Identification of correlated network intrusion alerts

Mirco Marchetti, Michele Colajanni, Fabio Manganiello
Department of Information Engineering
University of Modena and Reggio Emilia
Modena, Italy
{mirco.marchetti, michele.colajanni, fabio.manganiello}@unimore.it

Abstract—**Attacks to information systems are becoming more sophisticated and traditional algorithms supporting Network Intrusion Detection Systems may be ineffective or cause too many false alarms. This paper describes a new algorithm for the correlation of alerts generated by Network Intrusion Detection Systems. It is specifically oriented to face multistep attacks where multiple intrusion activities belonging to the same attack scenario are performed within a small time window. This algorithm takes as its input the security alerts generated by a NIDS and, through a pseudo-bayesian alert correlation, is able to identify those that are likely to belong to the same multistep attack scenario. The proposed approach is completely unsupervised and applicable to security alerts generated by any kind of NIDS.**

I. Introduction

Network Intrusion Detection Systems (NIDSs) are widely adopted network appliances that analyze network traffic looking for illicit activities. As soon as an intrusion is detected, a NIDS reacts by issuing a security alert that contains information about the intrusion, such as the endpoints of the attack, the type of alert, and a timestamp that identifies when the intrusion has been detected.

While alert generation is performed automatically, NIDSs provide only minimal aids to the network security analyst, because he has to manually analyze all the security alerts in order to discard false positives and identify few security threats that are relevant for his information system. Human expertise is especially required to correlate and link together multiple low-level security alerts belonging to the same complex multistep attack scenario, that is characterized by multiple correlated intrusion activities [1].

This paper proposes a novel algorithm for the identification of correlated security alerts, that are likely to belong to the same multistep attack scenario. Our approach, called *pseudo-bayesian alert correlation*, is inspired to Bayes' law about conditional probability. Since the proposed method is unsupervised, and does not rely on any a-priori knowledge of possible attacks and of the monitored environment, the value of the correlation index depends on previously analyzed security alerts. The underlying assumption is that correlated intrusion activities exhibit strong temporal locality [2], hence security alerts corresponding to intrusion activities that belong to the same multistep attack scenario are generated within a small time window. In particular, if previous history shows that alerts of two given types are often generated within a short time frame, then it is likely that instances of these two alert types are correlated. The correlation index is then weighted according to the time distance between the alerts (correlation grows with temporal locality) and to the number of security alerts analyzed so far. Hence the reliability of the correlation index grows as the number of analyzed alerts increases.

An important attribute is that pseudo-bayesian alert correlation does not require any knowledge about the security alerts and of the environment analyzed by the NIDS, it can be applied to security alerts produced by any NIDS and to heterogeneous network environments. Experimental evaluations are carried out through the widely deployed open source Snort NIDS [3] and applied to the Capture The Flag 2010 dataset [4]. They demonstrate that the proposed approach is able to identify correlated alerts, thus helping a security supervisor in the analysis of intrusion alerts produced by NIDSes.

Related works in the field of IDS alerts aggregation and correlation are described in Section II. An overview of the pseudo-bayesian alert correlation approach is given in Section III, while the details about the algorithms used to compute and to weight the correlation indexes among intrusion alert instances are described in Section IV. Section V describes the implementation of the proposed algorithm as a plugin for the open source Snort NIDS. This reference implementation has been used to evaluate the effectiveness of our proposal through several experiments, presented in the same section. Finally, Section VI concludes the paper and outlines future works.

II. Related work

The application of unsupervised machine-learning techniques, such as Bayesian networks, neural networks and clustering algorithms, for intrusion detection has been widely explored in the security literature. Several papers propose naïve Bayes classification and Bayesian networks as the main detection engine for the implementation of anomaly-based intrusion detection systems.

In particular, naïve Bayes classification has been proposed in [5] and [6] to implement anomaly based network intrusion detection systems, while in [7] the same approach has been used for the detection of malicious services (proxylets) deployed dynamically within active networks. More complex detection algorithms, leveraging complete Bayes networks for anomaly-based network intrusion detection have also been proposed in [8] and [9]. Another relevant work in the same field is represented by [10], in which authors propose the use of

Bayes network to fuse the results obtained by several different detection models, thus improving the overall detection rate of a multi-model host intrusion detection system.

Unlike previous literature oriented to anomaly detection, in this paper we design an algorithm that is inspired to Bayes' theorem of conditional probability for the post-processing of security alerts that have already been generated by a signature-based NIDS. Hence, our proposal relates to other papers focusing on techniques for NIDS alert correlation [11], rather than to anomaly detection. According to the comprehensive framework for NIDS alert correlation proposed in [1], the solution proposed in this paper can be classified as a *multistep correlation* component. In this context, a closely related work is [2].

In [2], authors propose an alert correlation algorithm based on naïve Bayesian networks. In particular, their approach tries to predict the target of a multistep attack based on previous alert history. Their algorithm starts by identifying a set of possible intrusion objectives, and then it analyzes the historical data to build a naïve bayes network for each of the intrusion objectives.

On the other hand, the work proposed in this paper does not aim to predict the future objectives of a possible multistep attack. Instead, the purpose of the proposed pseudo-bayesian alert correlation algorithm is to identify and highlight groups of intrusion alerts that, based on their detection time and on the past alert history, are more likely to be correlated and to belong to the same high-level multistep attack scenario. The final output of our aggregation algorithm is a graph in which nodes denotes alert instances, and correlations among them are represented by vertexes. This output allows a security analyst to easily identify correlated alerts, thus reducing the time spent in manual analysis of the intrusion alerts produced by the NIDS.

III. Pseudo-bayesian alert correlation algorithm

The pseudo-bayesian algorithm for alert correlation proposed in this paper is designed to correlate alerts generated by any NIDS. It takes two types of input:

- the alert log containing historical intrusion alerts generated by the NIDS;
- intrusion alerts generated by the NIDS.

The final output is a graph in which the nodes represent intrusion alerts, and the vertexes connect correlated intrusion alerts.

A. Architecture overview

A high-level overview of the architecture of the proposed approach for alert correlation is shown in Figure 1. It consists of two main processing steps.

The first processing step, called *pseudo-bayesian probability* in Figure 1, is inspired to Bayes' law of conditional probability. It determines the likelihood of any two intrusion alerts of being correlated, based on their type, on their timestamps, and on the intrusion alerts previously generated by the same NIDS. In particular, the previous alert history is analyzed

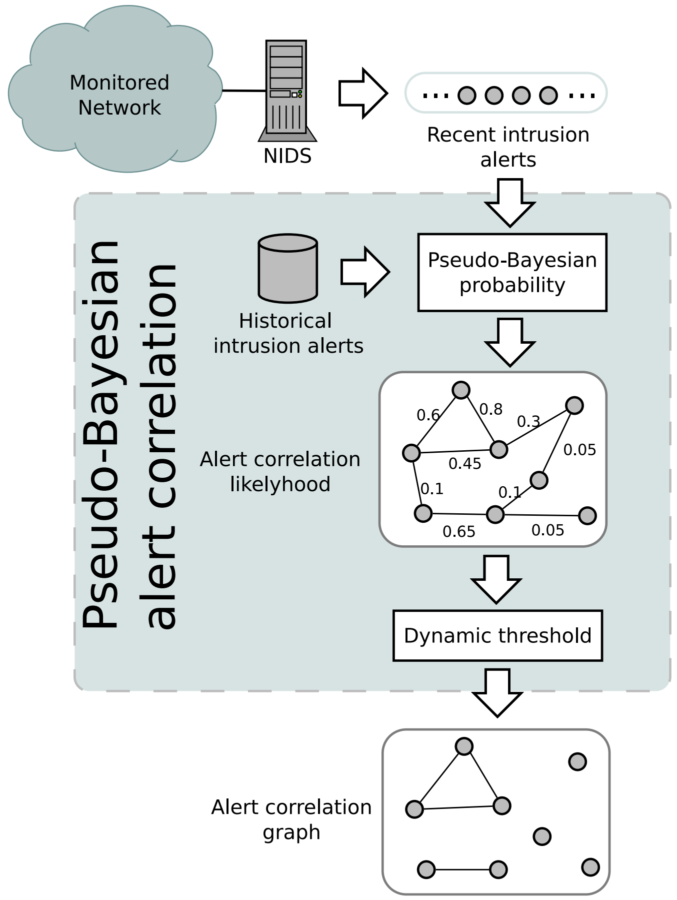

Monitored Network
NIDS
Recent intrusion alerts
Pseudo-Bayesian alert correlation
Historical intrusion alerts
Pseudo-Bayesian probability
Alert correlation likelyhood
0.6
0.8
0.3
0.05
0.45
0.1
0.1
0.65
0.05
Dynamic threshold
Alert correlation graph

Figure 1. High level architecture of the proposed pseudo-bayesian alert correlation scheme

periodically, while recent intrusion alerts are received from the NIDS and analyzed as soon as they are generated. The intermediate output of this first processing step is a weighted graph in which the nodes represent intrusion alerts, and the vertexes express the likelihood of two connected intrusion alerts to be correlated. The details of the *psuedo-bayesian probability* algorithm are described in Section IV.

This intermediate result represents the input of the second (and last) processing step, called *dynamic threshold* in Figure 1. This step prunes the correlation graph by removing the vertexes having a relatively low weight. Hence, our alert correlation algorithm highlights only the groups of alerts that are strongly correlated. The *dynamic threshold* algorithm is presented in Section IV-A.

IV. Pseudo-bayesian correlation probability

In the context of signature-based NIDS, different signatures trigger alerts of different alert types, and each alert instance is characterized by an identifier of the signature that has been triggered. Let A and B be two different types of alert, generated by two different attack signatures. They can be modeled as sets of homogeneous alert instances. Let a and b be two alert instances of type A and B, respectively. Then $a \in A$, and $b \in B$.

The two alert instances a and b are considered *correlated* if both the following conditions are verified:

1) Previous history shows that there is a high *pseudo-bayesian probability* that an alert of type A is triggered shortly after an alert of type B. We express this notion with the notation $P(A|B)$ (pseudo-bayesian probability of A given B).
2) Alert instances a and b have been generated within a short timeframe.

These conditions embed the underlying assumption that two intrusion activities that belong to the same multistep attack scenario exhibit a strong temporal locality. While the second condition can be verified easily by comparing the timestamps associated to the alert instances a and b, the pseudo-bayesian probability $P(A|B)$ can only be computed by analyzing the intrusion alerts that have been previously generated by the NIDS.

Given a finite set of intrusion alerts (alert log) L and any two alert types $A = \{a_1, a_2, a_3, \ldots, a_m\}$ and $B = \{b_1, b_2, b_3, \ldots, b_n\}$, such as $A \subseteq L$ and $B \subseteq L$, we define the set $Corr(A, B)$ containing all the potentially correlated instances of alert types A and B as

$$Corr(A, B) = \{(a, b) \in A \times B : |t_a - t_b| \leq T_{max}\} \quad (1)$$

where t_a and t_b are the timestamps of the alert instances a and b respectively, and T_{max} is a configurable parameter. T_{max} identifies the highest possible *time threshold* for considering two alert instances a and b potentially correlated. I.e., if the timestamps of alert instances a and b differ for more that T_{max}, than a and b will never be considered as correlated, independently of the alert history L.

We now define the set A_{corr} as the subset of A containing only elements that appear in $Corr(A, B)$:

$$A_{corr} = \{a \in A \mid \exists b \in B : (a, b) \in Corr(A, B)\} \quad (2)$$

We also define the two set $A_{\overline{corr}}$ as the subset of A containing only elements that do not appear in $Corr(A, B)$:

$$A_{\overline{corr}} = \{a \in A \mid \nexists b \in B : (a, b) \in Corr(A, B)\} \quad (3)$$

It is now possible to define the *pseudo-bayesian probability of having an intrusion alert of type A given an intrusion alert of type B*, or $P(A|B)$, as

$$P(A|B) = P(Corr(A, B)) - \frac{|A_{\overline{corr}}|}{|A|} \quad (4)$$

The term $|A_{\overline{corr}}|/|A|$ in equation 4 represents the ratio between the number of alerts of type A that are not correlated to any alert of type B and the total number of alerts of type A. This quantity ranges from 0 (if all instances of A are correlated to at least one instance of B) to 1 (if no instance of A is correlated to any instance of B), and it is subtracted to the value of $P(Corr(A, B))$. Hence, even if a large number of instances of alert type A are correlated to instances of alert type B, but the number of alert instances of type A that are not correlated to alert instances of type B is much larger, the probability $P(A|B)$ is significantly decreased.

The other term of equation 4, $P(Corr(A, B))$ is defined as follows:

$$P(Corr(A, B)) = \frac{1}{|Corr(A, B)|} \sum_{(a,b) \in Corr(A,B)} e^{-\frac{(t_a - t_b)^2}{K}} \quad (5)$$

Each term of the sum represents the time correlation between two alert instances a and b, where $(a, b) \in Corr(A, B)$. The time correlation value decreases exponentially (according to a Gaussian decay) as the difference between the timestamps of the two alert instances ($|t_a - t_b|$) increases. The decay is modulated by the coefficient K. For low values of K, the decay will be faster, and even small time differences will result in low correlation values. The coefficient K is computed as a function of the user-defined parameter T_{max}, already used in equation 1.

In particular, we want to compute K so that the value of the function

$$f(|t_a - t_b|) = e^{-\frac{|t_a - t_b|^2}{K}} \quad (6)$$

reaches an arbitrarily small *cutoff* value C when $x = |t_a - t_b| = T_{max}$. In our implementation $C = 0.01$, hence the correlation between two alert instances a and b whose timestamps differ by T_{max} will be 1%. To compute K we have to solve the following equation:

$$f(T_{max}) = C \longrightarrow e^{-\frac{T_{max}^2}{K}} = C \quad (7)$$

that is an an exponential equation in which K is the only variable. By solving it, we obtain

$$K = -\frac{T_{max}^2}{\log C} \quad (8)$$

The proposed approach is completely unsupervised, and does not rely on any a-priori knowledge about the nature of the intrusion alerts or about the network environment monitored by the NIDS. Information about likely alert correlations are only inferred from previously analyzed alerts. Hence, the reliability of the pseudo-bayesian correlation index will be influenced by the completeness of this knowledge base, that can be approximated by the number of alerts that are included in the alert log.

If the number of past intrusion alerts (that is, the size of the set L, or $|L|$) is low, then the correlation indexes will be computed on the basis of incomplete and possibly biased information. As $|L|$ grows the correlation index will become more reliable, and less sensitive with respect to perturbations caused by anomalous alert clusters.

To account for this phenomena, we multiply the value $P(A|B)$ by a weight factor w that is close to zero for small values of $|L|$, and that increases as $|L|$ grows by approaching asymptotically 1. Given an alert log L, the weight factor w

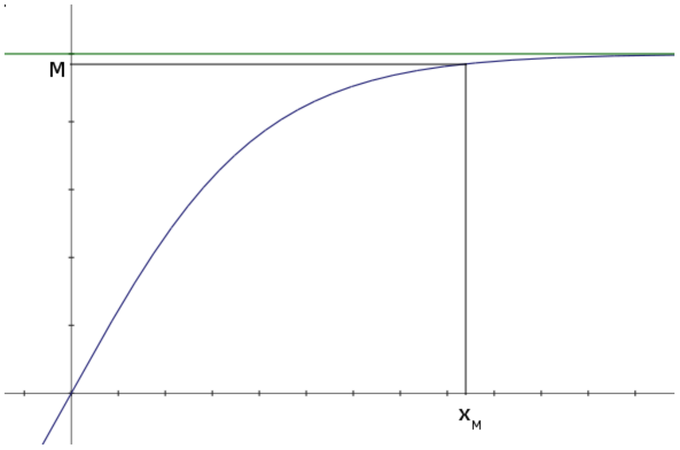


Figure 2. Plot of the hyperbolic tangent function $W(x) = \tanh\left(\frac{x}{k}\right)$, used as weight function for the *pseudo*-bayesian correlation index

is computed as $w = W(|L|)$, where W represents a suitable weight function.

In this paper we use the *hyperbolic tangent* $\tanh$ (see Figure 2) as the weight function:

$$W(x) = \tanh\left(\frac{x}{k}\right) = \frac{e^{\frac{x}{k}} - e^{-\frac{x}{k}}}{e^{\frac{x}{k}} + e^{-\frac{x}{k}}} \tag{9}$$

The parameter k allows us to tune the weight function by letting the user choose how fast it will approach unitary value as $|L|$ grows. It is computed as a function of two parameters, M and x_M. M is a value arbitrarily close to 1, and x_M represents the value for which $W(x_M) = M$. In our proposal, M is equal to 0.95, and the user can tune the weight function by choosing x_M, i.e. the cardinality of the alert set L for which $W(|L|) = 0.95$.

By setting a suitable value of x_M and by imposing $w = W(x_M) = M$, we get the following equation:

$$w = W(x_M) = M \longrightarrow \frac{e^{\frac{x_M}{k}} - e^{-\frac{x_M}{k}}}{e^{\frac{x_M}{k}} + e^{-\frac{x_M}{k}}} = M \tag{10}$$

in which k is the only variable. In particular, let $t = x_M/k$. For a fixed value of M, an approximated solution t_{sol} can be computed through a numerical approximation, as shown in [12]. For example, when $M = 0.95$ we have $t_{sol} \simeq 1.83178$

The parameter k can be computed by solving the following equation:

$$k = \frac{x_M}{t_{sol}} \tag{11}$$

The output of this processing step is a graph that expresses how strong is the correlation likelihood among different alert instances. Each alert instance is represented by a node in the graph. If two alert instances a and b have been generated within the same timeframe (that is, if their timestamps are less than T_{max} away) and if the pseudo-bayesian probability $P(A|B)$ is higher than 0, then a and b are connected through a vertex whose weight is $wP(A|B)$. This graph is then processed by the *dynamic threshold* algorithm.

A. *Dynamic threshold*

This processing step generates the final alert correlation graph by removing from the alert correlation likelihood graphs all the vertexes whose weight is relatively low, compared to the average weight of all the vertexes. This process serves two purposes. First of all, it removes false correlations, because it is possible for two alert instances to be connected by a vertex even though there is no real correlation nor causal relationships among them. This may occur just because alerts of the same type have been generated previously within the same time-frame by accident. The positive point is that false correlations are typically characterized by low correlation weights, that are removed by the dynamic threshold algorithm. Moreover, by preserving only the correlations that are characterized by a high weight, dynamic threshold highlights only the alert instances that are most likely to belong to the same multistep attack scenario, thus aiding the security analyst to identify them among the multitude of alerts generated by the NIDS.

A dynamic threshold algorithm removes from the alert correlation likelihood graph all the vertexes whose weight v does not satisfy the following condition:

$$v \geq \mu_{corr} + \lambda\sigma_{corr} \tag{12}$$

where μ_{corr} is the average value of the weights of all the vertexes in the graph, and σ_{corr} is its standard deviation. The user defined parameter $\lambda \in \mathbb{R}$ denotes how far from the average we want to shift in order to consider two alert instances as correlated. Feasible values of λ for our purposes are in the interval $\lambda \in [0, 2]$. For $\lambda \simeq 0$ we consider as correlated all the pairs of alert instances having a correlation value higher than the average one. This usually implies a large correlation graph, composed by many interconnected alert clusters. A value of $\lambda \simeq 2$ brings to a smaller correlation graph, that only contains the most strongly correlated pairs of alerts. Higher values for λ could result in empty or poorly populated correlation graphs, since the correlation constraint could be too strict.

V. EXPERIMENTAL RESULTS

We carried out several experiments using a prototype implementation of the proposed multistep alert correlation architecture. The software has been mainly developed in C, as a module for Intrusion Detection System software *Snort*. It also includes a Web-based user interface realized in Perl, HTML and AJAX.

Our goals are to demonstrate that the computational cost of the proposed solution is compatible with live traffic analysis and to verify the capability of the system to recognize and correlate alerts belonging to the same attack scenario.

A. *Performance evaluation*

To achieve high performance, several different processing steps have been implemented as concurrent threads, that run in parallel with traffic analysis. In particular, the training phase that relies on historical alerts in order to compute the pseudo-bayesian probability $P(A|B)$ (where A and B are sets of

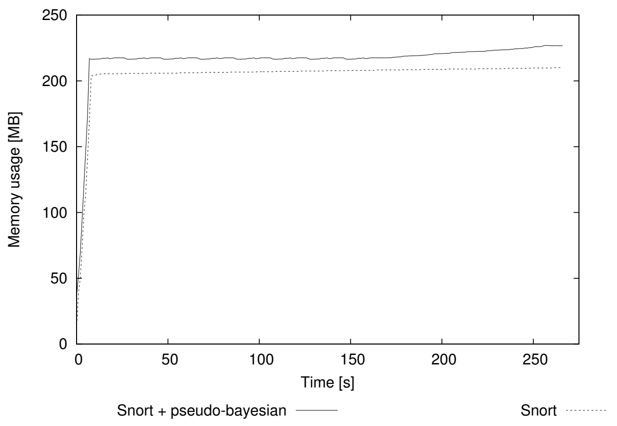
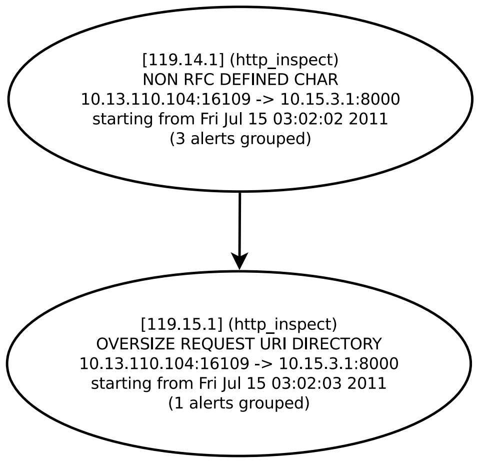


Figure 3. Memory usage of Snort with and without the pseudo-bayesian alert correlation module for multistep attack detection



[119.14.1] (http_inspect)
NON RFC DEFINED CHAR
10.13.110.104:16109 -> 10.15.3.1:8000
starting from Fri Jul 15 03:02:02 2011
(3 alerts grouped)
[119.15.1] (http_inspect)
OVERSIZE REQUEST URI DIRECTORY
10.13.110.104:16109 -> 10.15.3.1:8000
starting from Fri Jul 15 03:02:03 2011
(1 alerts grouped)

Figure 4. Example of a correlation sub-graph generated by the software using the *Capture the Flag* 2010 dataset, successfully identifying correlated alerts between the same endpoints

homogeneous alert instances) is performed in the background. As soon as the new values for $P(A|B)$ (for all the possible combinations of alert types A and B for which $Corr(A, B)$ is not an empty set) are computed, they are substituted to the previous values. This design choice allows us to leverage multi-core CPUs to perform these expensive operations while minimizing the impacts on the performance of Snort.

The two lines in Figure 3 show the memory usage of the Snort process while analyzing a 486MB traffic trace with (solid line) and without (dashed line) our pseudo-bayesian alert correlation module for multistep attack detection. When our module is active, the memory usage of Snort is slightly higher. However, in both the cases the analysis of the traffic dump requires about 267 seconds, hence the impact of our module on the time required by Snort to analyze the traffic trace is negligible.

B. Alert correlation

A preliminary set of experiments, aiming to verify all the features of our software, were conducted using small-scale traffic traces, including attack scenarios performed within controlled network environment.

Extensive experimental evaluation have then been carried out against the *Capture the Flag* (CTF) 2010 dataset, that includes 40 GB of network traffic in PCAP format and is publicly available [4]. The experimental results performed in this paper have been achieved by setting the parameter T_{max} to 1 hour, and x_M to 1000 alerts.

The last processing step, that computes the correlation index among clustered alerts belonging to the same attack scenario, is quite sensitive to the choice of the parameter λ in Eq. 12. Feasible values are in the interval $\lambda \in [0, 2]$, as discussed in Section IV-A. A value of λ close to zero produces a graph that contains many alert correlations (all the ones having a correlation value greater than the average one). This is useful when the user wants to know all the likely correlations. A value closer to 2 highlights the few "strongest" correlations, having high correlation values. For example, by setting $\lambda = 1.4$, we obtain two different attack scenarios, depicted in Figures 4 and 5, respectively.

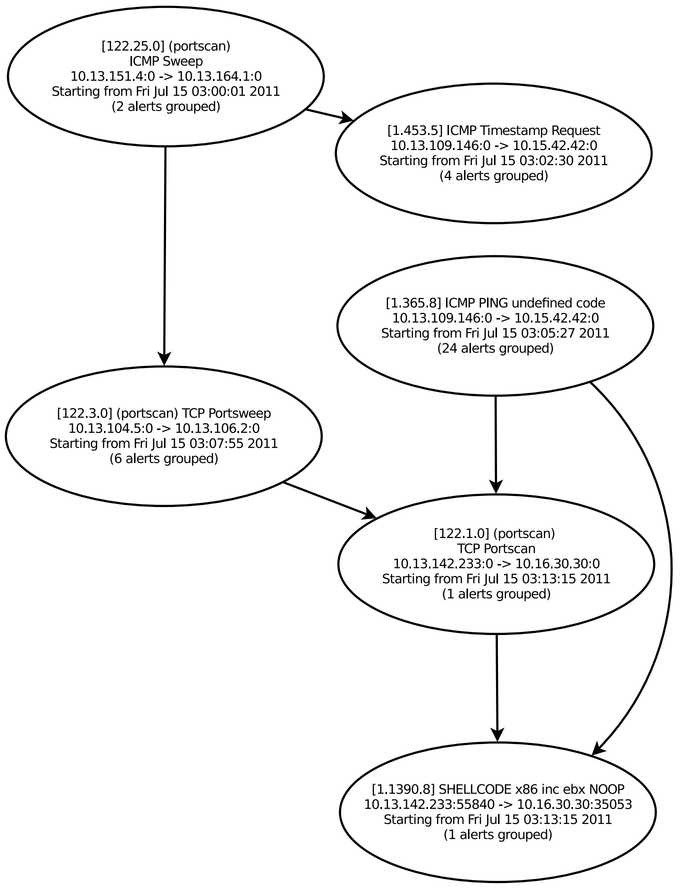
The scenario represented by the correlation graph in Figure 4 correctly identifies two different alerts produced by the Snort `http_inspect` module. Both alerts are related to the same attack scenario, in which a host repeatedly tries to exploit a vulnerability in the same victim Web server by issuing illicit Web requests. In particular, the first alert group (composed by 4 different alert instances) is related to requests whose URI contains characters that are not allowed in the RFC. These alerts are connected to an alert raised by a request characterized by an overly long URI (a likely attempt to trigger a buffer overflow in the Web server).

The correlation graph depicted in Figure 5 represents a more complex attack scenario, in which several scans and sweeps towards the same subnet in the same time window are correlated with an exploit attempt. Of particular relevance is the correlation between the two alerts at the bottom of Figure 5. These two alerts represent a portscan activity, correlated to an alert raised after the detection of a shellcode. Portscans are performed to identify vulnerable services offered by the target machine, hence it is very likely that the attacker tried to exploit a vulnerable service through a shellcode injection only after having identified a vulnerable service through a portscan.

On the other hand, by setting $\lambda = 1.1$ we obtain a larger number of likely scenarios. The XML file that contains about 40 alert clusters that represent different attack scenarios and the corresponding correlation graphs cannot be included and discussed in this paper.

VI. Conclusion

This paper presents a novel algorithm for the correlation of intrusion alerts generated by signature-based NIDS. The



[122.25.0] (portscan) ICMP Sweep 10.13.151.4:0 -> 10.13.164.1:0 Starting from Fri Jul 15 03:00:01 2011 (2 alerts grouped)
[1.453.5] ICMP Timestamp Request 10.13.109.146:0 -> 10.15.42.42:0 Starting from Fri Jul 15 03:02:30 2011 (4 alerts grouped)
[1.365.8] ICMP PING undefined code 10.13.109.146:0 -> 10.15.42.42:0 Starting from Fri Jul 15 03:05:27 2011 (24 alerts grouped)
[122.3.0] (portscan) TCP Portsweep 10.13.104.5:0 -> 10.13.106.2:0 Starting from Fri Jul 15 03:07:55 2011 (6 alerts grouped)
[122.1.0] (portscan) TCP Portscan 10.13.142.233:0 -> 10.16.30.30:0 Starting from Fri Jul 15 03:13:15 2011 (1 alerts grouped)
[1.1390.8] SHELLCODE x86 inc ebx NOOP 10.13.142.233:55840 -> 10.16.30.30:35053 Starting from Fri Jul 15 03:13:15 2011 (1 alerts grouped)

Figure 5. Another example of a multistep scenario identified by the software, showing a correlation between reconnaissance activities and an exploitation attempt through a shellcode injection

proposed algorithm, called *pseudo-bayesian alert correlation*, is inspired to Bayes' theorem of conditional probability, it is completely unsupervised, and aims to highlight correlations among intrusion alerts that belong to the same multistep attack scenario. Viability and efficacy of the proposed pseudo-bayesian alert correlation algorithm are demonstrated through a prototype, tested against recent and publicly available traffic traces. Experimental results show that the proposed algorithm is able to correlate intrusion activities belonging to the same attack scenario, thus helping security administrator in the analysis of alerts produced by a NIDS. Moreover, by leveraging modern multi-core architectures to perform training in parallel and non-blocking threads, the computational cost of our prototype is compatible with live traffic analysis. Since *pseudo-bayesian alert correlation* is completely unsupervised, and does not require any apriori knowledge about complex attack scenarios and the monitored environment, it can be immediately used to analyze security alerts generated by any signature-based NIDS.

Acknowledgements

The authors acknowledge the support of MIUR-PRIN project DOTS-LCCI "Dependable Off-The-Shelf based middleware systems for Large-scale Complex Critical Infrastructures".

References

[1] F. Valeur, G. Vigna, C. Kruegel, and R. A. Kemmerer, "A comprehensive approach to intrusion detection alert correlation," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, pp. 146–169, 2004.

[2] S. Benferhat, T. Kenaza, and A. Mokhtari, "A naive bayes approach for detecting coordinated attacks," in *Proc. of the 32nd IEEE International Annual Conference on Computer Software and Applications COMPSAC '08*, 2008.

[3] "Snort home page, available online at http://www.snort.org."

[4] "Capture the flag traffic dump, available online at http://www.defcon.org/html/links/dc-ctf.html."

[5] A. Valdes and K. Skinner, "Adaptive, model-based monitoring for cyber attack detection," in *Proc. of Third International Workshop on the Recent Advances in Intrusion Detection RAID 2000*, Toulouse, France, 2000.

[6] M. Panda and M. R. Patra, "Network intrusion detection using naïve bayes," *International Journal on Computer Science and Network Security*, vol. 7, no. 12, pp. 258–263, 2007.

[7] A. A. Sebyala, T. Olukemi, and L. Sacks, "Active platform security through intrusion detection using naive bayesian network for anomaly detection," in *Proc. of the London Communications Symposium*, 2002.

[8] M. Mehdi, S. Zahir, A. Anou, and M. Bensebti, "A bayesian networks in intrusion detection systems," *Journal of Computer Science*, vol. 3, no. 5, pp. 259–265, 2007.

[9] A. Cemerlic, L. Yang, and J. M. Kizza, "Network intrusion detection based on bayesian networks," in *Proc. of the 20th International Conference on Software Engineering and Knowledge Engineering (SEKE'08)*, Redwood City, CA, USA, 2008.

[10] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian event classification for intrusion detection," in *Proc. of the 19th Annual Computer Security Applications Conference (ACSAC '03)*, Las Vegas, NV, USA, 2003, p. 14.

[11] M. Colajanni, M. Marchetti, and M. Messori, "Selective and early threat detection in large networked systems," in *Proc. of the 10th IEEE International Conference on Computer and Information Technology (CIT 2010)*, 2010.

[12] "Solution of equation 10 obtained through the computational engine wolfram alpha. available online at http://goo.gl/Ozgc."