Application of Relevance Vector Machines in Real Time Intrusion Detection

Naveen N.C,
Associate Professor, Department of ISE,
R V College of Engineering, Bangalore
and Research Scholar, Dept of CSE,
SRM University, Chennai, India

Dr Natarajan.S,
Professor,
Department of ISE P E S I T,
Bangalore, India

Dr Srinivasan.R,
Professor Emeritus,
Department of Computer Science and Engineering,
M S R I T, Bangalore, India

Abstract*— In the recent years, there has been a growing interest in the development of change detection techniques for the analysis of Intrusion Detection. This interest stems from the wide range of applications in which change detection methods can be used. Detecting the changes by observing data collected at different times is one of the most important applications of network security because they can provide analysis of short interval on global scale. Research in exploring change detection techniques for medium/high network data can be found for the new generation of very high resolution data. The advent of these technologies has greatly increased the ability to monitor and resolve the details of changes and makes it possible to analyze. At the same time, they present a new challenge over other technologies in that a relatively large amount of data must be analyzed and corrected for registration and classification errors to identify frequently changing trend. In this research paper an approach for Intrusion Detection System (IDS) which embeds a Change Detection Algorithm with Relevance Vector Machine (RVM) is proposed. IDS are considered as a complex task that handles a huge amount of network related data with different parameters. Current research work has proved that kernel learning based methods are very effective in addressing these problems. In contrast to Support Vector Machines (SVM), the RVM provides a probabilistic output while preserving the accuracy. The focus of this paper is to model RVM that can work with large network data set in a real environment and develop RVM classifier for IDS. The new model consists of Change Point (CP) and RVM which is competitive in processing time and improve the classification performance compared to other known classification model like SVM. The goal is to make the system simple but efficient in detecting network intrusion in an actual real time environment. Results show that the model learns more effectively, automatically adjust to the changes and adjust the threshold while minimizing the false alarm rate with timely detection.

Keywords- Intrusion Detection; Change Point Detection; Relevance Vector Machine; Outlier Detection.

I. INTRODUCTION

Reasonable level of security is provided by static defense mechanisms such as firewalls and software updates. Dynamic mechanisms can also be used to achieve security such as IDS and Network Analyzers (NA). The main difference between IDS and NA is that IDS aims to achieve the specific goal of detecting attacks whereas NA aims to determine the changing trends in network of computers [1] [18]. Earlier work emphasized that data can be obtained by three ways using real traffic, sanitized traffic and simulated traffic. But in real time fast response with reduced false positives to external events within an extremely short time is demanded and expected [19]. Therefore, an alternative algorithm to implement real time learning is imperative for critical applications for fast changing environments. Even for offline applications, speed is still a need, and a real time learning algorithm that reduces training time and human effort to nearly zero would always be of considerable value. Mining data in real time is still a big challenge [21].

IDS involve automatic identification of unusual activity by collecting data, and comparing it with reference data. An assumption of IDS is that a network's normal behavior is distinct from abnormal or intrusive behavior, which can be a result of various attack/s. In this research work, flow analysis is used for network traffic analysis which searches for behavioral characteristics in a flow. There are various characteristics such as transferred bytes, packets, flow length, inter-arrival times, inter-packet gaps, etc that are monitored and computed. Data that is collected from flows can be used on high-speed networks as there is no deep packet inspection.

In this paper a hybrid approach for improving the performance of detection algorithm by building more intelligence to the system is proposed. In this direction CP detection is considered for discovering change points if properties of network behavior change. CP is the change in characteristics that occur very fast with respect to the sampling period of the measurements, if not instantaneously. The detection of changes refers to tools that help to decide whether such a change has occurred in the characteristics or not. Outlier Detection is a major step in Data Mining (DM) problem which discovers abnormal or deviating data points with respect to distribution in data [20]. Outliers are often considered as an error or noise although they may carry very important information.

A real time detection system is one in which network intrusion detection happens while an attack is occurring. A real time IDS captures the present network traffic data which is on line data. Bayesian learning algorithms, like RVM allow the user to specify a probability distribution over possible parameter values from the learned classifier. This will provide one solution to the over fitting problem as the algorithm can use prior distribution to regularize the classifier.

II. Related Work

Research shows that many Machine Learning (ML) techniques can be used for data classification. It is presented that popular supervised learning techniques gives high detection accuracy for IDS. A wide range of real world applications are discussed in the community of Statistical Analysis and DM [3] [13]. Statistical techniques usually assume an underlying distribution of data and require the elimination of data instances containing noise. Statistical methods though computationally intense can be applied to analyze the data [4]. Statistical methods are widely used to build behavior based IDS. The behavior of the system is measured by a number of variables sampled over time such as the resource usage duration, the number of processors, memory disk resources consumed during that session etc. The model keeps averages of all the variables and detects whether thresholds are exceeded based on the standard deviation of the variable. Very few on line (real time) network IDS approaches are proposed until now. Liao and Vemuri [5] develop a real time IDS using Self Organizing Maps (SOM) and preprocess their dataset with 10 features for each data record containing information of 50 packets. M. Al-Subaie [6] uses Hidden Markov Models over Neural Networks in anomaly intrusion detection to classify normal network activity and attack using a large training dataset. The approach was evaluated by analyzing how it affected the classification results. Ben Amor [7] designs a real time IDS using Naïve Bayes and Decision Trees and the results show that the Naive Bayes gives higher detection speed and detection rate than the Decision Trees. Authors in [8] [14] propose a hybrid intelligent systems using Decision Trees (DT), SVM and Fuzzy SVM for anomaly detection (unknown or new attacks). The results show that the hybrid DT–SVM approach improves the performance for all the classes when compared to a SVM approach.

SVM proposed by (Burges 1998, Cortes and Vapnik 1995) is a supervised learning algorithm that is used increasingly in IDS. The classification performance of SVM model is better than the classification methods, such as ANN [9]. The benefit of SVMs is that they learn very effectively with high dimensional data. Rung Ching Chen [10] uses Rough Set Theory (RST) and SVM to detect intrusions. Initially, RST is used to preprocess the data and reduce the dimensions. Later, the features selected by RST are sent to SVM model to learn and test. This method proves to be effective and also decreased the space density of data. The SVM is one of the most successful classification algorithms in the DM area [17].

RVM, proposed by Tipping [11] is a sparse machine learning algorithm that is similar to the SVM in many respects. It is capable of delivering a fully probabilistic output and it is proved to have nearly identical performance to, if not better than, that of SVM in several benchmarks. Di He [12] proposes an IDS approach based on the RVM where a Chebyshev chaotic map is introduced as the inner training noise signal. The result shows that the approach can reach higher detection probabilities under different kinds of intrusions and the computational complexity reduces efficiently. Li Rui [16] improves the generalization performance of RVM by an incremental relevance vector machine algorithm and the results are better than RVM and SVM. This guarantees the reliability of using RVM based approach for designing IDS. RVM has a better generalization performance than SVM due to the less support vectors.

III. Methodology

Over 90% of Internet traffic uses the Transmission Control Protocol (TCP). Because of its widespread use and its impressive growth, the research focuses on the detection of anomalous behavior within TCP traffic. Exploring the TCP packet attributes would enable a classifier to identify normal and abnormal activity on a packet-by-packet basis. From these attributes, a decision tree is built which will enable to identify and classify different attacks and violations. The process of building a classifier model using RVM is depicted in Fig. 1.

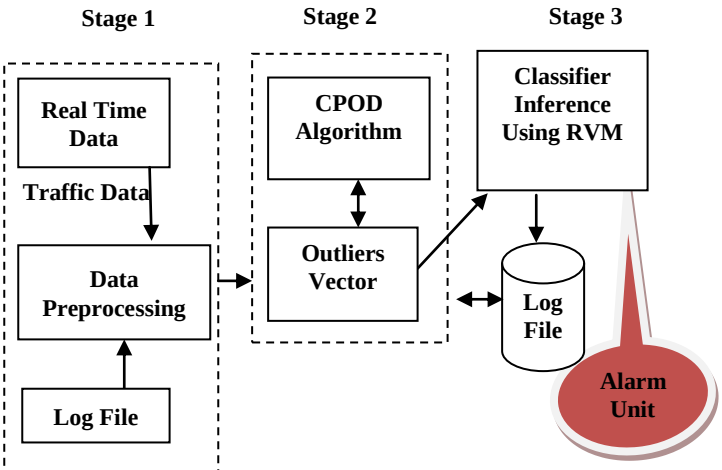


Figure 1. Architecture of the model

A. Dataset Description

The first stage of the implementation involves in training the system. For the present problem data is collected from the campus network to measure the accuracy and attacks. Data such collected is preprocessed and used to detect change point in network performance characteristics. Traffic in the network results in continuous change as the user's login and make use of Internet. For capturing the packets in real time JPCAP and WINPCAP tool is used to collect the information that is being transmitted. JPCAP provides facilities to capture and save raw packets live. It can automatically identify packet types and can generate corresponding Java objects for Ethernet, IPv4, IPv6, ARP/RARP, TCP, UDP, and ICMPv4 packets. Packets can also be filtered according to the user requirement. JPCAP is developed on LIBPCAP / WINPCAP, which is implemented in C and Java and is the industry-standard tool for link-layer network access. In Windows environment WINPCAP allows applications to capture and transmit network packets bypassing the protocol stack. The network data is collected from the interface which is capable of capturing information flowing within the local network. For example, anomalies can be detected on a single machine, a group of network a switch or a router. For the current research work the TCP/IP packet is collected in real time from the research lab network and dumped for further process. The Data Preprocessing phase handles the conversion of raw packet or connection data into a

format that algorithms can utilize and store the results in the knowledge base. Rather than operating on a raw network dump file, the algorithm uses summary information to perform the analysis. Data is preprocessed to generate summary lines about each connection found in the dump file. The resulting summary file is then parsed and processed by the algorithm to give a count to each data/each time point, with a higher score indicating a high possibility of being an outlier/a change point.

The Log File stores the data as rules produced by the detection algorithm for further mining process. It may also hold information for the preprocessor, such as patterns for recognizing attacks and conversion templates. This Training Data is responsible for generating the initial rule sets that needs to be used for deviation analysis. It can be triggered automatically based on time or the amount of pre-processed data available.

The proposed Outlier Detection Algorithm examines the network data and creates a description of differences and stores in the outlier vectors for further reference. If a deviation is detected it signals the alarm unit. A strategy for invoking the deviation analyzer is by querying periodically the outlier vectors for the new profiles. Also the profiler may signal when a new profile is added and the Alarm Unit is responsible for informing the administrator when the deviation analyzer reports unusual behavior in the network stream. This can be in the form of SMS, e-mails, console alerts, log entries etc.

In the data preprocessing step as shown in Figure 1, packets are captured using JPCAP library and information is extracted that includes IP header, TCP header, UDP header, and ICMP header from each packet. After that, the packet information is partitioned and formed into a record by aggregating information every 30 minutes. Each record consists of data features considered as the key signature features representing the main characteristics of network data and activities.

IV. Proposed Algorithm

A. Change Point Outlier Detection (CPOD):

To illustrate the problem, network data collected is observed with threshold values in the college local area network at regular intervals of time window for a certain period of time. Most of the threshold variations may be statistically regular, but once a while there may be an outlier point i.e. a marked deviation from the previous data. In general detecting such outliers is important because they may be caused by an anomaly within the network or from the external environment.

B. CP Distribution

Since the CP can occur randomly and at different times the current model assumes that the number of CP m is fixed and treated as an unknown random variable.

If there are n observations which are denoted by $o_1, o_2, \ldots, o_n$, and a set of CP denoted by $0 < c_1 < c_2 < \cdots < c_m < n$, where the number of CP m is unknown it is assumed that the CP occur at discrete time, $1, \ldots, n-1$. From this a prior distribution of the CP is considered.

C. Sampling stage for CP

1. At time n the algorithm starts
2. N data collected is sampled for time t without attack
3. Within this data sample if any CP is detected it is saved as a training data with parameter W_i
4. These values are updated to the training database

This algorithm has a computational cost of O(n). The following are the requirements of CPOD algorithm. The statistics should be on line meaning an outlier has to be detected as it appears. A change point has to be detected within some constant number of observations after the change happens. Specific assumptions regarding distributions are not done and hence the detection can be adaptive to a non-stationary time series and robust to a wide variety of distributions.

D. The CPOD Algorithm

We denote a data sequence as $\{x_i : i = 1, 2, \ldots N\}$, i is the time variable, p denotes the number of data points to be observed before initiating analysis. Cx_i denotes the current data point in the time series being considered for analysis. t and s denotes the thresholds for change point and outlier detection respectively. Threshold value is the mean magnitude of fluctuation allowed in the data points within which they won't be classified. Window size w, v denotes the number of data points to consider for computing the median and meaning respectively. The algorithm chooses median over a small window, as it is less sensitive to outliers and helps in localizing the deviation. $w < v$ in all cases, $(w/v) < 0.5$ is found optimal. We maintain a vector to signify the classification state of each data point. States could be {0,1,2,3} where '0' means neither outlier nor change point, '1' means outlier, '2' means outlier with high probability that previous point was the change point, and '3' means the change point done from previous two observations.

If the thresholds are well tuned, CPOD can detect maximum outliers and change points in a time series.

CPOD Algorithm (Input : p,s,t,w,v)

Step 1: The iteration for all data points is done after initializing i=p+1

$$\forall\ (i > p | p < w < v < (N - i))$$

Step 2: The median and mean over the windows v, w is computed respectively as in (1) and (2)

$$C\,\tilde{x}_i = \tilde{x}_{i-1}^{\,i-w} \qquad (1)$$

Mean of values in window 'v'

$$C\,\bar{x}_i = \bar{x}_{i-1}^{\,i-v} \qquad (2)$$

Step 3: Score1 is the ratio of absolute difference between the current data point from the median to the mean amplified by the threshold and calculated as

$$Score_{1i} = (|Cx_i - C\,\tilde{x}_i| * t) / C\,\bar{x}_i \text{ and} \quad (3)$$

Score 2 is the normalized ratio of two distance magnitudes i.e. the median from mean and the mean from the current data point and calculated as

$$Score_{2i} = (|C\,\tilde{x}_i - C\,\bar{x}_i|)/ (|C\,\bar{x}_i - C\,\tilde{x}_i|) * 100 \quad (4)$$

The median is over the short term window w and mean over the longer term window v. By taking ratio, makes the score robust to fluctuations that may happen in the mean over a long term. We heuristically found w/v < 0.5 to be the best choice. The resolution of detection is dependent on the threshold.

If Score1 > Score2, we classify the point as an outlier. Score2 is used as a data-dependent cutoff to classify Score1 as outlier or not. Score1 is sensitive to mean and to the deviation of current data point from median. Score2 is sensitive to deviation of current point from mean and to deviation of mean from median. In window v, Score1 will be greater than Score2, with the presence of outliers or with data points subsequent to a change point.

Step 4: A stronger possibility of current data point being an outlier or CP is indicated if Score1 is higher than Score2. To classify the current data point as CP an additional check is made to find if the point lies beyond a certain band around the median represented as LL_i and UL_i. The classification state is then saved in vector V.

$$\begin{cases} Score_{1i} > Score_{2i} \wedge (Cx_i < LL_i \vee Cx_i > UL_i) : V_i = 1 \\ \leq Score_{2i} \vee (LL_i > Cx_i > UL_i) : V_i = 0 \end{cases} \quad (5)$$

Step 5: State information in vector V is used to classify outlier and CP. If the current point has a higher Score1 as indicated in V_i the past three states are considered for classification. Vector V_{si} is used to express the sum state of V_i,V_{i-1} and V_{i-2}. V_{si} stores state of the current data point with respect to past two data points. If the value of V_{si} is 3 it indicates that outliers were detected in the past and current point could be the change point. We test the previous states to make sure there was no change point detected in the past two data points. If detected then the CP is inferred as an outlier. Similarly if the value of V_{si} is 1 then it is possible that current point is an outlier as shown in (6) and (7).

$$\mathbf{V}_i \begin{cases} = 1 \;:\; \mathbf{V_{si} = (V_i + V_{i-1} + V_{i-2})} \\ = 0 \;:\; \mathbf{V_{si} = 0} \end{cases} \quad \mathbf{(6)}$$

$$\mathbf{V_{si}} \begin{cases} = 3 \wedge (\mathbf{V_{s_{i-1}}} = 3 \vee \mathbf{V_{s_{i-2}}} = 3) \;:\; \mathbf{V_{s_i} = 1} \\ = 1 \wedge (\mathbf{V_{s_{i-1}}} = 1 \vee \mathbf{V_{s_{i-2}}} = 1) \;:\; \mathbf{V_{s_i} = V_{s_i} + V_{s_i} - 1} \end{cases} \quad \mathbf{(7)}$$

Finally Cx_i is classified depending on the values of V_{si}. If the state of current point is 0, then there is no significant deviation. If the current point is 1 or 2 and the current data point deviates more than s% threshold it is inferred as an outlier and signifies a higher possibility of Cx_i - 1 being a CP. If the state of current point is 3, with accuracy it is inferred that two points prior to current one is the CP.

$$\mathbf{V_{si}} \begin{cases} = 0 \text{ : No change , adapt LL, UL to } C\,\tilde{x}_i \\ = (1 \vee 2) \wedge (\mathbf{Cx}_i > (\mathbf{Cx}_i + s\,\%\,\mathbf{Cx}_i)) \text{ : Outlier} \\ > 2 \text{ : Change point, adapt LL, UL to change} \end{cases} \quad \mathbf{(8)}$$

Step 6: The classification of outliers and change points as signified in the V_{si} vector is reported. The scores and state elements of vector V, V_s for past data points N, N-1 and N-2 are persisted. Persistence of median and mean over the window sizes while classifying current data point enables online implementation.

Table I List of Network Dataset Features Collected

1. appName	10. direction
2. totalSourceBytes	11. sourceTCPFlagsDescription
3. totalDestinationBytes	12. destinationTCPFlagsDescription
4. totalDestinationPackets	13. source
5. totalSourcePackets	14. protocolName
6. sourcePayloadAsBase64	15. sourcePort
7. sourcePayloadAsUTF	16. destination
8. destinationPayloadAsBase6	17. destinationPort
9. destinationPayloadAsUTF	18. startDateTime
	19. stopDateTime

RVM is currently of much interest in the research community as they provide a number of advantages. RVM is based on a Bayesian formulation of a linear model with an appropriate prior that results in a sparse data representation. As a result, they can generalize well and provide inferences at very low computational cost. Many applications like object detection and classification, target detection in images, classification of micro calcifications from mammograms etc are developed. RVM produces a function which is comprised of a set of kernel functions also known as basis functions and a set of weights. This function represents a model for the system presented to the learning process from a set of training data set. The kernels and weights calculated by the learning process and the model function defined by the weighted sum of kernels are fixed. From this set of training vectors the RVM selects a sparse subset of input vectors which are deemed to be relevant by the probabilistic learning scheme. This is used for building a function that estimates the output of the system from the inputs. These relevant vectors are used to form the basis functions and comprise the model function.

In the classification phase each of the network data selected from the feature selection phase is classified as normal data or attack data. This phase consists of two main dataset which are used for training and testing. The log file contains four weeks of training data and one week of testing data. A total of 19 features are captured as listed in Table I. During the first phase training is performed using RVM with a set of network records with known answer classes. Based on the training the IDS model can

classify the data in each record into normal network activity or main attack types. Then the model is tested with new or untrained dataset where each record was captured in a real time environment in the college research lab.

For an input vector x, an RVM classifier models the Probability distribution of its class labeled C ε (1, +1} using logistic regression as

$$p\left((C=1\middle|x)=\frac{1}{1+\exp(-fRVM(x))}\right) \quad \textbf{(9)}$$

where $f_{RVM}(x)$ the classifier function is given by,

$$f_{RVM}(x) = \sum_{i=1}^{N} \alpha_i K(x,x_i) \quad \textbf{(10)}$$

where K(.,.) is a kernel function, and x_i, i = 1,2,...,N, are training samples. The parameters α_i, i 1, 2, ..., N, in $f_{RVM}(x)$ are determined using Bayesian estimation, introducing a sparse prior on α_i The parameters α_i are assumed to be statistically independent obeying a zero-mean Gaussian distribution with variance λ_i^{-1}, used to force them to be highly concentrated around zero, leading to very few nonzero terms in $f_{RVM}(x)$.

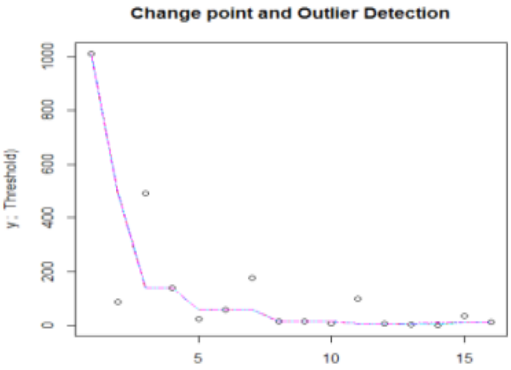
V. Experimental Results

Sample statistics of IP addresses and their count observed in our research laboratory for a time period of 30 minutes specified as window 'w_1', 'w_2', 'w_3'.

Table II Sample statistics of IP addresses

Source IP Address	*Time Window 'w_1'*	*Time Window 'w_2'*	*Time Window 'w_3'*
172.16.30.28	1010	1011	2000
172.16.30.91	86	86	90
172.16.30.75	415	492	512
172.16.30.108	140	140	140
172.16.30.70	24	24	24
172.16.30.92	58	58	58
172.16.30.68	175	175	179
172.16.30.69	14	14	14
172.16.30.96	14	14	14
172.16.30.35	7	7	7
172.16.30.95	100	100	100
172.16.30.101	11	12	12

For the real time statistics the data was collected for one week and the graph is as shown in Figure 2 and 3.



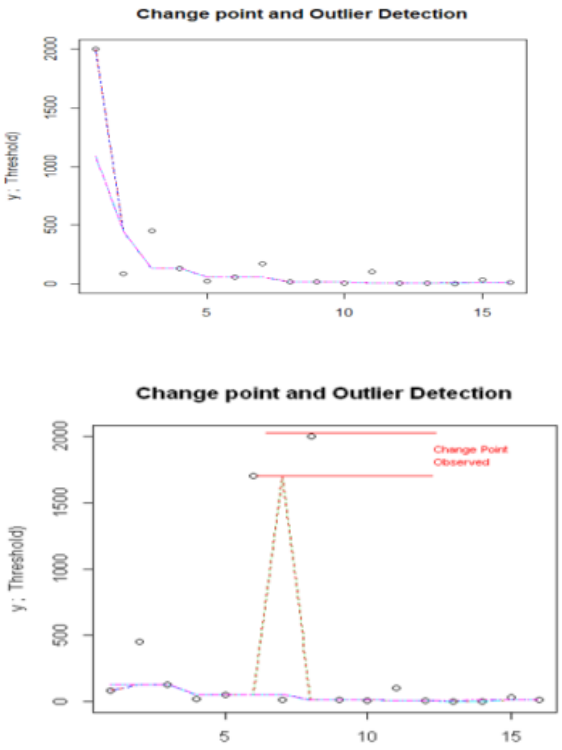
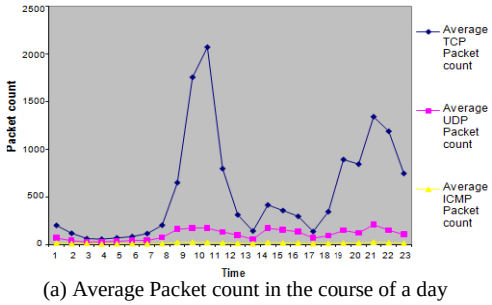
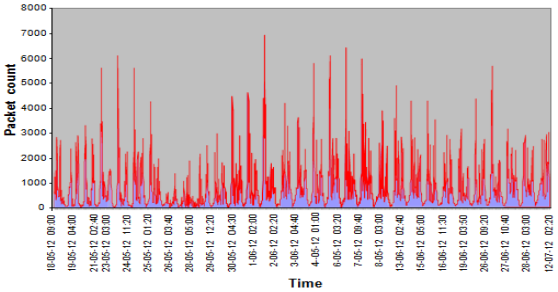


Figure 2. Plot of real time data and detection of change point and outliers



(a) Average Packet count in the course of a day



(b) Traffic Statistics in the course of a month

Figure 3. Plot of real time data collected for 2 weeks

Table III Comparison between RVM and SVM models

Model	*Number of training data*	*Number of vectors*	*Testing Performance*
SVM	100	25	0.71
	500	129	0.72
	1000	230	0.81
	5000	540	0.82

RVM	100	17	0.76
	500	109	0.81
	1000	170	0.86
	5000	240	0.88

Table III shows the comparison between the SVM and RVM models. The value of testing performance from RVM model is effectively same as that of SVM with lesser support vectors. The performance of the RVM is also better than the SVM.

VI. CONCLUSION

In this research work a solution for classifying outliers and change points from real time data is proposed which is addressed in two parts: scoring and classification. Scores are computed that reflect outliers and incrementally discover to keep the state of outliers in data series. The algorithm is characterized in its property to address outliers and change points at the same time. This enables to deal with frequent and fast changes in the source. The current implementation and usage indicates the success of the algorithm. This gives a unifying view of outlier detection and change point detection in real time network data.

Usage of RVM shows a competitive accuracy maintaining its sparseness ability. Experimental results show that the RVM model achieves essentially the same performance with a much sparser model as a previously developed SVM model. The much reduced computational complexity in RVM makes it more feasible for real time processing while designing IDS. The proposed method is competitive with respect to processing time and allows the use of selected training data set. The result shows an improvement in RVM classification performance. In this work, the design and successful implementation of a system with outlier detection was done.

REFERENCES

[1] Olin Hyde, Machine Learning For Cyber Security at Network Speed & Scale, 1st Public Edition: October 11, 2011

[2] Iftikhar Ahmad, Azween Abdullah and Abdullah Alghamdi, Towards the Selection of Best Neural Network System for Intrusion Detection, International Journal of the Physical Sciences Vol. 5(12), October, 2010 , pp. 1830-1839

[3] Fabio Pacifici, Change Detection Algorithms: State of the Art, v1.2, Earth Observation Laboratory, Tor Vergata University, Rome, Italy, Feb 28, 2007

[4] G. Mohammed Nazer, A. Arul Lawrence Selvakumar, Current Intrusion Detection Techniques in Information, European Journal of Scientific Research, EuroJournals Publishing, Inc. 2011, pp. 611-624

[5] Liao Y, Vemuri VR. Use of K-nearest Neighbor Classifier for Intrusion Detection. Computers & Security 2002;21: 439–48.

[6] M. Al-Subaie and M. Zulkernine. Efficacy of Hidden Markov Models Over Neural Networks in Anomaly Intrusion Detection. In 30th Annual International Computer Software and Applications Conference (COMPSAC'06), pages 325–332, 2006.

[7] N. Ben Amor, S. Benferhat and Z. Elouedi. Naive Bayes vs Decision Trees in Intrusion Detection Systems. In SAC '04: Proceedings of the 2004 ACM symposium on Applied computing, pages 420–424, New York, NY, USA, 2004. ACM. ISBN 1-58113-812-1.

[8] Sandhya Peddabachigari,Ajith Abraham, Crina Grosanc,Johnson Thomas, Oklahoma State University, Modeling Intrusion Detection System Using Hybrid Intelligent Systems, Journal of Network and Computer Applications, Elsevier Ltd, 2005

[9] Zhiqiang ZHANG, Jianzhong CUI, Network Intrusion Detection Based on Robust Wavelet RVM Algorithm, Journal of Information & Computational Science, 2011, pp. 2983–2989

[10] Rung-Ching Chen, Kai-Fan Cheng, Chia-Fen Hsieh, Using Rough Set and Support Vector Machine for Network Intrusion Detect**,** International Journal of Network Security & Its Applications (IJNSA), Vol 1, No 1, April 2009

[11] M. Tipping, "Sparse Bayesian Learning and the Relevance Vector Machine", Journal of Machine Learning Research, 2001, pp 211 - 214.

[12] Di He, Shanghai Jiao Tong University , Improving the Computer Network Intrusion Detection Performance Using the Relevance Vector Machine with Chebyshev Chaotic Map, IEEE, 2011

[13] Reza Sadoddin, Ali A. Ghorbani, "An Incremental Frequent Structure Mining Framework for Real-Time Alert Correlation", Computers & Security, 2009, pp 153 – 173

[14] Shaohua Teng, Hongle Du, Naiqi Wu, Wei Zhang, Jiangyi Su, "A Cooperative Network Intrusion Detection Based on Fuzzy SVMs", Journal of Networks, Vol. 5, No. 4, April 2010

[15] Phurivit Sangkatsanee, Naruemon Wattanapongsakorn, Chalermpol Charnsripinyo, "Practical Real-Time Intrusion Detection Using Machine Learning Approaches", Computer Communications , 2011, pp 2227–2235

[16] Li Rui, "Computer Network Attack Evaluation Based on Incremental Relevance Vector Machine Algorithm", Journal of Convergence Information Technology, JCIT, Volume7, Number1, January 2012

[17] Javier M. Moguerza and Alberto Munoz, "Support Vector Machines with Applications", Statistical Science, 2006, Vol. 21, No. 3, pp 322 – 336

[18] Chenfeng Vincent Zhou, Christopher Leckie, Shanika Karunasekera, "A Survey of Coordinated Attacks and Collaborative Intrusion Detection", Computers & Security, 2010, 124 – 140

[19] Georgios P. Spathoulas, Sokratis K. Katsikas "Reducing False Positives in Intrusion Detection Systems", Computers & Security, 2010, pp 35 – 44

[20] Anna Koufakou, Michael Georgiopoulos, "A Fast Outlier Detection Strategy for Distributed High-Dimensional Data Sets with Mixed Attributes", Data Mining Knowledge Discovery, 2010, pp 259–289

[21] Wu, S., & Yen, E., "Data mining-based intrusion detectors", Expert Systems with Applications, 2009, pp 5605–5612.