On the Sequential Pattern and Rule Mining in the Analysis of Cyber Security Alerts

Martin Husák
Institute of Computer Science and Faculty of Informatics
Masaryk University
Brno, Czech Republic
husakm@ics.muni.cz

Jaroslav Kašpar
Institute of Computer Science and Faculty of Informatics
Masaryk University
Brno, Czech Republic
kaspar@ics.muni.cz

Elias Bou-Harb
Cyber Threat Intelligence Laboratory
Florida Atlantic University
Boca Raton, USA
ebouharb@fau.edu

Pavel Čeleda
Institute of Computer Science
Masaryk University
Brno, Czech Republic
celeda@ics.muni.cz

ABSTRACT

Data mining is well-known for its ability to extract concealed and indistinct patterns in the data, which is a common task in the field of cyber security. However, data mining is not always used to its full potential among cyber security community. In this paper, we discuss usability of sequential pattern and rule mining, a subset of data mining methods, in an analysis of cyber security alerts. First, we survey the use case of data mining, namely alert correlation and attack prediction. Subsequently, we evaluate sequential pattern and rule mining methods to find the one that is both fast and provides valuable results while dealing with the peculiarities of security alerts. An experiment was performed using the dataset of real alerts from an alert sharing platform. Finally, we present lessons learned from the experiment and a comparison of the selected methods based on their performance and soundness of the results.

CCS CONCEPTS

• **Information systems** → **Data mining**; • **Security and privacy** → *Intrusion detection systems*;

KEYWORDS

data mining, cyber security, sequential pattern mining, sequential rule mining, alert correlation, attack prediction

1 INTRODUCTION

Data mining is gaining a lot of attention recently in the field of cyber security. A lot of recent research is dedicated to its application in intrusion detection, where the data mining is performed over network data such as packets or network flows [2]. Another interesting use case for data mining is one level of abstraction higher, in the processing of outputs of intrusion detection. In this case, the interest is in finding patterns and sequences of events that are observable via intrusion detection systems. Thus, we can learn about complex attacks including multi-stage attacks and advanced persistent threats. Due to the growing popularity of information sharing and collaborative intrusion detection, one may also observe large-scale attacks or spread of an attack in a large network [34]. Further, given a typical attack progression, one can employ such information to predict future events in the network traffic. In either case, data mining is used to learn about patterns in security alerts, which are otherwise hard to create. Namely, sequential pattern mining fits the needs of cyber security analytics as the sequences in the security alerts correspond to attack scenarios.

In this paper, we introduce our work on sequential pattern and rule mining of cyber security alerts. The three main research questions and contributions of this paper are as follows:

(1) *What are the use cases of sequential pattern mining in analysis of cyber security alerts?*

To this end, we survey related work to find the typical use cases of data mining of cyber security alerts and identified its main applications. We extract approaches, methods, and algorithms used in the literature to find out how deeply the cyber security research community understands data mining and if suitable and efficient methods are in use.

(2) *Which approaches are the most suitable and effective for mining sequences in security alerts?*

Given the use cases and a dataset, we evaluated the existing data mining approaches to find out which ones are most suitable from the perspective of efficiency and soundness of results. We used a dataset obtained from an alert sharing platform that contains a large number of real-world attacks and alerts. Thus, we can go through the whole process of sequence mining, from feature selection to evaluation of results, to generate insights into the peculiarities of cyber security data and its influence on sequence mining methods. An optimal sequence mining method should be effective and provide sound, yet not obvious, results.

(3) *What are the effects of optimizations and data reductions in sequence mining of security alerts?*

Finally, we would like to know which optimizations are worth considering when using sequence mining in cyber security. Cyber

ARES '17, August 29-September 01, 2017, Reggio Calabria, Italy

security data is known for their variability and incompleteness, which influences the results. Many methods were proposed to select only the most interesting results, e.g., by omitting patterns that contained in larger patterns or selecting only top-K results by a predefined parameter. Therefore, we evaluate and discuss these optimizations to see which are useful for the given data.

The remainder of this paper is organized into six sections. In Section 2, we present the related work and the use cases of sequential pattern mining in the analysis of network security alerts. Section 3 lists and evaluates sequential pattern mining methods from the perspective of the set use case. Section 4 describes the experimental evaluation of the methods. The results are presented and discussed in Section 5. Section 6 concludes the paper with lessons learned and proposals for future work.

2 RELATED WORK

We performed a systematic search for papers that mention data mining in cyber security, namely data mining applied on security alerts. Utilization of data mining in cyber security dates back to 1998, when Lee and Stolfo [22] proposed using data mining for intrusion detection. In this highly cited work, association rules and frequent episodes were used to discover patterns that describe program or user behavior. Recently, various use cases and utilizations of data mining and machine learning in the field of cyber security were surveyed by Buczak and Guven [2].

Many works have been published that use some data mining method to analyze cyber security data. However, these papers typically use basic or well-known data mining methods and approaches that are not necessarily the most suitable and efficient ones. To the best of our knowledge, there is no paper dedicated specifically to the problem of selecting the most suitable data mining method for the purpose of cyber security alert analysis. Based on the related work, we identified two main use cases for data mining in alert analysis, alert correlation and attack prediction. The use cases are presented in Table 1. For each related work, we include the proposed use case, approach, algorithm, expected outputs, and evaluation, if available. As we can see, association rule mining and frequent episode mining are the most popular approaches. If an existing algorithm or implementation is used, it is mostly Apriori or some of its variants.

2.1 Alert correlation

One use case of sequential pattern mining is alert correlation, an analysis process that takes the alerts produced by intrusion detection systems and produces compact reports [33] by grouping together individual alerts using logical relations between them [5]. Shin and Ryu [31] used association rule mining, frequent episode mining, and clustering for the purpose of alert correlation. Time window size and support values are discussed and the approaches are evaluated using DARPA 1998 dataset. Dasgupta et al. [4] used frequent episode mining to analyze security alerts from DARPA 2000 dataset. Some limitations, e.g., minimal confidence and maximal episode length, are mentioned, but the data mining component is otherwise described very briefly as a part of a larger system. Treinen and Thurimella [32] proposed a framework that applies association rule mining in large intrusion detection infrastructures to process alerts from intrusion detection systems. Shin and Jeong [30] presented a data mining framework, which includes association rule miner, frequent episode miner, and cluster miner. However, as stated also in [6], the authors used inefficient algorithms for such an application, e.g., Apriori algorithm to extract frequent episodes. Li et al. [25] used sequential pattern mining in real time to find out the frequency and sequence features of multi-stage attacks. In this work, AprioriAll algorithm is used to construct a pattern graph and the approach is evaluated using the DARPA 2000 dataset.

Alert filtering and sensor profiling are closely tied to alert correlation. Clifton and Gengo [3] used sequential association mining, specifically frequent episodes, to lower false positives in intrusion detection. Their goal is to identify sequences of alarms caused by normal operations based on preposition that a common sequence of alerts is probably not the result of actual intrusion attempt. Manganaris et al. [27] briefly mentions sequential patterns as an alternative to association rules in their work dedicated to adaptive alert filtering and sensor profiling. Sequential pattern mining was proposed as a future work.

2.2 Attack prediction

An advanced use case is an attack projection and prediction. In this task, the sequential patterns are used to create models of attackers' behavior. The attacks are then compared to the models and if a partial match is found, the possible future moves of an attacker can be predicted. The models can take many forms, e.g., attack graphs or Markov models. Li et al. [24] and Lei et al. [23] constructed attack graphs using data mining. Lie et al. used association rule mining, while Lei et al. used transaction database and sequential pattern mining. Both works describe used data mining algorithms. Farhadi et al. [6] used data mining to construct Markov model as a mean of attack prediction. Association rule mining was used and algorithms are provided. Similar approach was proposed by Katipally et al. [19], but without any further description. Kim and Park [20] used continuous association rule mining algorithm (CARMA) to create sequential rules by applying attack thread and attack session to sequential association rule for attack and threat prediction. Recently, Jiang et al. [18] used association rule mining on honeypot logs to predict suspicious network traffic present in the honeypots.

3 APPROACHES TO SEQUENCE MINING

Background on sequence mining was presented by Agrawal and Srikant in 1995 [1]. Since that time, many approaches and algorithms were presented in the literature as surveyed by Mabroukeh and Ezeife [26] or Mooney and Roddick [28]. With a growing number of methods and algorithms, it is hard to keep pace with the field and to select the right method for a given use case. In this section, we discuss how we enumerated a set of candidate methods to be evaluated in an experiment. The selected methods are briefly described.

3.1 Method selection

The first task was to select a set of candidate methods. We followed the map of data mining algorithms provided at the web page of

Table 1: Related work on using data mining in cyber security alert analysis

Authors	Use case	Approach	Algorithm/Implementation	Evaluated on
Clifton and Gengo [3]	Alert correlation	Frequent episode mining	Query Flocks	live data
Manganaris et al. [27]	Alert correlation	Association rule mining	custom	live data
Shin and Ryu [31]	Alert correlation	Association rule mining, Frequent episode mining	custom Apriori-based, custom	KDD Cup 1999, DARPA 1998 dataset
Dasgupta et al. [4]	Alert correlation	Frequent episode mining	custom	DARPA 2000 dataset
Treinen and Thurimella [32]	Alert correlation	Association rule mining	DB2 Intelligent Miner	live data
Li et al. [25]	Alert correlation	Sequential pattern mining	AprioriAll.	DARPA 2000 dataset, live data
Shin and Jeong [30]	Alert correlation	Association rule mining, Frequent episode mining	custom Apriori-based, custom	DARPA 1998 dataset
Li et al. [24]	Attack prediction	Association rule mining	custom	DARPA 1999, DARPA 2000 datasets
Lei et al. [23]	Attack prediction	Sequential pattern mining	custom	live data
Farhadi et al. [6]	Attack prediction	Association rule mining	custom	DARPA 2000 dataset
Kim and Park [20]	Attack prediction	Continuous association rule mining	-	-
Jiang et al. [18]	Attack prediction	Association rule mining	Apriori, FP-Growth	live data

SPMF library[1]. The map serves as a guide and a decision tree that helps to choose suitable data mining methods and algorithms.

On the top levels, we choose a motivation and the expected form of results. Both of our use cases direct in the same way; we want to discover interesting patterns in data and the patterns should be in a form of sequential patterns or sequential rules. By selecting only sequential patterns and rules, we omit the large area of association rule mining, which was used a lot in related work. Thus, our first proposal for the field of security alert analysis is switching from association rule mining to sequential rule and pattern mining. Similarly, sequence prediction algorithms were not considered for our use cases, even though an attack prediction is one of our use cases. Instead, sequential rule mining is proposed.

The next criterion by which we select candidate methods is whether we want the results to include confidence or probability. In this moment, our two use cases are diverging. While attack prediction requires such information and, thus, should use sequential rules, the alert correlation does not need such data and, thus, should use sequential patterns.

Sequential patterns are a suitable output of data mining for alert correlation. Further criteria include time constraints, dimensions, and the volume of results. Time might play a significant role in cyber security, so we decided to evaluate both algorithms with and without time constraints. On the other hand, multi-dimensionality is not a concert in this work as we yet cannot identify more dimensions of sequences consisting of security alerts. Thus, multi-dimensional sequential pattern mining methods are not included in the evaluation. Finally, the volume of results can be influenced by the selected method, either using a top-K approach, compressing sequential patterns or selecting other concise subset. We consider these options as highly relevant to the use cases as they can improve utilization of the outputs. Thus, top-K, compressing, closed, maximal, and generator patterns are selected for evaluation.

Sequential rules are a desired output of data mining for the purpose of attack prediction. Further criteria on selecting suitable algorithms are information about profit, time constraints, limitations to number of results, and redundancy of results. First, we have no requirements on information about profit. Thus, high-utility sequential rule mining methods were not considered. These methods could be used in a situation, in which we could calculate the cost of an attack and value of targets. This could be a topic of future work in the field of cyber situational awareness, but is not further considered in this paper. Second, methods with time constraints will be subject to experimental evaluation, thus they are added to the candidate method set. Similarly, top-K approaches filter out the most interesting rules. Finally, the problem of redundancy was not encountered in our or related work. Thus, method enhancements dealing with redundancy in results were not included in the experiment.

3.2 Methods in details

In the previous section, we selected a set of candidate methods, which are described here in details. In total, we identified seven sequential pattern mining methods and three sequential rule mining methods.

Sequential pattern mining is looking for sequential patterns (subsequences) which often occur in an input sequential database; these patterns are called *frequent* [26]. Sequential database contains set of sequences where sequence is consecutive list of itemsets. The *support* of sequential pattern is a number of sequences where the pattern occur divided by total number of sequences contained in an input database. Pattern is considered as frequent if support of the pattern is not smaller than a user defined minimal support.

[1] http://www.philippe-fournier-viger.com/spmf/map_algorithms_spmf_data_mining097.png

Top-K sequential patterns differs in the selection of patterns. Determining the value of minimal support can be difficult, because an unsuitable choice can cause algorithms to be very slow and generate extremely large amount of patterns or generate none or too few patterns. In the top-K algorithms, there is no need to define minimum support value. Top-K algorithms discover k most frequent patterns for a user defined k [8].

Sequential pattern mining often generates lots of patterns with redundant information. For example, for very long frequent pattern there has to be also generated every subsequence of the pattern as a unique pattern, because every subsequence has to have support at least same as the original pattern. There are several methods addressing this problem as closed, maximal and generators patterns, which generate concise subset of all frequent patterns.

Closed sequential pattern mining is an optimization of sequential pattern mining that filters results contained in other results. A pattern is called frequent closed pattern if there is no supersequence with the same support [7].

Maximal sequential pattern mining is another optimization that aims at reducing the number of results. A pattern is called frequent maximal pattern if there is no supersequence that is also frequent [14]. A detailed comparison of closed and maximal sequential pattern mining can be found in the work by Mabroukeh et al. [26].

Sequential generator patterns is another optimization aiming at filtering results within results. A pattern is called frequent generator pattern if it has no subsequence with the same support [10].

Compressing sequential patterns come with alternative interestingness measures that address the redundancy issue (many patterns with very similar meanings are generated). The GoKrimp algorithm generate non-redundant patterns based on minimum description length principle [21]. An advantage of this approach is that there is no need to set input parameters and it generates patterns that significantly compress the data.

Sequential pattern mining with time constraints extends patterns with information about interval between itemsets and allow users to define time constraints that reduce number of generated patterns. There are two kinds of item interval measurements, item gap and time intervals [17]. Item gap is defined as number of itemsets between successive itemsets in pattern. Time interval requires time extended database where every itemset has assigned value representing time of occurrence so the time interval between two itemsets is defined as length of time between the occurrences of the itemsets. Pattern mining with time intervals is able to generate more precise patterns than pattern mining with item gaps [17]. We used sequential pattern mining with time intervals, because the time information is available to us.

Sequential rule mining produces rules in the form of implication $A \Rightarrow B$ where set of items A is followed by set of items B [12]. Rules have support value same as patterns, but unlike patterns also offers the probability of confidence that items A will be followed by the items B. *Confidence* of a rule $A \Rightarrow B$ is number of sequences in which the rule appears divided by number of sequences in which appear items from itemset A. This is the reason why sequential rules are more useful for doing predictions. Mining sequential rules requires defining values for minimal support and minimal confidence on the basis of which will be generated rules.

Top-K sequential rule mining is similar to top-K sequential pattern mining. Determining values for input parameters as minimal support and minimal confidence can be difficult, because, based on the choice of these parameters, the algorithms can be very slow and generate extremely large amount of rules or generate none or too few rules. Top-K algorithms address this problem by generating top k rules with maximum support for user defined k [13].

Sequential rule mining with window constraint finds rules that appears within a sliding-window whose size is user defined [15]. This means that a rule occurs in a sequence only if the number of items in the sequence between the first and the last item from the rule is not bigger than the user defined window size.

4 EXPERIMENT SETUP

To compare the existing approaches, we performed an experiment in which we applied selected data mining approaches to the dataset of security alerts. In this section, we describe the dataset, the process of creation of the sequential databases, and the results of applying the methods presented in the previous section.

A general process of data mining goes as follows [2]. First, we have to identify class attributes (features) and classes from training data. Second, we have to identify a subset of the attributes necessary for classification (i.e., dimensionality reduction). Then, we can finally run the data mining algorithms. The two preliminary steps are demonstrated on a real-world dataset.

We created a set of scripts in python to process the dataset, build the databases, run the data mining algorithms, and format the results. The scripts are publicly available on GitHub[2].

4.1 Dataset

To evaluate the existing approaches, we used a dataset obtained via cyber security alert sharing platform SABU[3]. Our platform exchanges security alerts among participating networks, mostly campus networks associated with CESNET, national research and education network of the Czech Republic. Various alert sources are deployed in participating networks. Most of the alerts are raised by network intrusion detection systems and honeypots. Typical alerts include reports of network scanning, brute-force password attacks, exploit attempts, etc.

We extracted alerts shared during one week. In total, we got 16 million security alerts. The alerts are formatted in IDEA[4], an extensible data exchange format inspired by IDMEF and suited for the needs of SABU platform. Each alert contains at least a timestamp and a category of a reported event. Depending on the event type, information on event source, target, or both are provided. These information typically include IP addresses, ports, protocols, and other network identifiers.

It is worth noting that the dataset and IDEA format have several distinct features. The alerts were collected mostly from network-based intrusion detection systems. Thus, almost all the alerts include network identifiers of attack source and target, i.e., IP addresses and ports. Unlike other security alert formats, IDEA has its own categorization of events. A type of reported event is contained in

[2] https://github.com/CSIRT-MU/SecAlertSeqMining
[3] https://sabu.cesnet.cz
[4] https://idea.cesnet.cz

Table 2: Sequential databases

Database	Sequences	Unique items
Source and Target IPs with ports	9,571,703	76,131
Source and Target IPs without ports	9,571,703	7
Source IPs with ports	1,407,113	76,135
Source IPs without ports	1,407,113	10
Target IPs with ports	6,331	76,135
Target IPs without ports	6,331	7

the mandatory field. Another difference from other formats is the mandatory timestamp of an event. While timestamp of actual start of an event is generally preferred, IDEA considers the timestamp of detection as mandatory due to practical deployment. Many intrusion detection systems have a problem extracting actual starting time of an event, while the time of detection is always known.

4.2 Feature selection and database preparation

In the first step of pattern mining over the dataset, we selected the features to extract. We included two mandatory fields of alerts, category of an event and timestamp of event detection. Then, we added source and target identifiers, source and destination IP addresses and target port. Source IP address was present in almost every alert and 200,000 alerts missed target IP address. Source port is not particularly useful, but the target port was missing in 500,000 alerts. The aforementioned entries are the solid base for feature selection. Other data entries were considered as well, e.g., protocol and number of network connections, but they were sparsely present in the alerts than the basic 5-tuple. When the features to extract were selected, we created sequential databases, with which the sequential pattern mining algorithms can work.

Sequential databases can be viewed from several perspectives. In this work, we chose a perspective of source-target interaction. Thus, a sequential database contains either sequences of events between a source and a target, or sequences of events that share just one of these values. We created a sequential database for each of these cases. Summary of the resulting sequential databases is available in Table 2. The events were aggregated by the IP addresses. The sequences were then sorted by the event timestamps. Alerts with missing IP addresses or ports were not used. Items in the databases consisted either of event categories or pairs of event category and target port.

Sequential database with time intervals is a special case of sequential database, in which the itemsets in sequences are extended by a number representing information about time. For each sequential database presented above, we created its variant with time intervals. The numbers of sequences and unique items in the databases are the same as in the basic sequential databases presented in Table 2. The important step in creating the sequential database with time intervals is setting the unit of time. In our case, we chose 5 minutes, so the alerts that appeared within same 5 minute interval was aggregated into the same itemset. The reason for this is that majority of the intrusion detection systems contributing to the alert sharing platform uses 5-minute time window for raw data analytics. Having in mind that the alert timestamps represent the time of detection, i.e., the end of the processing time window, it is common to have time differences between alerts in the dataset in multiples of 5 minutes.

Table 3: Approaches and algorithms

Approach	Algorithm(s)
Sequential pattern mining	CM-SPADE [7]
Top-K sequential pattern mining	TKS [8]
Closed sequential pattern mining	CM-ClaSP [7]
Sequential generator pattern mining	VGEN [10]
Maximal sequential pattern mining	VMSP [14]
Compressing sequential pattern mining	GoKrimp [21]
Sequential pattern mining with time constraints	HirateYamana [17]
Closed sequential pattern mining with time constraints	Fournier08-Closed+time [11]
Sequential rule mining	RuleGrowth [12]
Sequential rule mining with window constraints	TRuleGrowth [15]
Top-K sequential rule mining	TopKRules [13]

Due to computational and memory complexity of some sequential pattern mining algorithms, we had to reduce the volume of sequential databases. Random sampling of sequences was used in favor of selecting shorter time interval of alert collecting. The motivation for this was to be able record long-term patterns. Additional test showed that, in our case, pattern sampling had only minimal effect on the results, i.e., found patterns and support values.

4.3 Data mining algorithms

In this work, we used SPMF [9], an open-source data mining library. SPMF specializes in pattern mining and implements all the methods we want to evaluate, unlike other data mining tools and libraries, e.g., Weka [16]. SPMF is a well-known library in the data mining community and can be considered as an exemplary implementation of the methods. We selected a set of suitable sequence mining methods in Section 3.2. All methods are implemented in SMPF, typically with more than one algorithm. Thus, we had to also select an algorithm and its implementation, typically we picked the most efficient one or at least the most recent one. A list of selected algorithms for each approach is presented in Table 3.

5 RESULTS AND DISCUSSION

In this section, we present and discuss results of an experiment. First, we show how the selection of database influenced algorithm efficiency. Then, a comparison of sequence mining algorithms is presented. Some interesting results are presented next to illustrate how sound the results are.

To begin with, we discuss several aspects of the dataset we used. Using the data from real environment instead of an artificial dataset showed some practical limitations of sequence mining of cyber security alerts. As we show in the following sections, many algorithms had troubles processing the full dataset and we had to

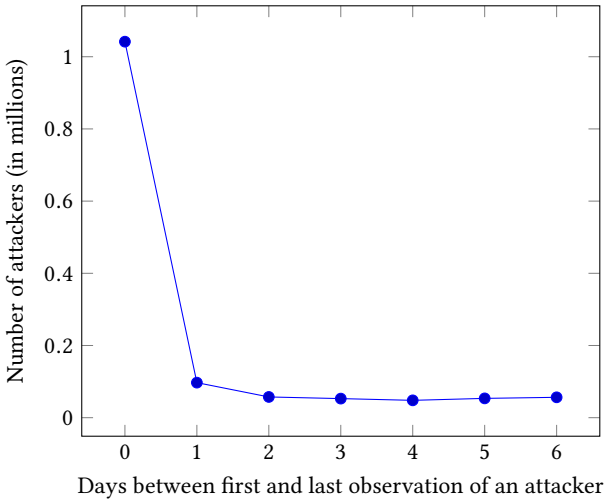

Figure 1: Persistence of attackers (unique IP addresses). The figure shows how many attackers were seen for how long.

reduce the database volume by sampling the data. We would like to emphasize that the data from a whole week were used to discover long sequences, e.g., attackers that are active for several days. As we can see on Figure 1, the majority of attackers were active for one day. However, the long tail of attackers active for several days should be taken into account. These long tails might to illustrate the differences between general sequence mining approaches and the approaches with time constraints. Thus, we went for random sampling instead of shortening the time interval in case of too long computing times. The detailed results, including computation times and database volume and sampling, are presented in Table 4.

5.1 Performance evaluation

The first criterion in our evaluation is performance of sequence mining approaches. We were looking for a method that computes the task in reasonable time on commodity hardware (Intel Xeon E5520, 8 threads, 16 GB RAM). No prior estimates were set, but during the experiment, we encountered computing times ranging from one minute to one hour. In many cases we had to reduce the database size to be able to finish computation in under an hour.

As we found out, performance is heavily influenced by composition of input data, which stresses the importance of feature selection. The first general observation is that the algorithms performed better over all the databases without ports compared to databases with ports. Naturally, adding ports into the selected features creates more unique items to work with.

The most complex database containing both sources and targets was evaluated first. When we used the database without ports, the proposed methods behaved quite well. Sequential pattern mining and sequential rule mining algorithms were able to finish the computation in reasonable time. When the database with ports was used, the reduced database had to be used for all the sequential pattern mining algorithms except sequential pattern mining with time constraints. Sequential pattern mining methods with time constraints are working with different databases, where the alerts are aggregated by 5-minute intervals, thus having shorter sequences in general. Sequential rule mining algorithms were able to process the full database with ports.

The databases with sources allowed slightly better computation times for all the rule mining methods, but significantly worse for compressing pattern mining and pattern mining with time intervals. Most of the algorithms performed well when the database without ports was used, but we had to apply database reduction for compressing sequential pattern mining. Closed sequential pattern mining with time constraints was distinctively slower in comparison to others methods. When we switched to the database with ports, we had to reduce the database for the sequential pattern mining algorithms except for pattern mining with time constraints. Sequential rule mining algorithms had no issues working with the database with ports.

The most problematic was the database containing sequences aggregated only by target IP address and port. No algorithm was able to finish in reasonable time (less than 1 hour), even after database reduction. The reason was that the database contained low number of long sequences. The input of 16 million events was grouped into 6,331 sequences.

A general observation is that sequential rule mining algorithms performed significantly better than sequential pattern mining algorithms. We did not have to reduce a single database for sequential rule mining. The algorithms behaved well in all cases and did not make a difference between the databases with and without ports. The only exception was the database containing targets with ports, on which all methods failed even with database reduction. If a database without ports was used, then the sequential pattern mining algorithms were able to finish in reasonable time. Sequential pattern mining methods varied in computing time, but the common trend is that with higher data complexity, the computing time is longer, often including database reduction. The most problematic was compressing data mining with the highest demands even after database reduction to 1 % of original size.

5.2 Comparison of sequence mining algorithms

The second criterion in the comparison of the sequence mining approaches is the usability and comprehensibility of their outputs. Generally speaking, we want the methods to produce adequate number of results with sufficient support and confidence values and without much duplicities and ambiguous information.

Results with low support and confidence values are not convincing for an application in practice. For example, attack prediction could effectively use sequential rules with confidence of at least 50 % to select the most possible future move of an attacker. On the other hand, confidence approaching 100 % would indicate obvious results that could be obtained by much simpler methods than data mining.

Similarly, the number of results and their ambiguity plays a significant role. If a low k in top-K methods or high support and confidence thresholds are used, then there is a risk of having very low number of obvious results. On the other hand, receiving too

Table 4: Performance-based usability of sequential pattern and rule mining approaches

Method	Database: Sources and Targets, without ports	Database: Sources and Targets, with ports	Database: Sources only, without ports	Database: Sources only, with ports	Database: Targets only, without ports	Database: Targets only, with ports
Sequential pattern mining	16 min, 100 %	<1 min, 1 %	2 min, 100 %	<1 min, 5 %	✖	✖
Top-K sequential pattern mining	<1 min, 100 %	<1 min, 10 %	<1 min, 100 %	<1 min, 10 %	✖	✖
Closed sequential pattern mining	3 min, 100 %	2 min, 20 %	2 min, 100 %	2 min, 50 %	2 min, 5 %	✖
Sequential generator pattern mining	<1 min, 100 %	<1 min, 10 %	<1 min, 100 %	<1 min, 10 %	6 min, 60 %	✖
Maximal sequential pattern mining	<1 min, 100 %	<1 min, 10 %	<1 min, 100 %	<1 min, 10 %	4 min, 60 %	✖
Compressing sequential pattern mining	15 min, 100 %	3 min, 1 %	18 min, 10 %	4 min, 1 %	<1 min, 1 %	✖
Sequential pattern mining with time constraints	5 min, 100 %	6 min, 100 %	16 min, 100 %	11 min, 100 %	<1 min, 100 %	✖
Closed sequential pattern mining with time constraints	11 min, 100 %	11 min, 100 %	57 min, 100 %	34 min, 100 %	2 min, 100 %	✖
Sequential rule mining	1 min, 100 %	3 min, 100 %	<1 min, 100 %	<1 min, 100 %	<1 min, 100 %	✖
Sequential rule mining with window constraints	2 min, 100 %	4 min, 100 %	1 min, 100 %	1 min, 100 %	<1 min, 100 %	✖
Top-K sequential rule mining	1 min, 100 %	3 min, 100 %	<1 min, 100 %	<1 min, 100 %	<1 min, 100 %	✖

The results show computation time in minutes and size of the input data in percentage of the original database. Computation that did not complete or that generated excessive amount of patterns are marked with ✖.

much results would prevent their application in practice, as there is high risk of bogus or misleading patterns. In addition, shorter patterns may be discovered alone and as a part of a larger pattern, thus creating uncertainty for a potential user of the results. Often we had to manipulate threshold of support value to get a reasonable number of patterns. We managed to find support thresholds that lead to at least 5, but no more than 50, patterns.

For **sequential pattern mining** algorithms, we were able to choose a reasonable minimal support value to discover around 40 patterns. The problem was with the target databases without ports, which contain large numbers of patterns (up to thousands) with the same support value. Thus, setting the minimal support was a difficult task as we either got too few or too much patterns. Even the top-K method was not able to address this problem as the results contained all the patterns from a group with the same support. However, closed, generator, and maximal sequential pattern mining methods were able to address this problem and generated reasonable number of results. As mentioned before, the database with targets with ports was problematic due to performance issues. All methods failed even if the database was significantly reduced. Nevertheless, these results serve only as a baseline for comparison of advanced approaches.

The problem of redundancy in sequential patterns has been mentioned in the method description. Many redundancies were found in the results, prime example is a continuous scanning activity of a single attacker. In this case, network scanning is performed over and over, often up to dozens of times in a row. Sequence mining methods thus create sequences consisting of a variable number of network scanning alerts and nothing else. The longer the sequences are, the rarer they are. Thus the support values of long sequences are decreasing. We illustrate this observation in Figure 2.

The problem with such sequences is that they describe similar behavior of an attacker in a large number of patterns. Often only one

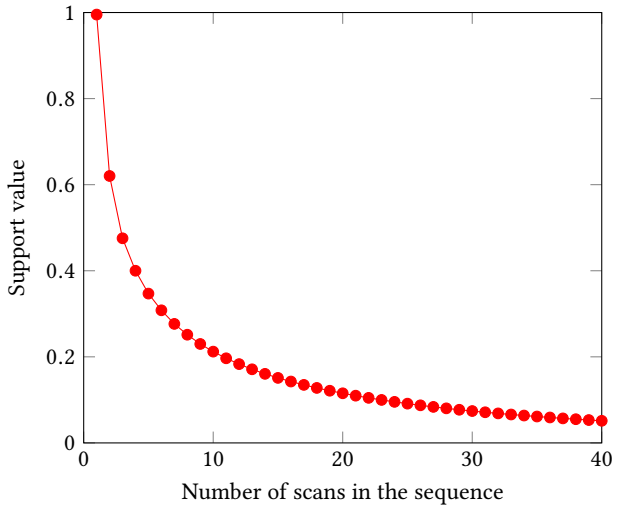

Figure 2: Decreasing *support* in closed sequential patterns containing network scanning alerts.

of the patterns is enough for practical use. Further, other interesting results may get lost in the numbers of such patterns when being processed by a human. Closed, maximal, and generator pattern mining algorithms should overcome this issue. Closed and generator pattern mining provided the same results over all the databases, even if we had to reduce the database due to performance issues. However, the results did not differ from the basic sequential pattern mining. On the other hand, maximal sequential pattern mining significantly reduced the number of patterns generated for same minimal support value, e.g., from 40 to 16 when mining from the database containing sources only. However the redundancy still exists in maximal patterns in the form of repeating the same event which is sometimes interleaved with different event. For example, there was a pattern containing six consecutive scans on port 23 which were interleaved with scanning on port 2323. For this case, we generated seven patterns that differ just in the position of scanning of port 2323. In conclusion, closed and generator sequential pattern mining methods did not prove to have an impact on the results, but maximal sequential pattern mining method filtered the results significantly, although not completely.

The particular values of minimal support differ among the databases. For databases with sources and targets, the minimal support of 0.01 was suitable. For databases with sources only, the adequate minimal support was different among databases with ports (0.1) and without ports (0.05). For database with targets only, we used minimal support values varying around 0.3.

Compressing sequential pattern mining algorithm had the biggest performance problems and in the half of the cases did not find any pattern. If some patterns were found, they were similar to patterns found by other methods. Still, the performance issues and lacking results make this method unsuitable.

Sequential pattern mining with time constraints adds more information about the timing in discovered patterns and allows users to define time constraints that reduce the number of generated patterns. As we are mainly interested in additional information about time, we do not set any constraints. We selected 5-minute interval as a unit of time due to typical settings of sensors contributing to our dataset, which usually detect security events in 5-minute time windows. We found around 40 patterns in every database, except databases with targets only. Typical time differences were either none (simultaneous events) or multiples of 12, i.e., units of hours. This is not surprising and confirms the soundness of the results. Otherwise, the results are very similar to results of methods that do not involve time intervals. No differences were found after comparing basic sequential pattern mining with time intervals with closed sequential pattern mining with time intervals. However, closed pattern mining was always slower than the basic pattern mining. In conclusion, knowing the time intervals between events in a pattern is an interesting feature to have, but minor performance issues were found, which may outweigh the benefits.

Sequential rule mining algorithms showed interesting dissimilarities in results. First of all, they discovered probably the most appealing results of all methods, especially when the ports were included in the database. In such case, various rules were found for attackers that perform scanning over a combination of ports. These results also had high support and confidence values. For example, top-10 sequence rule mining over the database with sources and ports had the following results:

```
Scan.1755  ==> Scan.1723 #SUP: 0.00025 #CONF: 0.69553
Scan.37777 ==> Scan.8000 #SUP: 0.00024 #CONF: 0.38748
Scan.1723  ==> Scan.1755 #SUP: 0.00023 #CONF: 0.35531
Scan.3392  ==> Scan.3391 #SUP: 0.00034 #CONF: 0.27006
Scan.3390  ==> Scan.3389 #SUP: 0.00024 #CONF: 0.10841
Scan.443   ==> Scan.80   #SUP: 0.00080 #CONF: 0.09309
Scan.80    ==> Scan.443  #SUP: 0.00066 #CONF: 0.02521
Scan.3389  ==> Scan.3390 #SUP: 0.00039 #CONF: 0.02226
Scan.2323  ==> Scan.23   #SUP: 0.00210 #CONF: 0.02031
Scan.23    ==> Scan.2323 #SUP: 0.00322 #CONF: 0.00461
```

Rules with confidence (#CONF) around 70 % were found in other databases as well. However, due to a disproportion between the number network scanning alerts and alerts of other events, there is also a gap in results between scanning and other types of events. There are high confidence rules for various combinations of port scanning, but there is very low confidence (<0.01) for rules beginning with scan and ending with other event type, e.g., Scan ==> Exploit. On the other hand, surprising rules with appreciable confidence values, e.g., Anomaly ==> Exploit with confidence around 0.13. In conclusion, sequential rule mining provides highly valuable results and has no performance issues.

We used window size of 5 in sequential rule mining with window size. In most cases, the results were exactly the same as in common sequential rule mining but, for the databases with sources, a smaller number of rules was generated. Unlike sequential rule mining, the longer rules were missing in the results.

5.3 Further results

Although we were primarily not interested in the actual results of data mining results, we list some of them in this section to show how accurate and predictable the results are. Relaying on time efficient, yet inaccurate, method with misleading results would cause more problems than good. Thus, we discuss how perspective are the results from the security practitioner's perspective.

A common result is a network scanning pattern in which a combination of ports is scanned by a single scanner. For example, a scan of port 80 is often accompanied by a scan of port 443. In this case, a scanner is looking for web servers running either HTTP or HTTPS. Similarly, port 23 and 2323 are scanned together due to system administrators, who often move a well-known service, e.g., Telnet, to an unusual port to avoid scanning. However, this practice is so common that scanners adapt to this by scanning both ports at once. We found such port combinations in the outputs of most sequence mining methods.

The most frequently found port combination in network scanning patterns included ports 23 and 2323. This combinations were found by most methods with high support and confidence. When the time intervals were taken into account, it turned out that the network scanning events were rather simultaneous then consequent. However, there are typically higher support values for patterns (23, 2323) then in an opposite direction, for example:

```
Scan.23   -> Scan.2323 #SUP: 0.00325
Scan.2323 -> Scan.23   #SUP: 0.00217
```

As for the other port combinations, ports 80 and 443 were found by most of the methods. Naturally, top-K methods found more

interesting combinations. For example, top-K sequential rule mining discovered combinations of ports 1755 and 1723 (confidence 69.55 %), 37777 and 8000 (confidence 38.75 %), and 3392 and 3391 (confidence 27 %).

Another interesting finding is the presence of patterns describing repeated scanning. Closed sequential pattern mining and sequential generator pattern mining excelled in this task and revealed that this is more common with not so often scanned ports. While ports like 80 and 23 were scanned mostly just once by an attacker, the sequential pattern mining revealed a group of ports that the attackers scanned up to six times, e.g., ports 666, 674, 922, and 930. These ports are used by various backdoors, which would explain the patterns. The scanner could be a botnet center trying to keep in touch with the bots. Even more interesting finding was that sequential rule mining revealed that the four ports are often scanned together from the same source. Although the support is low, confidence over 50 % is a promising result. An example output is as follows:

```
(Scan.922, Scan.674) ==> Scan.930
    #SUP: 0.02075 #CONF: 0.53690
(Scan.922, Scan.666) ==> Scan.930
    #SUP: 0.02003 #CONF: 0.53096
```

In a related work, Panjwani et al. [29] denote that 50 % of attacks is preceded by scanning. Many sequences (patterns and rules) were found to describe network scanning followed by exploit attempt of brute-force password attack. However, we could not confirm nor disprove the 50 % ratio due to significant overwhelm of scanning alerts in our dataset leading to low support and confidence values of corresponding patterns and rules. However, sequence mining raised further questions with regard to scanning and network attacks. For example, our results indicate that 62 % of attackers repeats scanning at least once. Thus, an interesting research question for future work would be the number of network scans before an intrusion attempt or other type of attack.

6 CONCLUSION AND FUTURE WORK

We surveyed and evaluated methods of sequential pattern and rule mining and applied them to the analysis of cyber security alerts. To the best of our knowledge, we are the first to evaluate suitability of various data mining methods for this task, focusing on sequence mining. Contrary to previous works, we also evaluated the advanced and optimized methods. This section concludes the paper with lessons learned from the experiment and recommendations for cyber security community on using sequence mining in practice. Subsequently, we propose the future work in the field. Source codes of the scripts used in the experiment are publicly available. Unfortunately, publishing the dataset is problematic as it is based on actual security alerts and, thus, requires proper anonymization and annotation as well as legal compliance.

6.1 Lessons learned

The lessons learned are divided into three groups according to the research questions stated in introduction.

1) Use cases and method selection are the first problems to approach when one wants to use data mining. Although association rule mining and frequent episode mining were used in most of the related work, we consider sequence mining as a better choice for processing cyber security alerts. We would recommend sequential pattern mining for alert correlation and sequential rule mining for attack prediction. Thus, to wrap up our recommendation:

- Sequential pattern mining is suitable for alert correlation use cases; the results are more comprehensive than with association rule mining and frequent episode mining.
- Sequential rule mining fits attack prediction use cases; confidence value can be directly used in a prediction model.

2) Performance, namely computation time, is a serious issue in security alert processing since these are often considered as big data, especially in collaborative environment. In our experiment, we found that the feature extraction influences the computation time and the results more than a selected method. The evaluated methods were able to efficiently mine patterns in sequences of alerts sharing the same source and target or source only. On the other hand, sequences of events with the same target were so long (thousands of itemsets) that they practically prevented pattern mining. If ports were included in the databases, we had to reduce the database size by random sampling to achieve reasonable computation times. Similarly, adding time criteria in the computation led to worse performance results. No effect on performance was observed when using specialized approaches like top-K or closed pattern mining. Sequential rule mining showed no performance issues with the only exception of database of targets with ports. Thus, our lessons learned are:

- Most of the sequential pattern mining methods provide similar results. The same applies for sequential rule mining.
- Feature selection makes the biggest difference; select them carefully.
- Beware of too long sequences; existing methods cannot work with them efficiently.
- Using optimized sequence mining methods (top-K, closed patterns, etc.) has positive impact on performance as well as on soundness of results.
- Sequence rule mining algorithms are much faster than sequential pattern mining algorithms. If efficiency is a requirement, we may consider using them even for alert correlation.

3) Soundness of results and used optimizations is hard to formalize and has to be approached with a good knowledge of the field of application. From the perspective of cyber security alert analysis, we have several observations and recommendations on how to improve soundness of results:

- Mining patterns in sequences of interaction between an attacker and a target is tempting, but following only the progress of an attacker is more rewarding. Processing only target interactions proved inefficient and results were not really useful.
- Including ports in the features is definitely useful, especially when mining patterns or rules of attacker's activity. However, it is more computationally demanding.
- Item intervals and other time-related criteria are useful and allowed interesting discoveries in attack timing.
- To avoid duplicities in discovered patterns, we recommend using some of the optimized pattern mining methods. In our experiment, maximal sequential pattern mining filtered the results better than other methods.

Nevertheless, it is important to continuously keep the desired use case and expected results in mind, especially in the cyber security field, where the data are often incomplete and prone to false positive alerts and other misleading information.

6.2 Future work

In our future work, we are going to inspect the incremental mining algorithms, a prospective approach to pattern mining that continuously updates the results. This approach has not yet been properly evaluated in the cyber security field, but seems promising with respect to peculiarities of cyber security alerts. It is generally acknowledged that cyber attacks are continuously evolving, but certain patterns in attackers' behavior may not change significantly, thus creating a use case for incremental mining. Finally, we are going to further inspect the serendipitously discovered patterns and examine the behavior of attackers.

ACKNOWLEDGMENTS

This research was supported by the Security Research Programme of the Czech Republic 2015 - 2020 (BV III / 1 VS) granted by the Ministry of the Interior of the Czech Republic under No. VI20162019029 The Sharing and analysis of security events in the Czech Republic.

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. 1995. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*. 3–14.

[2] Anna L. Buczak and Erhan Guven. 2016. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys Tutorials* 18, 2 (Secondquarter 2016), 1153–1176.

[3] Chris Clifton and Gary Gengo. 2000. Developing custom intrusion detection filters using data mining. In *MILCOM 2000 Proceedings. 21st Century Military Communications. Architectures and Technologies for Information Superiority (Cat. No.00CH37155)*, Vol. 1. 440–443 vol.1.

[4] Dipankar Dasgupta, Jose Rodriguez, and Sankalp Balachandran. 2006. Mining security events in a distributed agent society. In *Defense and Security Symposium*. International Society for Optics and Photonics, 62410A–62410A.

[5] Huwaida Tagelsir Elshoush and Izzeldin Mohamed Osman. 2011. Alert correlation in collaborative intelligent intrusion detection systems – A survey. *Applied Soft Computing* 11, 7 (2011), 4349 – 4365. Soft Computing for Information System Security.

[6] Hamid Farhadi, Maryam AmirHaeri, and Mohammad Khansari. 2011. Alert Correlation and Prediction Using Data Mining and HMM. *ISeCure* 3, 2, Article 3 (2011), 25 pages.

[7] Philippe Fournier-Viger, Antonio Gomariz, Manuel Campos, and Rincy Thomas. 2014. Fast vertical mining of sequential patterns using co-occurrence information. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 40–52.

[8] Philippe Fournier-Viger, Antonio Gomariz, Ted Gueniche, Espérance Mwamikazi, and Rincy Thomas. 2013. TKS: efficient mining of top-k sequential patterns. In *International Conference on Advanced Data Mining and Applications*. Springer, 109–120.

[9] Philippe Fournier-Viger, Antonio Gomariz, Ted Gueniche, Azadeh Soltani, Cheng-Wei Wu, and Vincent S. Tseng. 2014. SPMF: a Java Open-Source Pattern Mining Library. *Journal of Machine Learning Research (JMLR)* 15 (2014), 3389–3393. http://www.philippe-fournier-viger.com/spmf/

[10] Philippe Fournier-Viger, Antonio Gomariz, Michal Šebek, and Martin Hlosta. 2014. VGEN: fast vertical mining of sequential generator patterns. In *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 476–488.

[11] Philippe Fournier-Viger, Roger Nkambou, and Engelbert Nguifo. 2008. A knowledge discovery framework for learning task models from user interactions in intelligent tutoring systems. *MICAI 2008: Advances in Artificial Intelligence* (2008), 765–778.

[12] Philippe Fournier-Viger, Roger Nkambou, and Vincent Shin-Mu Tseng. 2011. RuleGrowth: mining sequential rules common to several sequences by pattern-growth. In *Proceedings of the 2011 ACM symposium on applied computing*. ACM, 956–961.

[13] Philippe Fournier-Viger and Vincent S. Tseng. 2011. Mining top-k sequential rules. In *International Conference on Advanced Data Mining and Applications*. Springer, 180–194.

[14] Philippe Fournier-Viger, Cheng-Wei Wu, Antonio Gomariz, and Vincent S Tseng. 2014. VMSP: Efficient vertical mining of maximal sequential patterns. In *Canadian Conference on Artificial Intelligence*. Springer, 83–94.

[15] Philippe Fournier-Viger, Cheng-Wei Wu, Vincent S. Tseng, and Roger Nkambou. 2012. Mining sequential rules common to several sequences with the window size constraint. In *Canadian Conference on Artificial Intelligence*. Springer, 299–304.

[16] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.* 11, 1 (Nov. 2009), 10–18.

[17] Yu Hirate and Hayato Yamana. 2006. Generalized Sequential Pattern Mining with Item Intervals. *Journal of Computers* 1, 3 (2006), 51–60.

[18] Ci-Bin Jiang, I-Hsien Liu, Yao-Nien Chung, and Jung-Shian Li. 2016. Novel intrusion prediction mechanism based on honeypot log similarity. *International Journal of Network Management* 26, 3 (2016), 156–175.

[19] Rajeshwar Katipally, Li Yang, and Anyi Liu. 2011. Attacker Behavior Analysis in Multi-stage Attack Detection System. In *Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research (CSIIRW '11)*. ACM, New York, NY, USA, Article 63, 4 pages.

[20] Yong-Ho Kim and Won Hyung Park. 2014. A study on cyber threat prediction based on intrusion detection event for APT attack detection. *Multimedia Tools and Applications* 71, 2 (2014), 685–698.

[21] Hoang Thanh Lam, Fabian Mörchen, Dmitriy Fradkin, and Toon Calders. 2014. Mining compressing sequential patterns. *Statistical Analysis and Data Mining* 7, 1 (2014), 34–52.

[22] Wenke Lee and Salvatore J. Stolfo. 1998. Data Mining Approaches for Intrusion Detection. In *7th USENIX Security Symposium*.

[23] Jie Lei and Zhitang Li. 2007. Using Network Attack Graph to Predict the Future Attacks. In *Communications and Networking in China, 2007. CHINACOM '07. Second International Conference on*. 403–407.

[24] Zhitang Li, Jie Lei, Li Wang, and Dong Li. 2007. A Data Mining Approach to Generating Network Attack Graph for Intrusion Prediction. In *Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery - Volume 04 (FSKD '07)*. IEEE Computer Society, Washington, DC, USA, 307–311.

[25] Zhitang Li, Aifang Zhang, Jie Lei, and Li Wang. 2007. Real-Time Correlation of Network Security Alerts. In *e-Business Engineering, 2007. ICEBE 2007. IEEE International Conference on*. 73–80.

[26] Nizar R. Mabroukeh and Christie I. Ezeife. 2010. A Taxonomy of Sequential Pattern Mining Algorithms. *ACM Comput. Surv.* 43, 1, Article 3 (Dec. 2010), 41 pages.

[27] Stefanos Manganaris, Marvin Christensen, Dan Zerkle, and Keith Hermiz. 2000. A data mining analysis of RTID alarms. *Computer Networks* 34, 4 (2000), 571 – 577. Recent Advances in Intrusion Detection Systems.

[28] Carl H. Mooney and John F. Roddick. 2013. Sequential Pattern Mining – Approaches and Algorithms. *ACM Comput. Surv.* 45, 2, Article 19 (March 2013), 39 pages.

[29] Susmit Panjwani, Stephanie Tan, Keith M. Jarrin, and Michel Cukier. 2005. An experimental evaluation to determine if port scans are precursors to an attack. In *2005 International Conference on Dependable Systems and Networks (DSN'05)*. 602–611.

[30] Moon Sun Shin and Kyeong Ja Jeong. 2006. *An Alert Data Mining Framework for Network-Based Intrusion Detection System*. Springer Berlin Heidelberg, Berlin, Heidelberg, 38–53.

[31] Moon Sun Shin and Keun Ho Ryu. 2003. Data mining methods for alert correlation analysis. *International Journal of Computer and Information Science* 4, 4 (2003).

[32] James J. Treinen and Ramakrishna Thurimella. 2006. A framework for the application of association rule mining in large intrusion detection infrastructures. In *International Workshop on Recent Advances in Intrusion Detection*. Springer, 1–18.

[33] Fredrik Valeur, Giovanni Vigna, Christopher Kruegel, and Richard A. Kemmerer. 2004. Comprehensive approach to intrusion detection alert correlation. *IEEE Transactions on Dependable and Secure Computing* 1, 3 (July 2004), 146–169.

[34] Emmanouil Vasilomanolakis, Shankar Karuppayah, Max Mühlhäuser, and Mathias Fischer. 2015. Taxonomy and Survey of Collaborative Intrusion Detection. *ACM Comput. Surv.* 47, 4, Article 55 (May 2015), 33 pages.