Suarez-Tangil, G., Palomar, E., De Fuentes, J. M., Blasco, J. & Ribagorda, A. (2009). Automatic rule generation based on genetic programming for event correlation. Advances in Intelligent and Soft Computing, 63, pp. 127-134. doi: 10.1007/978-3-642-04091-7_16

Automatic Rule Generation based on Genetic Programming for Event Correlation

G. Suarez-Tangil, E. Palomar, J. M. de Fuentes, J. Blasco, A. Ribagorda

Department of Computer Science – University Carlos III of Madrid
Avda. Universidad 30, 28911 Leganes, Madrid
gtangil@pa.uc3m.es, {epalomar, jfuentes, jbalis, arturo}@inf.uc3m.es

Abstract. The widespread adoption of autonomous intrusion detection technology is overwhelming current frameworks for network security management. Modern intrusion detection systems (IDSs) and intelligent agents are the most mentioned in literature and news, although other risks such as broad attacks (e.g. very widely spread in a distributed fashion like botnets), and their consequences on incident response management cannot be overlooked. Event correlation becomes then essential. Basically, security event correlation pulls together detection, prevention and reaction tasks by means of consolidating huge amounts of event data. Providing adaptation to unknown distributed attacks is a major requirement as well as their automatic identification. This positioning paper poses an optimization challenge in the design of such correlation engine and a number of directions for research. We present a novel approach for automatic generation of security event correlation rules based on Genetic Programming which has been already used at sensor level.

Key Words: Event Correlation, Rule Generation, Genetic Programming, Network Security Management

1 Introduction

Nowadays, network security management is a critical task which involves different security data sources, e.g. IDSs, firewalls, server logs, to name a few. One the one hand, these sources (known as sensors) produces high amounts of heterogeneous information, generally difficult to understand. Thus, cooperation among these sensors becomes essential for security information management. On the other hand, modern attacks pose a challenge to the network infrastructure as these attacks may be not noticed when inspecting each one separately. Event correlation was therefore conceived as a palliative for both problems.

Security information Management (SIM) systems help to gather, organize and correlate security network information, i.e. reducing the amount of time spent by security administrators. OSSIM [1] is an open source SIM implementation which centralizes the detection and monitoring of the security events within an organization. Nevertheless, correlation techniques included in OSSIM are not able to efficiently detect broad attacks such as Distributed Denial of Service (DDoS).

In this context, we are therefore facing the classical problem of providing optimization to a data mining analysis. Many disciplines exist within data mining; those are particular advantageous when connected to artificial intelligence algorithms. Indeed, Genetic Programming (GP) introduces several interesting properties to deal with building complex event correlation rules: Rules can be represented as computer programs with length variability.

Our contribution: In this positioning paper, we elaborate on applying GP to optimize log correlation methods. Specifically, we introduce the main aspects of our current work, which is aimed at building optimized OSSIM rules by means of the application of evolutionary computation.

The rest of the paper is organized as follows. In Section 2 we outline emergent problems related to current IDSs and event correlation techniques. Section 3 describes the foundations of our case study, i.e. automatic generation of correlation rules for the OSSIM. Finally, in Section 4 we establish the main conclusions as well as the immediate future work. As this is a work in progress, each section intents to describe the position of this line of research in each one of the corresponding areas.

2 Related Work

Several disciplines are related to security event correlation. Here we describe the most representatives.

2.1 Intrusion Detection

As stated in [2], IDSs such as SNORT are not enough to detect complex attacks over a network. Depending on where the IDS is placed they will detect some things and skip some others as well as valuable information logged in other devices such as firewalls, routers, web servers, operating system logs... is also missed.

2.2 Event Correlation Techniques

To solve the problem originated by the huge amount of events it is necessary to look into all the implicated devices over the enterprise network. Two possible solutions in this way have been proposed: Data Aggregation (centralization of the information) and Event Correlation (combination of multiple events into one relevant event).

Among its advantages, data aggregation provides a centralized management as well as a unification of the security events into a single format; whereas the high volume of the gathered information can be a disadvantage. On the other hand, we can distinguish the following event correlation mechanisms such as those based on statistics, bayesian inference, alert fusion and data correlation (Micro and Macro correlations).

In fact, the latter shows promise as a security event correlation since other models mentioned before are not especially appropriated [2].

Data correlation has to deal with performance issues in a similar way to data mining classic problems. Instead, the application of artificial intelligence techniques involves amount of properties such as flexibility, adaptability, pattern recognition, efficiency [3], to name a few.

An interesting approach to event correlation at the application layer is proposed in [4] in which involving Network, Kernel and Application Layer an efficient correlation is performed. Additionally, authors in [5] use probabilistic correlation techniques, i.e. Bayesian networks, to increase the sensitivity, reduce false alarms and improve log report quality. The correlation is carried out in both sensor and alert models.

2.3 OSSIM Correlation

OSSIM has the ability to correlate attacks using both sequence of events and heuristic algorithms. In the former, the correlation engine [6] generates alerts when a detected event matches the pattern described by a rule. These rules are called “detector rules”. Once a detector rule is triggered, OSSIM can start monitoring the sensors providing indicators to the correlation engine: this is done by using “monitoring rules”.

Contrarily, correlation engine based on heuristic algorithms generates an indicator or snapshot of the global security state of the network as result of event agregation.

Nevertheless, attack signatures must be defined in advance in both methods. Moreover, classic methods lack of efficiency on pattern recognition. For instance, open problems focus on the application of artificial intelligence to provide Data Mining optimization. Evolutionary algorithms are especially suitable for those problems in which a cost–effort trade-off exists. Event correlation is a good example –achieving the optimal solution is feasible, but it is not affordable at all–. Time and computational savings lead us to employ AI-based techniques.

Finally, more robust event correlation engine is provided by BITACORA [7].

2.4 Evolutionary Computation

Evolutionary Computation (EC) [8] is a branch of the Artificial Intelligence paradigm which is based on the species evolution theory. Each iteration of an evolutionary algorithm execution (also known as generation) represents a set of solutions. Afterwards, this set is evaluated, and the best individuals will be selected. The basic genetic operators are used over these individuals to produce next generations. This way, an evolutionary algorithm can be described as a supervised search technique.

Genetic programming [9] is a EC-based methodology, where individuals are represented as computer programs. This fact leads to interesting advantages, i.e. individuals can be defined in a flexible representation. The use of a variant of standard GP which is called Gene Expression Programming (GEP) has been proposed in [10]. In this work, authors state that size of populations and the number of generations are critical, sometimes too high, thus these incur a high cost in term of the search space and time. GEP, Linear GP and Multi-Expression Programming (MEP) are analyzed and used to extract IDS rules in [11]. Nevertheless, to the best of our knowledge, GP has not been applied to event correlation. Indeed, GP is especially suitable on this domain due to the intrinsic characteristics of correlation rules i.e. they have variable lengths and may be expressed by computer routines.

3 Applying Genetic Programming to Event Correlation

We now describe how to apply GP [12] to event correlation. In particular, we focus on the automatically generation of correlation rules, once security logs have been centralized in the system intrusion management (SIM) provided by OSSIM framework.

3.1 Experimental Environment

Briefly, we now describe the emulation environment. First, we collect SIM alerts that OSSIM Server generates as the result of distributed denial of service (DDoS) attacks. These attacks are launched from a botnet located at the same environment. The collected DDoS traces will be finally used as a training set to evaluate the individuals in each generation. Note that our simulations do not learn in real time. Next sections overview all the stages of our experiments.

The development environment is based on ECJ framework, i.e. a Java-based Evolutionary Computation toolkit [13]. ECJ assists in the deployment of genetic program setting the challenge to the definition of the individual, the fitness function and other parameters such as the number of individuals.

Specifically, we use Standard GP where an individual is represented as a "forest of trees", and the genome (i.e. individual's genetic code) is represented as a tree. Tree's nodes are operator functions whereas tree's terminals are the genome's selected attributes. For instance, from our simulations, results especially rely on three building blocks: (i) the representation of the solution, (ii) how the fitness function represents the objective of the problem and, (iii) how the crossover and mutation operators are defined. We further elaborate on this concerns in the following sections.

3.2 Preliminary Format Definition

As the result of the evaluation of the alerts stored on the OSSIM, we have selected the following attributes to be introduced on our Genetic Program. These are the following: Destination IP, Destination Port, Source IP, Source Port, Time stamp and Type of event.

Our goal is to design our Genetic Program as much general as possible. We would like to leave the door open for the future so we can extend our rule generator to other Security Information Manager such as BITACORA.

For this purpose we have decided to give a normalized format to the events. OSSIM output and our GP input are expressed in IDMEF (Detection Message Exchange Format) format [14].

Also, it is necessary to give a normalized type for the event type. The problem is that each IDS follows its own taxonomy. We have solved this problem by parsing the event type string and taking the main keywords that can be found on the event type. Examples on such keywords are ”FTP”,”NMAP”, etc.

3.3 Representation of the Individual

At first we represented the individuals with the parameters described in 3.2. These parameters come straight forward from the attributes used by some sensor. The use of this attributes provides us the ability of extending the correlations to other systems. A problem to be solved is that we need more attributes to compound a rule; some of the attributes are specific from OSSIM such as the reliability, the occurrence and many others.

As non terminal nodes we have selected the Boolean functions AND and OR. We present two open alternatives for terminal node representation. Terminal nodes are defined as ERC (Ephemeral Random Constants) nodes to guarantee the coherence of the randomly selected and crossed values.

First Approach: Rule as an ERC Let’s specify first the resultant work we are looking for. We want our genetic program to automatically generate valid directives for the OSSIM correlator engine.

One directive is compound by many rules in the relation: Directive ⟶ (Rule+).

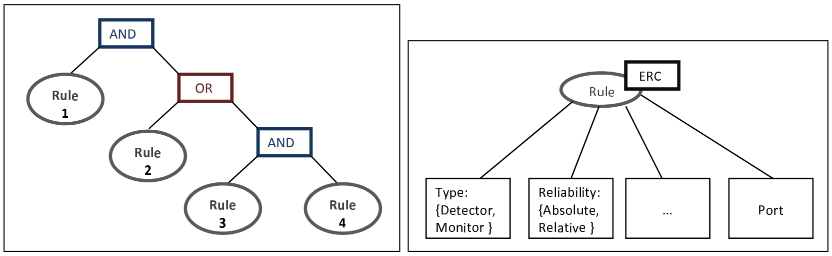
AND
Rule 1
OR
Rule 2
AND
Rule 3
Rule 4
ERC
Rule
Type: {Detector, Monitor }
Reliability: {Absolute, Relative }
...
Port

Fig. 1. (a) Genetic tree directive (b) A Rule as a ERC terminal element

As all the rules can have the same arguments and we want to generate rules that operate with other rules, (Figure 1 a) the best encoding of the attributes that we can do is to wrap them into an ERC data type. This ERC will define a Rule (Figure 1 b)

A major problem for this representation is that the all the attributes from the rule have to be randomly generated and is much more complicated (in terms of computation) to generate a ’good’ rule.

The search space is bigger since we have to find sets of attributes (a rule), with all its possible combinations, that have elements in common with other attributes of another rule.

Second Approach: Attributes as an ERC This second solution tries to solve the problem encountered on the first approach. In this case the terminal elements are the attributes.

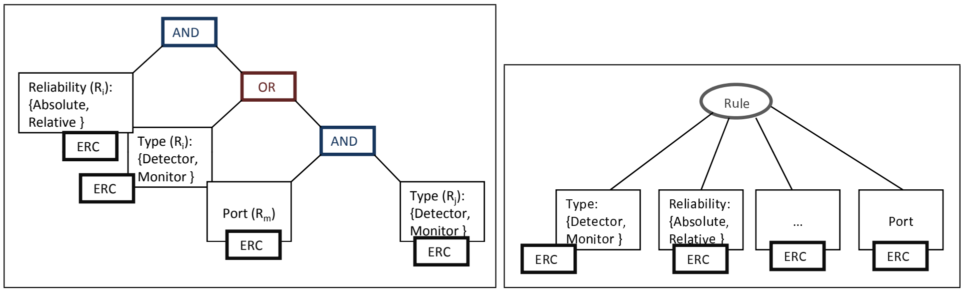

Fig. 2. (a) Attributes as an ERC terminal elements. (b)Genetic tree directive

A major problem for this representation is that we are treating the attributes as terminal elements so a possible solution is a tree where attributes of a rule N can operate with attributes of the rule M, where N can be equal to M; this is not always a valid OSSIM directive.

3.4 Genetic Operators

The following genetic operators are performed: *reproduction, mutation and crossover*. The *reproduction* operator is responsible of the clonation of the individuals: These cloned individuals are randomly modified in the *mutation* operator. During *mutation* and *crossover* operators we have use default ECJ framework values.

ECJ provides a simple way to implement a specific crossover and mutation operators. Depending on the accuracy of the results, this might be implemented.

3.5 Training and Fitness Function

Each round of the algorithm will be evaluated with a training set logged in our training environment which contains events from specific attacks.

Additionally, the fitness function will be based on the risk assessment as defined by OSSIM.

Therefore, detector rules follow a logical evaluation of the attributes. Each time an event is detected by a rule, its parameter *Reliability* is incremented. This change also affects the Risk calculation. In this situation, considering the Risk parameter for building the fitness function seems to be suitable. Since the monitoring rules (Eq. 1) follow a heuristic algorithm which is defined as the risk that the monitorized asset might be suffering for a specific thread (reliability of the possible attack) and the priority of the directive.

$$Risk = \frac{(Asset \cdot Priority \cdot Reliability)}{25} \tag{1}$$

4 Conclusions and Research Directions

In this positioning paper, we have studied the suitability of evolutionary computation techniques to improve the efficiency of current security event correlation mechanisms. In fact, genetic programming lets us manage the problem in a very natural way: correlation rules can be built based on the evolution of individuals as a mechanism to discover event "proximity". We have outlined our findings in building OSSIM correlation directives by means of GP. Nevertheless, our approach faces up some research problems, such as an appropriate population representation and the generation of a suitable training set.

In the short term, we further elaborate on the task of correlation and pattern matching which is still one for a strong analysis concern. Finally, as future work, we are evaluating the viability of the presented representation of the problem, which seems to be a completely original approach to tackle the security event correlation problem, by means of system emulation.

Our hope is that this paper will, directly or indirectly, inspire new directions in efficiently automatizing security event correlation.

5 Acknowledgements

This work is supported by CDTI, Ministerio de Industria, Turismo y Comercio of Spain in collaboration with Telefonica I+D, project SEGUR@ with reference CENIT-2007 2004

References

1. OSSIM: Open source security information management (2009) http://www.ossim.net/whatis.php.
2. Center for Education and Research in information Assurance and Security of Purde University: CERIAS Security Seminar Archive - Intrusion Detection Event Correlation: Approaches, Benefits and Pitfalls, Center for Education and Research in information Assurance and Security of Purde University (March 2007)
3. Tjhai, G.: Intrusion detection system: Facts, challenges and futures. http://www.bcssouthwest.org.uk/presentations/GinaTjhai2007.pdf (March 2007)
4. Rice, G., Daniels, T.: A hierarchical approach for detecting system intrusions through event correlation. In: IASTED International Conference on Communication, Network, and Information Security, Phoenix, USA (November 2005)

5. Valdes, A., Skinner, K.: Probabilistic alert correlation. In: Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection. (2001) 54–68
6. Karg, D.: OSSIM Correlation engine explained. (2004) `http://www.ossim.net/docs/correlation_engine_explained_rpc_dcom_example.pdf`.
7. Bitacora: System of centralization, management and exploitation of a company's events `http://www.s21sec.com/productos.aspx?sec=34`.
8. Fogel, L.J., Owens, A.J., Walsh, M.J.: Artificial Intelligence through Simulated Evolution. John Wiley, New York, USA (1966)
9. Koza, J., Poli, R.: Introductory Tutorials in Optimization and Decision Support Techniques. Springer, UK (2005)
10. Tang, W., Cao, Y., Yang, X., So, W.: Study on adaptive intrusion detection engine based on gene expression programming rules. In: CSSE International Conference on Computer Science and Software Engineering, Wuhan, China (December 2008)
11. Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S.: A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. In: Applications of Data Mining in Computer Security, Kluwer (2002)
12. Mukkamala, Srinivas, Sung, Andrew, H., Abraham, Ajith: Modeling intrusion detection systems using linear genetic programming approach. In: IEA/AIE'2004: Proceedings of the 17th international conference on Innovations in applied artificial intelligence, Ottawa, Canada, Springer Springer Verlag Inc (2004) 633–642
13. Luke, S., Panait, L., Balan, G., Paus, S., Skolicki, Z., Popovici, E., Sullivan, K., Harrison, J., Bassett, J., Hubley, R., Chircop, A.: A java-based evolutionary computation research system `http://cs.gmu.edu/~eclab/projects/ecj/`.
14. Debar, H., Curry, D., Feinstein, B.: Ietf rfc 4765 - the intrusion detection message exchange format (March 2007) `www.ietf.org/rfc/rfc4765.txt`.