International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163
Volume 1 Issue 10 (November 2014) www.ijirae.com

Intrusion Detection System Using Supervised Learning Vector Quantization

Ezat mahmoud soleiman
Dept of ICT Eng, Maleke Ashtar Univ. of Tech

Abdelhamid fetanat
Dept of ICT Eng, Maleke Ashtar Univ. of Tech

Abstract— Attacks on computer infrastructure are becoming an increasingly problem nowadays, and with the rapid expansion of computer networks during the past decade, computer security has become a crucial issue for protecting systems against threats, such as intrusions. Intrusion detection is an interesting approach that could be used to improve the security of network system. Different soft-computing based methods have been proposed in recent years for the development of intrusion detection systems. This paper presents a supervised Learning Vector Quantization artificial neural network to detect intrusion. A Supervised Learning Vector Quantization (LVQ) was trained for the intrusion detection system; it consists of two layers with two different transfer functions, competitive and linear. Competitive (hidden) and output layers contain a specific number of neurons which are the sub attack types and the main attack types respectively. The experiments and evaluations of the proposed method have been performed using the NSL-KDD 99 intrusion detection dataset.

Keywords— intrusion detection system,IDS, learning vector quantization, LVQ

I. INTRODUCTION

Learning Vector Quantization is a supervised classification scheme that was introduced by Kohonen in 1986 [1]. The approach still enjoys great popularity and numerous modifications of the original algorithm have been proposed. The classifier is parameterized in terms of a set of labelled prototypes which represent the classes in the input space in combination with a distance measure d (•, •). To label an unknown sample, the classifier performs a nearest prototype classification, i.e. the pattern is assigned to the class represented by the closest prototype with respect to d(•,•). Nearest prototype classification is closely related to the popular k-nearest neighbour classifier (k-NN) [2], but avoids the huge storage needs and computational effort of k-NN. LVQ is appealing for several reasons: The classifiers are sparse and define a clustering of the data distribution by means of the prototypes. Multi-class problems can be treated by LVQ without modifying the learning algorithm or the decision rule.

Similarly, missing values do not need to be replaced, but can simply be ignored for the comparison between prototypes and input data; given a training pattern with missing features, the prototype update only affects the known dimensions. Furthermore, unlike other neural classification schemes like the support vector machine or feed-forward networks, LVQ classifiers do not suffer from a black box character, but are intuitive. The prototypes reflect the characteristic class-specific attributes of the input samples. Hence, the models provide further insight into the nature of the data. The interpretability of the resulting model makes LVQ especially attractive for complex real life applications, e.g. in bioinformatics, image analysis, or satellite remote sensing [3],[4]. A collection of successful applications can also be found in Bibliography on the Self-Organizing Map (SOM) and Learning Vector Quantization (LVQ).

The first LVQ algorithm (LVQ1) to learn a set of prototypes from training data is based on heuristics and implements Hebbian learning steps. Kohonen addition- ally proposed optimized learning-rate LVQ (OLVQ1) and LVQ2.1, two alternative (heuristic) training schemes which aim at improving the algorithm with respect to faster convergence and better approximation of Bayesian decision boundaries. LVQ variants which are derived from an explicit cost function are especially attractive alternatives to the heuristic update schemes. Proposals for cost functions to train LVQ networks were introduced in Sato and Yamada (1996); Seo and Obermayer (2003); Seo et al. (2003). The extension of cost function based methods with respect to a larger number of adaptive parameters is especially easy to implement. Furthermore, mathematical analysis of these algorithms can be investigated based on the respective cost function [5]. In [6], it has been shown that LVQ aims at margin optimization, i.e. good generalization ability can be expected. A theoretical analysis of different LVQ algorithms in simplified model situations can also be found in [7] and [4].

II. MATRIX LEARNING IN LVQ

Discriminative vector quantization schemes such as learning vector quantization (LVQ) are very popular classification methods due to their intuitivity and robustness: they represent the classification by (usually few) prototypes which constitute typical representatives of the respective classes and, thus, allow a direct inspection of the given classifier. The use of Euclidean distance, for instance, corresponds to the implicit assumption of isotropic clusters.

Such models can only be successful if the data displays a Euclidean characteristic. This is particularly problematic for high-dimensional data where noise accumulates and disrupts the classification or heterogeneous data sets where different scaling and correlations of the dimensions can be observed. Thus, a more general metric structure would be beneficial in such cases.

The field of metric adaptation constitutes a very active research topic in various distance based approaches such as unsupervised or semi-supervised clustering and visualization [8] , k-nearest neighbour approaches[9] and learning vector quantization [10], [11] .

We will focus on matrix learning in LVQ schemes which accounts for pairwise correlations of features, i.e. a very general and flexible set of classifiers. On the one hand, we will investigate the behaviour of generalized matrix LVQ (GM-LVQ) in detail, a matrix adaptation scheme for GLVQ that is based on a heuristic, though intuitive cost function. On the other hand, we will develop matrix adaptation for RSLVQ, a statistical model for LVQ schemes, and thus we will arrive at a uniform statistical formulation for prototype and metric adaptation in discriminative prototype-based classifiers. We will introduce variants that adapt the matrix parameters globally based on the training set or locally for every given prototype or mixture component, respectively.

Interestingly, depending on the data, the methods show different characteristic behaviour with respect to prototype locations and learned metrics. Although the classification accuracy is in many cases comparable, they display quite different behaviour concerning their robustness with respect to parameter choices and the characteristics of the solutions. We will point out that these findings have consequences on the interpretability of the results. In all cases, however, matrix adaptation leads to an improvement of the classification accuracy, despite a largely increased number of free parameters.

III. INTRUSION DETECTION SYSTEM

An intrusion detection system (IDS) inspects all inbound and outbound network activity or computer system events and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. IDSs are producing many alerts each day that many of them are false positive alerts. Big amount of the false positive alerts crowd and cover true positive alerts from security experts. Also identifying true positive from false positives are time consuming and error prone therefore IDS alert management system are introduced to manage generated IDS alerts. IDSs can be used as active or passive.

In passive usage of IDS, it analyses traffics or events in offline mode but active IDSs work in online mode. To manage alerts concurrently with alerts generation, active alert management systems are used. Active alert management systems same work in online mode as active IDSs. These types of alert management systems should have little amount of alert analyse time to be used in online mode. Some of problems of IDS are: huge amount of generated alerts and high rate of false positive alert among generated alerts. Also most alert management system has low speed.

There are several ways to categorize an IDS according to the literature. Misuse detection vs. anomaly detection: in misuse detection, the IDS analyses the information it gathers and compares it to large databases of attack signatures. Essentially, the IDS looks for a specific attack that has already been documented. Like a virus detection system, misuse detection software is only as good as the database of attack signatures that it uses to compare packets against [12].

In anomaly detection, the system administrator defines the baseline, or normal, state of the networks traffic load, breakdown, protocol, and typical packet size. The anomaly detector monitors network segments to compare their state to the normal baseline and look for anomalies [13], [14], [15], [16] and [17].

Network-based vs. host-based systems: in a network-based system, or NIDS, the individual packets flowing through a network are analysed. The NIDS can detect malicious packets that are designed to be overlooked by firewalls simplistic filtering rules. In a host-based system, the IDS examines at the activity on each individual computer or host.

Passive system vs. reactive system: in a passive system, the IDS detects a potential security breach, logs the information and signals an alert. In a reactive system, the IDS responds to the suspicious activity by logging off a user or by reprogramming the firewall to block network traffic from the suspected malicious source.

Though they both relate to network security, an IDS differs from a firewall in that a firewall looks out for intrusions in order to stop them from happening. The firewall limits the access between networks in order to prevent intrusion and does not signal an attack from inside the network. An IDS evaluates a suspected intrusion once it has taken place and signals an alarm. An IDS also watches for attacks that originate from within a system.

An IDS needs only to detect threats and as such is placed out-of-band on the network infrastructure, meaning that it is not in the true real-time communication path between the sender and receiver of information. Rather, IDS solutions will often take advantage of a TAP or SPAN port to analyse a copy of the inline traffic stream (and thus ensuring that IDS does not affect inline network performance).

IDS was originally developed this way because at the time the depth of analysis required for intrusion detection could not be performed at a speed that could keep pace with components on the direct communications path of the network infrastructure [18] , [19].

As explained, the IDS is also a listen-only device. The IDS monitors traffic and reports its results to an administrator, but cannot automatically take action to prevent a detected exploit from taking over the system. Attackers are capable of exploiting vulnerabilities very quickly once they enter the network, rendering the IDS an inadequate deployment for prevention device.

IV. REQUIREMENTS

When building an IDS, a certain number of requirements must be fulfilled in order for the system to be efficient. Before listing the requirements of an IDS, it is necessary to introduce four important terms.

- False Positive (FP): represents the number of instances that are classified by the IDS as being anomalous when in fact they are legitimate.
- True Positive (TP): represents the number of instances that are classified by the IDS as being anomalous and that really are anomalous.
- False Negative (FN): represents the number of instances that are classified by the IDS as being legitimate when in fact, they are anomalous.
- True Negative (TN): represents the number of instances that are classified by the IDS as being legitimate and that really are legitimate.

FP and FN cause big problem for IDSs. Indeed, an FN represents an attack that was misclassified by the IDS as being a legitimate action. This is the worst possible outcome because it means that the IDS failed to detect a potentially harmful attack on the network that it was supposed to protect. FP seems to be a smaller problem, but in practice an IDS with a high FP rate will be as useless as an IDS with a high FN rate. For instance, if we consider a 10Gbit/s Ethernet network, the number of packets per second that an IDS should be able to handle vary between 812,740 and 14,880,960. If the IDS misclassifies one packet every second (or every 14,880,960 packets) as being anomalous when it is not, this means that the network administrator will have to deal with 86,400 false alerts at the end of the day. In this case, the IDS becomes a nuisance for the administrator who will most probably not use it any more even though the primary objective of the IDS was to help him.

An intrusion detection system has to full field the following requirements.

- Accuracy: Also referred as soundness, this property ensures that the IDS does not classify legitimate instances as anomalous. As mentioned above, the problem of false positives limits the use of IDS using anomaly detection in real-world applications.
- Performance: An IDS must be able to classify the traffic without adding a noticeable overload to the network. More details about training and testing times of an IDS are given in next section.
- Completeness: This property is the core of the IDS. It states that an IDS should be able to detect all intrusion attempts leading to a false negative rate equal to 0. In practice, this property is very hard to achieve because the IDS must be able to detect known attack as well as unseen ones.
- Fault Tolerance: An IDS must itself be resistant to attacks.
- Scalability: An IDS must be able to process the traffic of the network in real-time without dropping any packets because of a higher bandwidth than what the IDS can handle.

V. SUPERVISED LEARNING VS. UNSUPERVISED LEARNING

Machine learning algorithms can be divided into two major classes depending on their learning technique: supervised and unsupervised. Another interesting type of learning method which lays in-between the two main types is called semi-supervised learning because it uses techniques of both supervised and unsupervised learning. This learning paradigm has been studied intensively in the last few years, but not much in the field of IDS.

Supervised learning implies to obtain a training dataset in which every entry is labelled. A label indicates which class the example belongs to. For example, each entry in the KDD99 dataset is originally labeled with the type of attacks it belongs to or Normal when the example corresponds to a harmless packet. However, they could also be labelled with only two deferent labels: “attack “or “normal”. Using only two classes allow binary classifiers such as support vector machines (SVM) to learn from the dataset. Another option would be to label the examples according to the class of attack they belong to: DoS, Probe, R2L and U2R.

The idea of supervised algorithms is very similar to the process in which a child learns to recognize objects with the help of his parents. One of the parents shows him deferent objects and pointing at one of them tells the word corresponding to the object. The child keeps track of information such as the shape of the object, its colour, etc. These properties of the object are the variables contained in a dataset. A set of values of these properties represents an example in a dataset. The child then associates a label (the word told by his parent) to the corresponding example in his memory which can be seen as his personal “dataset”. [20], [8] , [21]. The UCI KDD Archive (1999), [22], [23] and [24]. The next time that the child will see an object similar to the ones in his “dataset", hopefully, he will be able to utter the word corresponding to the label of the set of examples found. A supervised machine learning algorithm works very similar to this analogy. The algorithm builds a model which should be able to separate the examples with different labels. For example, logistic regression can find decision boundaries to separate the data from different classes. There are many supervised learning algorithms such as artificial neural networks (which can also be unsupervised), support vector machines, decision tree, etc.

The most obvious disadvantage of supervised learning is the need for a labelled dataset. This is a big problem for the IDS research community because the only available dataset with labels is the KDD99 which was created in 1999. Since then, many new attacks have been developed and for this reason mainly, this dataset is considered sometimes obsolete. Nevertheless, it is still widely used because of its uniqueness and because useful information can be extracted from it.

In fact, obtaining data is cheap whereas obtaining labels for the data is very expensive both in terms of time and money, because one or more experts must go through millions of examples and assign them a label. Apart from this main drawback, supervised learning has also some advantages. [25], [26], [27].

The first one is the ease of use and interpretation of the results. Indeed, the output of the classifier belongs to one of the classes defined by the labels of the dataset. The second advantage of supervised learning is its accuracy to classify similar examples. However, this accuracy drops significantly when the new examples are not so similar to the ones in the training set.

Unsupervised learning algorithms do not need the dataset to be labelled. The most popular technique of unsupervised learning is called clustering. In this case, the algorithm exploits the similarity of the examples in order to form clusters or groups of instances. Examples belonging to the same cluster are assumed to share similar properties and belong to the same class. In contrast to supervised learning, disadvantages of unsupervised learning include the manual choice of the number of cluster that the algorithm must form, the low accuracy of the prediction and the fact that the meaning of each cluster must be interpreted to understand the output. However, unsupervised learning is more robust to big variations than supervised learning. This is a very important advantage that unsupervised learning has over supervised learning for the problem of intrusion detection because it means that unsupervised learning is able to generalize to new types of attacks much better than supervised learning. This property could be very useful to detect zero-day vulnerabilities [26], [27], [28], [29].

In conclusion, the lack of proper labelled datasets and the speed at which hackers invent new attacks could make unsupervised and semi-supervised learning good candidates for future development of IDSs. In fact, an IDS should be trained with traffic found on the network where it will be deployed in order to take into account the particular configuration of the network.

VI. LVQ NEURAL NETWORK ARCHITECTURE

As we said in previous section, learning vector quantization is a nearest neighbour pattern classifier based on competitive learning. Kohonen originally suggested it. The basic architecture of LVQ neural network is shown in Fig.1.

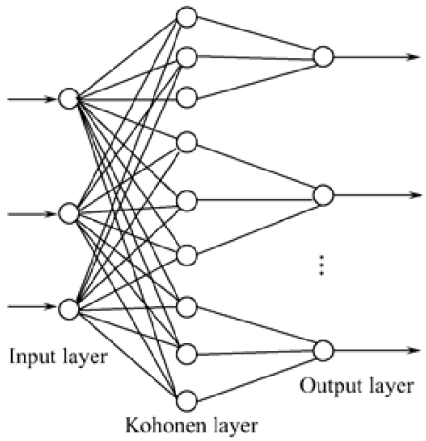

Fig.1 The architecture of LVQ neural network

In Fig.1, LVQ network contains an input layer, a Kohonen layer which learns and performs the classification, and an output layer. The input layer contains one node for each input feature; the Kohonen layer contains equal numbers of nodes for each class; in the output layer, each output node represents a particular class. Its main idea is to divide the input space Rn into a number of distinct regions, called decision regions, and for each region one reference vector is assigned. Classification is performed based on the vicinity of the input vector X to the reference vectors; X will be classified as the label of its nearest neighbour among reference vectors. During the training, the reference vectors and consequently the borders of decision regions are adjusted through an iterative process.

VII. OVERVIEW ON INTRUSION DETECTION BASED ON LVQ METHOD

The goal of intrusion detection is to build a system that would automatically scan network activity and detect all kinds of intrusion attacks. Once an attack is detected, the system administrator could be informed and thus take corrective action. The intrusion detection method based on LVQ algorithm is described as follow:

Feature Selection and Data Normalization Processing: In complex classification domains, some data may hinder the classification pro cess. Features may contain false correlations, which hinder the process of detecting intrusions. Further, some features may be redundant since the information they add is contained in other features.

Extra features can increase computation time, and can affect the accuracy of intrusion detection system. Feature selection improves classification by searching for the subset of features, which best classifies the training data. The subset of selected features is then used to detect intrusions.

Training Phase: In the training process of LVQ neural network, the instances, which are selected at random, are used as training examples. The Euclidean distance of a training vector to each node's weight vector in Kohonen layer is calculated. The nearest node is the winner whose weight vector is adjusted according to whether the winning node is in the class of training vector. After modifying the weight vector, it is moved towards or away from the input vector.

VIII. RESEARCH QUESTION

We detailed the Research Question from two filed, at first from Dataset View. As the research presented in this paper developed, it naturally formed three main parts: analysis of the

KDD cup '99 data set, ENNs for improved performance on imbalanced data, and multi-objective evolution of LVQ classifiers and classifier combinations. As the next view, we want to examine the best classification algorithm for IDS detection & Analysis using online LVQ.

-What has caused the contradictory findings with the KDD Cup '99 data set reported in the literature?

- Is the poor detection of some classes of intrusion due to issues with learning from imbalanced data?

- How can one utilise machine learning to better learn from imbalanced data?

- Can classifier combination be adopted to better learn from IDS data?

- How does the selection of base classifiers affect the performance of the resultant ensemble?

- How effect the selection of LVQ online classifiers affect the performance of the IDS?

IX. CONTRIBUTION AND NOVELTY

Ensemble approaches using bagging techniques are modular systems as discussed through this paper after the acceptance. A framework should be developed for ensemble approaches applied to intrusion detection. This would facilitate the replacement of any algorithm of the ensemble and ease the addition of more algorithms to each class of attacks. An object-oriented architecture exploiting heritage and polymorphism could be a good candidate for this kind of framework. Different machine learning algorithms could be added as modules to the framework and be represented in the program by an object initialized at run time. For example, all algorithms could inherit from the class Predictor. The number of algorithms as well as the type of each algorithm for each class of attacks could be determined on a user interface in an initialization step in proposed proposal. Additionally, the framework should exploit multicore processors in order to increase the speed of the computation. With the help of the previously described framework, it would be easy to experiment many combinations of algorithms for each class of attacks in order to build the most accurate system.

Another novelty would be to apply a signature-based detection system with a database of signatures of limited size first to alter known attacks. If an attack is detected in this first step, a response module should be triggered to stop the attack. If no attack is detected, the data would be transferred to the ensemble described in this paper. Finally, if the ensemble detects an attack, a signature generator could be triggered to update the signature database.

As introduced above, there are three empirical parts to this proposed proposal. These parts have made contributions to both the intrusion detection and machine learning domains. Although the focus of this proposal is on the application of machine learning to intrusion detection, several contributions have been made to the general machine-learning domain. The first part of the proposal makes the greatest contribution to the intrusion detection domain. First, identifying discrepancies in the findings reported in the literature. This has led to an empirical investigation of the KDD Cup '99 data set, which has uncovered several underlying causes of the discrepancies. Furthermore, an important contribution of this part of the proposal is a discussion of, and recommendations for, future research using this data set with an online LVQ that using our proposed method.

In addition, Learning from imbalanced data has been identified as one of the reasons for poor detection of certain classes of intrusion. This has not been considered an issue in this domain previously, and the empirical research conducted in the second part of the paper demonstrates how commonly adopted techniques such as ANNs and DTs perform poorly for this reason. An alternative approach to training ANNs has been proposed, which demonstrates that the issues with learning from imbalanced data are due to using accuracy or a general measure of error as a performance metric in the training process. Offering a different approach to training, this research demonstrates that ANNs are capable of learning from imbalanced data, and, therefore, detecting more intrusions.

The ensemble approach is a relatively new trend in artificial intelligence in which several machine learning algorithms are combined. The main idea is to exploit the strengths of each algorithm of the ensemble to obtain a robust classifier. Moreover, ensembles are particularly useful when a problem can be segmented into sub problems. In this case, each module of the ensemble, which can include one or more algorithms, is assigned to one particular sub problem. Network attacks can be divided into four classes: denial of service, user to root, remote to local and probe.

One module of the ensemble will be designed in this work is itself an ensemble of decision trees and is specialized on the detection of one class of attacks. We proposed the inner structure of each module uses bagging techniques to increase the accuracy of the IDS.

X. LITERATURE REVIEW

This paper provides a survey of the state-of-the-art in the field of ensemble approaches Applied to intrusion detection systems. LVQ based system are surveyed next. Additionally, this work has shown that each class of attacks should be treated separately. In fact, at least one algorithm should be assigned to detect one class of attacks instead of using a single algorithm to detect all classes of attacks.

For further information on the AI techniques discussed in this related work. Due to the scope of this review, there are techniques and approaches that are not considered here. However, that does not imply that they are not used in modern IDSs. Statistical techniques have been widely used (Biermann et al. 2001), particularly for anomaly detection, and still form parts of many hybrid IDSs. Applications of Bayesian networks are included here, but other model based approaches, and state based applications such as STATL and USTAT [12], are not. Pattern matching, particularly string matching, has also been successfully applied to this domain, with proposed algorithms such as ExB [30] and [31].

The following texts are suggested as complementary reading: [32] for a survey of intrusion detection and response, [13] for a survey of alert correlation, [14] for a survey of coordinated attacks and collaborative intrusion detection, [15] for a survey of anomaly detection techniques, [16] and [17] on AI applied to intrusion detection, in which the former focuses on knowledge representation and the latter on computational intelligence.

One of the most common forms of Rule Based Systems (RBSs) that have been applied to intrusion detection is expert systems [33. In the last decade, fuzzy logic and rule induction has increasingly been applied to intrusion detection. Although most RBSs do not facilitate effective anomaly detection. There is a range of event correlation tools created with rule based systems, all of which operate similarly. The different tools have been somewhat specialised for different environments, allowing different types of rules.

One tool that has been around for approximately two decades is the Production-Based Expert System Toolset (P-BEST), which has been integrated into several IDSs with a focus on handling SYN flooding and buffer overruns [34] briefly describe four IDSs that P-BEST have been used in: MIDAS, IDES, NIDES and EMERALD eXpert, and in [35], a fifth, eXpert-BSM; all systems being suitable for real-time misuse detection. The first three systems are host based, whilst the latter two have achieved support for distributed networks. Although eXpert-BSM was developed to analyse Sun Solaris audit trials on a host, it can be distributed by employing an alert collection application referred to as an eftunnel, which will produce a single event stream. [34] Highlight some drawbacks of P-BEST, such as being poor at dealing with uncertainty and missing data due to being strictly forward chaining.

XI. REFERENCES

[1] Kohonen, T.: 1998, Learning vector quantization, The handbook of brain theory and neural networks, MIT Press, Cambridge, MA, USA, pp. 537–540.

[2] Cover, T. and Hart, P.: 1967, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13(1), 21–27.

[3] Biehl, M., Pasma, P., Pijl, M. and Petkov, N.: 2006, Classification of boar sperm head images using learning vector quantization, in M. Verleysen (ed.), European Symposium on Artificial Neural Networks, Bruges, Belgium, pp. 545–550.

[4] Biehl, M., Ghosh, A. and Hammer, B.: 2007, Dynamics and generalization ability of LVQ algorithms, Journal of Machine Learning Research 8, 323–360.

[5] Sato, A. and Yamada, K.: 1998, An analysis of convergence in generalized lvq, in L. Niklasson, M. Bod´en and T. Ziemke (eds), Proceedings of the International Conference on Artificial Neural Networks, Springer, pp. 170–176.

[6] Crammer, K., Gilad-Bachrach, R., Navot, A. and Tishby, A.: 2003, Margin analysis of the lvq algorithm, Advances in Neural Information Processing Systems, Vol. 15, MIT Press, Cambridge, MA, USA, pp. 462–469.

[7] Ghosh, A., Biehl, M. and Hammer, B.: 2006, Performance analysis of lvq algorithms: a statistical physics approach, Neural Networks 19(6), 817–829.

[8] Arnonkijpanich, B., Hammer, B., Hasenfuss, A. and Lursinap, A.: 2008, Matrix learning for topographic neural maps, International Conference on Artificial Neural Networks, Prague, Czech Republic, pp. 572–582.

[9] Strickert, M., Witzel, K., Mock, H.-P., Schleif, F.-M. and Villmann, T.: 2007, Supervised attribute relevance determination for protein identification in stress experiments, Machine Learning in Systems Biology, pp. 81–86.

[10] Hammer, B. and Villmann, T.: 2002, Generalized relevance learning vector quantization, Neural Networks 15(8-9), 1059–1068.

[11] Schneider, P., Biehl, M. and Hammer, B.: 2009a, Adaptive relevance matrices in learning vector quantization, Neural Computation 21(12), 3532–3561.

[12] Ilgun. USTAT: A Real-Time Intrusion Detection System for UNIX. In SP '93: Proceedings of the 1993 IEEE Symposium on Security and Privacy, page 16, Washington, DC, USA, 1993. IEEE Computer Society.

[13] Sadoddin and A. Ghorbani. Alert correlation survey: framework and techniques. In PST '06: Proceedings of the 2006 International Conference on Privacy, Security and Trust, pages 1–10, New York
[14] Zhou, C. Leckie and S. Karunasekera. 2010. A survey of coordinated attacks and collaborative intrusion detection. Computers & Security, 29, 124–140., NY, USA, 2006. ACM. ISBN 1-59593-604-1.
[15] Patcha and J-M. Park. 2007. An overview of anomaly detection techniques: Existing solutions and latest technological trends. Comput. Netw., 51, 3448–3470. ISSN 1389-1286.
[16] Gagnon and B. Esfandiari. Using Artificial Intelligence for Intrusion Detection. In Proceeding of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering, pages 295–306, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press. ISBN 978-1-58603-780-2.
[17] Wu and E. Yen. 2009. Data mining-based intrusion detectors. Expert Syst. Appl., 36, 5605–5612. ISSN 0957-4174.
[18] Vaarandi. SEC - A Lightweight Event Correlation Tool. In Proceedings of the IEEE Workshop on IP Operations & Management, pages 111–115, 2002.
[19] Tillapart, T. Thumthawatworn and P. Santiprabhob. Oct 2002. Fuzzy Intrusion Detection System. AU J.T., 6, 109–114.
[20] Tajbakhsh, M. Rahmati and A. Mirzaei. 2009. Intrusion detection using fuzzy association rules. Appl. Soft Compute., 9, 462–469. ISSN 1568-4946.
[21] Florez, S.M. Bridges and R.B. Vaughn. An Improved Algorithm for Fuzzy Data Mining for Intrusion Detection. In NAFIPS. 2002 Annual Meeting of the North American Fuzzy Information Processing Society, 2002.
[22] Ghosh and A. Schwartzbard. A Study in Using Neural Networks for Anomaly and Misuse Detection. In Proceedings of the 8th USENIX Security Symposium, 1999.
[23] Jeya and K. Ramar. 2007. Rule Based Network Intrusion Detection System Based on Crossover and Mutation. Asian Journal of Information Technology, 6, 896–901.
[24] Selvakani and R.S. Rajesh. 2007. Genetic algorithm for framing rules for intrusion detection. IJCSNS International Journal of Computer Science and Network Security, 7, 285–290.
[25] Goss, M. Botha and R. von Solms. Utilizing fuzzy logic and neural networks for effective, preventative intrusion detection in a wireless environment. In SAICSIT '07: Proceedings of the 2007 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries, pages 29–35, New York, NY, USA, 2007. ACM. ISBN 978-159593-775-9.
[26] Yuan and D. Guanzhong. 2007. An Intrusion Detection Expert System with Fact-Base. Asian Journal of Information Technology, 6, 614–617.
[27] Depren, M. Topallar, E. Anarim and M.K. Ciliz. 2005. An Intelligent Intrusion Detection System for Anomaly and Misuse Detection in Computer Networks. Expert systems with Applications, 29, 713–722.
[28] Zhou, M. Heckman, B. Reynolds, A. Carlson and M. Bishop. 2007. Modeling network intrusion detection alerts for correlation. ACM Trans. Inf. Syst. Secure., 10, 4. ISSN 1094-9224.
[29] Hanemann. A Hybrid Rule-Based/Case-Based Reasoning Approach for Service Fault Diagnosis. In Proceedings of the 2006 International Symposium on Frontiers in Networking with Applications (FINA 2006), Vienna, Austria, 2006.
[30] Markatos, S. Antonatos, M. Polychronakis and K.G. Anagnostakis. ExB: Exclusion Based signature Matching for Intrusion Detection. In Proceedings of Communications and Computer Networks, MIT, USA, 2002.
[31] Antonatos, M. Polychronakis, P. Akritidis, D. Kostas, K. G. Anagnostakis and E. P. Markatos. Piranha: Fast and Memory Efficient Pattern Matching for Intrusion Detection. In Proceedings of the 20th International Information Security Conference (IFIP/SEC 2005), May 2005.
[32] Kabiri and A.A. Ghorbani. September 2005. Research on Intrusion Detection and Response: A Survey. International Journal of Network Security, 1, 84–102.
[33] Cannady. Artificial Neural Networks for Misuse Detection. In Proceedings of the National Information Systems Security Conference. NIST, Washington, DC, 1998.
[34] Lindqvist and P. A. Porras. Detecting Computer and Network Misuse Through the Production-Based Expert System Toolset (P-BEST). In IEEE Symposium on Security and Privacy, pages 146–161. IEEE Computer Society Press, 1999.
[35] Lindqvist and P. A. Porras. eXpert-BSM: A Host Based Intrusion Detection Solution for Sun Solaris. In Proceedings of the 17th Annual computer Security Applications Conference, pages 240–251, New Orleans, USA, Dec 2001.

