Recent Trends in Statistical Analysis of Event Logs for Network-Wide Intrusion Detection

Corentin Larroche, Johan Mazel, Stéphan Clémençon

Recent Trends in Statistical Analysis of Event Logs for Network-Wide Intrusion Detection

Corentin Larroche[1,2(✉)], Johan Mazel[1], and Stephan Clémençon[2]

[1] French National Cybersecurity Agency (ANSSI), Paris, France
first.last@ssi.gouv.fr
[2] LTCI, Télécom Paris, Institut Polytechnique de Paris, France
first.last@telecom-paris.fr

Abstract. Event logs are information-rich and complex data that keep track of the activity taking place in a computer network, and can therefore contain traces of malicious activity when an intrusion happens. However, such traces are scarce and buried under considerable volumes of unrelated information, making the use of event logs for intrusion detection a challenging research topic. We review some recent contributions to that area of research, focusing on the application of statistical analysis to various types of event logs collected over a computer network. Emphasis is put on the formalism used to translate the data into a collection of mathematical objects suited to statistical modelling.

Keywords: Event logs · Intrusion detection · Anomaly detection.

1 Introduction

With their often impressive volume as well as the abundant and valuable information they contain, event logs appear as an obvious candidate for big data and statistical learning methods. Among the possible benefits of their automated analysis, intrusion detection raises outstandingly great expectations: what if a model could make sense of this immense wealth of data, thereby gaining a subtle understanding of the normal behavior of a computer network and spotting hints of suspicious activity? Although such a seamless workflow hardly seems realistic in practice, this perspective has motivated a significant amount of research.

Our work aims to review some interesting contributions to the field of statistical analysis of event logs for network-wide intrusion detection that were published in the last ten years. We only consider detection methods which rely on statistical tools and attempt to find evidence of malicious behavior in high-level event logs. In particular, fine-grained analysis of the behavior of a host, using for instance system calls or information flow tracking between system-level objects, is out of our scope. In addition to listing what we consider to be the most relevant contributions, we focus on one specific aspect whose importance we seek to emphasize, namely the transformation that is applied to the raw event logs in order to obtain a collection of mathematical objects suited to statistical modelling. Two main paradigms are identified: aggregation-based methods,

which represent the network as a set of independent entities whose behavior can be reduced to the events in which they appear, and interaction-based methods, which consider each event as an interaction between two or more entities and analyze the high-order relationships that emerge from these interactions. Building upon this dichotomy, we propose a classification of statistical intrusion detection methods based on event logs. To the best of our knowledge, this aspect of the intrusion detection workflow has not been emphasized in related surveys (e.g. [23,6]), which mainly focus on the algorithms used downstream.

The rest of this paper is structured as follows: we formally define the notions of event logs and intrusion detection in section 2, then give an overview of aggregation-based and interaction-based methods in sections 3 and 4, respectively. We conclude with some suggested research directions in section 5.

2 Definitions and Problem Statement

We first give a formal definition of the data and the problem that are considered here, and then introduce the data representation step, which underpins the classification presented in sections 3 and 4.

2.1 Event Logs and Intrusion Detection – Formal Definitions

Consider a computer network, defined as a set of entities of various types (e.g. users, hosts). An event is an interaction between two or more entities, associated with a timestamp, an event type and a dictionary containing additional information. In practice, this definition encompasses various data sources: authentications can for instance be represented as interactions between users and hosts, with additional information such as the authentication package used. Likewise, a NetFlow record can be seen as an interaction between two hosts, further characterized by the protocol used, the number of packets exchanged, etc.

Given a sequence of events, intrusion detection can be broadly defined as looking for a subset of this sequence corresponding to malicious activity. Note that the sequence can either be observed as a stream or readily available in full. Since finding the exact subset of malicious events is a rather unrealistic goal in practice, detection algorithms mostly aim to extract a collection of suspicious events (or event sets), preferably ranked by their probability of being malicious. The following assumptions can typically be made to better specify the problem:

Assumption 1 *Malicious events are scarce compared to benign ones.*

Assumption 2 *Malicious event sets are distinguishable from benign ones.*

Assumption 3 *Malicious events are connected with each other with respect to the entities they involve.*

This last assumption reflects the idea that an intruder typically uses already compromised entities (such as hosts and user credentials) to propagate further

into the network. Thus, a new event resulting from the intruder's actions has to involve at least one entity that already appears in a previous malicious event. Under these assumptions, intrusion detection can be phrased as a case of anomaly detection: given a high volume of data mostly reflecting normal (usual) activity, the goal is to find a small but cohesive subset that deviates from the norm.

2.2 Data Representation – A Crucial Step

At this point, it should be noted that event logs are peculiarly complex data. Three important aspects stand out: first, time is a fundamental dimension here since the moment when an event appears, as well as the events that precede it, significantly matter in deciding whether it is normal or anomalous. Secondly, the notion of normality of an event is tightly related to the likelihood of the involved entities being associated with each other, and events involving some common entities may actually be disseminated clues of a single deliberate sequence of actions (as per assumption 3). Therefore, this combinatorial dimension has to be taken into account. Finally, entities and events can be of several types with different semantics (e.g. users and hosts, authentications and process creations), and this heterogeneity adds another layer of complexity to the data.

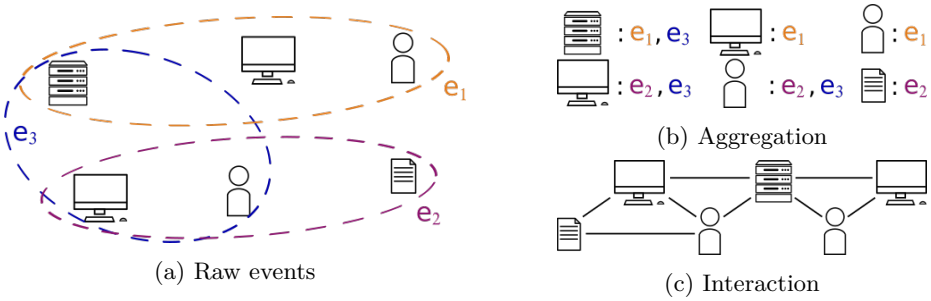
These specific characteristics make it nontrivial to apply standard anomaly detection algorithms to event logs. Indeed, even though such algorithms exist for various kinds of data (e.g. vectors in Euclidean spaces [7], discrete sequences [8], graphs [2]), none of these perfectly fits the complex nature of the input considered here. An intermediary representation is thus needed to translate the logs into a collection of simpler mathematical objects, while preserving enough information to enable detection of malicious activity. Due to its critical importance, this representation is the main criterion we use to categorize the contributions that are reviewed here. Two central paradigms are identified: the first one relies on aggregation, essentially treating entities as mutually independent and summarizing the events in which they are involved to model their behavior. In contrast, the second paradigm attempts to preserve the relational nature of event logs by directly modelling how entities interact with each other. These two paradigms are illustrated in figure 1 and described in more detail in the next two sections.

3 Divide and Conquer – Aggregation-Based Approach

We begin with the aggregation-based paradigm, first explaining the main intuition underlying it, and then exploring its practical applications in more detail.

3.1 General Definition

The fundamental idea behind aggregation-based models is that the set of events involving a given entity can be understood as a history of this entity's behavior. Therefore, summarizing this set into a mathematical object enables comparison of an entity's activity during a given time window with, for instance, its own



(a) Raw events (b) Aggregation (c) Interaction

Fig. 1: Illustration of the two representation paradigms: given events viewed as interactions between entities (a), aggregation-based models (b) represent each entity as the set of events involving it, while interaction-based models (c) consider high-order relationships between entities.

past activity or that of presumably similar entities. Such an object should be carefully defined to preserve evidence of malicious behavior when it exists, while being simple enough to be fed to a standard anomaly detection algorithm.

The anomaly detection language introduced in [18] can be taken as a general framework for aggregation-based methods. In particular, it introduces the notions of *extent* and *baseline*. The former is defined as the conjunction of an entity or group of entities (*entity extent*) and a time period (*temporal extent*), whereas the latter sets the scope of the comparison: an entity's behavior can be compared with the behavior of other entities in the same temporal extent (*cross-sectional baseline*), with the behavior of the same entity in other temporal extents (*longitudinal baseline*) or with both (*simultaneous baseline*). Together, these two notions are the high-level inputs of the anomaly detection process.

3.2 Usual Aggregation Keys and Mathematical Objects

In order to instantiate the general idea of aggregation-based modelling, the first important step is to define how to aggregate events (i.e. the entity extent to use). We now describe some widely used aggregation keys, along with the usually associated mathematical objects (see table 1 for a summary).

First of all, a common intrusion detection scheme consists in monitoring the activity of each user in order to spot compromised accounts. This amounts to aggregating events by user, and the aggregated events can then be summarized into a feature vector, with features often made out of event counts [10,14,29]. In that case, the activity of a given user during a period of time is simply characterized by the number of actions of each possible type taken by this user. Emphasis can alternatively be put on the order in which these actions happen, thus representing the set of events as a discrete sequence [25,5]. More complex sequence-based methods can also leverage the inter-arrival times of events [32]. Finally, some authors picture event sets as graphs, which can for instance be defined by focusing on a user's authentications: each vertex represents a host,

Table 1: Aggregation-based methods, grouped by aggregation key and mathematical object.

Agg. key \ Object	Scalar or vector	Sequence	Graph
User	[10,12,14,29]	[25,5,32]	[16,24]
Host	[26,31,4]	[11,21]	-
Entity pair	[20,27,17]	[1]	-

and directed edges between vertices stand for authentications from source hosts to destination hosts. The obtained graph can then be transformed into a feature vector [12] or directly analyzed using graph-oriented tools [24]. Another kind of graph can also be built by associating a vertex with each event and adding links between events sharing some specific traits [16].

An alternative to user-based aggregation is its host-based counterpart, which aims to model the usage pattern of a host rather than the behavior of a user. Abstracting the data into vectors of event counts remains a popular method in this context [26,31], but other ideas have also been proposed. In particular, graph-based features can be derived for hosts as well, using network traffic metadata to build a host communication graph [4]. Communications between hosts can also be treated from the point of view of a single host as a discrete sequence of sources or destinations [11,21].

Finally, a more fine-grained understanding of the activity contained in the logs can be achieved by aggregating events by user-host or host-host pair. This is a first step towards handling the combinatorial nature of events, while still considering each pair separately from the others. The usual multivariate count-based approach is still relevant here [27], and discrete sequences can be used as well [1]. However, because the mere existence of an interaction between two entities already carries significant information, it is possible to use even simpler mathematical objects (e.g. a single interaction count [20] or a boolean [17]).

4 All Intertwined – Interaction-Based Approach

We now turn to the interaction-based paradigm, once again starting with a general explanation before reviewing its applications.

4.1 General Definition

Unlike aggregation-based methods, which essentially break the complex entanglement of events into a collection of subsets that are then treated as independent, the interaction-based approach attempts to capture the intricate patterns of association between entities. In particular, the relationships between different interactions (e.g. events that involve some common entities) are explicitly taken into account, possibly unveiling high order dependencies between entities and cohesive sets of anomalous events (as defined in assumption 3). This makes

Table 2: Interaction-based methods and the objects they rely upon.

Object	**Graph**	**Bipartite graph**	**Knowledge graph**	**Hypergraph**
References	[13]	[9,30,19,22]	[15]	[28,3]

interaction-based modelling different from aggregation by entity pair, which only compares the activity of a given pair with its own past activity or with that of other pairs, without looking for dependencies between them.

This paradigm thus implies a more global view of the activity happening inside a computer network, and as a consequence, it can lead to more complex objects and models. The next section reviews some of these (see table 2 for a summary of interaction-based methods and associated mathematical objects).

4.2 Existing Models and Underlying Objects

Unsurprisingly enough, graphs are a popular tool for abstracting event logs in an interaction-based formalism. More specifically, events can typically be translated to user-item interactions, which in turn yields a bipartite user-item interaction graph. Practical examples include bipartite authentication graphs, whose vertices are users and hosts, with each edge indicating an authentication of a user on a host. Communications between hosts can also be understood as a graph.

Detection models in this setting can for instance rely on the community structure of the graph, marking an unusually high number of inter-community edges as a network-wide anomaly [19]. Alternatively, a similarity measure between vertices can be designed using the structure of the graph, enabling the search for outliers [9]. Suspicious entities can then be identified, providing more fine-grained information than a global model. However, even more targeted alerts can be obtained by building a statistical model of historical interactions, which can then be used to predict how likely two given entities are to interact with one another. This can typically be done through collaborative filtering, a method relying on the intuition that if user A usually interacts with the same items as user B, then A should be more likely to interact with a new item if B already has. Having built such a model, one can then assign probabilities to new interactions, raising alerts on highly improbable ones [30]. This method can be further enriched by integrating a temporal aspect [13,22] or using attributes of the entities as additional input [22].

Graphs, however, can only handle dyadic interactions. This can be a problem when working with events that involve more than two entities (e.g. a user running an executable on a host). A way to circumvent this limitation is to depict the events themselves as vertices which are linked to the entities they involve, resulting in a heterogeneous graph (also called knowledge graph) carrying more fine-grained information. Looking for outliers in this graph can then reveal suspicious events [15]. Alternatively, events can be explicitly described as polyadic interactions (i.e. hyperedges in a hypergraph) and analyzed as such with dedicated tools, including pattern mining [28] and representation learning [3].

5 Conclusion – Research Directions

Having reviewed the main trends in the literature, we now conclude with some possible research directions. As for aggregation-based methods, one of the main challenges lies in correlating anomalies detected for different entities (based on assumption 3). Indeed, having separated the events into independent subsets, it is then nontrivial to assess whether different subsets that appear anomalous on their own can be traced back to a single trail of malicious activity. Interesting ideas regarding this problem can be found in [20,26,17]. Another area for improvement is the use of entities' roles to build better models: instead of only using past activity of a single entity to determine whether its current behavior is anomalous, it can be interesting to include events related to presumably similar entities. However, defining groups of similar entities, especially with limited background information about the network, is challenging (see [10] for a purely behavioral method and [14] for an approach relying on organisational roles).

Finally, possible research directions in the interaction-based paradigm include more accurate modelling of temporal dynamics: even though some recent contributions [13,22] have tackled this issue, the highly complex temporal variations of activity in computer networks (including periodic automated behavior, seasonal patterns and long-term drift) still provide room for improvement. The heterogeneity of interactions also calls for richer models, as most published methods only consider a single kind of interaction, thus ignoring the relationships that could exist between events of different types. Despite these limitations, interaction-based modelling seems to be a promising approach, and we think it deserves increased attention from the research community.

References

1. Adilova, L., Natious, L., Chen, S., Thonnard, O., Kamp, M.: System misuse detection via informed behavior clustering and modeling. In: DSN-W (2019)
2. Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. Data Min. Knowl. Discov. **29**(3), 626–688 (2015)
3. Amin, M.R., Garg, P., Coskun, B.: Cadence: Conditional anomaly detection for events using noise-contrastive estimation. In: AISec (2019)
4. Bohara, A., Noureddine, M.A., Fawaz, A., Sanders, W.H.: An unsupervised multi-detector approach for identifying malicious lateral movement. In: SRDS (2017)
5. Brown, A., Tuor, A., Hutchinson, B., Nichols, N.: Recurrent neural network attention mechanisms for interpretable system log anomaly detection. In: MLCS (2018)
6. Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun. Surveys Tuts. **18**(2), 1153–1176 (2015)
7. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM comput. surv. **41**(3), 1–58 (2009)
8. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: A survey. IEEE Trans. Knowl. Data Eng. **24**(5), 823–839 (2010)
9. Chen, Y., Malin, B.: Detection of anomalous insiders in collaborative environments via relational analysis of access logs. In: CODASPY (2011)

10. Eldardiry, H., Bart, E., Liu, J., Hanley, J., Price, B., Brdiczka, O.: Multi-domain information fusion for insider threat detection. In: S&P Workshops (2013)
11. Heard, N., Rubin-Delanchy, P.: Network-wide anomaly detection via the dirichlet process. In: ISI (2016)
12. Kent, A.D., Liebrock, L.M., Neil, J.C.: Authentication graphs: Analyzing user behavior within an enterprise network. Comput. Secur. **48**, 150–166 (2015)
13. Lee, W., McCormick, T.H., Neil, J., Sodja, C.: Anomaly detection in large scale networks with latent space models. arXiv preprint arXiv:1911.05522 (2019)
14. Legg, P.A., Buckley, O., Goldsmith, M., Creese, S.: Automated insider threat detection system using user and role-based profile assessment. IEEE Syst. J. **11**(2), 503–512 (2015)
15. Leichtnam, L., Totel, E., Prigent, N., Mé, L.: Sec2graph: Network attack detection based on novelty detection on graph structured data. In: DIMVA (2020)
16. Liu, F., Wen, Y., Zhang, D., Jiang, X., Xing, X., Meng, D.: Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within enterprise. In: SIGSAC (2019)
17. Liu, Q., Stokes, J.W., Mead, R., Burrell, T., Hellen, I., Lambert, J., Marochko, A., Cui, W.: Latte: Large-scale lateral movement detection. In: MILCOM (2018)
18. Memory, A., Goldberg, H.G., Senator, T.E.: Context-aware insider threat detection. In: AAAI Workshops (2013)
19. Moriano, P., Pendleton, J., Rich, S., Camp, L.J.: Insider threat event detection in user-system interactions. In: MIST (2017)
20. Neil, J., Hash, C., Brugh, A., Fisk, M., Storlie, C.B.: Scan statistics for the online detection of locally anomalous subgraphs. Technometrics **55**(4), 403–414 (2013)
21. Passino, F.S., Heard, N.A.: Modelling dynamic network evolution as a pitman-yor process. Found. Data Sci. **1**(3), 293 (2019)
22. Passino, F.S., Turcotte, M.J., Heard, N.A.: Graph link prediction in computer networks using poisson matrix factorisation. arXiv preprint arXiv:2001.09456 (2020)
23. Patcha, A., Park, J.M.: An overview of anomaly detection techniques: Existing solutions and latest technological trends. Comput. netw. **51**(12), 3448–3470 (2007)
24. Powell, B.A.: Detecting malicious logins as graph anomalies. J. Inf. Secur. Appl. **54**, 102557 (2020)
25. Rashid, T., Agrafiotis, I., Nurse, J.R.: A new take on detecting insider threats: exploring the use of hidden markov models. In: MIST (2016)
26. Sexton, J., Storlie, C., Neil, J.: Attack chain detection. Stat. Anal. Data Min. **8**(5-6), 353–363 (2015)
27. Shashanka, M., Shen, M.Y., Wang, J.: User and entity behavior analytics for enterprise security. In: BigData (2016)
28. Siadati, H., Memon, N.: Detecting structurally anomalous logins within enterprise networks. In: SIGSAC (2017)
29. Tuor, A., Kaplan, S., Hutchinson, B., Nichols, N., Robinson, S.: Deep learning for unsupervised insider threat detection in structured cybersecurity data streams. In: AAAI Workshops (2017)
30. Turcotte, M., Moore, J., Heard, N., McPhall, A.: Poisson factorization for peer-based anomaly detection. In: ISI (2016)
31. Veeramachaneni, K., Arnaldo, I., Korrapati, V., Bassias, C., Li, K.: ai^2: training a big data machine to defend. In: BigDataSecurity (2016)
32. Yuan, S., Zheng, P., Wu, X., Li, Q.: Insider threat detection via hierarchical neural temporal point processes. In: BigData (2019)