surviving groups in consecutive windows using the Jaccard index as discussed in section III-H. We consider two groups to form a pair, or rather to be the same group if the Jaccard index (a ratio of intersection over union) between two groups from w_1 and w_2 is greater than 0.5. In some cases, the groups evolve rather quickly, and a step equal to t_w is too large to pair such groups reliably. Decreasing the step (i.e., making the windows overlap) can be thought of as decreasing a sampling period and also allows more precise pairing. Therefore, we select to slide the window only by a duration of a single slot t_s to achieve the highest resolution for our evaluation, i.e., SECT is run every 15 minutes. This assumption is demonstrated by an experiment illustrated by Fig. 4 where we can see that the number of discovered clusters decreases significantly with a longer step. Nevertheless, for the practical deployment, it is still safe to use the step of 6 hours as we are interested in long-lasting groups of larger size. The Fig. 4 shows that the six-hour step does not affect the number of clusters with a duration longer than 24 hours and affects the number of clusters with a size larger than four IPs only slightly.

The p_{min} parameter is similar to the p-value of the statistical hypothesis tests. It is a threshold of the probability that the correlation between two IPs is purely coincidental, i.e., the IPs are not coordinated. Our experiments (see table 1) based on real data showed that a reasonable value of p_{min} represents a trade-off between the probability of coincidental correlation and preserving a sufficient portion of vectors in our data. We consider the probability of coincidental correlations $p_{min} = 0.01$ sufficiently low. In the first window of our dataset, activity vectors with 1, 2, 3, 4, 5, 6, 93, 94, 95, and 96 active slots do not meet this threshold and are removed. Using the selected p_{min} value, about 78 % of activity vectors are removed on average per window.

As mentioned in section III-F, hierarchical algorithms do not have a notion of noise. Even points that are not merged with any other point are considered to be clusters of size

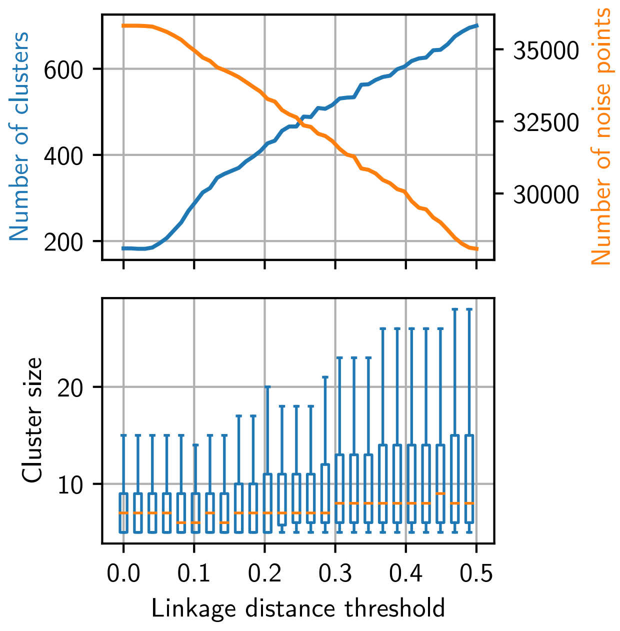

FIGURE 6. **Relationship between the linkage distance threshold parameter and the number of noise points, clusters, and cluster sizes.**

one. To eliminate these clusters, we utilize the basic cluster feature, the cluster size, to remove clusters containing less than five IPs. A graph in Fig. 5 visualizes a distribution of clusters according to their size (number of members) and shows the influence this parameter has on the total number of clusters considered large enough.

The relationship between the linkage distance threshold and threshold T makes it one of the most crucial parameters. Fig. 6 depicts how the number of clusters and the number of clusters considered to be noise change with the growing T (the higher the T, the more activity vectors are clustered which are less similar). Note that the threshold value of zero is a special case: the metric function is bypassed, and only equivalent activity vectors are clustered. Up to a value of 0.05, the clusters change only slightly, from 0.05 the total number of clusters starts increasing linearly while the total number of noise points starts decreasing linearly. The box plot shows that the clusters are getting bigger with the higher threshold, but the median cluster size remains approximately constant. For the rest of our experiments, we use the special linkage distance threshold value of zero due to better interpretability of the results (in such a case, each cluster can be represented by a single activity vector). We can increase T up to 0.05 and achieve very similar results as with $T = 0$. For example, $T = 0.05$ allows a one-bit difference between IPs that are active in twenty slots.

C. CLUSTER ANALYSIS OF A SINGLE TIME WINDOW

In this section, we discuss our findings regarding the discovered clusters. We use calculated features and metrics to improve data clarity and interpretability. In addition, we also inspect the raw data belonging to the discovered clusters to validate if the raw data supports the clustering results.

The cluster analysis of a single time window is a basic building block of the entire method, and a comprehensive analysis of the output is essential. For the demonstration, we have chosen the first time window of the dataset, i.e., the first 24 hours. On average, it takes less than five minutes to process one time window completely.

Using the autocorrelation function of the aggregate activity vector, we can classify the clusters into three categories: (1) clusters with a periodic activity, (2) clusters with a burst activity, and (3) clusters without any obvious behavioral pattern. Fig 7 to 10 reveal how the category can be determined by utilizing autocorrelation. The upper chart shows an aggregate activity vector reshaped into a matrix (6 rows $\times$ 16 columns $=$ 96 slots) to fit the page (black square marks an active slot). The lower graph is an autocorrelogram, a graphical representation of the autocorrelations versus the delay (also called a lag, values close to one indicate nearly perfect autocorrelation, and the respective delay is the period). Apart from the visual representation, we also provide numerical features like absolute and relative extrema of the autocorrelation function, which are more suitable for automated evaluation and cluster selection.

A demonstration of the first category is in Fig. 7. It shows a periodic activity with a period of one hour (i.e., four 15-minute-long time slots), which is confirmed by a strong positive autocorrelation for the delay/lag of four slots. This is the most common period of activity in our dataset, but other periods are not rare either. For example a six-hour-long period (24 time slots) is shown in Fig. 8. A representative of the second category is depicted in Fig. 9. The activity vector displays a sudden, intense, and continuous effort, which is typical for DDoS attacks. Its autocorrelogram exhibits a declining positive autocorrelation stabilizing around zero. The most interesting category is the third one, the activity vectors without any obvious pattern. The aggregate activity vector of the cluster shown in Fig. 10 looks arbitrary, and the autocorrelogram shows values no larger than 0.2, which indicates that there are no signs of periodicity hidden to the human eye.

To summarise the clustering results, the first time window contained 74176 unique IP addresses, which were reduced to 37448 (21.5 %) due to a high coincidental correlation probability. SECT discovered 34870 clusters (most of the clusters contained only one IP address), out of which only 183 (0.5 %) fulfilled the minimal cluster size condition. The 183 groups contained a total of 1630 IP addresses. The classification uncovered 45 clusters with a periodic activity, 33 clusters with a burst activity, and 105 clusters without any obvious behavioral pattern.

To further validate our hypothesis that the IPs exhibiting a temporal correlation of their alerts are coordinated, we also analyze IP flow records of grouped IPs. The flow records are, in comparison with security alerts, less aggregated. They reveal the behavior of an IP address in detail, such as the

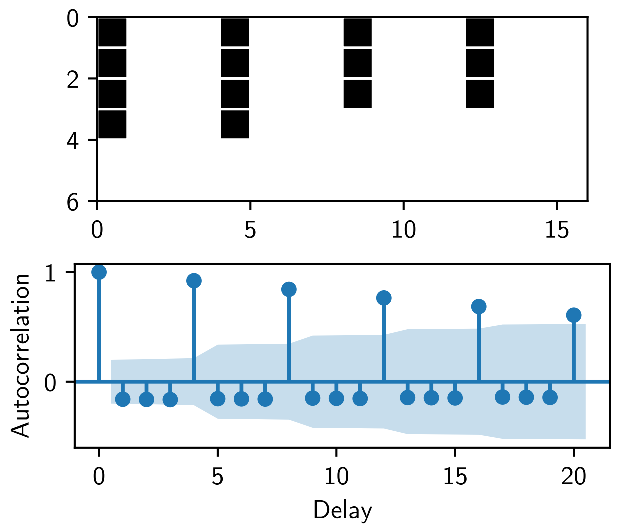

FIGURE 7. A cluster with a periodic activity with a one-hour-long period.

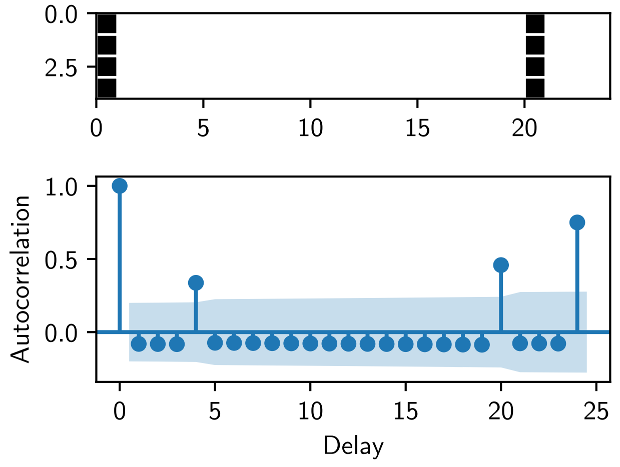

FIGURE 8. A cluster with a periodic activity with a six-hour-long period.

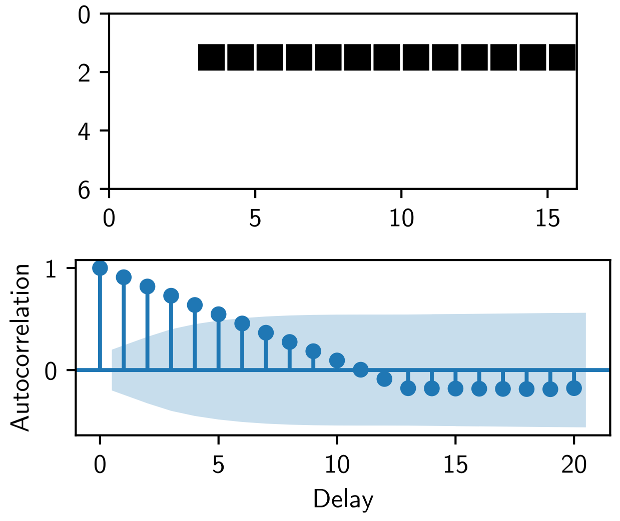

FIGURE 9. A cluster with a burst activity.

start and the end timestamps of each flow, the communicating parties, ports and protocol being used, and the number of bytes and packets. Also, flow records are collected for the entire communication, i.e., not only for the malicious but also for the benign traffic mix.

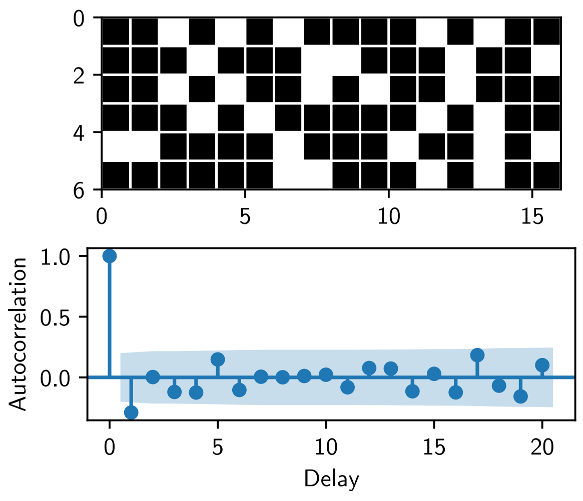

FIGURE 10. A cluster without any activity pattern.

For this paper, we selected three representative clusters, one per category, which we analyze and discuss in detail. Fig. 11 to 13 display a visual comparison between the activity of an IP address and the number of its flow records. The upper chart shows an aggregate activity vector; the lower graph shows the number of flows per slot. The horizontal axes are aligned to make it easier to see the correlation. All the IPs are anonymized using prefix-preserving anonymization for the purpose of presentation in this paper.

The cluster in Fig. 11 is the same cluster as in Fig. 7. All eight members of this group scan the same two ranges of IP addresses, targeting port 445 (protocol SMB). The number of connections initiated by the IPs in the group is rising at the same pace until a certain time when all members cease their communication. Based on this information, we conclude that this cluster (and many other clusters belonging to the first category) is comprised of slow coordinated network scanners. All the IPs in the group add a new IP address to the pool of IP addresses that they scan per some predefined period (one hour in this particular case), which results in the periodicity in the aggregated activity vector. Moreover, the observed flow data provides further support for the coordination of these IPs.

The cluster in Fig. 12 is the same cluster as in Fig. 9. All 11 IPs belonging to this group send TCP SYN packets towards more than 133 thousand destination IP addresses, all targeting port 23 (the Telnet protocol). The activity takes about four hours with a constant rate of about 50 flows per second. All members belong to the same autonomous system; each of them begins and ceases its activity at the same time (± one minute). The respective alerts that reported the 11 source IPs are categorized by the detectors themselves as a scanning activity. The clustering of these IPs draws a different picture of the suspicious activity. This activity should have

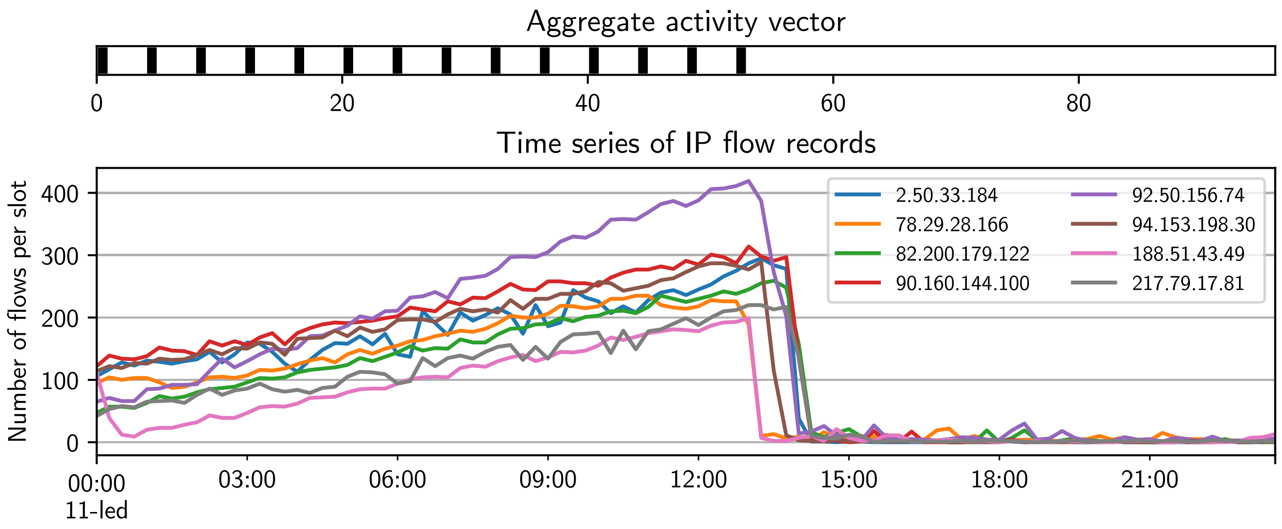

FIGURE 11. A cluster with periodic behavior.

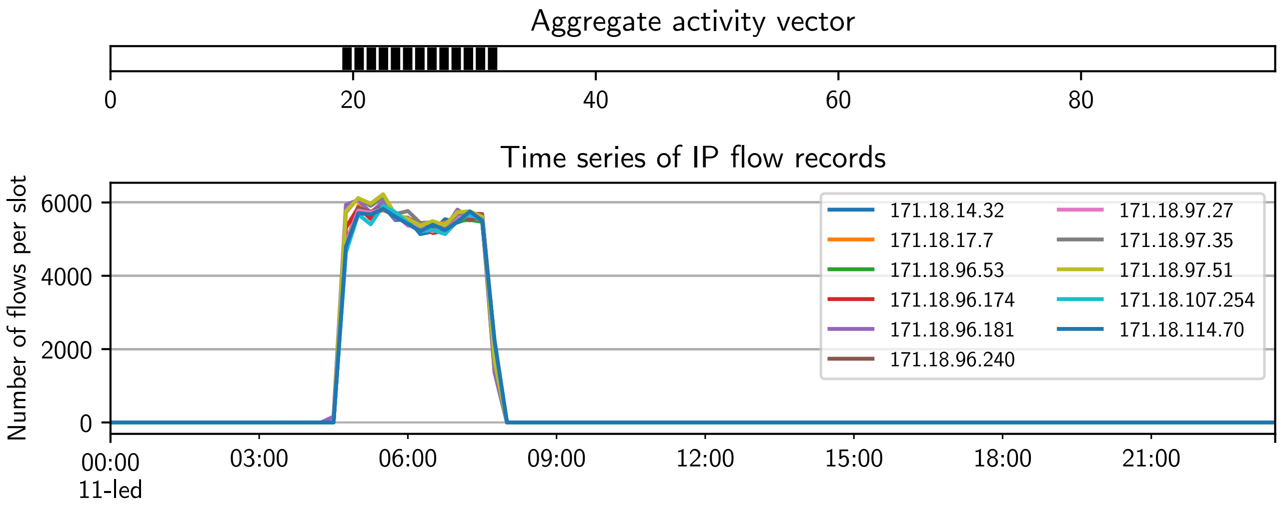

FIGURE 12. A cluster with bursts of activity.

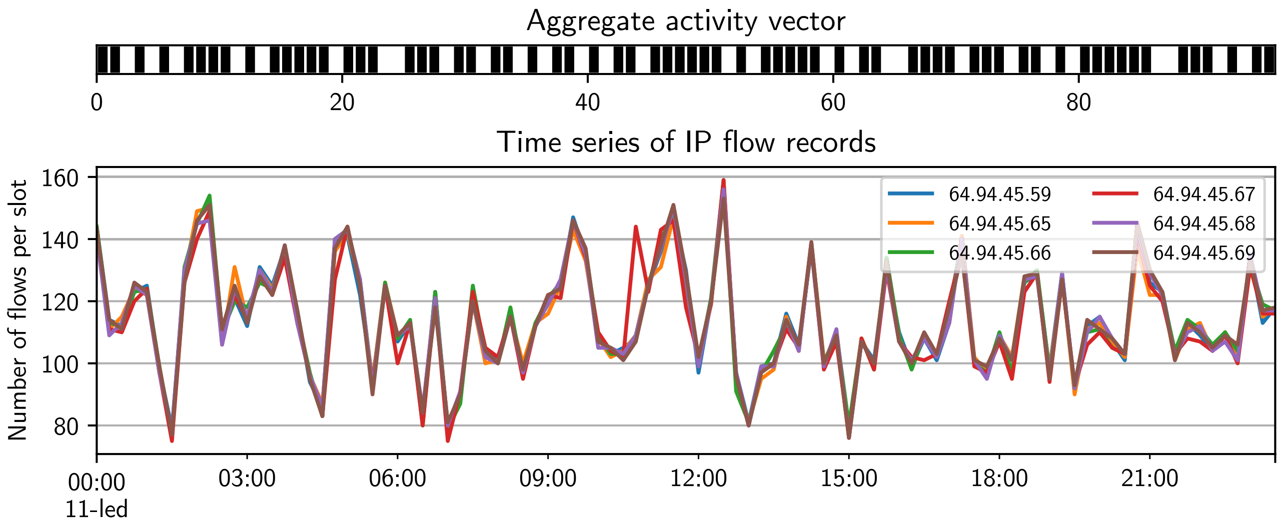

FIGURE 13. A cluster without any activity pattern.

been categorized as DDoS, in particular, the revival of TCP reflection attacks as described, for example, in [57].

The cluster in Fig. 13 is the same cluster as in Fig. 10. The six source IPs (all from a single /24 network) send ICMP

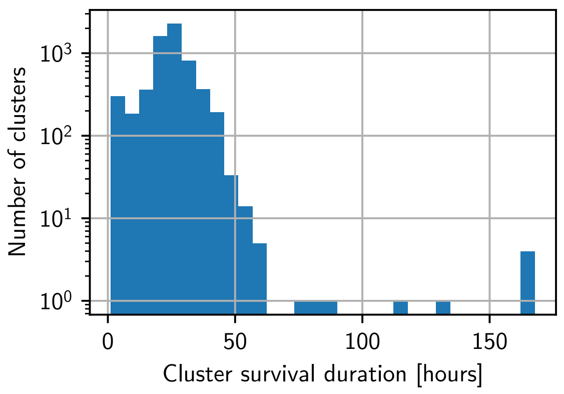

FIGURE 14. **Cluster survival duration histogram.**

echo packets towards several ranges of target IP addresses – a certain case of a distributed ICMP scan. It is remarkable that although the activity is long-term and without any visible pattern, it is almost the same for each member. This can be seen on the flow activity graph, where the lines overlap almost perfectly and provide evidence of coordinated activity.

D. CLUSTER ANALYSIS OF THE ENTIRE DATASET

In contrast to the relatively short-term perspective of the previous section, this section focuses on a long-term perspective of the entire week-long dataset. In other words, we investigate clusters with a lifetime exceeding one time window.

Employing the pairing technique from section III-H, SECT discovered 206,741 clusters (including the noise points) in the whole one-week dataset. However, only 6,160 clusters fulfill the minimal cluster size condition. A mean cluster survival duration is one day and 37 minutes, and the distribution of this feature is depicted in Fig. 14. The histogram shows the maximum number of clusters around the duration of 24 hours. There are also four clusters surviving over the entire week. One of them is the same cluster as we analyzed in the previous section (Fig. 13). Its members and behavior remain unchanged during the entire observation interval.

V. CONCLUSION

In this article, we introduced Security Event Correlation in Time (SECT), a novel method for grouping IP addresses that repetitively participate in coordinated security events. The method works with the vast number of alerts generated by the network monitoring systems. It does not reduce the number of alerts as such, but it infers complementary information for cybersecurity analysts to improve their situational awareness about coordinated IP addresses. SECT is built on clustering algorithms that correlate activity vectors of IP addresses.

Evaluation of SECT on a real dataset demonstrated how SECT performs from the perspective of the number of clusters and their size with respect to the evaluated SECT parameters. The evaluation further explained what are the typically discovered groups and categorized them based on their characteristic. The comparison between the selected groups of IPs and their network flow data supported our hypothesis about the possibility of utilizing network alerts to reveal such a relationship between IPs and demonstrated the applicability of the SECT method to find these groups. Last but not least, we observed the life of groups during the whole week, which produced significant insight that only a few groups survive. The surviving groups are not only interesting from the quality perspective but also from the perspective of situational awareness as the knowledge about these groups is applicable not only once. The method is currently running online and is integrated into the production reputation system NERD [8], [58], [59], which is being used by CESNET-CERT to model the reputation of suspicious network hosts.

Concerning our future work, we will further investigate the coordinated groups of IP addresses from a longer-term perspective, as well as how to automate the selection of the most relevant coordinated groups based on their characteristics. Moreover, we plan to explore if SECT is applicable to other data domains (such as DNS monitoring data or a selected subset of network flow monitoring data).

REFERENCES

[1] M. Drašar, M. Vizváry, and J. Vykopal, "Similarity as a central approach to flow-based anomaly detection," *Int. J. Netw. Manage.*, vol. 24, no. 4, pp. 318–336, Jul. 2014.

[2] S. Salah, G. Maciá-Fernández, and J. E. Díaz-Verdejo, "A model-based survey of alert correlation techniques," *Comput. Netw.*, vol. 57, no. 5, pp. 1289–1317, Apr. 2013.

[3] Z.-H. Tian, B. Li, and H.-L. Zhang, "An attack-feedback-based approach for verifying the success of intrusion attempts," in *Proc. Int. Conf. Comput. Intell. Secur.*, Nov. 2006, pp. 629–632.

[4] T. Sommestad and U. Franke, "A test of intrusion alert filtering based on network information," *Secur. Commun. Netw.*, vol. 8, no. 13, pp. 2291–2301, Sep. 2015.

[5] M. Husák, M. Čermák, M. Laštovička, and J. Vykopal, "Exchanging security events: Which and how many alerts can we aggregate?" in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, May 2017, pp. 604–607.

[6] K. Alsubhi, E. Al-Shaer, and R. Boutaba, "Alert prioritization in intrusion detection systems," in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, Apr. 2008, pp. 33–40.

[7] S. A. Mirheidari, S. Arshad, and R. Jalili, "Alert correlation algorithms: A survey and taxonomy," in *Cyberspace Safety and Security* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2013, pp. 183–197.

[8] A. L. E. CESNET. (2019). *Network Entity Reputation Database (NERD)*. [Online]. Available: https://nerd.cesnet.cz/

[9] H. Debar and A. Wespi, "Aggregation and correlation of intrusion-detection alerts," in *Proc. Int. Workshop Recent Adv. Intrusion Detection*. Berlin, Germany: Springer, 2001, pp. 85–103.

[10] F. Cuppens and A. Miege, "Alert correlation in a cooperative intrusion detection framework," in *Proc. IEEE Symp. Security Privacy*, May 2002, pp. 202–215.

[11] F. Valeur, G. Vigna, C. Kruegel, and R. A. Kemmerer, "Comprehensive approach to intrusion detection alert correlation," *IEEE Trans. Dependable Secure Comput.*, vol. 1, no. 3, pp. 146–169, Jul. 2004.

[12] H. Debar, D. Curry, and B. Feinstein, *The Intrusion Detection Message Exchange Format (IDMEF)*, document RFC 4765, Internet Engineering Task Force, Mar. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4765.txt

[13] A. A. Ramaki, M. Khosravi-Farmad, and A. G. Bafghi, "Real time alert correlation and prediction using Bayesian networks," in *Proc. 12th Int. Iranian Soc. Cryptol. Conf. Inf. Security Cryptol. (ISCISC)*, Sep. 2015, pp. 98–103.

[14] S. Mathew, C. Shah, and S. Upadhyaya, "An alert fusion framework for situation awareness of coordinated multistage attacks," in *Proc. 3rd IEEE Int. Workshop Inf. Assurance (IWIA)*, 2005, pp. 95–104.

[15] R. Sadoddin and A. Ghorbani, "Alert correlation survey: Framework and techniques," in *Proc. Int. Conf. Privacy, Security Trust Bridge Gap Between PST Technol. Bus. Services (PST)*, 2006, pp. 1–10.

[16] G. Jiang and G. Cybenko, "Temporal and spatial distributed event correlation for network security," in *Proc. Amer. Control Conf.*, vol. 2, 2004, pp. 996–1001.

[17] R. Anbarestani, B. Akbari, and F. Fathi, "An iterative alert correlation method for extracting network intrusion scenarios," in *Proc. 20th Iranian Conf. Electr. Eng. (ICEE)*, May 2012, pp. 684–689.

[18] Z. Wu, D. Xiao, M. Xiao, and X. Peng, "Using multilevel correlation in a unified platform of network security management: Design and implementation," in *Proc. Int. Symp. Electron. Commerce Security*, 2008, pp. 402–406.

[19] S. Benferhat, T. Kenaza, and A. Mokhtari, "A naive Bayes approach for detecting coordinated attacks," in *Proc. 32nd Annu. IEEE Int. Comput. Softw. Appl. Conf.*, Jul. 2008, pp. 704–709.

[20] Z. Li, A. Zhang, J. Lei, and L. Wang, "Real-time correlation of network security alerts," in *Proc. IEEE Int. Conf. e-Business Eng. (ICEBE)*, Oct. 2007, pp. 73–80.

[21] O. Dain and R. K. Cunningham, "Fusing a heterogeneous alert stream into scenarios," in *Applications of Data Mining in Computer Security* (Advances in Information Security). Berlin, Germany: Springer, 2002, pp. 103–122.

[22] D. Yu and D. Frincke, "A novel framework for alert correlation and understanding," in *Proc. Int. Conf. Appl. Cryptography Netw. Security*. Springer, 2004, pp. 452–466.

[23] B. Zhu and A. A. Ghorbani, "Alert correlation for extracting attack strategies," *J. Netw. Secur.*, vol. 3, no. 3, pp. 244–258, Nov. 2006.

[24] M. Kim, Y. Park, I. Han, and S. Cho, "A fast alert correlation method with Bayesian feature constraints," in *Proc. 15th Int. Conf. Cyber Warfare Security (ICCWS)*. New York, NY, USA: Academic, 2020, p. 277.

[25] K. Zhang, F. Zhao, S. Luo, Y. Xin, and H. Zhu, "An intrusion action-based IDS alert correlation analysis and prediction framework," *IEEE Access*, vol. 7, pp. 150540–150551, 2019.

[26] C. V. Zhou, C. Leckie, and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," *Comput. Secur.*, vol. 29, no. 1, pp. 124–140, 2010.

[27] H. T. Elshoush and I. M. Osman, "Alert correlation in collaborative intelligent intrusion detection systems—A survey," *Appl. Soft Comput.*, vol. 11, no. 7, pp. 4349–4365, Oct. 2011.

[28] G. Meng, Y. Liu, J. Zhang, A. Pokluda, and R. Boutaba, "Collaborative security: A survey and taxonomy," *ACM Comput. Surv.*, vol. 48, no. 1, pp. 1–42, Jul. 2015.

[29] E. Vasilomanolakis, S. Karuppayah, M. Mühlhäuser, and M. Fischer, "Taxonomy and survey of collaborative intrusion detection," *ACM Comput. Surv.*, vol. 47, no. 4, p. 55, May 2015.

[30] J. Steinberger, A. Sperotto, M. Golling, and H. Baier, "How to exchange security events? Overview and evaluation of formats and protocols," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 261–269.

[31] J. Steinberger, B. Kuhnert, A. Sperotto, H. Baier, and A. Pras, "Collaborative DDoS defense using flow-based security event information," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp. (NOMS)*, Apr. 2016, pp. 516–522.

[32] F. Skopik, *Collaborative Cyber Threat Intelligence: Detecting and Responding to Advanced Cyber Attacks at the National Level*. Boca Raton, FL, USA: CRC Press, 2018.

[33] ENISA. (Nov. 2014). *Standards and Tools for Exchange and Processing of Actionable Information*. [Online]. Available: https://www.enisa.europa.eu/publications/standards-and-tools-for-exchange-and-processing-of-actionable-information/at_download/fullReport

[34] M. A. Azad, S. Bag, F. Ahmad, and F. Hao, "Sharing is caring: A collaborative framework for sharing security alerts," *Comput. Commun.*, vol. 165, pp. 75–84, Jan. 2021.

[35] C. Fachkha and M. Debbabi, "Darknet as a source of cyber intelligence: Survey, taxonomy, and characterization," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1197–1227, 2nd Quart., 2016.

[36] E. Bou-Harb and C. Fachkha, "On inferring and characterizing large-scale probing and DDoS campaigns," in *Computer and Network Security Essentials*. Cham, Switzerland: Springer, 2018, pp. 461–474.

[37] E. Bou-Harb, C. Assi, and M. Debbabi, "CSC-detector: A system to infer large-scale probing campaigns," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 3, pp. 364–377, May 2018.

[38] E. Bou-Harb, M. Husák, M. Debbabi, and C. Assi, "Big data sanitization and cyber situational awareness: A network telescope perspective," *IEEE Trans. Big Data*, vol. 5, no. 4, pp. 439–453, Dec. 2019.

[39] S. Torabi, E. Bou-Harb, C. Assi, E. B. Karbab, A. Boukhtouta, and M. Debbabi, "Inferring and investigating IoT-generated scanning campaigns targeting a large network telescope," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 402–418, Jan. 2020.

[40] P. Richter and A. Berger, "Scanning the scanners: Sensing the internet from a massively distributed network telescope," in *Proc. Internet Meas. Conf.* New York, NY, USA: ACM, Oct. 2019, pp. 144–157.

[41] A. DasGupta, "The matching, birthday and the strong birthday problem: A contemporary review," *J. Statist. Planning Inference*, vol. 130, nos. 1–2, pp. 377–389, 2005.

[42] S. Baraty, D. Simovici, and C. Zara, "The impact of triangular inequality violations on medoid-based clustering," in *Foundations of Intelligent Systems* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2011, pp. 280–289.

[43] X. Qin and W. Lee, "Statistical causality analysis of INFOSEC alert data," in *Int. Workshop Recent Adv. Intrusion Detection*. Winter Garden, FL, USA: International Institute of Informatics and Cybernetics, 2003, pp. 73–93.

[44] S. Choi, S.-H. Cha, and C. Tappert, "A survey of binary similarity and distance measures," *J. Syst. Cybern. Inform.*, vol. 8, no. 1, pp. 43–48, 2009.

[45] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1. Berkeley, CA, USA: Univ. California Press, 1967, pp. 281–297. [Online]. Available: https://projecteuclid.org/euclid.bsmsp/1200512992

[46] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining (KDD)*. Palo Alto, CA, USA: AAAI Press, 1996, pp. 226–231.

[47] R. J. G. B. Campello, D. Moulavi, A. Zimek, and J. Sander, "Hierarchical density estimates for data clustering, visualization, and outlier detection," *ACM Trans. Knowl. Discovery Data*, vol. 10, no. 1, pp. 1–51, Jul. 2015.

[48] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and S. Jörg, "OPTICS: Ordering points to identify the clustering structure," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*. New York, NY, USA: ACM Press, 1999, pp. 49–60.

[49] B. Claise, B. Trammell, and P. Aitken, *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information*, document RFC 7011, Internet Engineering Task Force, Sep. 2013. [Online]. Available: https://www.ietf.org/rfc/rfc7011.txt

[50] A. L. E. CESNET. (2017). *Warden*. [Online]. Available: https://warden.cesnet.cz/en/index

[51] A. L. E. CESNET. (2017). *Intrusion Detection Extensible Alert*. [Online]. Available: https://idea.cesnet.cz

[52] T. Liston. (2003). *LaBrea: Sticky Honeypot and IDS*. [Online]. Available: https://labrea.sourceforge.io/

[53] D. Putri, "Honeypot cowrie implementation to protect SSH protocol in ubuntu server with visualisation using kippo-graph," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 8, no. 6, pp. 3200–3207, Dec. 2019.

[54] R. Hofstede, P. Celeda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras, "Flow monitoring explained: From packet capture to data analysis with NetFlow and IPFIX," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 2037–2064, 4th Quart., 2014.

[55] T. Cejka, V. Bartos, M. Svepes, Z. Rosa, and H. Kubatova, "NEMEA: A framework for network traffic analysis," in *Proc. 12th Int. Conf. Netw. Service Manage. (CNSM)*, Oct. 2016, pp. 195–201.

[56] M. Žádník, E. Sabik, and V. Bartos, "Detection of network flow timestamp reliability," in *Monitoring and Securing Virtualized Networks and Services*. Berlin, Germany: Springer, 2014, pp. 147–159.

[57] D. Smith. (2019). *Radware*. [Online]. Available: https://blog.radware.com/security/attack-types-and-vectors/2019/11/tcp-reflection-attacks-then-and-now

[58] V. Barto, "NERD: Network entity reputation database," in *Proc. 14th Int. Conf. Availability, Rel. Security*, Aug. 2019, pp. 1–7.

[59] V. Bartos, M. Zadnik, S. M. Habib, and E. Vasilomanolakis, "Network entity characterization and attack prediction," *Future Gener. Comput. Syst.*, vol. 97, pp. 674–686, Aug. 2019.

MARTIN ZADNIK is the Deputy Leader with the Department of Tools for Security and Administration, CESNET Association of Legal Entities. He has been the project leader in many national and European projects related to network security, cyber threat intelligence, and network monitoring at high speeds.

JAN WRONA is a Researcher and a Developer with CESNET Association of Legal Entities, where his interests are related to the network security and cyber threat intelligence. He is the Main Researcher and a Developer of the "GRIP" Technology that aims to lookup suspicious groups of network entities that perform similar correlated behavior.

KAREL HYNEK is currently pursuing the Ph.D. degree with FIT, CTU in Prague. His Ph.D. study is related to network security and network monitoring. Since his bachelor's study Ing., he has worked on several research projects and he is currently a Key Member of Network Traffic Monitoring Research Team with FIT.

TOMAS CEJKA is an Assistant Professor with FIT, CTU in Prague. He teaches network security course, and supervises a research group of bachelor's/master's/Ph.D. students. He is also a Researcher and the Team Leader with CESNET Association of Legal Entities, the operator of the Czech national research and education network.

MARTIN HUSÁK is a Researcher with the Institute of Computer Science, Masaryk University and a member of the University's Security Team (CSIRT-MU). His Ph.D. thesis addressed the problem of early detection and prediction of network attacks. His research interests include cyber situational awareness and threat intelligence with a special focus on the effective sharing of data from honeypots and network monitoring.