International Journal of Security and Its Applications
Vol.9, No.9 (2015), pp.321-336
http://dx.doi.org/10.14257/ijsia.2015.9.9.28

Intrusion Detection System based on Hidden Conditional Random Fields

Jun Luo[1] and Zenghui Gao[2]

Key Laboratory of Optoelectronic Technology & System, Ministry of Education, Chongqing University, Chongqing 400030, China
[1]luojun@cqu.edu.cn, [2]gzengh@163.com

Abstract

Intrusion detection is an important way to ensure the security of computers and networks. In this paper, a new intrusion detection system (IDS) is proposed based on Hidden Conditional Random Fields (HCRFs). In order to optimize the performance of HCRFs, we bring forward the Two-stage Feature Selection method, which contains Manual Feature Selection method and Backward Feature Elimination Wrapper (BFEW) method. The BFEW is a feature selection method which is introduced based on wrapper approach. Experimental results on KDD99 dataset show that the proposed IDS not only has a great advantage in detection efficiency but also has a higher accuracy when compared with other well-known methods.

Keywords: *Backward Feature Elimination Wrapper, HCRFs, Intrusion Detection System, Network Security, Two-stage Feature Selection*

1. Introduction

The emergence and development of the network has changed the life of human. Unfortunately, we have to confront a variety of risks of network intrusion while enjoying the convenience brought by the network. In order to reduce and avoid the risks, a series of intrusion prevention techniques like firewalls have been presented. However, with the development of intrusion methods, these traditional intrusion prevention techniques failed to protect computers and networks well gradually. Intrusion detection system (IDS), which was first proposed by Denning in 1987, has developed very well, because researchers value its effective ability to ensure the security of computers and networks.

The IDS monitors the events happening in a system dynamically, and decides whether these events are intrusions or just normal behaviors. IDSs can be classified as anomaly based or signature based according to the attack detection method. The anomaly-based systems are trained by means of the normal data only, and they are able to detect unknown intrusions. However, the anomaly-based systems are difficult to discover boundaries between normal and abnormal behavior, because they lacking abnormal instances during training. The signature-based systems learn specific characteristics from known attacks which have been found previously, and they can detect well-known intrusions both efficiently and accurately. But the signature-based systems fail to handle unknown intrusions. Another method for intrusion detection is to make use of both the normal and the abnormal behavior for training an IDS. This combines the advantages of both the anomaly-based and the signature-based methods. And such IDSs are known as the Hybrid Systems. One of them is introduced by Gupta et al. based on Conditional Random Fields (CRFs) [1].

Compared with other commonly used methods, CRFs performed more accurately. However, the detection efficiency, which is a very important element in real-time systems, is still too low. In this paper, a new intrusion detection system is proposed based on Hidden Conditional Random Fields (HCRFs). Unlike CRFs, HCRFs label each instance instead of each feature. This greatly improves the detection efficiency. We also proposed the Two-stage Feature Selection (TSFS) method to optimize the performance of HCRFs. And the experiments show that the proposed IDS not only improves the detection rate significantly, but also has a great advantage in detection efficiency.

The remainder of this paper is organized as follows: Section 2 provides a brief review of some notable related works. The HCRFs are described in Section 3, and the TSFS method is described in Section 4. Section 5 presents the framework of the proposed IDS. Experiments and results are presented in Section 6. Finally, in Section 7, we make a conclusion and discuss future works.

2. Related Works

As an important barrier to protect computers and networks from intrusions, intrusion detection system must be capable of detecting network intrusions accurately and processing large amounts of network traffic efficiently. In order to handle these two issues, researchers have brought forward a variety of methods, and have made unremitting efforts. Here we briefly review some related methods and frameworks.

Bouzida et al. [2] discussed the use of Decision Trees (DTs) for intrusion detection. They mentioned that among the various kinds of DTs algorithms, the C4.5 algorithm proposed by Quinlan generates better generalization accuracy since it finds high accuracy hypotheses based on pruning rules. The detection efficiency of DTs is generally very high, and the accuracy is also competitive.

In [3], the authors applied Support Vector Machines (SVMs) to intrusion detection. SVMs are capable of handling these datasets which have high dimension feature space. Although the feature space has very large dimension, infinite sometimes, they still present good generalization performances. SVMs can be used for both multiclass classification and binary-class classification.

The Naive Bayes (NB) method is also applied to intrusion detection [4]. However, there is a conditional independence assumption imposed on this method. Koc et al. [5] used hidden Naive Bayes (HNB) method for detecting intrusions. It relaxed the assumption of NB, and got a better result for the detection of DoS attacks.

Some paper used Hidden Markov Models (HMMs) for intrusion detection [6]. However, HMMs have the problem of label bias, and also have a conditional independence assumption like NB. Lafferty et al. [7] introduced Conditional Random Fields (CRFs), it solved the problems of HMMs. Gupta et al. [8] applied CRFs to intrusion detection, and achieved good detection results. However, the efficiency of this method is too low. In [1], Gupta et al. introduced layered approach into CRFs to improve the efficiency. From their experimental results, we can see that it achieved the intended effect, and the accuracy is also improved. However, compared with NB and DTs, layered CRFs have better detection accuracy, but it still failed in efficiency. And the accuracy of CRFs is also need to be improved further, especially R2L attacks.

HCRFs were first introduced by Quattoni et al. [9], they applied HCRFs to object and gesture recognition and their experiments show that the recognition accuracy of HCRFs is higher than CRFs. However, Quattoni et al. didn’t describe the efficiency of HCRFs. In fact, since HCRFs do not label each feature like CRFs, they can also achieve much higher detection efficiency. In this study we apply the HCRFs to intrusion detection to enhance network security. Paper [1] execute Manual Feature Selection (MFS) on each type of attacks, and their experimental results show that the MFS performs better than some commonly used automatic feature selection methods such as Neural Network and PCA. However, the number of features selected by MFS is still too much. Hence, we proposed the TSFS method which contained MFS and Backward Feature Elimination Wrapper (BFEW) to select relevant features and eliminate useless redundancies. And TSFS optimized the performance of HCRFs.

3. Hidden Conditional Random Fields for Intrusion Detection

By making use of hidden variables in the model, Hidden Conditional Random Fields get the ability to learn latent structures. Therefore, they can be more accurate than CRFs.

Our task of intrusion detection is to tab each network record *r* with a state label *l*. Each *r* is a vector of features $r=\{r_1,\cdots,r_m\}$. As shown in Figure 1, in a CRF model, every feature r_j has a corresponding label l_j, so each 'label' *l* can also be expressed as $\{l_1,\cdots,l_m\}$. The label sequences can be observed completely on training instances [9]. However, in a HCRF model, every label *l* is an unstructured label of attack types, and there is no direct relationship between *l* and each feature r_j. Instead, for each instance *r*, a vector of hidden variables $h=\{h_1,\cdots,h_m\}$ is assumed. However, they can not be observed on training instances.

The conditional probabilistic model of HCRFs can be defined as the following form:

$$p(l,h\,|\,r,\vartheta)=\frac{\exp(\varphi(l,h,r;\vartheta))}{\sum_{l'h}\exp(\varphi(l',h,r;\vartheta))}. \tag{1}$$

Where *r* are the instances, *l* are the labels, ϑ are the parameters, and $\varphi(l,h,r;\vartheta)$ is a potential function. The form of $P(l\,|\,r,\vartheta)$ is as follows:

$$p(l\,|\,r,\vartheta)=\sum_h P(l,h\,|\,r;\vartheta)=\frac{\sum_h\exp(\varphi(l,h,r;\vartheta))}{\sum_{l'h}\exp(\varphi(l',h,r;\vartheta))}. \tag{2}$$

HCRFs can be modeled as an undirected graph structure *G*=(*V*,*E*). The hidden variables h_j correspond to vertices *V*, and the set of graph edges $(j,k)\in E$ correspond to the lines between vertices h_j and h_k.

The potential function $\varphi(l,h,r;\vartheta)$ can be defined as follows:

$$\varphi(l,h,r;\vartheta)=\sum_{j=1}^{m}\sum_{v\in V_f} f_{1,v}(j,l,h_j,r)\vartheta_{1,v}+\sum_{(j,k)\in E}\sum_{e\in E_f} f_{2,e}(j,k,l,h_j,h_k,r)\vartheta_{2,e}. \tag{3}$$

Where V_f, E_f are the sets of vertices and edge features, $\vartheta_{1,v}$, $\vartheta_{2,e}$ are the components of ϑ. $f_{1,v}$, $f_{2,e}$ are feature functions. The f_1 and f_2 features depend on single and pairs hidden variable values, respectively.

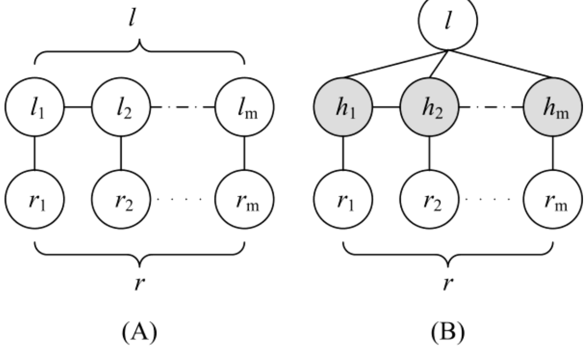


Figure 1. (A) The Model of CRF; (B) The Model of HCRF

As an undirected graph, parameter estimation of the HCRF model can be carried out with the use of standard graphical model algorithms. Paper [10] shows L-BFGS method performs better than other parameter optimization methods such as BFGS, B-L, PQN and CONMIN. Therefore, we use L-BFGS method to estimate the parameters of HCRFs.

For the training, the structures of HCRFs are generally very complicated and unpredictable. So it is meaningless to discuss the time complexity for training HCRFs. Since HCRFs have the ability to learn latent structures, the training time of them is supposed to be longer than CRFs. However, the experimental results, which will be given in Section 6.2, show that the number of iterations of HCRFs is less than CRFs. And the actual result is that the training time of the two methods are basically the same. For the test, the time complexity of HCRFs is $O(L^2N)$, where L is the number of labels, N is the number of instances. However, for binary-class classification, classifiers only need to determine whether an instance is normal (anomaly), so the time complexity can be simplified to $O(LN)$. Under the same condition, CRFs take $O(TL^2N)$ during inference, where T is the number of features [1]. Hence, theoretically, the detection efficiency of HCRFs is TL times the CRFs.

For intrusion detection, we made several experiments with all the 41 features of KDD99 dataset [11] in the beginning, but the performances are invalid. Then we use the features selected by MFS, and obtained remarkable results. The efficiency of HCRFs is much higher than CRFs, and except DoS attacks, the performances of HCRFs are also better than CRFs. This shows that HCRFs are more suitable for processing data with smaller number of features. In order to gain the optimal results, we make backward feature elimination after MFS. Feature selection method will be introduced next.

Table 1. All the 41 Features of KDD99 Dataset

(Num.)Feature name		
(1) duration	(15) su_attempted	(29) same_srv_rate
(2) protocol_type	(16) num_root	(30) diffsrv_rate
(3) service	(17) num_file_creations	(31) srv_diff_host_rate
(4) flag	(18) num_shells	(32) dst_host_count
(5) src_bytes	(19) num_access_files	(33) dst_host_srv_count
(6) dst_bytes	(20) num_outbound_cmds	(34) dst_host_same_srv_rate
(7) land	(21) is_hot_login	(35) dst_host_diff_srv_rate
(8) wrong_fragment	(22) is_guest_login	(36) dst_host_same_src_port_rat
(9) urgent	(23) count	(37) dst_host_srv_diff_host_rate
(10) hot	(24) srv_count	(38) dst_host_serror_rate
(11) num_failed_logins	(25) serror_rate	(39) dst_host_srv_serror_rate
(12) logged_in	(26) srv_serror_rate	(40) dst_host_rerror_rate
(13) num_compromised	(27) rerror_rate	(41) dst_host_srv_rerror_rate
(14) root_shell	(28) srv_rerror_rate	

4. Two-stage Feature Selection

Paper [12-14] demonstrate that feature selection can improve both efficiency and accuracy of intrusion detection. Hence, we propose a proper feature selection method which named Two-stage Feature Selection to optimize the performance of HCRFs. The TSFS method contains Manual Feature Selection and Backward Feature Elimination Wrapper. Each instance of KDD99 dataset has 41 different features, and Table 1 shows the number and name of each feature. For better use, the nominal features such as *protocol_type*, *service* and *flag* are converted into numeric features. Take the *protocol_type* for example, we replace its three states {'udp', 'tcp', 'icmp'} with {1, 2, 3}. There are four attack types (DoS, Probe, R2L, and U2R) in KDD dataset. However, the most relevant features in each type are not the same. So it can make IDSs perform better by selecting features for every type of attacks separately rather than selecting the same features for all types.

4.1. Manual Feature Selection

Gupta *et al.* introduced Manual Feature Selection (MFS) method in paper [1], and their experimental results show that the MFS performs better when compared with some commonly used automatic feature selection methods such as Neural Network and PCA. MFS selects correlate features by analyzing the characteristics of each type of attacks as follows:

- **Denial of Service (DoS):** As a very commonly found type of attacks with many varieties, DoS attacks attempt to block access to certain services by flooding them with illegitimate requests. Therefore, features about traffic and packet level are significant for DoS attacks.
- **Probe:** Attacks of this type aim to gain useful information about hosts by scanning the network in various modes. So, basic connection level features are important for Probe attacks.
- **Remote to Local (R2L):** Attacks of this type attempt to gain access to the remote victim machine without having an account on it. This behavior affects the value of both the network level and the host level features. The R2L attacks are extremely difficult to detect, almost all IDSs failed to perform well on it.
- **User to Root (U2R):** U2R attacks are also one of the most difficult attacks to detect. They aim to gain system's superuser privileges. Hence, the U2R attacks involve the details of lexemes. And they are usually based on the content and the target application.

Table 2 shows the features selected by MFS. Experimental results in Section 6.2 can demonstrate that it will get very good performance when applying these selected features to HCRFs classifier. In order to gain optimal results, we propose a new method to reduce redundancies of these selected features further after MFS.

Table 2. The Features Selected by MFS

	Features (Num.)								
DoS	1	2	4	5	23	34	38	39	40
Probe	1	2	3	4	5				
R2L	1	2	3	4	5	10	11	12	13
	17	18	19	21	22				
U2R	10	13	14	16	17	18	19	21	

4.2. Backward Feature Elimination Wrapper

In theory, features selected by MFS are related to their corresponding type of attacks. However, except Probe attacks, the number of selected features are still too much and they may contain redundancies which need to be eliminated. Hence, we propose a new method to reduce the redundancies and optimize the performance of HCRFs.

Filter methods and wrapper methods are two main methods that deal with feature selection [15-16]. Their characteristics are as follows:

- **Filter methods:** Rely on the general characteristics of the training data; select features with independence of any classifier; usually computationally less expensive than wrapper methods.
- **Wrapper methods:** Involve a classifier as part of the selection process; tend to give better results than filter methods.

The wrapper methods will not spend too much time since the number of features has been reduced by MFS. Therefore we choose to use the wrapper methods to eliminate the redundancies.

Depending on the initial state, the wrapper methods have two forms: forward selection and backward elimination. The forward selection method begins at the empty set of features, and may spend less time to build classifiers in the beginning than backward elimination. However, the backward elimination method, which begins at the full set of features, can capture interacting features more easily [15]. Hence, we choose to use backward elimination method. And we apply the greedy hill-climbing search engine to our method since algorithms with this search engine are simple to implement and fast in producing results [16].

We name our feature elimination method as Backward Feature Elimination Wrapper (BFEW), and its structure is shown in Figure 2. Firstly, get initial detection result Q with HCRFs classifier and all the *n* features selected by MFS. Then get every Q(*i*) with HCRFs classifier and *n*-1 features which don't contain the *i*th feature. If the maximal Q(*i*) is greater than Q, start a new cycle. If not, output the *n* features.

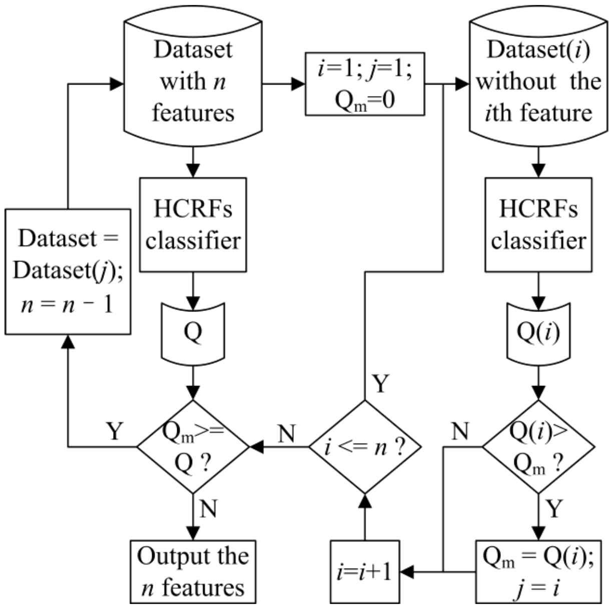


Figure 2. The Structure of the Backward Feature Elimination Wrapper Method; Q is an Evaluation Index of Detection Performance

5. Framework of the IDS

Paper [17] introduced a framework with three layers corresponding to three aspects of security, viz. availability, confidentiality and integrity. In [1], Gupta et al. applied the layered approach to CRFs classifier, and this improved the efficiency of the overall system. In this text, we propose a TSFS method to select the most targeted features for each layer, and use the HCRFs for detecting intrusions. Figure 3 shows the framework structure of the IDS.

In this framework, we select features for every layer with the TSFS feature selection method which contains of MFS and BFEW. In each layer we only select data of the corresponding attack class and normal class. We use HCRFs to train detection models for each layer. And corresponding intrusions can be identified from the testing dataset by means of these models. If one instance is recognized as an intrusion in a layer, it will be labeled with the relevant type of attacks. Those instances which pass through the detection models of all layers will be labeled as Normal. The order of layers in our framework relies on the number of each type of attacks in the training dataset. For example, DoS attacks have the largest number, so it is arranged in the first layer. It is worth noting that the arrangement of each layer can be adjusted according to the specific situations. Experimental results in the next section demonstrate that the proposed IDS have both high detection efficiency and high detection accuracy.

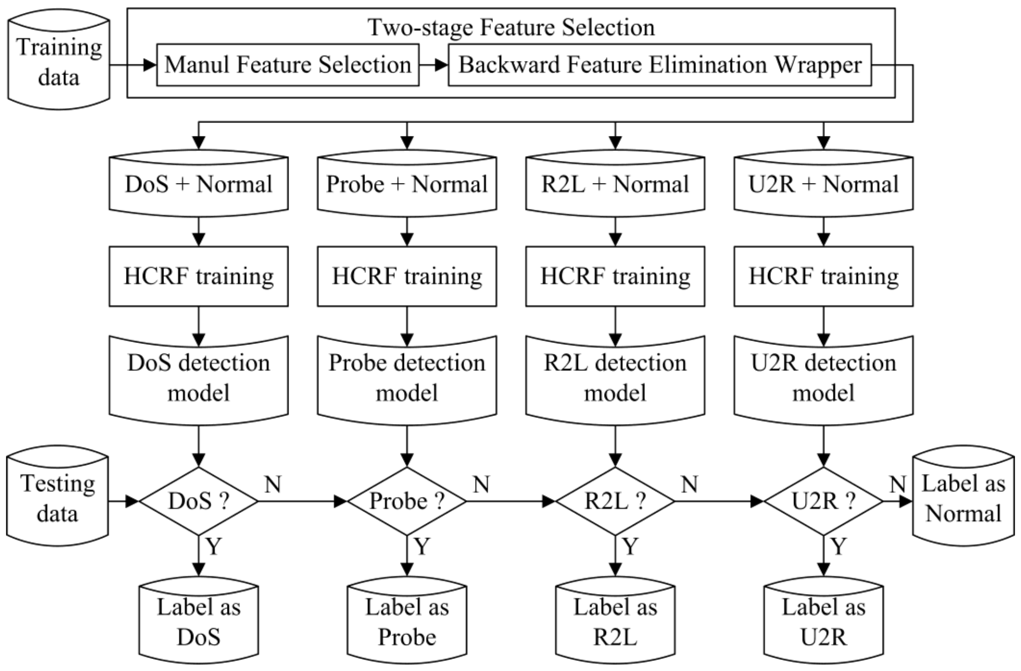


Figure 3. The Framework of the Intrusion Detection System based on Hidden Conditional Random Fields

6. Experiments and Results

The KDD99 dataset is the most widely used dataset in the intrusion detection experiments. In order to ensure the comparability and credibility of the experimental results, we use 10 percent of the total training data and 10 percent of the test data (with corrected labels) of the KDD99 dataset [11] as training dataset and testing dataset. In the training dataset, every instance is either labeled as normal or as one of the 24 different kinds of attack. And the 24 kinds of attack can be divided into four types (DoS, Probe, R2L, and U2R). The testing dataset contains 311,029 instances. And these instances are also either labeled as normal or as one of the attacks belonging to the four attack types. Distribution of instances in training dataset and testing dataset is shown in Table 3.

For experiments with HCRFs, we use HCRF2.0b [18]. And we use the CRF tool of crfsuite0.12 [19] to perform experiments with CRFs. All experiments are done on the same computer running with Pentium (R) 4, CPU 2.8 GHz, and 1.48-GB RAM.

Table 3. The Distribution of Instances in Training Dataset and Testing Dataset

	Training dataset	Testing dataset
Normal	97,278	60,593
DoS	391,458	229,853
Probe	4,107	4,166
R2L	1,126	16,347
U2R	52	70
Total	494,021	311,029

Precision, Recall, and F-Value are useful indexes for evaluating classifiers. They are defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}.$$

$$\text{Recall} = \frac{TP}{TP + FN}.$$

$$\text{F-Value} = \frac{(1+\beta^2) * \text{Recall} * \text{Precision}}{\beta^2 * (\text{Recall} + \text{Precision})}.$$

Where *TP* are the number of True Positives, *FP* are the number of False Positives, and *FN* are the number of False Negatives. And β refers to the relative importance of Precision versus Recall and is usually set to 1 (In this case, the F-Value could be called F-1 Value).

However, every experiment has more than one label, and every label gets a set of Precision, Recall, and F-Value in the result. Paper [1] only use the set of the attack label and ignore the normal, when there are only these two labels in an experiment. This cannot effectively reflect the overall detection results. To handle this problem, we apply Macro-Average to the classifier evaluation. It can be defined as the following form:

$$\text{Macro-Average}(s) = (s_1 + s_2 + \cdots + s_n) / n.$$

Where s_i represents an evaluation index of the results of each class. The Macro-Average weights classes equally, the smaller class can get enough attention, even if the proportion of these classes in the test data is not balanced [20]. We use Macro-Average Precision, Macro-Average Recall, and Macro-Average F-Value for performance evaluation. For convenience, we still present them as Precision (Prec.), Recall, and F-Value (F-1).

6.1. Feature Selection with BFEW

The BFEW method has been described in detail in Section 4.2. Here we use F-Value which can be got by HCRFs classifier as the detection result Q(F), and make feature selection with BFEW for DoS, Probe, R2L, and U2R attacks, separately. For every type of attacks we randomly select data of the corresponding attack class and normal class only as the training data. Since the number of instances of DoS attacks in testing dataset is too much, we only use part of DoS attacks and part of normal instances for testing. While for other types we use all corresponding instances. The detailed data is shown in Table 4. In order to ensure high selection efficiency and minimize interference, for each type of attacks, we make all feature selection experiments with the same dataset.

The feature elimination process is shown in Figure 4. Each loop in BFEW eliminates a feature until Qm(F)<*Q*(F). Take the feature elimination process of DoS attacks, for example. In the first loop, the F-Value (Q(F*i*)) can obtain the maximum value when the feature with the serial number 40 in Table 1 (feature 40) is eliminated, and Q(F*i*)>=Q(F). Hence, we eliminate the feature 40 firstly. In the sixth loop, eliminating the feature 38 can obtain the maximal Q(F*i*), but Q(F*i*)<Q(F), so feature 38 cannot be eliminated and the feature elimination process of DoS attacks is over. All the F-Value here is the best one in five repeated experiments.

For DoS class, the F-Value improves notably when feature 5 and feature 23 are eliminated. However, no matter which feature is eliminated in Probe class, the F-Value will drop. We can also observe that most of the features eliminated in R2L class have almost no impact on F-Value. The F-Value in U2R class can reach its highest point after eliminating 8 features.

Table 4. The Data Sets for Feature Selection with BFEW

	Training data		Testing data	
	Corresponding attacks	Normal	Corresponding attacks	Normal
DoS	2,000	2,000	16,000	4,000

Probe	4,107	4,107	4,166	60,593
R2L	1,126	1,126	16,347	60,593
U2R	52	100	70	60,593

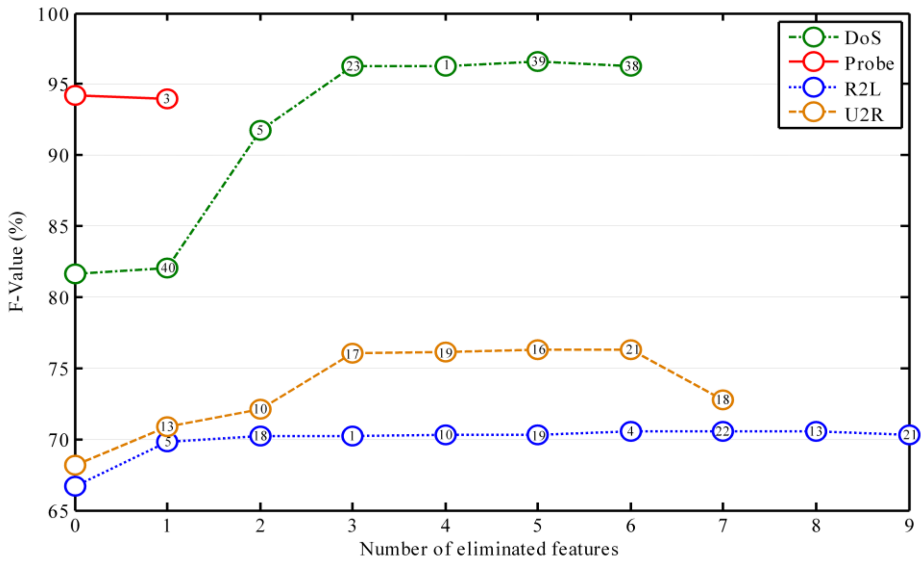


Figure 4. The Feature Elimination Process; The Numbers on the Polyline Represent the Corresponding Features which are Eliminated

Table 5. The Features Selected with TSFS

	Feature number					
DoS	2	4	34	38		
Probe	1	2	3	4	5	
R2L	2	3	11	12	17	21
U2R	14	18				

The final results of the four attack types with the TSFS including MFS and BFEW are shown in Table 5. From this table, we can see that compared with the status in Table 2, features of DoS, R2L and U2R have reduced more than a half, while features of Probe attacks have no change since it is already few after MFS.

6.2. Detection Models Building for Each Layer

In this section we use the HCRFs and the final selected features with TSFS to build detection models for every layer of the IDS. We still use the data shown in Table 4 for training and testing, except in DoS layer we make use of all the 229,853 instances of DoS attacks and 60,593 normal instances for testing. Under the same condition, for each layer we build the other two models, CRFs with MFS and HCRFs with MFS. Results comparisons of these three models are shown in Table 6. For fair comparison, all results are the average of ten repeated experiments. And we re-selected training data for each experiment.

As is shown in Table 6, for DoS attacks, when both perform with MFS, CRFs possess better detection results than HCRFs, while the latter perform faster. However, the detection results of HCRFs with TSFS improve a lot, and they are almost similar to the results of CRFs. For Probe attacks, since the BFEW did not eliminate any feature selected by MFS, there are only one set of results of HCRFs here. Obviously, HCRFs perform better than CRFs. For R2L attacks, CRFs with MFS get the best Precision while it also have the worst Recall and F-Value, the best Recall and F-Value are got by HCRFs with TSFS. And for U2R attacks, CRFs with MFS have the best Recall, while HCRFs with MFS perform better than CRFs on the aspects of Precision and F-Value. However, the best results on these two aspects are got by HCRFs with TSFS.

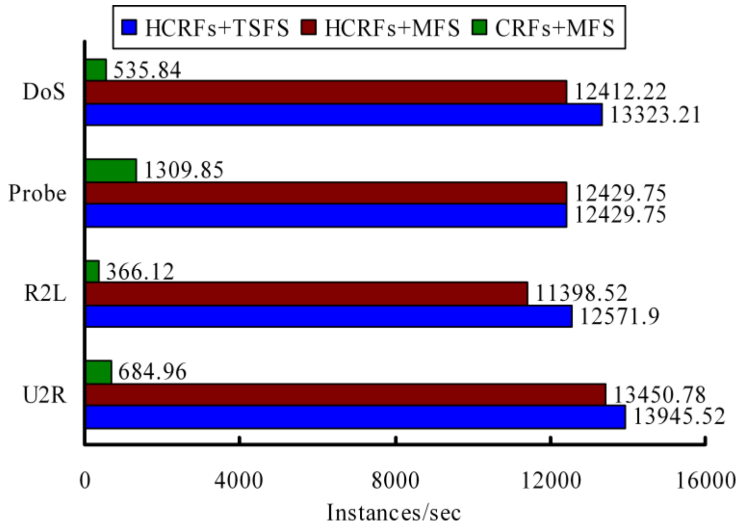


Figure 5. The Comparison of Detection Efficiency

From Table 6, we can also observe that under the same condition of MFS, the time for training has no significant change. Although HCRFs consumed more time in each iteration, the number of iterations of HCRFs is also less. The detection efficiency which is one of the most important elements of real-time IDSs has changed qualitatively. Furthermore, the changes further exacerbate when HCRFs perform with TSFS. The detection efficiency of HCRFs is much higher than CRFs. By comparing the number of instances detected by these models per sec, Figure 4 shows the efficiency improvement of HCRFs more intuitive. In Section 3, we have mentioned that the efficiency of HCRFs is TL times the CRFs theoretically. Under the same condition of MFS, CRFs can only manage several hundred instances per second, while this number of HCRFs can reach more than ten thousand.

Table 6. Results Comparisons of Different Models in Each Layer

		Prec. (%)	Recall (%)	F-1 (%)	Train (s)	Iterations	Test (s)
DoS	CRFs + MFS	**95.2622**	**98.5420**	**96.7805**	**23.88**	118.0	542.04
	HCRFs + MFS	79.4619	90.0294	82.1171	51.30	**38.3**	23.40
	HCRFs + TSFS	94.6817	98.0874	96.2511	30.23	46.5	**21.80**
Probe	CRFs + MFS	87.4077	97.6270	91.7783	**22.70**	111.0	49.44
	HCRFs + MFS/ HCRFs + TSFS	**90.0132**	**98.9603**	**93.9488**	49.28	**19.9**	**5.21**
R2L	CRFs + MFS	**88.5818**	62.4824	65.3174	26.08	161.4	210.18
	HCRFs + MFS	74.3459	64.1033	66.4430	65.13	100.8	6.75
	HCRFs + TSFS	79.7947	**67.2046**	**70.3686**	**23.33**	**54.1**	**6.12**
U2R	CRFs + MFS	56.2318	**84.0085**	60.4246	1.40	68.0	88.56
	HCRFs + MFS	59.5123	83.0390	64.4324	1.03	30.5	4.51

HCRFs + TSFS	**78.7839**	74.2651	**76.3314**	**0.75**	**24.3**	**4.35**

In [6], the authors compared the test time of CRFs, Naive Bayes and Decision Trees. The comparison of detection efficiency of HCRFs and these methods is presented in Table 7. Each value in the table is calculated as the test time of corresponding method over the test time of CRFs. Table 7 shows that the CRFs are the slowest because all values are greater than 1. The Decision Trees are more efficient than Naive Bayes. HCRFs have a great advantage in detection efficiency. Particularly they per-form over 30 times faster than CRFs for detecting R2L attacks.

Table 7. The Detection Efficiency Comparison

	DoS	Probe	R2L	U2R
HCRFs	**23.16**	**9.49**	**31.13**	**19.64**
Naïve Bayes	2.33	1.81	1.99	1.46
Decision Trees	3.92	2.04	4.17	2.87

Each value in the table is the ratio of test time of the corresponding method and CRFs.

6.3. The Overall IDS Performance

The detection models for every layer have been built with HCRFs and TSFS described in Section 6.2, here we choose the best models in each layer to build our overall IDS. We use all the 311,029 instances in the 10 percent of the test data to test our IDS. In the first layer, each instance is labeled as ether DoS or Normal, then these Normal instances are sent to the next layer. Part of these instances are labeled as Probe, the other instances with the Normal label continue to be sent to the next layer. The same process is repeated at the later two layers. The confusion matrix in Table 8 shows the results of this experiment.

In Table 8, the values refer to the percent detection with respect to each of the five classes. We can see from the confusion matrix that 80.34 percent of Probe attacks are labeled as DoS in the first layer. There is no problem because it is extremely important to detect an intrusion as soon as possible to minimize its impact in reality. And experimental results in Section 6.2 have proved that one type of attacks can be well detected by the detection model of itself. Therefore, for a type of attacks, no matter detected in which layer, all of the detected instances should be count on when calculating the detection rate. The detection rate can be defined as the following form:

$$\text{Detection rate} = \frac{Detected}{Total}.$$

Where *Detected* refers to the number of all the detected instances of a class, and *Total* refers to the total number of instances of this class.

The detection rate of each class is shown in the final column of Table 8. We can observe that our IDS does a great job. 100 percent of Probe and U2R attacks are detected, and DoS attacks are detected 97.37 percent. The detection rate of R2L attacks achieved 37.95 percent, which improves 28.21 percent when compared with layered CRFs [6].

Table 8. Confusion Matrix

Actual	Predict					Detection rate
	DoS	Probe	R2L	U2R	Normal	
DoS	97.1434	0.0009	0.2210	0	2.6347	97.3653
Probe	80.3409	19.6111	0.0480	0	0	100

R2L	1.1011	25.5888	11.2620	0.0122	62.0358	37.9642
U2R	0	10.0000	90.0000	0	0	100
Normal	0.5182	1.4804	2.3452	0.0281	95.6282	95.6282

In Section 5 we have explained that we use the detection model of DoS attacks as the first layer of the IDS because the number of instances of DoS attacks in the training dataset is larger than others. This idea has universal significance, because we don't know which type of attacks in number is larger in the real environment. However, in the testing dataset, R2L attacks has the larger number of instances than Probe attacks, so we adjust our IDS by exchanging the positions of the detection models of R2L and Probe to make it more targeted. And we remove the last layer since all the U2R attacks can be detected in the front layers. For convenience, we call the IDS before adjusted as Universal IDS, and call the adjusted IDS as Targeted IDS. The confusion matrix of the Targeted IDS is shown in Table 9.

Table 9. The Confusion Matrix of Targeted IDS

Actual	Predict				Detection rate
	DoS	R2L	Probe	Normal	
DoS	97.1434	0.2210	0.0009	2.6347	97.3653
R2L	1.1011	36.8508	0	62.0481	37.9519
Probe	80.3409	9.4575	10.2016	0	100
U2R	0	100	0	0	100
Normal	0.5182	2.9822	0.8433	95.6563	95.6563

From Table 9, we observe that the detection rate of each type of attacks of Targeted IDS is almost the same with Universal IDS which is shown in Table 8. However, the efficiency of the overall IDS has been improved dramatically. This can be observed in Table 10 which shows the percentage of attacks detected in each layer and the time each layer spend. Attacks detected in Layer 2 of the Targeted IDS are much more than the same layer of the Universal IDS. The Layer 4 of the Universal IDS almost has no contribution to the detection effect, but it still spend a lot of time. More than 90 percent of attacks are detected in Layer 1 of both the two IDSs. By the end of the Layer 2, the Targeted IDS has not only detected more attacks than the Universal IDS, but spent a shorter time. Finally, the whole time spent by the Targeted IDS is more than 13 percent shorter compared with the Universal IDS, while its overall detection result is almost as good as the other.

Table 10. The Performance of Each Layer

		Layer 1	Layer 2	Layer 3	Layer 4	Total
Universal IDS	Attacks (%)	90.5677	2.0001	0.9639	0.0008	93.5325
	Time (sec)	23.06	6.70	6.28	5.34	41.38
Targeted IDS	Attacks (%)	90.5677	2.7935	0.1705	\	93.5317
	Time (sec)	23.06	6.66	5.93		35.65

Table 11. The Detection Rate Comparison with Other IDSs

	DoS	Probe	R2L	U2R	Norma l	Macro-Average
Proposed IDS	97.37	**100**	37.95	**100**	95.66	**86.20**
XCS-ANN[21]	98.80	90.50	34.60	88.50	99.10	82.30

Layered CRFs[1]	97.40	98.60	29.60	86.30	98.62	82.10
LSSVM+MMIFS[14]	78.69	86.46	**84.85**	30.70	95.15	75.17
SVM + BIRCH[3]	**99.53**	97.55	28.81	19.73	99.30	68.98
Pocket Mcl[13]	97.65	86.79	11.45	54.38	91.86	68.43
NB+PKID+Cons[13]	95.12	96.67	13.85	11.40	96.13	62.63
5FNs-Poly[13]	96.77	85.96	29.43	8.33	92.45	62.59
SVM RBF[13]	97.30	79.26	18.29	25.88	91.83	62.51
IP+FAR[22]	95.80	79.20	12.60	21.10	92.90	60.32
KDD'99 Winner[13]	97.10	83.30	8.40	13.20	**99.50**	60.30
DSSVM[23]	97.20	87.50	6.30	3.10	98.40	58.50

Ranked by the Macro-Average value.

6.4. Comparison with Other IDSs

We have compared the proposed IDS with the IDS based on CRFs layer by layer in Section 6.2, and the results show that our IDS not only have a better detection effect, but also more efficient. In this section, we compare the proposed method with KDD99 Winner and some other methods which also use 10 percent of the test data (with corrected labels) of the KDD99 dataset for testing [1, 3, 13, 21, 22, 23]. It is not possible to use the evaluation indexes employed in the prior comparisons in Section 6.2, and only the detection rate of attacks for each class is provided (the last column of Table 9). Table 11 shows the comparison of these IDSs with detection rate in percent.

There are five classes in the KDD99 dataset including Normal class. And the number of instances of different class has huge disparities. In this case if we don't consider each class separately, the test result of the larger class will affect the overall testing results excessively. Therefore, the detection rate of each class is presented separately in Table 11. And the table shows obviously that an IDS always gets a good performance in a class but fails in another. In order to evaluate the overall system fairly, we make use of Macro-Average detection rate to evaluate the IDSs. As is mentioned in the beginning of Section 6, the Macro-Average which takes enough attention to the smaller class can weight classes equally, even if the proportion of these classes in the test data is not balanced.

From Table 11, we observe that our IDS provides the best detection rate for Probe and U2R attacks. And both of them were detected 100 percent. The R2L attacks are extremely difficult to detect. The proposed IDS detected 37.95 percent of R2L attacks, which gets the second place. In this item, the LSSVM with MMIFS method did amazingly well, detected more than 80 percent of R2L attacks. However, several other test results of it are not satisfactory. For DoS attacks the SVM with BIRCH method gets the best result. And the KDD99 Winner performs best on the Normal class. Overall, although the proposed IDS does not get the best results in all categories, its performance is very stable and not has any obvious shortcomings. In terms of the Macro-Average detection rate, the proposed IDS performs better than all other methods.

7. Conclusion and Future Works

In this paper, a new IDS was proposed based on HCRFs. In order to optimize the performance of HCRFs, we introduced the TSFS feature selection method which contained MFS and BFEW. And the BFEW method is proposed by means of the wrapper approach. We made use of KDD99 dataset to train and test our IDS. Since the distribution of attack types in the test dataset is not the same as the training dataset. We adjusted our IDS for making it more targeted and efficient. The experimental results show that the proposed IDS not only has great advantage in detection efficiency, but also has better accuracy when compared with other well-known methods such as Naïve Bayes, SVMs and ANN.

Under the same classification method, the performance can be improved notably when making use of a better feature selection method. Therefore, in the future works, feature selection algorithm is a very meaningful research direction.

Acknowledgments

This work was supported by the project of Scientific Research Plan of Chongqing Economic Information Committee, under grant 10-cxy-02.

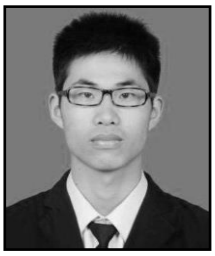
References

[1] KK Gupta, B Nath and K Ramamohanarao, "Layered approach using conditional random fields for intrusion detection", IEEE Transactions on Dependable and Secure Computing, vol. 7, no. 1, **(2010)**, pp. 35-49.
[2] Y Bouzida and F Cuppens, "Neural networks vs. decision trees for intrusion detection", IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation (MonAM), **(2006)**.
[3] SJ Horng, MY Su, YH Chen, TW Kao, RJ Chen, JL Lai and CD Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines", Expert Systems with Applications, vol. 38, **(2011)**, pp. 306-313.
[4] S Mukherjeea and N Sharma, "Intrusion Detection using Naive Bayes Classifier with Feature Reduction", Procedia Technology, vol. 4, **(2012)**, pp. 119-128.
[5] L Koc, TA Mazzuchi and S Sarkani, "A Network Intrusion Detection System Based on A Hidden Naive Bayes Multiclass Classier", Expert Systems with Applications, vol. 39, no. 18, **(2012)**, pp. 13492-13500.
[6] PH Wang, L Shi, BZ Wang, YQ Wu and YB Liu, "Survey on HMM based anomaly intrusion detection using system calls", 5th International Conference on Computer Science and Education (ICCSE), **(2010)** Aug. 24-27, pp. 102-105.
[7] J Lafferty, A McCallum and F Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", In Proceedings of the Eighteenth International Conference on Machine Learning, **(2001)**, pp. 282-289.
[8] KK Gupta, B Nath and K Ramamohanarao, "Conditional Random Fields for Intrusion Detection", 21st International Conference on Advanced Information Networking and Applications Workshops, **(2007)** May 21-23, pp. 203-208, Niagara Falls, Canada.
[9] A Quattoni, M Collins and T Darrell, "Conditional random fields for object recognition", In proceeding of: Advances in Neural Information Processing Systems 17, **(2004)** Dec. 13-18, pp. 1097-1104, Vancouver, British Columbia, Canada.
[10] J Vlcek and L Luksan, "Generalizations of the limited-memory BFGS method based on the quasi-product form of update", Journal of Computational and Applied Mathematics, vol. 241, **(2013)**, pp. 116-129.
[11] KDD99 Cup Dataset, http://kdd.ics.ics.uci.edu/database/kddcup99/kddcup99.html (accessed Dec. 2014), **(1999)**.
[12] Li, J Xia, S Zhang, J Yan, X Ai and K Dai, "An efficient intrusion detection system based on support vector machines and gradually feature removal method", Expert Systems with Applications, vol. 39, no. 1, **(2012)**, pp. 24-430.
[13] V Bolón-Canedo, N Sánchez-Maroño and A Alonso-Betanzos, "Feature selection and classification in multiple class datasets: an application to KDD Cup 99 dataset", Expert Systems with Applications, vol. 38, no. 5, **(2011)**, pp. 5947-5957.
[14] F Amiri, MR Yousefi, C Lucas, A Shakery and N Yazdani, "Mutual information-based feature selection for intrusion detection systems", Journal of Network and Computer Applications, vol. 34, no. 4, **(2011)**, pp. 1184-1199.
[15] R Kohavi and G John, "Wrappers for feature subset selection", Artificial Intelligence Journal, vol. 97, no. 1-2, **(1997)**, pp. 273-324.
[16] H Liu and L Yu, "Toward Integrating Feature Selection Algorithms for Classification and Clustering", IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 4, **(2005)**, pp. 491-502.
[17] KK Gupta, B Nath and R Kotagiri . "Network Security Framework". International Journal of Computer Science and Network Security, vol. 6, no. 7, **(2006)**, pp. 151-157.
[18] HCRF2.0b, http://sourceforge.net/projects/hcrf.html (accessed Dec. 2014), **(2011)**.
[19] O Naoaki, "CRFsuite: a fast implementation of Conditional Random Fields (CRFs)", http://www.chokkan.org/software/crfsuite (accessed Dec. 2014), **(2007)**.
[20] DD Lewis, "An evaluation of phrasal and clustered representations on a text categorization task", In Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, **(1992)** June 21-24, pp. 37-50, Copenhagen, Denmark.

[21] W Alsharafat, "Applying Artificial Neural Network and eXtended Classifier System for Network Intrusion Detection", The International Arab Journal of Information Technology, vol. 10, no. 3, **(2013)**, pp. 230-238.
[22] L Zhang, LG Meng and CJ Hou, "Intrusion Detection Based on Immune Principles and Fuzzy Association Rules", Intelligence Computation and Evolutionary Computation, vol. 180, **(2013)**, pp. 31-35.
[23] C Guo, YJ Zhou, Y Ping, ZK Zhang, GL Liu and YX Yang, "A distance sum-based hybrid method for intrusion detection". Appl Intell, vol. 40, **(2014)**, pp. 178-188.

Authors

Jun Luo, he is a Professor of the School of Optoelectronic Engineering of Chongqing University, China. He received his M.E degree from Chongqing University in 1990. In 1994 he joined the School of Optoelectronic Engineering and engaged in teaching and research work. Now he is the director of Embedded System Laboratory. His interests include computer and network security, trusted embedded systems, and trusted computing, etc.



Zenghui Gao, he received his B.E degree from Shandong Polytechnic University, China, in 2012. He is currently working toward a master degree in the School of Optoelectronic Engineering of Chongqing University. His research interests include computer and network security, trusted embedded systems.