Intelligent Automation And Soft Computing, 2019

Vol.25,no.1,131–141

The Design and Implementation of a Multidimensional and Hierarchical Web Anomaly Detection System

Jianfeng Guan*, Jiawei Li, and Zhongbai Jiang

*P.O. Box 202, Beijing University of Posts and Telecommunications, Haidian District, Beijing, 100876, China.

State Key Laboratory of Networking and Switching Technology
Beijing University of Posts Telecommunications, Beijing, 100876, China
jfguan@bupt.edu.cn, ljwemls@gmail.com, zbjiang@bupt.edu.cn

ABSTRACT
The traditional web anomaly detection systems face the challenges derived from the constantly evolving of the web malicious attacks, which therefore result in high false positive rate, poor adaptability, easy over-fitting, and high time complexity. Due to these limitations, we need a new anomaly detection system to satisfy the requirements of enterprise-level anomaly detection. There are lots of anomaly detection systems designed for different application domains. However, as for web anomaly detection, it has to describe the network accessing behaviours characters from as many dimensions as possible to improve the performance. In this paper we design and implement a Multidimensional and Hierarchical Web Anomaly Detection System (MHWADS) with the objectives to provide high performance, low latency, multi-dimension and adaptability. MHWADS calculates the statistical characteristics, and constructs the corresponding statistical model, detects the behaviour characteristics to generate the multidimensional correlation eigenvectors, and adopts several classifications to build an ensemble model. The system performance is evaluated based on realistic dataset, and the experimental results show that MHWADS yields substantial improvements than the previous single model. More important, by using 2-fold Stacking as the ensemble architecture, the detection precision and recall are 0.99988 and 0.99647, respectively.

KEY WORDS: Web Anomaly Detection, Ensemble model, Statistical Model, Multi-dimension

1 INTRODUCTION

THE web applications are booming with the rapid increase of Internet services and users which results in the shift of network paradigm (Zhang et al., 2016; Guan et al., 2017). As a result, the evolving web-based attacks also become more and more complicated (Yao et al., 2016b, a), which spurs lots of researches around the web detection such as misuse detection and anomaly detection. Misuse detection system constructs the models of known attacks which has an attractive detection performance. For example, by using attack signatures it can help the Intrusion Detection System (IDS) to find out all kinds of known attacks. However, with the development attack skills, it cannot detect the new unknown attacks, which makes it inappropriate to the enterprise-level deployment.

Misuse detection system has following problems: (1) Adaptability: the judgement criteria of similar regular expressions are easily evaded, and the system cannot effectively detect unknown attack types. (2) Requirement: the signature-based detecting method mainly depends on the experience and judgement of security researchers, which has a high demand for the expertise level of developers, and therefore results in a high study threshold. (3) Operation Cost: the rule-base needs to be periodically updated and maintained by experts, while each modification of on-line system introduces high cost.

Contrast to misuse detection, anomaly detection first defines the normal behaviors and then it detects the anomaly ones, which can detect the unknown

CONTACT Jianfeng Guan jfguan@bupt.edu.cn

attacks. Once detecting the different behaviors, it will take them as certain attacks. The essence is the anomaly detection techniques which has an important impact on the detection performance. Lots of anomaly detection algorithms have been proposed (Chandola et al., 2009; Yu, 2012), and some of them focused on dimension reduction (Wang et al., 2011), feature selection (Chen and Huang, 2011). In this paper, we focus on design and implement a web anomaly detection system to improve the detection precision and recall.

The rest of this paper is organized as follows. Section 2 analyzes the related work of anomaly detection. Section 3 describes the architecture design of the whole system and depicts its main subsystems and related algorithms. Especially, the detection model part introduces the single model and ensemble model, respectively. Section 4 evaluates each subsystem and integrated detection model based on realistic dataset. Finally, Section 5 concludes this paper and outlines further work.

2 RELATED WORK

2.1 Anomaly Detection Techniques

CHANDOLA et al. (2009) investigated the different anomaly detection techniques in multiple domains including intrusion detection, fraud detection, medical and public Health anomaly detection, industrial damage detection, image processing, text data, and sensor network and so on. Besides, they classified the anomaly detection techniques into neural network-based, Bayesian network-based, Support Vector Machine (SVM)-based and Rule-based algorithms. After that, Yu (2012) focused on the anomaly detection in computer system, and reviewed the anomaly detection techniques based on statistics, machine learning, neural network, computer immunology, and data mining techniques. More specifically, Baddar et al. (2014) investigated several categorizations of anomaly detection solutions which have adopted the machine learning approach. However, learning-based intrusion detection systems may result in many false positives which degrade its precise. Therefore, several works are focusing on improving its performance.

The early stage research (Lin and Ying, 2012) adopted SVM, simulated annealing and decision trees to select features and obtain decision rules to help identify new attacks. Then, Dorj et al. (2013) proposed a Bayesian Hidden Markov Model-based approach which consists of models and expresses inter-model dependencies in order to further enhance performance and availability. Juvonen and Hamalainen (2014) proposed a log anomaly detection framework which is based on dimensionality reduction by using random projection and uses Mahalanobis distance to find outliers. Besides, Huang and Huang (2013) proposed a growing hierarchical self-organizing map-based system to analyze the network traffic data and visualize the distribution of attack patterns. However, the system detects the overall anomaly which results in a relatively sophisticated detection algorithm and therefore, is hard for online detection.

More recently, Ippoliti et al. (2016) proposed a SVM-based on adaptive anomaly detection and correlation mechanism for flow analysis without a priori alert classification. Furthermore, it developed a lightweight evolving alter aggregation method. The evaluation results shows that it can maintain high accuracy without the need of offline training. Other methods such as Shannon entropy (Gautam and Om, 2015) which adopts the Shannon entropy analysis to detect malicious attacks based on the assumption that these attacks should have more complexity than common access requests. Besides, Barani (2014) proposed GAAIS algorithm which is based on Genetic Algorithm (GA) and Artificial Immune System (AIS) for dynamic intrusion detection in ad hoc networks. Fiore and Palmieri (2013) developed a self-learning anomaly detection approach called Discriminative Restricted Botlzman Machine (DRBM) which combines strong generative modelling with classification accuracy in the network's traffic.

Besides, inspired by model ensemble, Louvieris et al. (2013) adopted K-means for feature extraction, Naive Bayes and Kruskal-Wallis test for feature selection, and C4.5 decision tree for final classification. However, the simple combination of these algorithms are difficult to achieve both a high accuracy and recall for that most of them are in the pursuit of classification accuracy, therefore, the overall effect on the abnormal behavior is relatively poor. By improving k-means algorithm, Li (2013) proposed a system for clustering anomalies and classifying clusters to detect intrusion behaviors including system or users' non-normal behavior and unauthorized use of computer resources. However, these kinds of systems can only report that there is an attack without any supporting description of the anomaly that has been detected.

2.2 Web Anomaly Detection Systems

In terms of web anomaly detection, the early research adopted the signature-based intrusion detection method which cannot detect the unknown intrusions. Therefore, Jang and Won (2007) proposed a method using traffic analysis and URI information analysis by updating the detection rules in Snort. After that, Liang presented an immune-based active defense model (Liang, 2008) based on the clone selection and hyper-mutation, which can quantize the risk of web attacks to provide active defense for unknown attacks. Similarly, Lau et al. (2009) proposed an unsupervised anomaly detection based on vertebrate immune system by dynamically updating detection model. Subsequently, Fan (2012) adopted hidden Markov model to detect the web attacks, and Xie and Tang

(2012) set up a dynamic hidden semi-Markov model with the ability of online updating parameters to detect the DDoS attacks. Recently, Kozik and Choras (2015) propose an ensemble of one-class classifiers (Decision Stump and Reduced Error Pruning Tree) based on packet segmentation mechanism to detect the attacks. Besides, Parhizkar and Abadi (2015) introduced BeeSnips algorithm to prune the initial ensemble of one-class SVM classifiers to reduce false alarms in anomaly detection. More recently, Bronte et al. (2016) adopted the cross entropy for parameter, value, and data type to detect web anomaly. More recently, Yuan et al. (2017) considered the semantic information and proposed the DEP-SSEC solution which uses the deep learning enabled subspace spectral ensemble clustering approach to detect anomalies and their types.

In our previous work (Li et al., 2016), we have proposed a hierarchical anomaly detection system that combines feature generating with prediction system, which has a good detection performance and time complexity. The whole system constructs a number of detection subsystems based on statistical characteristics and uses classification algorithm to remap the new feature space. However, the generalization capability of the single classifier is limited, which could be further improved by model ensemble. Therefore, this paper proposes a comprehensive detection system against Web anomaly, which combines multidimensional feature generating system and hierarchical classification system, and makes a comprehensive assessment to the web-based access behavior.

The main contributions of this paper are shown as follows. (1) MHWADS synthetically combines the misuse detection and anomaly detection to further improve the system performance. (2) MHWADS generates multidimensional correlation eigenvectors and adopts multiple classification algorithms to detect abnormal records. (3) MHWADS ensembles multiple single classifier models by 2-fold stacking model to improve detection precision and recall.

3 MHWADS DESIGN

THE data used in our system is firstly processed by the statistical anomaly detection and rule anomaly detection as shown in Figure 1. First, the raw data and detection logs from network devices are captured and stored in Big Data platform, and then these information will be filtered by both statistical anomaly detection and rule anomaly detection. Finally, to further detect the omissive anomaly behaviors, all these filtered data will be proceeded by the MHWADS.

Figure 2 shows operation flow of the MHWADS which mainly consists of four subsystems: data pre-processing subsystem, statistical model construction subsystem, multidimensional subsystems and

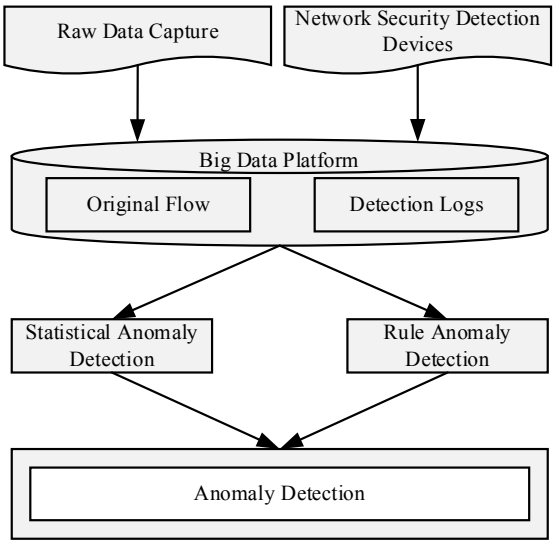

Figure 1. The overall architecture of MHWADS.

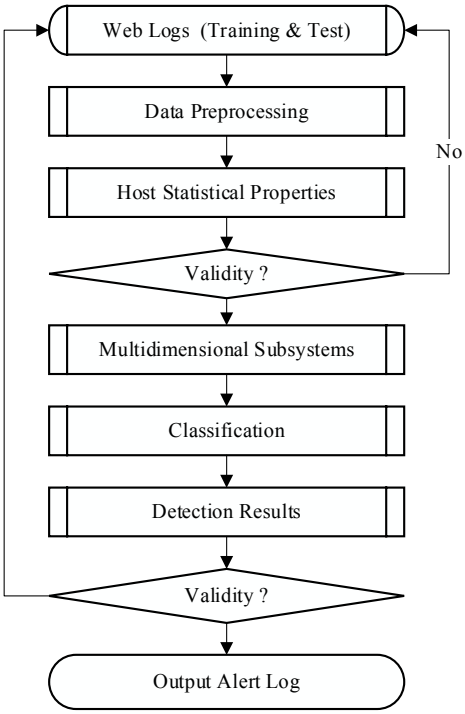

Figure 2. The operation flow of MHWADS.

detection subsystem. In addition, the detection subsystem is divided into single module and ensemble module. The procedure of MHWADS is shown as follows: (1) MHWADS constructs a separate statistical model which is based on large quantities of HTTP access records under each specific domain; (2) According to the calculated statistical characteristic parameters, the behaviour characteristics of each record in the network access data set are detected from different dimensions, which results in correlation

eigenvectors; (3) MHWADS remaps the results of each detecting subsystem to the new feature space and uses classification algorithm to judge anomaly; (4) MHWADS adopts the Stacking architecture to build the ensemble model of different classifiers to improve the detection performance of system.

3.1 Data Pre-processing Subsystem

This subsystem mainly processes error, absent or repetitive data, and classifies log records according to different host domains. The input of this system is Web log (parsed HTTP packets). More specifically, a complete access record should at least include the field of time stamp, source IP, source port, destination IP, destination port, method, URI, host domain, origin, cookie, REFERER, user agent, and data. In this stage, the dataset is appropriately divided into two sets, one is for training and the other is for test.

3.2 Statistical Model Construction Subsystem

This subsystem calculates all the related statistical parameters according to the log records under a certain domain output by data preprocessing module, and constructs a statistical model under specific domains. The following four categories are included.

(1) Hierarchical Path Frequency List (HPFL): It counts the frequency of every hierarchical path and the two neighboring hierarchical paths according to the path field of each URI.

(2) Bag of Parameter Subset (BPS): First, as for each record under the same path, it extracts all the parameters $(p_1, p_2, ..., p_k)$ that appear in the parameter field and forms a subset of parameters $S_i = \{\{p_1, p_2, ..., p_k\}\}$. Then it gathers the different parameter subsets and forms a BPS under a certain path, which is $S = S_1 \bigcup S_2 \bigcup ... \bigcup S_n$, where n is the number of record under a specific path.

(3) Parameter Rule Set (PRS): First, it extracts parameters and composes a directed graph according to each ordered sequence from every URI under the same path. Secondly, it traverses all the parameters (nodes) and calculates the links between the nodes to make sure whether they are connected. Finally, it constructs PRS (S). Take parameter x and y for example:

1) If node x can reach node y, but node y cannot reach node x, which means node x is surely before node y, then add (y, x) to S.

2) If node y can reach node x, but node x cannot reach node y, which means node y is surely before node x, then add (x, y) to S.

3) All the other conditions cannot conclude the fixed sequence, nor can they change the rule set S.

(4) EPS (Enumeration Parameter Set): The enumeration parameter refers to that parameter value under certain variables all come from a fixed finite EPS (Kruegel and Vigna, 2003), such as content identification or index. EPS firstly introduces two auxiliary functions $f(k)$ and $g(k)$ as shown in Functions (1) and (2). Suppose a certain parameter q under a path has n values $(q_1, q_2, ..., q_n)$ which come from the n URIs under the same path.

$$f(k) = k \qquad k = 1, 2, 3...n \tag{1}$$

$$g(k) = \begin{cases} 0 & , k = 0 \\ g(k-1)+1 & , q_k \notin (q_1, q_2, \cdots, q_{k-1}) \\ g(k-1)-1 & , q_k \in (q_1, q_2, \cdots, q_{k-1}) \end{cases} \tag{2}$$

Then $f(x)$ and $g(x)$ are concluded from the actual data. The correlation coefficient ρ of the two functions can be calculated. We can judge whether a certain variable name belongs to enumeration parameter set by using the following rules:

1) If $\rho < 0$ (negative correlation), variable name q is enumeration;

2) If $\rho > 0$ (positive correlation), variable name q is stochastic;

3) If $\rho \approx 0$, it is hard to judge the type of variable name q. To avoid false positive, it can be dealt as stochastic type.

EPS contains all parameters that are judged as enumeration.

In addition to these statistical characteristics such as the mean value and standard deviation of parameter length, the special character, are also considered in the following section.

3.3 Multidimensional Subsystem

Based on the related statistical parameters, this subsystem detects the behaviors characters of each record from different dimensions, and then remaps the multi-dimensional detected results into the vector space of records to generate the related multi-dimensional eigenvectors for final classification. This subsystem will detect the certain dimension of a single record, and generates a probability that the record can be regarded as a normal behavior.

As shown in Figure 3, the multidimensional subsystem consists of path, parameter name and parameter value. (1) Path model: MHWADS adopts Bigram algorithm to calculate the normal probability of different paths, and adjusts them by regularization algorithm. (2) Path name association model: It contain two parts, the first one is parameter composition which is used to test whether or not the recorded parameter is in the parameter pool of corresponding path, while the parameter order is used to verify whether or not the sequence of parameter name is against the parameter naming rules. (3) Parameter value model: It is consisted of Length Distribution,

Special Symbol, Enumeration Parameters and Character Distribution, which are used to measure the network accessing behaviors.

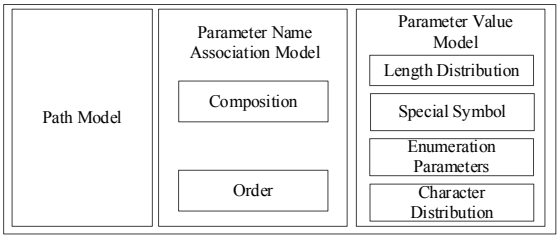

Figure 3. The modules of multidimensional subsystem

3.3.1 Path

Based on N-grams (Brants et al., 2007) (we choose Bi-gram here), we can calculate the normal probability of the path in URI.

For a given path: $a/b/c?x=1\&y=2$, the conditional probability and maximum likelihood estimation $P(b|a)$ can be expressed as Equation 3.

$$P(b|a)=\frac{P(a,b)}{P(a)}=\frac{Count(a,b)}{Count(a)} \tag{3}$$

Where $Count(a,b)$ represents the frequency that path a appears before path b. $Count(a)$ stands for the frequency that Path a occurs. The specific number can be acquired from HPFL in previous module. As for the imbalance caused by different lengths of paths, we adopt the related regularization term to revise it. Table 1 shows the process to calculate path probability.

Table 1. The process of calculating path probability

Step	Operation
URI	$a/b/c?x=1\&y=2$
path	$/a/b/c$
path depth	3
add location identifier	$HEAD/a/b/c/END$
normal probability	$\sqrt[3]{\log[p(a\|HEAD)*P(b\|a)*P(c\|b)*P(END\|c}$
maximum likelihood estimation	$\frac{1}{3}\{\log[P(a\|HEAD)]+\log[P(b\|a)]+\log[P(c\|b)]+\log[P(END\|c)]\}$

3.3.2 Variable Correlation

Considering that in most cases users do not manually input all parameters of URI in the browser but click the page links or buttons to open a web page, the URI generated from those links or buttons is pre-defined by client programs, scripts or HTML forms. In this case, this mechanism makes the parameters that have a strong regularity in composition, number and order, etc. Therefore, based on the parameter name, the abnormal behaviors can be detected if there exists the conditions of the excess, lack, disorder of the relevant parameters. More specifically, the variable correlation can be performed in terms of variable composition and variable order.

• Variable Composition

As for the variable composition extracted from a record's URI which does not belong to its corresponding BPS, this record is judged to be anomaly and noted as 0. Otherwise, it is noted as 1.

• Variable Order

If the order of the record's variable is included in the corresponding PRS, which indicates that the variable sequence is abnormal, the record will be noted as 0. Otherwise, it is noted as 1.

3.3.3 Parameter Value

When normal users access Web services, the query parameter values are automatically generated by web scripts or users' input through HTML forms. Therefore, the string length of input parameters is either constant or variable in a small range. However, many malicious attacks contain parameter values that are often far beyond the normal range. For example, buffer overflow attack often contains malicious binary code with hundreds of bytes, while XSS attack will be embedded with a large number of invasive scripts. Meanwhile, under normal conditions, the character distribution of the specific parameter values has a regular structure. As for anomaly records, there will be a completely different character distribution. For example, some exploratory behaviors before the attacks will contain a large number of "." symbols so as to achieve traverse routing space. There are four metrics to evaluate the parameter value including the length distribution, special symbols, enumeration and character distribution.

• Length Distribution

The normal probability of the length distribution is calculated based on revised Chebyshev inequality. For a given variable x_i, assuming that μ_i and σ_i are the means of the parameter length and standard deviation, respectively. Let l_i represent the parameter value length of a record. If $l_i \le \mu_i$, then output 1. Otherwise, as shown in Equation (4), it calculates the normal probability $P(x_i)$ with distribution of variable x_i as output.

$$P(x_i)=\frac{1}{1+2\varepsilon_i^2/\sigma^2} \quad ,\varepsilon_i=l_i-\mu \tag{4}$$

• Special Symbols

The special symbol sets S={"!", "@", "#", "%", "^", "(", ")", "{", "}"...}. The probability of all special symbols contained in parameter value of the whole model space is used as its normal probability.

For a specific record, we get each normal probability of the special symbol contained in the parameter value and take its minimum as variable's normal probability. Then take the minimum normal probability of each different variable in the record as the output of the module.

• Enumeration

For a given record, its behavior will be asserted as anomaly if its variable is judged as enumeration parameter and does not belong to the relevant EPS (noted as 0). Otherwise, the subsystem output 1.

• Character Distribution

The character distribution contains direct character distribution test (Kruegel and Vigna, 2003) and aggregated character distribution test. The former is based on the single ASCII code for interval division as shown in Table 2, and the latter is based on the character categories (uppercase letters, lowercase letters, control character, digit, unprintable characters, and out-of-range characters) as shown in Table 3. Combined with the probability expectation among specific ranges of characters, the Chi-square test is used to calculate the normal probability of character distribution under both cases.

Table 2. Direct character distribution test

Interval No. (position of ordered probability sequence)	X	Y	……
1 (0)	x_1	y_1	
2 (1-3)	x_2	y_2	
3 (4-6)	x_3	y_3	
4 (7-11)	x_4	y_4	
5 (12-15)	x_5	y_5	
6 (16-255)	x_6	y_6	

Table 3. Aggregated character distribution test

Character type	X	Y	……
Uppercase letter	x_1	y_1	
Lowercase letter	x_2	y_2	
Control character	x_3	y_3	
Digit	x_4	y_4	
Unprintable character	x_5	y_5	
Out-of-range character	x_6	y_6	

As shown in Table 2 and Table 3, x_i and y_i represent the probability expectations that the frequency of characters in a given interval $(\sum_i x_i = 1, \sum_i y_i = 1)$, respectively.

Taking the direct character distribution test for example, the calculation process of a specific record is shown as follows. Assuming the URI is "$a/b?x=123@mm\,\&\,y=nn\#1\#2$", the parameter value of variable x is $123@mm$, and the frequency of the ASCII code is shown in Table 4.

Table 4. ASCII code frequency of variable *x*

ASCII Code	Frequency	ASCII Code	Frequency
0	0	…	…
1	0	64	1
…	…	…	…
49	1	109	2
50	1	…	…
51	1	255	…

Without taking the character value into account, the frequency is rearranged by the order from high to low, then the ordered frequency (2,1,1,1,1,0,0,…,0) is acquired. The sequence includes 256 elements, and the corresponding index is (0,1,2,…,255). Then according to the partition of Table 2, the frequency of the sequence in the corresponding interval is summed up to get the following interval distribution as shown in Table 5.

Table 5. Character distribution of variable *x*

Interval No. (position of ordered probability sequence)	Interval frequency
1 (0)	2
2 (1-3)	3
3 (4-6)	1
4 (7-11)	0
5 (12-15)	0
6 (16-255)	0

The interval frequency distribution are tested with Chi-square (DOF is 5). $P(\chi_x^2 \mid 5)$ is the normal probability for the character distribution corresponding to the variable x. The calculation process of variable y is similar to the above, eventually the normal probability of the record is the minimum of the two, that is $\min(P(\chi_x^2 \mid 5), P(\chi_y^2 \mid 5))$.

• Generation of Feature Vector

According to the detection result of each subsystem, each record is mapped into an 8 dimensional feature vectors as shown in Table 6, which is used as the input features of detection model.

$x^{(1)}, x^{(4)}, x^{(5)}, x^{(7)}, x^{(8)}$ are the floats from zero to one, $x^{(2)}, x^{(3)}, x^{(6)}$ are Boolean parameter which is zero or one.

Table 6. An example of feature vector

Feature	$X_{8\times1}$
Path	$x^{(1)}$
Variable composition	$x^{(2)}$
Variable order	$x^{(3)}$
Value length	$x^{(4)}$
Special symbols	$x^{(5)}$
Enumeration	$x^{(6)}$
Value distribution	$x^{(7)}$
Value distribution 2	$x^{(8)}$

3.4 Detection model

On the whole, the detection model is divided into single model and ensemble model. Firstly, according to the calculated statistical characteristic parameters, it remaps the results of every detecting subsystem to the new feature space and uses classification algorithm for judging anomaly. Then, in order to improve the detection performance of the system, it further uses Stacking architecture to build the ensemble model of different classifiers.

3.4.1 Single Model

Based on the multidimensional feature vector above, classification algorithms (i.e. KNN, Logistic Regression, SVM, Decision Tree, Random Forest, GBDT, Xgboost) were used to detect anomaly on web logs.

The input matrix of training data is shown as Matrix (5):

$$\begin{bmatrix} x_1^{(1)},x_1^{(2)},x_1^{(3)},x_1^{(4)},x_1^{(5)},x_1^{(6)},x_1^{(7)},x_1^{(8)},y_1 \\ x_2^{(1)},x_2^{(2)},x_2^{(3)},x_2^{(4)},x_2^{(5)},x_2^{(6)},x_2^{(7)},x_2^{(8)},y_2 \\ \cdots \\ x_m^{(1)},x_m^{(2)},x_m^{(3)},x_m^{(4)},x_m^{(5)},x_m^{(6)},x_m^{(7)},x_m^{(8)},y_m \end{bmatrix}_{m\times9} \quad (5)$$

where m represents the total record number of training dataset, $x_a^{(b)}$ represents the b^{th} $(1\le b\le 8)$ dimension of a^{th} record in the training dataset, y_k represents the real label of k^{th} record (-1 represents anomaly, +1 represents normal record).

The input matrix of testing data is shown as Matrix (6).

$$\begin{bmatrix} \hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)} \\ \hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)} \\ \cdots \\ \hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)} \end{bmatrix}_{n\times8} \quad (6)$$

where n represents the total record number of testing dataset, $\hat{x}_a^{(b)}$ represents the b^{th} $(1\le b\le 8)$ dimension of a^{th} record in the testing dataset.

The output matrix of detection model is shown as Matrix (7).

$$\begin{bmatrix} \hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{x}_1^{(1)},\hat{y}_1 \\ \hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{x}_2^{(1)},\hat{y}_2 \\ \cdots \\ \hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)},\hat{x}_n^{(1)} \end{bmatrix}_{n\times9} \quad (7)$$

where n and $\hat{x}_a^{(b)}$ represent the same meaning in Matrix (6). Alternatively, y_k represents the predicted label of k^{th} record (-1 represents anomaly, +1 represents normal record).

3.4.2 Ensemble Model

Model ensemble is a very powerful technique to increase accuracy on a variety of machine learning tasks. Stacked generalization was introduced by Wolpert (1992). The basic idea behind stacked generalization is to use a pool of base classifiers, then adopt another classifier to combine their predictions, with the objective to reduce the generalization error. Figure 4 shows the procedure of 2-fold stacking, where m and n represent the dimension and size of the training dataset, respectively.

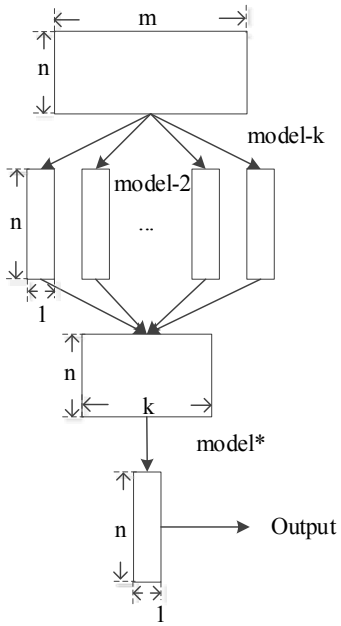

Figure 4. Flow chart of 2-fold stacking.

The whole ensemble process consists of five steps:

(1) Split the training set into 2 parts: A-part and B-part;

(2) Fit a first-stage model on A-part and create predictions for B-part;

(3) Fit the same model on B-part and create predictions for A-part;

(4) Fit the model on the entire training set and create predictions for the test set;

(5) Train the second-stage stacker model on the probabilities from the first-stage models.

Compared with isolated training, the stacker model gets more information on the problem space by using the first-stage predictions as features. 2-fold Stacking is a means of non-linearly combining generalizers which makes a new generalizer and optimally integrates what each of the original generalizers has to say about the learning set. The more each generalizer has to say (which isn't duplicated in what the other generalizer's have to say), the better the resultant stacked generalization. Therefore, in order to select the optimal model set for constructing ensemble network, we use forward search algorithm to realize the model selection.

4 PERFORMANCE EVALUATION

4.1 Dataset

THE dataset adopted in this paper is a one-month web logs (parsed HTTP packets) under a certain domain (xiaoshuo.360.com) in company *QIHU360*. This dataset contains 883,355 access records, with the proportion of normal records to anomaly being 8:1. Each access record contains 14 fields which are delimited by tabs. They are time-stamp, source IP, source port, destination IP, destination port, method, URI, host, origin, cookie, user-agent, reference, data and label (labelled as either normal, or as an attack, with exactly one specific attack type). Attacks fall into six main categories: XSS (Cross Site Scripting), SQL Injection, XML Injection, CRLF Injection, LFI (Local File Include) and Directory Traversal. Due to the intercompany scanners used for emulating attacks, the input dataset contains various typical attack types and the total anomaly proportion is larger than conventional access logs.

4.2 Single Model

Since the features of classifier come from the statistical model which is constructed at the first stage, the performance of the whole system depends heavily on the validity of the statistical model. Meanwhile, the statistical model relies on modelling the normal access behavior, so the first 60% of normal records (according to the time order) are adopted to construct statistic models (471,123), while the rest of the whole records (314,082) are split medially into training and testing data for the detection model. The details are shown in Table 7.

In order to optimize the performance of detection model, in addition to selecting different classification algorithms, it needs to optimize the parameters of the classifiers. We mainly use KNN, Logistic regression, SVM, decision tree, random forest, GBDT and Xgboost as the classifier to validate the detection performance. For simplicity, in Table 8, we only list the performance of each model under the optimal parameters. And in order to reflect the magnitude of each model, the corresponding accuracy and recall are shown in Table 8.

Table 7. The record number of input dataset

Records	Statistic model	Training data	Testing data
Normal records (785,205)	471,123	157,041	157,041
Abnormal records (98,150)	0	49,075	49,075

Table 8. The detection performance of each single model

Model	Precision	Recall
Logistic Regression	0.94652	0.63212
SVM	0.97334	0.76735
KNN	0.99061	0.94924
Random Forest	0.99373	0.95137
GBDT	0.99376	0.96613
Xgboost	0.99625	0.97422

It can be observed from the Table 8 that the precision and recall of Xgboost is the highest, reaching 0.99625 and 0.97422, respectively. At the same time, KNN, decision tree, random forest, GBDT all have a high detection property after parameter tuning, which are maintained at least 95%. However, the recall of Logistic regression and SVM are only 63% and 76%, which are difficult to achieve the practical application requirements.

4.3 Ensemble Model

The ensemble framework adopts 2-fold Stacking, and uses the forward search strategy to select the base model of first stage. The second stage of the prediction algorithm generally uses the single model with the best detection performance that is Xgboost. Furthermore, in order to prevent the entire ensemble model from overfitting, Xgboost is no longer considered in the base model set of first stage. The overall flow of the model selection process is shown in Figure 5.

As shown in the Figure 5, the set of single model includes the prediction results of test data by different classifiers according to varies of sampling, parameters or algorithms. Whole selection process starts from the optimal result of single model, and add one column of prediction results to the ensemble result every time.

The selection criterion is that making prediction algorithm of second stage has optimal detection performance. By analogy, the selection process is carried out until the detection performance of the classifier after a round of ensemble is smaller than the threshold value T (here, we takes T = 0.0001) compared with the previous round. Finally, all single models corresponding to the ensemble model are found, then they are output as the parameters of the ensemble model together with the prediction algorithm of second stage. Specifically, based on the testing dataset of our experiment, the first stage consists of six models: two KNN models, three random forest models and one GBDT model. The structure of ensemble model is shown in Figure 6.

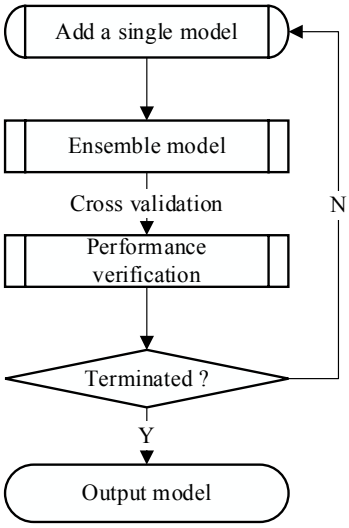

Figure 5. **Flow chart of model selection.**

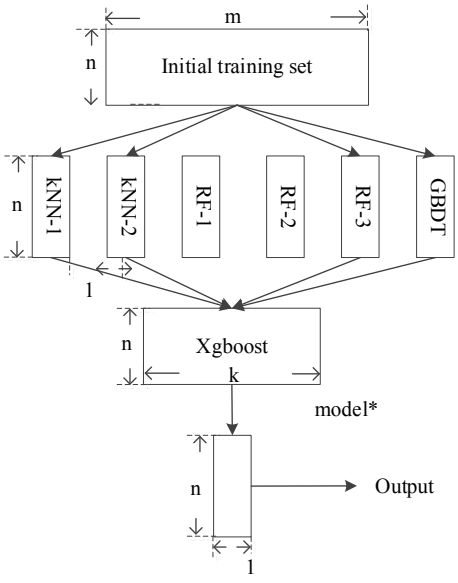

Figure 6. **The structure of ensemble model.**

The above experimental results show that based on the good but different single classifiers, and stacked generalization to integrate them, the precision and recall of the ensemble model are 0.99988 and 0.99647, respectively. Compared with the previous single model, the detection performance is improved obviously. It indicates that the generalization ability of the proposed ensemble model has been considerably enhanced.

5 CONCLUSION

TO deal with the limitation of misuse detection system and traditional intrusion detection system, we propose a comprehensive detection system by combining feature generating with prediction algorithms to achieve an integrated assessment on behavioral pattern of the web access records. In order to improve the detection performance of the system, we further build the ensemble model of different classifiers. Especially, using 2-fold Stacking as ensemble architecture, the detection precision and recall will reach 0.99988 and 0.99647 that are really remarkable.

In the future, we plan to continue the use of hierarchical approaches to detect web-based intrusions in unsupervised way, and we are going to concentrate on the improvement of detection performance by expanding the layers of ensemble architecture and adding self-adaption threshold to automatic update domain's statistical characteristics.

6 DISCLOSURE STATEMENT

NO potential conflict of interest was reported by the authors.

7 ACKNOWLEDGEMENTS

THIS work was partially supported by the National Basic Research Program of China (973 Program) under Grant No. 2013CB329102, in part by the National Natural Science Foundation of China (NSFC) under Grant No. 61372112 and 61003283. We would like to thank QIHU360 Corporation which made it possible to test our detection system on abundant labelled web logs from various web sites.

8 REFERENCES

S. A.-H. Baddar, A. Merlo, and M. Migliardi, (2014). Anomaly detection in computer networks: A state-of-the-art review. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications,* 5(4):29-64.

F. Barani, (2014). A hybrid approach for dynamic intrusion detection in ad hoc networks using genetic algorithm and artificial immune system. In *2014 Iranian Conference on Intelligent Systems,* pages 1-6.

T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean, (2007). Large language models in machine

translation. In *2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 858-867. Association for Computational Linguistics.

R. Bronte, H. Shahriar, and H. Haddad, (2016). Information theoretic anomaly detection framework for web application. In *2016 IEEE 40th Annual Computer Software and Applications Conference,* volume 2, pages 394-399. IEEE.

V. Chandola, A. Banerjee, and V. Kumar, (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1-15:58.

Z. X. Chen and H. Huang, (2011). A hybrid method for intrusion detection with ga-based feature selection. *Intelligent Automation & Soft Computing*, 17(2):175-186.

E. Dorj, C. Chen, and M. Pecht, (2013). A bayesian hidden markov model-based approach for anomaly detection in electronic systems. In *2013 IEEE Aerospace Conference,* pages 1-10.

W. K. G. Fan, (2012). An adaptive anomaly detection of web-based attacks. In *2012 7th International Conference on Computer Science Education,* pages 690-694. IEEE.

U. Fiore and F. Palmieri, (2013). Network anomaly detection with the restricted boltzmann machine. *Neurocomputing,* 122:13-23.

S. K. Gautam and H. Om, (2015). Anomaly detection system using entropy based technique. In *2015 1st International Conference on Next Generation Computing Technologies,* pages 738-743.

J. F. Guan, Z. W. Yan, S. Yao, C. Q. Xu, and H. K. Zhang, (2017). GBC-based caching function group selection algorithm for SINET. *Journal of Network and Computer Applications,* 85(C):56-63.

S. Y. Huang and Y. N. Huang, (2013). Network traffic anomaly detection based on growing hierarchical som. In *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks,* pages 1-2.

D. Ippoliti, C. Jiang, Z. Ding, and X. Zhou, (2016). Online adaptive anomaly detection for augmented network flows. *ACM Transactions on Autonomous and Adaptive Systems,* 11(3):17:1-17:28.

S. M. Jang and Y. H. Won, (2007). Web anomaly detection system for mobile web client. In *2007 Future Generation Communication and Networking,* volume 2, pages 543-547. IEEE.

A. Juvonen and T. Hamalainen, (2014). An efficient network log anomaly detection system using random projection dimensionality reduction. In *2014 6th International Conference on New Technologies, Mobility and Security,* pages 1-5. IEEE.

R. Kozik and M. Choras, (2015). Adapting an ensemble of one-class classifiers for a web-layer anomaly detection system. In *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pages 724-729. IEEE.

C. Kruegel and G. Vigna, (2003). Anomaly detection of web-based attacks. In *10th ACM Conference on Computer and Communications Security,* pages 251-261, New York, NY, USA. ACM.

H. K. Lau, J. Timmis, and I. Bate, (2009). Anomaly detection inspired by immune network theory: A proposal. In *2009 IEEE Congress on Evolutionary Computation,* pages 3045-3051. IEEE.

J. Li, J. Guan, and Z. Jiang, (2016). Multidimensional and hierarchical anomaly detection system of web attacks. In *2016 International Symposium on Mobile Internet Security,* pages 1-11.

H. Li and Q. Wu, (2013). Research of clustering algorithm based on information entropy and frequency sensitive discrepancy metric in anomaly detection. In *2013 International Conference on Information Science and Cloud Computing Companion,* pages 799-805.

G. Liang, (2008). Modeling unknown web attacks in network anomaly detection. In *2008 Third International Conference on Convergence and Hybrid Information Technology,* volume 2, pages 112-116. IEEE.

S. W. Lin and K. C. Ying, (2012). An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. *Applied Soft Computing,* 12(10):3285-3290.

P. Louvieris, N. Clewley, and X. Liu, (2013). Effects-based feature identification for network intrusion detection. *Neurocomputing,* 121(18):265-273.

E. Parhizkar and M. Abadi, (2015). Oc-wad: A one-class classifier ensemble approach for anomaly detection in web traffic. In *2015 23rd Iranian Conference on Electrical Engineering,* pages 631-636. IEEE.

D. W. Wang, Y. B. Xue, and Y. F. Dong, (2011). Anomaly detection using neighborhood negative selection. *Intelligent Automation & Soft Computing,* 17(5):595-605.

D. H. Wolpert, (1992). Stacked generalization*. *Neural Networks,* 5(2):241-259.

Y. Xie and S. Tang, (2012). Online anomaly detection based on web usage mining. In *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops PhD Forum,* pages 1177-1182. IEEE.

S. Yao, J. F. Guan, and H. K. Zhang, (2016a). Survivable strategy set design for malicious attack propagation in nemo scenario. *EURASIP Journal on Wireless Communications and Networking,* 2016(1):234.

Y. Yao, Z. Zhang, W. Yang, and F. X. Gao, (2016b). Influence of removable devices on worm propagation under pulse quarantine strategy. *Intelligent Automation & Soft Computing,* 22(4):651-666.

Y. Yu, (2012). A survey of anomaly intrusion detection techniques. *Journal of Computing*

Sciences in Colleges, 28(1):9-17.

G. Yuan, B. Li, Y. Yao, and S. Zhang, (2017). A deep learning enabled subspace spectral ensemble clustering approach for web anomaly detection. In *2017 International Joint Conference on Neural Networks,* pages 3896-3903.

H. K. Zhang, W. Quan, H.-C. Chao, and C. M. Qiao, (2016). Smart identifier network: A collaborative architecture for the future internet. *IEEE Network,* 30(3):46-51.

9 NOTES ON CONTRIBUTORS

Jianfeng Guan is an associate professor in the Institute of Network Technology at Beijing University of Posts and Telecommunications (BUPT). He received his B.S. degree from Northeastern University of China in July 2004, and received the Ph.D. degree in communications and information system from Beijing Jiaotong University in Jan. 2010. His research interests include Mobile Internet, network security and future network. He has published more than 80 papers in international journals and conferences including IEEE TMC, IEEE TVT, IEEE TOB, WCNC, ICC etc. He is involved in several TPCs.

Jiawei Li received his B.S. degree in Telecommunications Engineering from the Beijing University of Posts Telecommunications (BUPT) of China in June 2014. He is currently working toward the M.S. degree in the Institute of Network Technology at BUPT. His research interests include data mining, machine learning and network security.

Zhongbai Jiang received his B.S Degree in Zhengzhou University, China, in 2014. He is currently working toward a Ph.D. degree at the Institute of Network Technology, Beijing University of Posts and Telecommunications, China. His current research interests are in the areas of high-speed railway networks, wireless communications and wireless networking.