Learning the Daily Model of Network Traffic

Costantina Caruso, Donato Malerba, and Davide Papagni

Dipartimento di Informatica, Università degli Studi di Bari,
Via E. Orabona 4 - 70126 Bari - Italy
{caruso, malerba}@di.uniba.it, davidepapagni@libero.it

Abstract. Anomaly detection is based on profiles that represent normal behaviour of users, hosts or networks and detects attacks as significant deviations from these profiles. In the paper we propose a methodology based on the application of several data mining methods for the construction of the "normal" model of the ingoing traffic of a department-level network. The methodology returns a daily model of the network traffic as a result of four main steps: first, daily network connections are reconstructed from TCP/IP packet headers passing through the firewall and represented by means of feature vectors; second, network connections are grouped by applying a clustering method; third, clusters are described as sets of rules generated by a supervised inductive learning algorithm; fourth, rules are transformed into symbolic objects and similarities between symbolic objects are computed for each couple of days. The result is a longitudinal model of the similarity of network connections that can be used by a network administrator to identify deviations in network traffic patterns that may demand for his/her attention. The proposed methodology has been tested on log files of the firewall of our University Department.

Keywords: Anomaly detection, daily model, data mining, machine learning, symbolic data analysis, network traffic analysis.

1 Introduction and Related Work

Intrusion detection is the process [1] of monitoring the events occurring in a computer system or network and analysing them for signs of intrusions, defined as the attempts to bypass the security mechanisms of a computer or a network.

The two analysis strategies are *misuse detection* and *anomaly detection*. While the former is based on extensive knowledge, provided by human experts, of patterns associated with known attacks, the latter is based on profiles that represent normal behaviour of users, hosts or networks and detects attacks as significant deviations from these profiles. The misuse detection strategy is based on matching the input streams to a signatures database, so as it can detect only known attacks and its effectiveness strongly depends on how frequently the database is updated. On the contrary, anomaly detection is potentially able to recognize new types of attacks, but it requires to define first what can be considered a *normal* behaviour for the observed entity (which can be a computer system, a particular user, etc.), and thereafter to decide how to label a specific activity as abnormal [2].

M.-S. Hacid et al. (Eds.): ISMIS 2005, LNAI 3488, pp. 131 – 141, 2005.

To overcome difficulties met in manually defining a model of normality for anomaly detection, it has been proposed to apply data mining and machine learning methods to clean data [3] [4], that is, data certainly containing no anomalies. However, this approach presents at least two issues. Firstly, clean data are difficult to obtain because, if we use real data, it is very hard to guarantee that there are no attacks in them, while simulated data represent only a limited view of the behaviour of a real system (computer or network) and the model trained over them will be difficult to update. Secondly, it is difficult to make anomaly detection systems adaptive because they cannot be trained in real-time given that they require clean data.

In the literature, various methods have been proposed for constructing normal models from *simulated* data, either clean (i.e. normal) or anomalous. In [5] a normal sequences model is built by means of look ahead pairs and contiguous sequences. Lee & Stolfo [6] build a prediction model for anomalies by training decision trees over normal data, while Ghosh & Schwartzbard [7] use a neural network to obtain the model. In [8] unlabeled data for anomaly detection are analyzed by looking at user profiles and by comparing an intrusive activity vs. a normal activity. A survey of techniques used for anomaly detection is given in [9].

When dealing with large volumes of network or system data collected in real-time, it is impracticable to apply supervised learning methods, which require a preclassification of activities as normal or anomalous. This justifies the interest for some unsupervised learning techniques and, more in general, for descriptive data mining methods, whose aim is that of finding the intrinsic structures, relations, or affinities in data but no classes or labels are assigned a priori.

In this paper we propose a methodology for the construction of a daily model of the network traffic, that starts with the application of a clustering (unsupervised) method to feature vectors representing daily network connections reconstructed from TCP/IP packet headers passing through a firewall. Clusters are then described as sets of rules generated by a supervised inductive learning algorithm. The last steps of the proposed methodology consist of the transformation of rules into symbolic objects and in the subsequent computation of similarities between symbolic objects for each couple of days. The result is a longitudinal model of the similarity of network connections that can be used by a network administrator to identify deviations in network traffic patterns that may demand for his/her attention.

The paper is organized as follows. In the next section the collection and preprocessing of real log files of a firewall is illustrated, while in Section 3 the generation of descriptions, expressed as sets of rules, for daily network connections is explained. The transformation of rules into symbolic objects and the formulation of the dissimilarity measure used in our analysis are both described in Section 4. A discussion on discovered longitudinal models for the traffic of our departmental network concludes the paper.

2 Data Collection and Preprocessing

The source data set is formed by four successive weeks of firewall logs of our Department, from May 31st to June 27th , 2004. A transaction is a single logged packet, each of which is described by the following attributes: counter (integer), date, time,

protocol (tcp/udp/icmp), direction (inbound/outbound), action (accept, drop, rejected, refused), source IP, destination IP, service port, source port, length, rule, diagnostic message, sys_message and others. Files have been cleaned from our service traffic generated by internal servers or internal broadcast. No missing or incorrect values have been observed in extracted data.

The target dataset is built by reconstructing the entire connections from single packets. In our analysis we have used only accepted ingoing connections and we have a file for every day that contains all the connections opened and closed in that day. There are very few connections between two days that have been discarded. Since the goal is to create the daily description of our connections, we have chosen to work with few but fundamental attributes, namely:

a. *StartHalfHour* (integer): The original time field of a packet has the format hh:mm:ss. We have chosen to map this value to the range [0, 47] by dividing a day into 48 intervals, each of half an hour: a connection belongs to the time interval it starts in.
b. *Protocol* (nominal): udp, tcp; few icmp values have been dropped when generating training data.
c. *DestinationIP* (integer): this field, which is composed of four bytes, can only have the value of our network addresses, so we have considered only the last byte (values 0..255).
d. *SourceIP* (nominal): this field has a very high cardinality because it represents the entire IPv4 space.
e. *ServicePort* (nominal): the requested service (http, ftp, smtp and many other ports).
f. *NumPackets* (integer): the number of connection packets.
g. *Length* (integer): the time length of the connection.
h. *NationCode* (nominal): the two digit code of the nation the source IP belongs to.
i. *NationTimeZone* (integer): the time zone of the nation the source IP belongs to, expressed as difference from GMT value; in some cases this is a mean value (e.g. for Russia or USA).

The features set above describes quite precisely a connection, however the high cardinality of *SourceIP*, if treated as nominal feature, may cause some problems in subsequent data mining steps. In a set of preliminary experiments performed on the first week of available data, we tested the importance of *SourceIP* when the source of a connection is also described by two redundant, but less precise, features such as *NationCode* or *NationTimeZone*. We observed that *SourceIP* behaves as a *masking variable* [10] and adds a bit of noise to clustering, therefore we decided to drop it in subsequent experiments. At the end of the pre-processing step, two different feature sets have been considered, one including only *NationCode* and the other including only *NationTimeZone*.

3 Describing Daily Network Connections

Clustering. Clustering is a global model of discovery and summarization used as a basis for more precise models [11] and it widely employed in different sciences. In

this work two different clustering algorithms, namely K-means and EM, have been considered, so that a comparison of possibly different results is possible [10].

K-means is very simple and reasonably efficient. It distributes instances between clusters deterministically by trying to minimize the distance (in our case the Euclidean one) between an instance and its cluster's centroid. The EM (expectation-maximization) algorithm distributes instances probabilistically. It tries to maximize likelihood looking for the most feasible parameters' values, in particular, the number of clusters. Both K-means and EM are implemented in the data mining package Weka [12] which we used in our experiments.

One of the most important problems in a clustering task is to determine the optimal number k of clusters. The parameter k has been determined by applying a cross-validation procedure on clustering results generated by the EM algorithm. More precisely, data are randomly divided in 10 subsets containing almost the same number of instances, and EM is run on every subset; k is initially set to 1 and incremented by 1 if the log-likelihood averaged over the ten runs increases.

For a fair comparison of the two clustering methods, EM is executed first, keeping the results for the best k obtained by cross-validation, and then K-means is tested for the same k value of EM.

Classification Rules. Both clustering techniques presented above have the disadvantage that they do not provide intensional descriptions of the clusters obtained. Intensional descriptions are essential in our context, since they provide the network administrator with human-interpretable patterns of network traffic. To overcome this limitation, several conceptual clustering techniques have been proposed in the literature [13], [14]. However, conceptual clustering methods are known to be rather slow. Therefore, we adopt a two-stepped approach. First, we apply a non-conceptual clustering algorithm, either EM or k-means, in order to decrease the size of the problem. Then we generate a set of rules whose consequents represent the cluster membership, that is, rules discriminate the resulting clusters.

The rules, which provide an intensional description of clusters, are generated by means of the algorithm PART [15]. PART combines two rule learning paradigms, namely divide-and-conquer and separate-and-conquer, and is characterized by high efficiency. It adopts the separate-and-conquer strategy in that it builds a rule, removes the instances it covers, and continues creating rules recursively for the remaining instances until none are left. However, the generation of each rule differs from the standard separate-and-conquer strategy, since it is based on the rule conversion of the path to the leaf with the largest coverage in a pre-pruned decision tree built for the current set of instances (hence the combination with the divide-and-conquer strategy). The time taken to generate a rule set of size r is $O(r \cdot a \cdot N \cdot \log N)$, where a is the number of attributes or features and N is the number of training examples. It is noteworthy that no pruning of the resulting rule set is performed, since our goal is that of precisely describing the phenomena rather than increasing predictive accuracy.

By testing two distinct clustering algorithms on two different data sets corresponding to the two feature sets (one with *NationCode* and one with *NationTimeZone*), we collect four sets of rules for each day. Some statistics are reported shown on Table 1 for three days of June 2004. The average value of the number of generated rules is generally greater than one hundred for the first features set while it is more manage-

able for the second set. It is worthwhile to notice that in this way we can drastically reduce data to treat and to store: daily model of network traffic is given by few hundreds of rules vs. thousands of network connections. The proposed daily model is light and at the same time adaptive because it is easily able to follow network variability; a network traffic change implies that one of its components can disappear or appear and, in our approach, this means that a rule disappears or appears.

By examining the distribution of instances per cluster, we notice that while EM tends to create one big cluster and many other significantly smaller, K-means tends to evenly distribute instances among clusters.

Table 1. Values of instances, clusters and rules obtained on June 4th , 5th , and 6th , 2004

Day	Instances	Clusters EM/K-means		Rules generated by PART			
		1st feature set	2nd feature set	1st feature set		2nd feature set	
				EM	K-means	EM	K-means
June 4th, 2004	20,130	4	4	156	143	138	38
June 5th, 2004	40,270	5	3	98	188	10	22
June 6th, 2004	16,302	2	6	179	90	45	33

4 Building a Longitudinal Model of Similarities

The rule set generated by PART for every day is a *static* representation of daily network traffic. However, the network administrator needs to compare patterns of network traffic over time in order to:

- understand which events have to be considered recurrent or random
- observe how characteristics of network connections vary upon time.

To build a longitudinal model of network traffic from daily sets of rules, we propose to compute similarities between rules derived for a user-defined time window. Rules correspond to homogeneous classes or groups of connections, that is, to *second-order objects* according to the terminology commonly used in symbolic data analysis (*first-order objects* correspond to the description of individual connections) [16].

Second order objects are described by *set-valued* or *modal* variables. A variable Y defined for all elements k of a set E is termed *set-valued* with the domain $\mathcal{Y}$ if it takes its values in $\boldsymbol{P}(\mathcal{Y})=\{U \mid U \subseteq \mathcal{Y}\}$, that is the power set of $\mathcal{Y}$. When $Y(k)$ is finite for each k, then Y is called *multi-valued*. A single-valued variable is a special case of set-valued variable for which $|Y(k)|=1$ for each k. When an order relation $\prec$ is defined on $\mathcal{Y}$ then the value returned by a set-valued variable can be expressed by an interval

[α,β], and Y is termed an *interval* variable. More generally, a *modal* variable is a set-valued variable with a measure or a (frequency, probability or weight) distribution associated to $Y(k)$. The description of a class or a group of individuals by means of either set-valued variables or modal variables is termed *symbolic object* (SO). Specifically, a *Boolean symbolic object* (BSO) is described by set-valued variables only, while a *probabilistic symbolic object* (PSO), is described by modal variables with a relative frequency distribution associated to each of them.

A set of symbolic data, which involves the same variables to describe different (possibly overlapping) classes of individuals, can be described by a single table, called *symbolic data table*, where rows correspond to distinct symbolic data while columns correspond descriptive variables. Symbolic data tables are more complex to be analysed than standard data tables, since each item at the intersection of a row and a column can be either a finite set of values, or an interval or a probability distribution. The main goal of the research area known as *symbolic data analysis* is that of investigating new theoretically sound techniques to analyse such tables [16].

Many proposals of dissimilarity measures for BSOs have been reported in literature [17]. They have been implemented in a software package developed for the three-years IST project ASSO (Analysis System of Symbolic Official Data) (http://www.assoproject.be/).

Therefore, to provide the network administrator with a dynamic representation of network traffic, we propose to transform rules into symbolic objects and then to compute the dissimilarities between different SOs of different days by means of the ASSO software. The most similar and recurrent SOs in a fixed time window represent the dominant behaviour of network, similar and less frequent events are its secondary aspects while symbolic objects very different from other ones are anomalies.

Transformation of rules into symbolic objects is straightforward. For instance, the following rule, obtained by K-means on the first features subset on June 4th, 2004:

```
ServicePort=10131 AND StartHalfHour>41 AND Nation-
Code=DE : c3
```

is transformed in this SO:

```
[StartHalfHour=[42,47]] ∧ [Protocol=*] ∧ [Destina-
tionIP=*] ∧ [service=10131] ∧ [NumPackets=*] ∧
[Length=*] ∧ [NationCode=DE].
```

The symbol "*", which represents the whole domain, is assigned to variables not present in the rule antecedent.

To execute this step, the DISS module of the ASSO software has been used; the dissimilarity measure selected for our analysis is that proposed by Gowda and Diday [18].

The rules found for a given day are compared with rules of all previous days within a time window. To simplify computation at this stage, the ten rules with highest support have been chosen for each day. The result is a daily dissimilarity matrix, that is, a table of dissimilarities values between a day and its previous ones. Every element is a dissimilarity measure D and it is represented by "DxRy" where x is the day and y is the number of the rule with respect to the same day. The matrix has as many columns

Table 2. Structure of daily dissimilarity matrix

	$\mathbf{so1g_i}$	$\mathbf{so2g_i}$	…	$\mathbf{soN_ig_i}$
$\mathbf{so1g_1}$	$D(so1g_1,so1g_i)$	$D(so1g_1,so2g_i)$	…	$D(so1g_1,soN_ig_i)$
$\mathbf{so2g_1}$	$D(so2g_1,so1g_i)$	$D(so2g_1,so2g_i)$	…	$D(so2g_1,soN_ig_i)$
…	…	…	…	…
$\mathbf{soN_1g_1}$	$D(soN_1g_1,so1g_i)$	$D(soN_1g_1,so2g_i)$	…	$D(soN_1g_1,soN_ig_i)$
$\mathbf{so1g_2}$	$D(so1g_2,so1g_i)$	$D(so1g_2,so2g_i)$	…	$D(so1g_2,soN_ig_i)$
…	…	…	…	…
$\mathbf{soN_2g_2}$	$D(soN_2g_2,so1g_i)$	$D(soN_2g_2,so2g_i)$	…	$D(soN_2g_2,soN_ig_i)$
…	…	…	…	…
$\mathbf{so1g_{i-1}}$	$D(so1g_{i-1},so1g_i)$	$D(so1g_{i-1},so2g_i)$	…	$D(so1g_{i-1},soN_ig_i)$
…	…	…	…	…
$\mathbf{soN_{i-1}g_{i-1}}$	$D(soN_{i-1}g_{i-1},so1g_i)$	$D(soN_{i-1}g_{i-1},so2g_i)$	…	$D(soN_{i-1}g_{i-1},soN_ig_i)$

as the SOs of the referenced day while the number of its rows is equal to the sum of the SOs of all previous days. The structure of this matrix for the i^{th} day (i>2) is shown in Table 2, where N_i represents the number of SOs of the i^{th} day.

To identify similar rules between the present day and the previous ones a numerical threshold (dissMax) has been fixed; given a rule *r,* all the rules whose dissimilarity from *r* is less then dissMax are considered similar to *r*. An example of the possible output is given in the following, where rules similar to Rule 1 of the 4^{th} day (June 3^{rd}, 2004) are shown:

```
Day 4 similar rules found (dissMax = 2.0)

>> Rule 1: /* of the 4th day */

DestinationIP <= 87 AND NationTimeZone >-4 AND Star-
tHalfHour > 26 : c1 /* cluster identifier */

- Similar rules (1):

> Rule 3 of day 3 (diss=1.7528): NationTimeZone > -6
AND StartHalfHour > 25: c0.
```

By consulting the lists of similar rules the network administrator is able to recognize sets of connections that are recurrent and other less frequent or random sets. In this way, the network administrator can learn more and more detailed information about network traffic.

Further information about a rule or its corresponding symbolic object $\mathbf{so_ng_m}$ (*n*-th rule of *m*-th day) can be obtained by computing its dissimilarity mean value:

$$avgDiss(so_ng_m) = \frac{\sum_{i=m-w}^{m-1}\sum_{j=1}^{n_i} D(so_ng_m, so_jg_i)}{\sum_{i=m-w}^{m-1} n_i} \tag{1}$$

where *w* is the chosen time window (the number of previous days considered) and n_i is the number of rules of the j^{th} day. The dissimilarity mean value measures "how

much" a rule is similar to all the rules previously seen. By using a graphical representation (see Fig. 1), the high values of the calculated means can identify rules very dissimilar from other ones while the low values put in evidence rules similar to the previous ones in the fixed time window *w*.

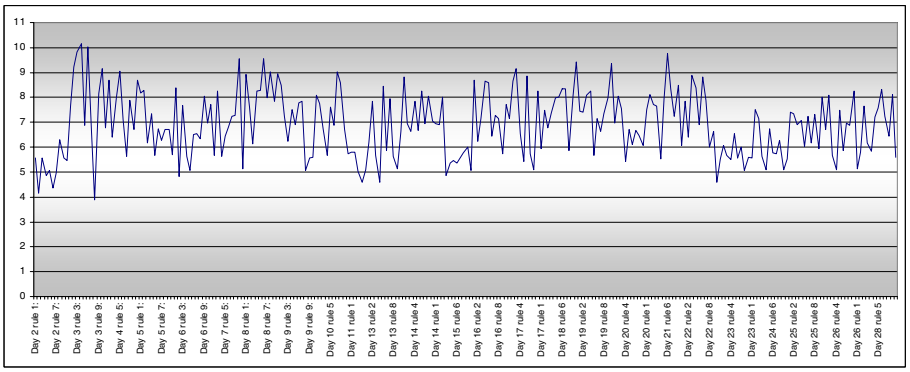

Fig. 1. The dissimilarity mean value of the rules generated by PART for clusters results obtained by EM algorithm and 1st feature set

Another interesting parameter is the minimum value of dissimilarity of a rule; indeed, a rule with high mean dissimilarity could be similar to few others but very dissimilar from the remaining ones and the mean value is not able to capture this situation. Therefore we compute, with the formula (2), the minimum value of dissimilarity amongst a fixed rule and all the rules of the previous days:

$$minDiss(so_n g_m) = \min_{i=m-w}^{m-1} \min_{j=1}^{n_i} D(so_n g_m, so_j g_i) \tag{2}$$

Dominant aspects of the network (i.e. frequent rules) can be reasonably represented by a set of recurrent similar rules whose mean and minimal dissimilarity values are low; secondary (but legal) aspects can have high mean dissimilarity values, while anomalies where domain expert attention must concentrate on, should have high values for both parameters.

5 Analysis of Results and Discussion

The main aim of this work is to investigate the potentialities of a new methodology in order to identify and learn patterns in the network traffic. The experiments have been made on the network traffic of our department but the features used are general and so this methodology can be applied to the traffic of every network active device.

One of the problem in learning network traffic is the huge quantity of data which pass daily through the network. It is important to reduce treated data by representing them with relatively few patterns. In our experiments we managed to compress the original firewall logs of 950 MB total size in 27 small dissimilarity matrices. For our

experiments, we used a Pentium IV with 512 MB Ram, 80 GB HD, CPU 2,4 GHz. The cleaning and preprocessing phases, the clustering phase, the rules generation phase and the DISS phase needed respectively 60 seconds, 15 minutes (EM) or 8 minutes (K-means), 15 minutes and 1 minute for each daily files.

We cannot report here all the graphs obtained by graphical representation of the mean and minimal dissimilarities but we try to summarize the main quantitative peculiarities observed in using K-means or EM as clustering algorithm and SP3.1 or SP3.2 as a feature subset. In our graphs, the mean dissimilarity fluctuates in a limited range of values since the fourth day (we chose w = 3). The mean values of extracted rules vary in a wider range when data are clusterized by EM rather than by K-means. In fact, the mean dissimilarity values obtained by K-means are generally lower of 1 unit than those obtained by EM, so that rules obtained by EM seems "more different" each other. The other aspect to underline is that the graphs obtained by EM and K-means have similar behaviour along the time: they generally show high and low peaks nearly at the same points. The first three days represent a transient period, for both the clustering algorithms, when the SP3.1 features subset is used, while this phenomenon does not show up for SP3.2. Although the graphs obtained with the two feature subsets have similar behaviour, mimimum and maximum points are respectively higher and lower with SP3.2 subset than with SP3.1. This suggests that the feature *Nation-TimeZone* puts more in evidence very similar or dissimilar rules.

The behaviour of mean dissimilarity and minimal one is the same: they respectively grown or decrease at the same time but the graphical representations obtained by the difference of mean and minimal dissimilarity show that this value is higher in the experiments with EM algorithm.

These results confirm that the proposed methodology is promising as to the most interesting aspect, that is, learning of network behaviour patterns along time by building a longitudinal model. A limit of this work is that we considered only syntactic dissimilarity measures according to which the more syntactically different two rules are the more dissimilar they are. However it may happen that two rules with several different attribute-value pairs represent the same network event. Therefore, it is important to investigate dissimilarity measures based on their actual coverage, that is, the observations that they daily share.

6 Conclusions and Future Work

For a network administrator it is important to have a complete description of the connections behaviour so to understand the development of his/her own network. This aspect is becoming more and more relevant and in fact commercial firewalls include modules which, though the computation of simple descriptive statistics, try to inform the security officer on the qualitative nature of network traffic. Firewalls have no means to give information about the mass of connections. The built-in modules permit to analyse every aspect of packet streams; some firewalls also possess an SQL dialect to query its own logs but SQL queries give answers about something the user already know or "suspect".

Personal experience as netadmins teaches us that some types of attack strategies against the network are discovered by chance. Often, netadmins have to read the fire-

wall logs to notice anomalous connections neither intrusion detection systems nor firewall itself would be able to notice. Although a firewall offers a privileged viewpoint with respect to other points in the network because of its concentration of all the pass-through traffic, the network security can only be guaranteed by agents distributed in different points that collaborate each other; a firewall is just one of this point.

These considerations justify some interest towards the application of data mining techniques to firewall logs in order to develop better tools for network security. In the paper we have proposed a methodology based on the application of several data mining methods for the construction of the "normal" model of the ingoing traffic of a department-level network. The methodology returns a daily model of the network traffic as a result of four main steps: 1) description of daily network connections; 2) clustering; 3) generation of sets of rules for each cluster; 4) conversion of rules into symbolic objects and computation of dissimilarities between symbolic objects.

As future work, we intend to achieve better computational performance and a better degree of automation. Moreover, to reach a better degree of automation we intend to investigate the problem of automatically selecting the threshold *dissMax* with respect to which rule similarity is compared. All results obtained as mean and minimum values will be further analysed by means of mathematical techniques to gain more insights on the dynamics of curves. Another research direction concerns the consideration of other essential information sources, like network servers logs or systems logs, for the tasks of intrusion detection and intrusion prevention. We also intend to investigate the use of coverage-based dissimilarity measures as alternative to syntax-based measures. Finally, we intend to develop further research on how to transform daily experience in effectual and stable knowledge which is able to distinguish among different types of anomalies.

Acknowledgments

The work presented in this paper is partial fulfillment of the research objective set by the ATENEO-2003 project on "Metodi di apprendimento automatico e di data mining per sistemi di conoscenza basati sulla semantica ".

References

1. Lazarević, A., Srivastava, J., Kumar, V.: Tutorial on the Pacific-Asia Conference on Knowledge Discovery in Databases (2003)
2. Axelsson, S.: IDS: A Survey and a Taxonomy (2000)
3. Bridges, S., Vaughn, R.: Intrusion Detection via Fuzzy Data Mining (2000)
4. Barbara, D. et al.: ADAM: A Testbed for Exploring the Use of Data Mining in Intrusion Detection, SIGMOD 2001
5. Hofmeyr, S.A., Forrest, S., Somayaji, A.: Intrusion Detection Using Sequences of System Calls. Journal of Computer Security, (1998) 6:151-180
6. Lee, W., Stolfo, S.J.: Data Mining approach for Intrusion Detection. Proceedings of the 1998 USENIX Security Symposium (1998)
7. Ghosh, A., Schwartzbard, A.: A Study in Using Neural Networks for Anomaly and Misuse Detection. Proceedings of the 8th USENIX Security Symposium (1999)

8. Lane, T., Brodley, C.E.: Sequence Matching and Learning in Anomaly Detection for Computer Security. AAAI Workshop: AI Approaches to Fraud Detection and Risk Management, pages 43-49. AAAI Press (1997)
9. Warrender, C., Forrest, S., Pearlmutter, B.: Detecting Intrusions Using Systems Calls: Alternative Data Models. IEEE Symposium on Security and Privacy. IEEE Computer Society (1999) 133-145
10. Milligan G.W., Clustering Validation: Results and Implications for Applied Analyses. World Scientific Publications, River Edge, NJ, USA (1996)
11. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. Advances in knowledge discovery and data mining. AAAI Press/ The MIT Press (1996)
12. http://www.cs.waikato.ac.nz/ml/weka
13. Michalski, R. S., Stepp, R. E.: Learning from Observation: Conceptual Clustering. In R. S. Michalski, J. G. Carbonell and T. M. Michell (Eds.), Machine Learning: An Artificial Intelligence Approach. Morgan Kauffmann, San Mateo, CA (1983) 331-363
14. Fisher, D. H. Knowledge Acquisition via Incremental Conceptual Clustering. Machine Learning (1987) 2:139-172
15. Witten I., Frank E., Generate Accurate Rule Sets Without Global Optimisation. Machine Learning: Proceedings of the 15th International Conference, Morgan Kaufmann Publishers, San Francisco, USA (1998)
16. Bock, H.H., Diday, E.: Symbolic Objects. In Bock, H.H, Diday, E. (eds.): Analysis of Symbolic Data. Exploratory Methods for extracting Statistical Information from Complex Data, Series: Studies in Classification, Data Analysis, and Knowledge Organisation, Vol. 15, Springer-Verlag, Berlin (2000) 54-77
17. Esposito, F. , Malerba, D., Tamma, V.: Dissimilarity Measures for Symbolic Objects. Chapter 8.3 in in H.-H. Bock and E. Diday (Eds.), Analysis of Symbolic Data. Exploratory methods for extracting statistical information from complex data, Series: Studies in Classification, Data Analysis, and Knowledge Organization, vol. 15, Springer-Verlag:Berlin, (2000), 165-185.
18. Gowda, K. C., Diday, E.: Symbolic Clustering Using a New Dissimilarity Measure. In Pattern Recognition, Vol. 24, No. 6 (1991) 567-578