Data Security Analysis using Unsupervised Learning and Explanations

G. Corral[1], E. Armengol[2], A. Fornells[1], and E. Golobardes[1]

[1] Grup de Recerca en Sistemes Intel·ligents
Enginyeria i Arquitectura La Salle, Universitat Ramon Llull
Quatre Camins, 2, 08022 Barcelona (Spain)
[2] IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Barcelona (Spain)
[1]{guiomar, afornells, elisabet}@salle.url.edu, [2]eva@iiia.csic.es

Abstract. Vulnerability assessment is an effective security mechanism to identify vulnerabilities in systems or networks before they are exploited. However manual analysis of network testing and vulnerability assessment results is time consuming and demands expertise. This paper presents an improvement of *Analia*, which is a security system to process results obtained after a vulnerability assessment using artificial intelligence techniques. The system applies unsupervised clustering techniques to discover hidden patterns and extract abnormal device behaviours by clustering devices in groups that share similar vulnerabilities. The proposed improvement consists in extracting a symbolic explanation for each cluster to help security analysts to understand the clustering solution using network security lexicon.

Keywords: Network Security, Unsupervised Learning, Clustering, Explanations.

1 Introduction

The significant growth of networks and Internet has lead to increase security risks. As networks and networked systems become essential in any corporation, also their vulnerabilities become a main concern. Vulnerability assessment is the process of identifying and quantifying vulnerabilities in a system and it pursues two main goals: test everything possible and generate a concise report [1]. However, time and cost can limit the depth of a vulnerability assessment. These limitations justify the automation of the processes involved in a vulnerability assessment, not only those related to the testing phase, but also those related to the analysis of test results, as a thorough network security test generates extensive data quantities that need to be audited.

Logs collection, network traffic capture and potential threat identification are tasks difficult to handle when managing large data sets. Thus it is prohibitively expensive to classify it manually [2]. Artificial Intelligence (AI) techniques can help managing all this information and identifying subjacent patterns of behaviour. More specifically, unsupervised learning is suitable to handle vulnerability results as it does not need previous knowledge of data and can find relationships between tested devices.

In previous works we demonstrated the utility of clustering vulnerability assessment results [3,4], including partition methods [5] and soft computing solutions, like Self-Organization Maps (SOM) [6]. These solutions have been combined in *Consensus*, an integrated computer-aided system to automate network security tests [3,7], and this new system is called *Analia*. Security analysts obtain a configuration of different clusters, and every cluster contains tested devices with similar vulnerabilities and behaviours. When identifying and solving the vulnerabilities of a device, the same process can be applied to all devices in that cluster. Also main efforts can be focused first on the most critical clusters without having to analyze the whole data set. Analysts can evaluate results with clustering validity indexes. However they do not know the reasons of that clustering and have no explanation of the obtained solution.

This paper presents an improvement of *Analia* based on constructing explanations for clusters. This solution focuses on creating generalizations based on the anti-unification concept [8] to characterize each cluster. These generalizations permit describing a cluster with the same representation language used to describe the elements; therefore security analysts can more easily understand results. The analyst will obtain a solution where all tested devices have been grouped in different clusters regarding their vulnerabilities and will know the reason of these clustering results.

This paper is organized as follows: Section 2 surveys related work about the application of AI in network security. Section 3 explains the concept of symbolic description. Section 4 details the *Analia* system. Section 5 describes the experiments. Finally, Section 6 presents conclusions and further work.

2 Related work

Considerable data quantities are compiled after performing a network security test, and therefore a manual classification becomes an arduous work. AI techniques are useful in the analysis phase to handle vulnerability assessment results. In particular unsupervised learning techniques are appropriate in this environment, where no previous knowledge of network behaviour and data results is required.

Clustering permits dividing data space into regions based on a similarity metric. Among many approaches, *K*-means [5] and SOM [6] highlight over the rest. *K*-means has been applied to group similar alarm records [9], for intrusion or anomaly detection [2,10], and for network traffic classification [11]. On the other hand, SOM has been used to detect anomalous traffic, intrusions, and classify attacks [12,13,14].

Another important aspect to be considered is the comprehension of the relationship between elements of an obtained cluster. For this reason, extraction of explanations from results is a key point. Symbolic descriptions [8] have been used in Case-based Reasoning systems to produce explanations on their performance and to organize their case memory [15,16]. In the present paper we adapt this idea to generate explanations to justify the clusters produced using an unsupervised technique.

3 Explanations

When clustering in a network security environment, not only the distribution of tested devices in groups regarding their vulnerabilities is useful information for security analysts, but also the reason of that organization. This paper proposes a value-added service by including symbolic descriptions that explain why a subset of cases has been grouped together. These descriptions are based on the *anti-unification* concept introduced in [8] although with some differences. The anti-unification of two objects is defined as the most specific generalization of both objects and it is a description with the attributes shared by both objects whose value is the most specific one. However, this paper considers the shared attributes among a set of objects, without using the most specific generalization of the values of these attributes.

Let C_i be a cluster and let $e_1, \ldots, e_n$ be the set of elements of that cluster after the application of a clustering algorithm included in *Analia*, like *K*-means, *X*-means, SOM or Autoclass. Each element e_j is described by a set of attributes *A*, where each attribute $a_k \in A$ takes values in V_k. We propose to describe the cluster C_i using a symbolic description D_i, with the following rules:

– D_i contains the common attributes to all the elements in C_i. Those attributes with unknown value in some element $e_j \in C_i$ are not considered in constructing D_i.
– Let a_k be an attribute common to all the elements in C_i taking values on a set V_k. The attribute a_k is not included in the description of D_i when the union of the values that a_k takes in the elements in C_i is exactly V_k.

Let us illustrate how to build the description D_i with an example. Let C_i be the cluster formed by the three elements and D_i its explanation, shown in Table 1. Attributes common to all elements with values different from 0 are included in D_i. Notice that attribute 'port 25' is not in D_i because it takes all its possible values.

Table 1. Description of three elements from *Consensus* dataset and their explanation D_i.

	Ports						**W2000**		**XP**	**W2003**	**Security Notes**
	21	**25**	**53**	**80**	**135**	**445**	**SP3**	**SP4**	**SP2**	**Server**	
e_1	1	0	0	0	1	1	41%	41%	41%	41%	3
e_2	1	2	1	0	1	1	41%	41%	41%	41%	6
e_3	1	1	0	1	1	1	41%	41%	41%	41%	6
D_i	1	--	--	--	1	1	41%	41%	41%	41%	(3,6)

4 Analia

Analia is the Analysis module of *Consensus* that introduces AI techniques to improve the results analysis after a network security test. *Consensus* automates the security testing procedures to reliably verify network security [7]. *Analia* uses *Consensus* to gather data and then applies unsupervised learning to help security analysts.

The main goal of *Analia* is to help finding hidden patterns in tested devices. After a security test, analysts must focus on every device in order to find abnormal behaviours, incorrect configurations or critical vulnerabilities. If the list of devices is extensive some behaviours could become unnoticeable, some patterns could be masked or maybe the most vulnerable devices could be the last to be checked. *Analia* aids analysts to find similarities within data resulting from security tests. Then unsupervised learning helps analysts extracting conclusions without analyzing the whole data set. Next, best results are selected by using validity indexes when evaluating different executions. Finally, explanations justify clustering results.

This modular architecture offers different advantages. First, there is a separation between data collection and analysis. Network tests can be regularly planned and data is stored after every test. Whenever necessary, an analysis of the tested devices can be performed. *Analia* is a complementary module, so analysts can choose a basic report or process data using the AI module to refine results. *Analia* works with the same database where *Consensus* stores all data, avoiding data duplication. Another feature is its knowledge representation flexibility. Different representations can be configured for the same data set as inputs to the clustering algorithms. Also the incorporation of new unsupervised learning techniques can be easily performed. Configuration files are obtained from the parameters selected by the analyst in the *Analia* web interface.

The main goal of clustering is to group elements with similar attribute values into the same class. In the clustering task, classes are initially unknown and they need to be discovered from data. The clustering process usually involves the following steps:

Pattern representation. Patterns are multidimensional vectors, where each dimension is a feature. In a clustering context, with lacking class labels for patterns, the feature selection process is necessarily ad hoc. In *Analia*, every element is a tested device and features are characteristic data from a security test on that device. The main goal is to obtain groups of devices with similar vulnerabilities. Thus features should be related to data associated to vulnerabilities. Not all data stored in *Consensus* is suitable, as many testing tools return long data strings difficult for parametrization. Port scanning and operating system (OS) fingerprinting have been selected [3], as these processes determine most of the system vulnerabilities. Different knowledge representations can be configured in *Analia*. An example is shown in Table 2: OS features depict the reliability percentage of having that OS installed, and port features show when an open port (1), a filtered port (2), or no response (0) has been detected.

Pattern proximity measure. The pattern proximity measure has been defined in every clustering algorithm. All of them implement the Euclidean distance, a special case of the Minkowski metric. It measures the distance between two tested devices.

Clustering. Four clustering algorithms have been included in *Analia*: *K*-means [5], *X*-means [3], SOM [6] and AutoClass [17]. The goal is not to perform a system comparison, but to determine how well these methods perform over security data sets. Previous experiments have shown effective results [3,18]. Clustering process eases the analysis work compared to a raw data set. Also analysts do not need previous knowledge about these techniques as they only have to select an algorithm and the number of clusters to obtain. Although algorithms can report different clustering results, all they permit deriving behaviour patterns and discovering modified devices.

Table 2. Knowledge representation with OS and open ports.

Ports					**Operating Systems**				
23	**25**	**53**	**80**	**…**	**Linux**	**Solaris**	**XP SP1**	**XP SP2**	**…**
1	0	0	1	…	0.67	0.2	0.0	0.0	…

Data abstraction. This process extracts a simple and compact representation of a data set. A summary description of each cluster can be easy to comprehend and intuitively appealing for a security analyst. Generalizations have been obtained based on the anti-unification concept [8] to characterize each cluster, where a cluster is described using the same representation language that depicts data elements. For instance, *K-means* represents a cluster by its centroid and SOM, by its director vector. However, neither centroids nor director vectors is understandable information for security analysts. On the other hand, explanations use the same vocabulary of feature characterization.

Cluster validation. This process assesses a clustering procedure's output. It can not rely on given patterns as they do not exist in unsupervised domains. Different cluster validation techniques have been integrated in *Analia* in order to help analysts to check clustering results and compare different executions: Dunn [19], DB [20] and Silhouette [21] indexes. Also Cohesion indexes [18] have been designed ad hoc.

These steps have been analyzed and implemented in *Analia* in order to cluster data from a network security test in groups of devices with similar vulnerabilities.

5 Experiments

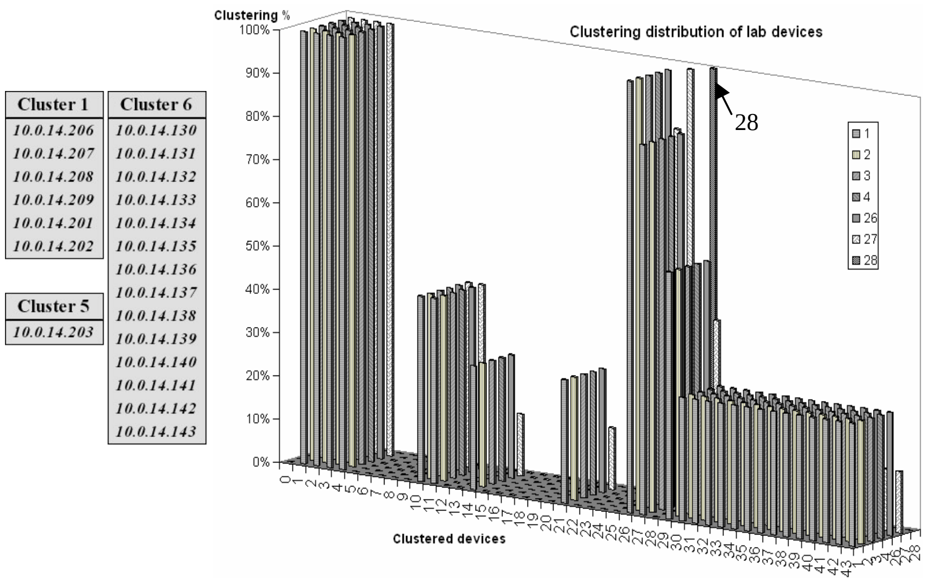
The main goal of the experimentation is to corroborate that not only clustering but also explanations let analysts obtain groups of devices with similar vulnerabilities and understand each cluster characterization. *Analia* can also help detecting unauthorized changes. When testing similar devices, clustering should group them. If an element is in another cluster, descriptions explain the changes between the modified and the rest.

Security tests have been executed over the university network to obtain data from real working servers, alumni laboratories and staff computers. Alumni lab computers should follow the same pattern as any other software installation is forbidden. Therefore they all should be in the same cluster if nobody has illegally installed new software or changed their configuration. 44 devices have been tested: 21 (14+7) from two different labs, 9 public servers, 11 internal servers and 3 staff computers.

The pattern representation of Table 2 has been selected. *K*-means has been executed with *K* from 3 to 8, where *K* is the number of clusters to obtain. *X*-means has been configured to find the best *K* between 3 and 8. SOM has been configured with size maps of 2×2, 3×3, 4×4, 5×5 and 6×6. Autoclass has been configured to calculate the number of clusters automatically. Also 10 different random seeds have been used. Best executions have been selected by using validity indexes (see previous section) as a decision factor.

Figure 1(a) shows clustering results related to lab devices. Other cluster results have been omitted for security purposes as they have public IP addresses. Lab computers should belong to two different groups, as they have two different configurations. However three clusters represent these devices. Analysts can detect at a glance the suspicious device in cluster 5. Another perspective is shown in Fig. 1(b). After obtaining clustering results from all executions, a statistic of how many times a device has been clustered with every other device for all executions has been calculated. The arrow in Fig. 1(b) shows that device 28 (with IP address 10.0.14.203) has never been clustered with any other tested device for all selected executions. On the other hand, devices 1, 2, 3, 4 and 26 have been clustered always together and device 27 has been clustered with them in more than the 80% of the executions. They correspond to devices of cluster 1. Thus analysts can easily detect an untrusting device but this information does not explain why this device is in a different cluster. However symbolic description of every cluster will give analysts this valuable information to quickly narrow the scope, isolate the device and apply the correction measures as soon as possible.

Table 3 shows the explanation of the studied clusters. These descriptions show that the OS of cluster 5 has not been modified with respect to cluster 1. However cluster 5 contains a high number of filtered ports (value=2). A filtered port means that a firewall, filter or some other network obstacle is blocking the port and preventing the system from determining whether it is open. Also ports 135 and 445 are open, which means that MSRPC[1] and Microsoft-DS[2] services have been altered. These ports should be blocked as they can be easily used to attack Windows computers. Therefore, analysts can promptly act to solve this situation without having to analyze the vulnerabilities of all lab computers.



Cluster 1	Cluster 6
10.0.14.206	10.0.14.130
10.0.14.207	10.0.14.131
10.0.14.208	10.0.14.132
10.0.14.209	10.0.14.133
10.0.14.201	10.0.14.134
10.0.14.202	10.0.14.135
	10.0.14.136
	10.0.14.137
	10.0.14.138
	10.0.14.139
	10.0.14.140
	10.0.14.141
	10.0.14.142
	10.0.14.143

Cluster 5
10.0.14.203

Fig. 1. Lab devices: (a) Clustering results, (b) Clustering distribution.

[1] MSRPC: Microsoft Remote Procedure Call.

[2] Microsoft-DS: Port used for file sharing in Microsoft Windows.

Table 3. Symbolic description of clusters containing lab devices.

	Ports					**W2000**		**XP**		**2003**	
Cluster	135	139	445	781-807	904-930	WS_SP4	S_SP2	SP1	SP2	Standard	Enterp.
1	---	1	---	---	---	67%	67%	67%	67%	70%	70%
5	1	1	1	2	2	67%	67%	67%	67%	70%	70%

6. Conclusions

Network vulnerability assessments generate many data and its analysis becomes a laborious work. By contrast, AI techniques are useful when dealing with large data sets. Unsupervised learning helps in the extraction of implicit, previously unknown and potentially useful information from data. Thus they are worthy to automate the classification of security data, thereby reducing the human effort required.

This paper has proposed *Analia*, a new system that includes unsupervised learning to cluster data obtained from a network vulnerability assessment. The system also introduces the concept of symbolic description to represent the main characteristics of the obtained clusters. The clustering process pursues grouping devices with similar vulnerabilities and discovering pattern behaviours regarding these vulnerabilities. Unsupervised learning can be very useful not only to identify similar devices, but also to find devices that unexpectedly appear separated and have new and different vulnerabilities. Explanations permit analysts to understand why some devices are in the same cluster or why they have been separated. This separation can be easily identified when a compact cluster with all the similar elements was expected. Also the relevant attributes of the clusters are easily identified in order to detect the common vulnerabilities of all the devices included in a cluster. Therefore, when studying an element of a cluster, the obtained conclusions can be applied to the rest of its neighbours. Special efforts can be primarily focused on the most vulnerable clusters or the clusters where the most critical devices are included.

Further work considers a deeper study on knowledge representation. Also an analysis of relations among the explanations of the clusters will be studied in order to get profit of that information for the system.

Acknowledgments

This work has been supported by the MCYT-FEDER project called MID-CBR (TIN2006-15140-C03-01 and TIN2006-15140-C03-03) and *Generalitat de Catalunya* for the support under grants 2005SGR-302 and 2007FIC-00976. We would also like to thank *Enginyeria i Arquitectura La Salle* (*Universitat Ramon Llull*) for the support to our research group.

References

1. T.R. Peltier, J. Peltier, and J. Blackley. Managing a Network Vulnerability Assessment. Auerbach Publishers Inc., 2003.
2. E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data, 2002.
3. G. Corral, E. Golobardes, O. Andreu, I. Serra, E. Maluquer, and A. Martinez. Application of clustering techniques in a network security testing system. AI Research and Development, IOS Press, 131:157–164, 2005.
4. A. Fornells, E. Golobardes, D. Vernet, and G. Corral. Unsupervised case memory organization: Analysing computational time and soft computing capabilities. In 8th European Conference on CBR, LNAI, 4106:241-255. Springer-Verlag, 2006.
5. J. Hartigan and M. Wong. A k-means clustering algorithm. In Applied Statistics, 28:100–108, 1979.
6. T. Kohonen. Self-Organization and Associative Memory, volume 8 of Springer Series in Information Sciences. Springer, Berlin, Heidelberg, 1984. 3rd ed. 1989.
7. G. Corral, A. Zaballos, X. Cadenas, and A. Grané. A distributed security system for an intranet. In 39th IEEE Int. Carnahan Conf. on Security Technology, 291-294, 2005.
8. E. Armengol and E. Plaza. Bottom-up induction of feature terms. Machine Learning, 41(1):259-294, 2000.
9. K. Julisch. Clustering intrusion detection alarms to support root cause analysis. ACM Transactions on Information and System Security, 6:443–471, 2003.
10. K. Leung and C. Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In Proc. 28th Australasian CS Conf., volume 38, 2005.
11. D. Marchette. A statistical method for profiling network traffic. In First USENIX Workshop on Intrusion Detection and Network Monitoring, 119–128, 1999.
12. M. Ramadas, S. Ostermann, and B. C. Tjaden. Detecting anomalous network traffic with SOMs. 6th Symposium on Recent Advances in Intrusion Detection, 2820: 36–54, 2003.
13. M.O. Depren, M. Topallar, E. Anarim, and K. Ciliz. Network-based anomaly intrusion detection system using SOMs. IEEE 12th Signal Processing and Communications Applications Conference, 76–79, 2004.
14. L. DeLooze. Classification of computer attacks using a self-organizing map. In Proc. of the 2004 IEEE Workshop on Information Assurance, pages 365–369, 2004.
15. E. Armengol and E. Plaza. Symbolic Explanation of Similarities in CBR. Computing and Informatics. Vol. 25, 1001-1019, 2006.
16. A. Fornells, E. Armengol, and E. Golobardes, Semi-supervised Strategy for Case Memory Organization. In 17th Europeen Conference on Machine Learning, 2007 (submitted).
17. P. Cheeseman and J. Stutz. Bayesian classification (autoclass): Theory and results. In Advances in Knowledge Discovery and Data Mining, pages 153–180, 1996.
18. G. Corral, A. Fornells, E. Golobardes, and J. Abella. Cohesion factors: improving the clustering capabilities of Consensus. In Intelligent Data Engineering and Automated Learning, in LNCS, 4224:488–495. Springer, 2006.
19. J.C. Dunn. Well separated clusters and optimal fuzzy partitions. In Journal of Cybernetics, 4:95–104, 1974.
20. D.L. Davies and D.W. Bouldin. A cluster separation measure. In IEEE Transactions on Pattern Analysis and Machine Learning, 4:224–227, 1979.
21. P.J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational Applications in Math, 20:53–65, 1987.