Computer Communications 35 (2012) 1368–1379

Contents lists available at SciVerse ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

# Multi-layer episode filtering for the multi-step attack detection

Mahbobeh Soleimani [a,*], Ali A. Ghorbani [a]

[a] Information Security Centre of Excellence, Faculty of Computer Science, University of New Brunswick, Fredericton, New Brunswick, Canada

ARTICLE INFO

*Article history:*
Received 2 November 2009
Received in revised form 2 April 2012
Accepted 2 April 2012
Available online 18 April 2012

*Keywords:*
Alert correlation
Multi-step attack
Intrusion detection system

ABSTRACT

The discovery of sophisticated attack sequences demands the development of significantly better alert correlation algorithms. Most of the proposed approaches in the area of multi-step attack detection have limited capabilities because they rely on various forms of predefined knowledge of attacks or attack transition patterns using attack modeling language or pre-and post-conditions of individual attacks. Therefore, those approaches cannot recognize a correlation when an attack is new or the relationship between attacks is new. In this research, we take a different view and consider alert correlation as the problem of inferring an intruder's actions as alert patterns that are constructed progressively. The work is based on a multi-layer episode mining and filtering algorithm. A decision-tree-based method is used for learning specifications of each attack pattern and detecting them in alert streams. We also used a Correlation Weight Matrix (CWM) for encoding correlation strength between attack types in the attack scenarios. One of the distinguishing features of our proposed technique is detecting novel multi-step attack scenarios, using a rule prediction method. The results have shown that our approach can effectively discover known and unknown attack strategies with high accuracy. We achieved more than 90% reduction in the number of discovered patterns while more than 95% of final patterns were actual patterns. Furthermore, our rule prediction capability showed a precise forecasting ability in guessing future alerts.



## 1. Introduction

Intrusion Detection Systems (IDSs) trigger too many alerts that usually contain false alerts that show either normal traffic or failed intrusion attempts. To make things worse, only 1% of the enormous amount of alerts generated by most IDSs corresponds to unique attacks [21].

Alert management and correlation improve the accuracy of IDS significantly. Decreasing false positives and improving the knowledge about attacks provides a more global view of what is happening in a network. Correlating alerts only based on the similarities between their attributes cannot discover the real reasons why alerts are correlated. On the other hand, considering previously specified scenarios to perform the correlation is limited to discovering only known scenarios and a great hand work is needed to specify each of them. It is also important to correlate alerts in real time in order to have a more effective response time.

However, there are some approaches that try to find the correlation between alerts based on the frequency of patterns in the alert streams. Despite the fact that some types of multi-step attack scenarios have a frequent behaviour, leaning on frequent patterns can result in missing those scenarios that do not occur frequently but are critical. Moreover, a large number of discovered frequent patterns still need to be inspected by administrators manually in order to find the meaningful and critical ones.

In this work, we used an unsupervised pattern generation method that can create all of the possible combinations of alerts in an alert stream. In each step of the pattern generation method, we applied a machine learning algorithm to filter those patterns that follow specifications of a multi-step attack scenario. The pattern generation algorithm applies flexible sliding windows for detecting steps of attacks. Hence, steps of attacks can occur consequently with a pause between each pair of steps. Behaviour of attackers in performing steps of attacks is learned gradually to consider a reasonable duration to complete each step.

Besides, an attack tree data structure is defined for storing the multi-step attacks that are discovered. In this tree, information about the frequency of each step is retained. For discerning critical attack patterns and false positives, we suggested a new learning-based episode reduction method. In the process of learning, specification of episodes rather than alerts is used for learning multi-step attack scenarios.

A Correlation Weight Matrix (CWM) is also used for encoding correlation strength between attack types in the attack scenarios. The attack tree data structure keeps information about the frequency of each step. As attack tree gets updated, rules that predict forthcoming attacks get generated. Each rule's degree of confidence is calculated based on a normalized distance deviation function

* Corresponding author. Tel.: +1 604 374 8423.
*E-mail address:* m.soleimani@unb.ca (M. Soleimani).

0140-3664/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comcom.2012.04.001

that measures the degree of similarity between a discovered attack pattern and a known multi-step attack. Rules are used in the attack prevention phase to guess the next step of an attack.

During each step of pattern filtering, a rule-based classification method decides about each discarded pattern. If steps of a discarded pattern have strong correlation, discovery of a new attack scenario can be inferred. If the correlation between steps of the pattern is frail, the pattern is probably a false one. Novel discovered attack scenarios will be used to update the CWM and learning algorithm.

The results of the study have shown that our approach can effectively discover known and unknown attack strategies with high accuracy. We achieved more than 90% reduction in the number of discovered patterns while more than 95% of final patterns were actual patterns. Furthermore, our rule prediction capability showed a precise forecasting ability in guessing future alerts.

The rest of the paper is organized as follows. In Section 2, some of the related works in the area of alert correlation are reviewed. Section 3 provides the details of the proposed framework for mining multi-step patterns in alert streams. Section 4 reports experiments that are conducted for evaluating the proposed technique. Finally, the conclusions and some suggestions for future work are given in Section 5.

## 2. Related works

Because IDSs usually trigger high volume of alerts, causality analysis is essential in providing a higher level view of the actual attacks. The causal relationships between alerts might represent how they are related to each other in an attack scenario. Viinikka et al. [17] aggregated individual alerts to alert flows, and then processed the flows instead of individual alerts to find the correlation between alerts for coping with the large quantity of alerts. Some of the previous works in this category have used formal models for specifying attack scenarios, like LAMBDA [1], STATL [2], ADeLe [3], [5,9,10]. However, some correlation research works are based on pre-defined attack scenario rules [4,7,8,23].

Valuable research work has been done in the area of attack strategy detection that mostly uses powerful machine learning methods to find multi-step attack scenarios. For example, in [6], two machine learning techniques (Multilayer Perception and Support Vector Machine) have been used to estimate the *correlation probability* between alerts. Moreover, Goldman et al. [7] built a correlation system based on Bayesian reasoning, and [20,16] used an unsupervised learning method for network alert correlation.

The idea of Correlation Weight Matrix (CWM) which is developed in this paper is similar to ACM which is defined in [6]. However, there is a difference between these two matrixes. The CWM gets dynamically updated during both training and test phases. But, in ACM correlation weights are incrementally updated during training process.

Developing a real-time alert correlation method is one of the goals of this research. There are some other research work that used different methods for real-time alert correlation. For example, Giannella et al.'s idea about mining time-sensitive data streams is used in [11] for real-time alert correlation and attack strategy detection. Core of the proposed framework is a new algorithm (FSP Growth) for mining frequent patterns of alerts considering their structures. Moreover, Wang et al. [19] developed an algorithm to correlate isolated alerts into the attack scenarios efficiently in terms of memory usage. Besides, in [12], a real-time alert correlation system is proposed to detect an ongoing attack and predict the upcoming next step of a multi-step attack in a timely manner. In [15] an attack graph analysis method which aggregates attack paths according to underlying network regularities is used to prioritize IDS alerts and reduce the complexity of analysis.

In this paper, we took advantage of episode generation method and Decision Tree (DT) algorithm. The frequent episode and episode rule mining problem was first introduced by Mannila [13]. Given a sequence of events, episode mining aims to find all of the episodes with occurrence frequencies satisfying the user-specified minimum support. However, almost all of them are based on the Apriori property which states that any super-pattern of a non-frequent pattern cannot be frequent. Most of apriori-like algorithms generate, potentially, huge sets of candidate episodes, and require as many full sequence scans as the longest episode.

Telecommunication network Alarm Sequence Analyzer (TASA) [14] applied the frequent episode mining algorithm in developing an interactive knowledge discovery system for the events triggered by network devices. TASA discovers frequent episode rules from alarm streams in an off-line mode. Actually, the administrator is able to see its favourite episodes by specifying some thresholds and templates. Therefore, the only criteria that is applied for controlling number of generated episodes and filtering the most important ones is support and confidence thresholds that both are based on the frequency of occurrence. However, there are some critical episodes that are not frequent and cannot be detected by this approach. We will address this problem in our work using a multi-layer episode verification approach.

In [12,14,18], patterns or rules are extracted from a large number of alerts. In these series of studies, they paid less attention to the number of patterns and verifying their criticality. While, an applicable and real-time alert correlation system needs an efficient algorithm for discovering patterns and verifying their criticality.

## 3. Framework

The system is composed of two modes. It first works in off-line mode and then it starts working in on-line mode. In the off-line mode, as is shown in Fig. 1, system receives a stream of training alerts that are labelled. First, labelled alerts get aggregated and then episode generation component uses them to generate different lengths of episodes. Next, learning component learns real multi-step attacks and false combination of alerts by generating decision trees. In the on-line mode, as is illustrated in Fig. 2, system continuously receives the on-line stream of alerts. First, alert aggregation component aggregates the alerts and sends them to the episode generation component. Second, episode generation component generates different lengths of episodes. Up to here, off-line and on-line modes work exactly alike. Next, episode generation component feeds the episodes into the detection component to filter them based on their degree of criticality, which is learned in off-line mode by learning component. Filtered critical episodes go to the prediction component to predict future steps of attack. Moreover, filtered uncritical episodes go to the new strategy detec-

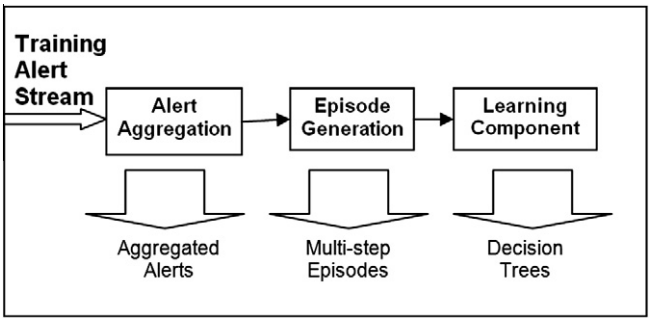


**Fig. 1.** Framework in the off-line mode.