Security Incident Detection Using Multidimensional Analysis of the Web Server Log Files

Grzegorz Kołaczek and Tomasz Kuzemko

Wroclaw University of Technology,
Wybrzeze Wyspianskiego 27 str. 50-370 Wroclaw, Poland
Grzegorz.Kolaczek@pwr.edu.pl

Abstract. The paper presents the results of the research related to security analysis of web servers. The presented method uses the web server log files to determine the type of the attack against the web server. The web server log files are collections of text strings describing users' requests, so one of the most important part of the work was to propose the method of conversion informative part of the requests, to numerical values to make possible further automatic processing. The vector of values obtained as the result of web server log file processing is used as the input to Self-Organizing Map (SOM) network. Finally, the SOM network has been trained to detect SQL injections and brute force password guessing attack. The method has been validated using the data obtained from a real data center.

Keywords: web server, security, log files, intrusion detection.

1 Introduction

The security breaches are very frequent events in Internet and they are not limited only to big enterprises and the most popular web servers. Each day, there are hundreds of new attacks which are performed using newly discovered vulnerabilities and new types of malware and hacking tools. Because the hacking tools are widely available and also there are many tools which automate the computer system exploitation every Internet user should be prepared not only to protect himself/herself but also it is important to detect if the applied security countermeasures have not been broken [1]. One of the most widely known and discussed example of the degree of the threats related to web servers may be a heartbleed vulnerability discovered recently [2].

The diversity of the threats that may impact users' data security makes that classical protective mechanisms as firewalls, antivirus systems etc. must be combined with intrusion detection and prevention systems (IDS/IPS). Because of rapid changes in protection systems as well as in attack methods, the companies developing IDS/IPS solutions started to combine typical signature based solutions with methods related to soft computing (e.g neural networks, Support Vector Machines, etc.) [3].

The main aim of the research presented in this paper was to develop a new method for improving web servers security. The basic element of the proposed method is

D. Hwang et al. (Eds.): ICCCI 2014, LNAI 8733, pp. 663–672, 2014.

based on artificial neuron network which processes the complex data sets characterizing web server user's behavior. The data sets are derived from the web server status logs and the neural network type is Self Organizing Map (SOM).

2 Related Works

There are several works which present the applicability of SOM networks to solve the problem of security breaches in computer networks [4] [5] [6] [7] [8]. In most cases the Self Organizing Map (SOM) networks are used to process the numerical data sets. This means that there is no need to preprocess data before the SOM can be applied. For example, in [6], the network traffic analyzer has been presented. For each ISO/OSI network layer a separate SOM network has been trained. The final decision about security related event detection comes as a result of fusion of data provided for each ISO/OSI layer by dedicated SOM network separately. The similar, multilayer SOM approach has been presented in papers [5] [7][11]. In this approach, the authors used a lot of SOM network trained to determine the similarity for each specialized groups of features. These SOM networks constitute the first layer of the intrusion detection model. Then, the SOM from the second layer combines the data from the first stage of data processing and produces the final decision about system security. Next element which has been derived from the previous researches and applied in the approach which has been presented this paper, is the sliding window for SOM network. However, to the authors' best knowledge there are no publications considering the application of SOM networks to the web server log analysis and especially there is no defined method to transform web server logs into the numerical values which can be processed by SOM network.

The rest of the paper is organized as follows. The next section presents the background of the log analysis problem and defines the detection method of attacks against web server. After that, the experimental evaluation of the proposed method has been presented. The last section contains conclusions and describes future works.

3 Server Log Analysis

The main directions in research related to web servers security incident detection focus on the network traffic and server log analysis [1] [6] [9]. As each HTTP request to the web server can be recorded in server's access log file it becomes a natural source of the information about the server's healthiness as well security. The correct user request as well as the invalid or related to attacks against server will be recorded at access log. Then, the logs can be analyzed and symptoms of the security incidents can be recognized. Due to log file size it cannot be analyzed thoroughly by system administrator. So, some additional tools supporting the system administrator should be provided. The final solution must be a compromise between computational complexity, detection precision and speed. The proposed method is offline processing of the recorded user activity available in web server log files. This assumption gives

greater possibility for data preprocessing optimization. It also is more flexible and easy to be applied in real web server environment.

The main steps of the web server incident detection method are presented in Fig. 1.

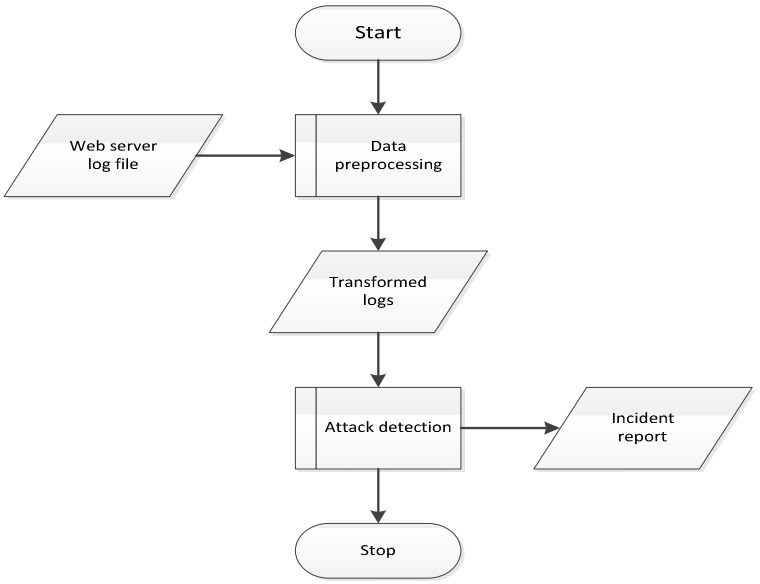


Fig. 1. Steps of Web Server incidents detection

Incident detection is performed in two main phases. The first one is web server log file processing. This phase is responsible for:

- sorting the access logs data by IP address and time stamp
- features selection
- transformation of selected features into numerical values
- encoding sequence number (in learning phase only)
- encoding the number of event/attack class (in learning class only)
- saving the data into CSV file

Next phase is respectively training the SOM network (in learning phase only) or attack detection in preprocessed log file. The network is trained in supervised mode so it is required that the data have information about class events. The learning phase is performed one time before the detection of the security incidents in web server log files. Important advantage of the proposed solution is that there is no association with specific types of attacks. The system using the information about the event class learns to recognize the characteristics of a given class of incidents. So, it is possible to use the approach to detect new and unknown attacks. The only element is required is extractor of characteristic attack features which will be able to recognize the information sufficient to distinguish new type of attack from other classes of traffic.

3.1 Data Preprocessing

The original web server log files are processed in two step procedure. The first step is log records aggregation. The predefined FIFO buffer has been used as a the sliding-window for data aggregation algorithm. In this method, only the information about a sequence of events is maintained. This means that the information about the time gaps between successive events is lost. Nevertheless, according to [4] hidden representation of time allows to get better detection results while using SOM networks than explicit time representation.

As data collected in the sliding window buffer is aggregated, the method must be proposed to set the particular traffic class for the buffer (in the training phase). This is due to the fact that the entire window is treated as one sample of data. Let all the classes contained in the window at the moment will be marked by a set *S*. The function *v* assigns the number of occurrences of each class of the records in the current time window.

$$\underset{s \in S}{v}: s \to \mathbb{N} \tag{1}$$

Then the time window may be defined as the multiset:

$$M = \langle S, v \rangle \tag{2}$$

The class assigned to the window is defined by *m*, which is set by finding the class which is the most frequent in this window.

$$m: \underset{m \in S}{max}(v(m)) \tag{3}$$

For example, in the window containing the following sequence of traffic classes: (*normal, normal, sqlinjection,sqlinjection,normal*), the window class will be set to *normal*.

Categorical variables are variables that can assume values only from a limited set. Some of the data in web server logs are categorical so there must be algorithm defined to transform categorical variables into numerical values. One of the frequently used approaches is to assign individual values of categorical variables to consecutive integers. This method does not work, however when it is necessary to determine distance between two values. For example, the calculated Euclidean distance will be larger for extreme values than the neighboring. An alternative method of representation of categorical variables in numerical form is to convert them to binary form [10]. With such representation calculated Euclidean distance between all acceptable values that will be constant. The example of representation of the binary categorical variable "HTTP method" for the value of GET and HEAD are presented in the **Table 1**.

Table 1. Categorical variable binary representation

Value	GET	POST	HEAD	OPTIONS
GET	1	0	0	0
HEAD	0	0	1	0

Finally, log file should be preprocessed to extract the most informative part of the record. The general idea of HTTP request/response interpretation has been presented in Fig. 2. The text string representing particular request is divided into a few separate parts which are interpreted by so called "extraction modules".

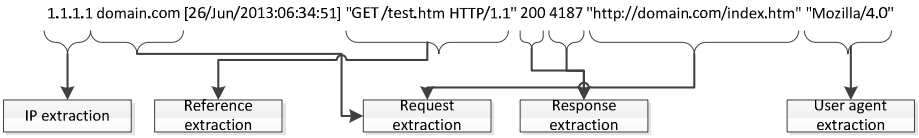


Fig. 2. Log record interpretation

The most important operations performed during features extraction phase include: identification and numerical representation of the continent related to IP address, representation of HTTP method and version as a binary vector, calculation of the URL request depth, calculation of the no alphanumerical characters number in the request, hash value calculation from the file extension, change of user-agent name into corresponding ID value, etc. After this phase the vector of numerical values describing the recorded in log file request is passed to the next module.

3.2 Network Training

To be able to use the system for the detection of attacks, it is necessary to train SOM network on the training data containing information about the class of the event. This phase is crucial, because the system is able to identify only those attacks that learned to recognize.

The training data must possess information about the class of the event. For example, normal traffic can be assigned to the class of *normal*, while attacks classes with names which correspond names of attacks like: *sqlinjection*, *pathtraversal* and so on. Label with name of the class should be added to each record of access logs used to train the system. It should apply the following format [CLASS], where CLASS is the name of the class to which is assigned to the record. Label in this form should be appended to the end of each row.

The detection system is multi-layered SOM network. Multilayer network consists of multiple layers of standard Kohonen network. Each of them is trained to determine the similarity of the analyzed record to the characteristics of a single attack category. At the input of the network is presented a single data sample. It was isolated three groups of features: *Request*, *Response*, and *Referer*. For each group there is a separate Kohonen network layer.

3.3 Attacks Detection

Trained Kohonen network system is ready to perform the classification of previously unknown records of access logs. For this purpose, it is necessary to convert logs into a metrics number. The neuron whose weight vector is most similar to the input is called the best matching unit (BMU). BMU of the sample is calculated by finding the node with the minimum distance to it. Distance sample to a node is calculated as the sum of

the partial-weighted distances for each layers. Weight for each layer is defined as a parameter before network training. The classification is made on the basis of class BMU given sample. In the training phase to each node is assigned one class. This information is used in step of anomaly detection to determine previously unknown class of the sample. After finding the BMU for a given sample the information on the BMU class is determined. The class name is returned as the final decision on the classification.

4 Experimental Results

The full set of access log which has been used during experimental method evaluation, contains more than 600 thousand records from the server of one hosting company. Each record contains one task per client and HTTP server response. During the observation period, clients' requests coming from more than 100 thousands different IP addresses have been recorded. For the experiment subset comprising the first 10000 records was selected. As a result of the classification procedure performed by the expert two different types of attacks have been identified: SQL injection and brute force password scan. Finally, together with the normal requests the three classes of network traffic has been defined.

Table 2. Default configuration of the algorithm

Parameter name	Value
Features	request, response, refer, user-agent, geoip
FIFO-size	5
Rows	7
Columns	7
Gridtype	Hexagonal
Weight	1
Training-ratio	0.5
Numer of iterations	50

The implemented method using SOM networks allows setting the values of the following parameters in the detection algorithm: set of log features defining the SOM dimensions, the sliding window size, number of rows and columns of SOM network, gridtype, weights of features, training ration and number of iterations. The default values of these parameters have been presented in

In the first performed experiment the correlation of the algorithm parameters values and the detection precision (eq. 4) has been investigated.

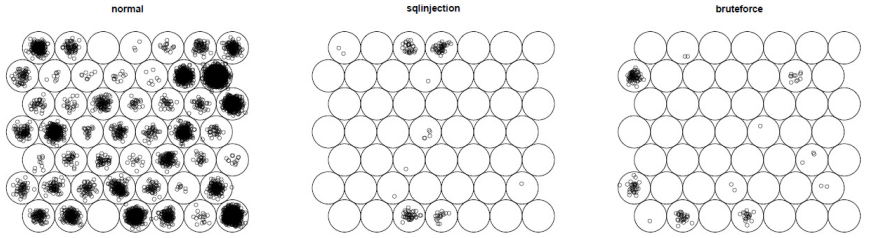
$$precision = \frac{\sum diag(M_{conf})}{\sum(M_{conf})} \tag{4}$$

where M_{conf} is a confusion matrix.

The results for the default values of algorithm parameters has been presented in Table 3 and Fig. 3.

Table 3. Confusion matrix for the default parameters algorithm

	Bruteforce	Normal	Sqlinjection
Bruteforce	41	47	0
Normal	27	4955	1
Sqlinjection	0	15	58
Precision	98,25%		



(a) normal (b) sqlinjection (c) bruteforce

Fig. 3. Mapping for each class of attack to the default SOM parameter values

Next experiment was intended to check the impact of feature selection on detection precision. This parameter determines which features describing HTTP requests and responses are taken into account when training SOM network and during classification. The experiment concerned the following combinations of features set values

Table 4. .Cofusion matrix for futures={request}

	Bruteforce	Normal	Sqlinjection
Bruteforce	60	28	0
Normal	3	4962	18
Sqlinjection	0	20	53
Precision	98,66%		

Table 5. Confusion matrix for futures={request,response}

	Bruteforce	Normal	Sqlinjection
Bruteforce	0	88	0
Normal	0	4968	15
Sqlinjection	0	29	44
Precision	97,43%		

Table 6. Confusion matrix for futures={request,response,refer}

	Bruteforce	**Normal**	**Sqlinjection**
Bruteforce	55	33	0
Normal	19	4922	19
Sqlinjection	0	25	48
Precision	97,69%		

As the results presented in Table 4, Table 5, Table 6 show, more features used to generate metrics does not necessarily turns into better precision. Paradoxically, the best results were achieved using only one element set *feature=request*. The results are even better than in the case of default parameter values, using all five available features extractors. However, the main element responsible for the highest level of precision is extremely good classification of brute force attack. Meanwhile, sqlinjection attack has been identified slightly worse than in the case of default parameter values. An interesting results of brute force attack classification appear when using set of features consisting of *request* and *response*. In this case, no events appears that could have been classified as a brute force attack. All attack samples were incorrectly classified as normal. The main result of the experiment is the observation that a group of features selected for classification must be chosen individually in the context of a particular attack that we want to detect with the greatest accuracy. Alternatively, one can achieve relatively good classification precision of all attack types by selecting all available features.

The third experiment investigated the influence of the fifo-size for the classification precision of attacks against web servers.

Table 7. Confusion matrix for fifo-size=1

	Bruteforce	**Normal**	**Sqlinjection**
Bruteforce	0	85	0
Normal	0	4997	0
Sqlinjection	0	64	0
Precision	97,10%		

Table 8. Confusion matrix for fifo-size=10

	Bruteforce	**Normal**	**Sqlinjection**
Bruteforce	43	53	0
Normal	24	4932	14
Sqlinjection	0	13	62
Precision	97,98%		

Table 9. Confusion matrix for fifo-size=20

	Bruteforce	**Normal**	**Sqlinjection**
Bruteforce	39	52	0
Normal	26	4937	14
Sqlinjection	0	20	48
Precision	97,69%		

From the obtained results of experiments with different parameter values fifo-size noticeable is the influence of this parameter on the accuracy of classification. First of all, when the sliding window is off (set to 1) none attack was not detected. The best total result was ever achieved with the default parameter values (fifo-size = 5). Brute force attacks are detected with slightly better precision for longer sliding window values . In turn, the attack *sqlinjection* is best recognized at the value fifo-size = 10.

On the basis of the experiment results can be concluded that the optimal length of the queue depends on the kind of attack we anticipate with the best precision. One should also remember that the selection of longer sliding window increases the probability that the information about a particular attack related to only a single request can be lost. This is due to the way in which traffic class is determined for the whole window.

5 Conclusions

Application of soft computing methods to recognition of attack against web server extends the possibilities of the security incident detection. Also previously unknown attacks can be detected only if they are described by similar set of features. The proposed method uses SOM networks for event type classification where events are defined by the records in web server log files. The approach defines also the method of transformation of log files records into corresponding numerical vectors which can be processed by SOM network.

Finally, the several experiments on real data sets have been performed. The results of the experiments are promising for security incidents detection precision. The average detection precision for the proposed method is about 98%.

Further work will focus on improving the rate of attacks detection. Despite the achievement of high values of precision indicator, the number of correctly detected brute force attacks are relatively low. This is widely known problem in computer attack detection where the number of normal events is much greater than the number of events related to attacks. The next steps will be dedicated to the improvement of learning phase to provide the better brute force attack recognition.

References

1. Multi-agent platform for security level evaluation of information and communication services. Grzegorz, Kołaczek. Springer, Berlin
2. Egeber, P.: Background on Heartbleed (2014)

3. Gudkov, O.: Calculation Algorithm for Network Flow Parameters Entropy in Anomaly Detection. Kaspersky Lab (2012), http://www.kaspersky.com/images/Oleg%20Gudkov.pdf
4. Lichodzijewski, P., et al.: Host-based intrusion detection using self-organizing maps. In: Neural Networks, pp. 1714–1719 (2002)
5. Heywood, M.I.: Dynamic intrusion detection using self-organizing maps (2002)
6. Rhodes, C.: Multiple self-organizing maps for intrusion detection. In: 23rd National Information Systems Security Conference (2000)
7. Stevanovic, D., Vlajic, N.: Detection of malicious and non-malicious website visitors using unsupervised neural network learning. Applied Soft Computing 13(1), 698–708 (2013)
8. Łukasz, B., Katarzyna, N., Michał, A., Grzegorz, K.: SOM-based system for anomaly detection in network traffic. Wroclaw University of Technology, Wroclaw (2013)
9. Kolaczek, G., Juszczyszyn, K.: Traffic pattern analysis for distributed anomaly detection. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Waśniewski, J. (eds.) PPAM 2011, Part II. LNCS, vol. 7204, pp. 648–657. Springer, Heidelberg (2012)
10. Singh, N., Jain, A., Raw, R.S., Raman, R.: Detection of Web-Based Attacks by Analyzing Web Server Log Files. In: Mohapatra, D.P., Patnaik, S. (eds.) Intelligent Computing, Networking, and Informatics. AISC, vol. 243, pp. 101–109. Springer, Heidelberg (2014)
11. Budka, K.C., Deshpande, J.G., Thottan, M.: Network Security. In: Communication Networks for Smart Grids, pp. 209–225. Springer, London (2014)