Software Abnormal Behavior Detection Based on Hidden Markov Model

Jingling Zhao[1,2], Guoxiao Huang[1,2(✉)], Tianyu Liu[4],
and Baojiang Cui[2,3]

[1] School of Computer Science,
Beijing University of Posts and Telecommunications, Beijing, China
{zhaojingling,hgx}@bupt.edu.cn
[2] National Engineering Laboratory for Mobile Network Security, Beijing, China
[3] School of CyberSpace Security,
Beijing University of Posts and Telecommunications, Beijing, China
cuibj@bupt.edu.cn
[4] China Futures Association, Beijing, China
liutianyu@cfachina.org

Abstract. In order to ensure the normal operation and security of the software, this paper presents an abnormal behavior state recognition system based on Hidden Markov model, using the local regularity of the trace of the system call to extract the intrinsic sequence pattern, and construct the normal behavior model through the HMM algorithm. This system is capable of establishing the system call model during normal operation and judge whether the software is running normally by the matching between the actual system call and the normal model. Experiments show that, compared with only using system call sequence information of classical hidden Markov method, the method of training time is only 10% of the traditional method, and can obtain lower false positive rate and false negative rate.

1 Introduction

Abnormal behavior detection technology is a kind of active and dynamic security protection technology for the host, is a powerful complement to traditional firewalls, data encryption and other static defense technology. The goal is to detect those non authorized or unauthorized internal and external intrusion or abnormal behavior. The Host-based anomaly behavior detection system through the process of monitoring system to realize the protection of key software, since most of the attacks will eventually be achieved by illegally change the system call execution traces, so through the system call sequences of the privileged process monitoring can timely detect and prevent intrusion behavior, to achieve the protection of computer systems.

Forrest et al. uses the short sequences of system calls to express the characteristics of the process, and then establish an intrusion detection model [1]. Lee et al. Following the work of Forrest, using RIPPER from the system call sequence mining normal and abnormal patterns, in order to rule the form to describe the system's running state, establish a more simple and effective system to normal model [2]. Wespi et al. Used the Teiresias algorithm of mining mode in biology to extract the indefinite length model of

L. Barolli and T. Enokido (eds.), *Innovative Mobile and Internet Services in Ubiquitous Computing*, Advances in Intelligent Systems and Computing 612,
DOI 10.1007/978-3-319-61542-4_94

the process [3]. In 2011, Fujian Ming et al. from Wuhan University proposed the detection based on object software behavior model [4, 5], the model can resolve the system object from the system call parameters, and take the system object state change as the detection basis, so that it can effectively detect the control flow attacks, imitate the attacks [6, 7], based on semantic attacks. Since Warrender et al. first proposed the application of HMM to the intrusion detection based on system calls in the University of New Mexico [8], there are many improved algorithms and models based on: In 2008, Xing Zhou and others designed a double layer HMM model for intrusion detection, and the training methods used in these two methods have been used in the practical application of [9]. In 2012, Li Cong designed a network intrusion detection method based on node growth Markov distance K mean and HMM [10]. In recent years, the research on this aspect is still not less [11, 12].

In this paper, according to the system call sequences generated by software running always show a strong local integrity of the characteristics, to analyse the system call intrinsic sequence mode and as a basis for the state layer. After extracting the intrinsic mode of the sequence, a state identification model is needed, which is able to convert the system call sequence into the state sequence according to the standard state derived. There are many ways to establish state identification model, the most intuitive approach is directly to match the system call sequence, but this method is too rough, it is difficult to tolerate minor differences between sequences, and has a high false alarm rate. Therefore, in this paper, we do not use this method, considering the accuracy and feasibility of the method, we choose the Markov Model Hidden (HMM) to identify the state. HMM is a classical probability statistical model, which is widely used in video audio signal processing, and the model is accurate, so it is very suitable for the derivation of the state sequence.

The main structure of this paper is as follows: the second section introduces the definition of hidden Markov models and its three classical problems. The third section records the three algorithms for state recognition in this paper. The implementation scheme of the method is recorded in the fourth section. The methods of this paper are tested and the experimental results are obtained in the fifth section. And the sixth section summarizes the main points of the paper.

2 Hidden Markov Model

2.1 Definition

Hidden Markov model is a kind of signal statistical theory model, which can identify and predict the random time-series data, based on the theory of Markov processes. Using the distribution function to describe the Markov process: Suppose I is a state space of the random process $\{X(t), t \in T\}$. If for any n numbers of event t, $t_1 < t_2 < t_3 < \ldots < t_n$, n >= 3, t_i >= T, exactly:

$$P\{X(t_n) \leq x_n | X(t_1) = x_1 \ldots X(t_{n-1}) = x_{n-1}\} = P\{X(t_n) \leq x_n | X(t_{n-1}) = x_{n-1}\}, x_n \in R \tag{1}$$

When the conditional distribution function of $X(t_n)$ under the condition $X(t_i) = x_i$ is equal to the conditional distribution function of $X(t_n)$ under the condition $X(t_{n-1}) = x_{n-1}$, we call the process $\{X(t), t \in T\}$ having the Markov property, and this process is called the Markov process.

Hidden Markov is the further development of Markov model. Markov model can be expressed by Fig. 1 image. It always takes a random process as a series of states of the continuous transfer. Time t is represented by a state q_t, it can be any one in collection of N states $S = \{s_1, s_2, \ldots, s_N\}$. The characteristic of Markov model is represented by the transition probability. The probability of occurrence of the latter state is determined only by its former state.

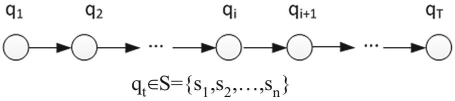

Fig. 1. Markov model chain

Hidden Markov model is considered to be unobservable (hidden), the only observed is that some observations it exhibits. The observation in t moments record as o_t, When the observation is discrete, o_t is one of the total observation set $M = \{m_1, m_2, \ldots, m_k\}$, as shown in Fig. 2. (Note: K is not necessarily equal to N).

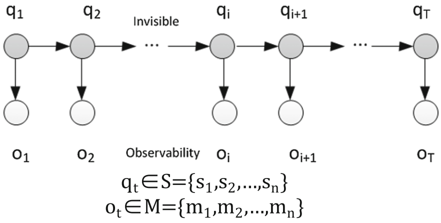

Fig. 2. Hidden Markov model chain

The Hidden Markov model is a doubly stochastic processes, it consists of two aspects of composition. One of it is implicit description of the state transition Markov chain, another is to describe the state of the observed sequence of events corresponding to the statistical relationship between the general random processes. The Hidden Markov model can be a quintuples (N, M, A, B, π) to show, its description of the parameters as shown in Table 1.

Table 1. Hidden Markov model parameters description

Parameter	Description
N	States, The state sets: S = $\{S_1, S_2, \ldots, S_N\}$
M	Possible observations, Observation set: V = $\{V_1, V_2, \ldots, V_m\}$
A	State transition probabilities, A = $\{a_{ij}\}$, $a_{ij} = P(q_{t+1} = j \mid q_t = i), 1 \le i, j \le N$
B	Output probabilities, B = $\{b_j(k)\}$, $b_j(k) = P(o_t = v_k \mid q_t = j)$, $1 \le k \le M$
π	The initial state probability, $\pi = \{\pi_i\}, \pi_i = p(q_1 = i), 1 \le i \le N$

2.2 Modeling Algorithm

Once we have an HMM, there are three problems of interest.

The first problem is to calculate the probability of a sequence of observations, which can be used to determine which model is better, and the problem is generally calculated by forward and backward algorithms.

The second problem is that when the model is known, the most likely sequence of States is to be used to establish the possible state sequences. Solving the decoding problem of the classical algorithm Viterbi algorithm.

The third problem is to give an observation value sequence o, and possible model space (a different model with different parameters of the model), how to find the output sequences of the model (to identify model parameters), which can be used as parameters from existing data to train the model. Baum-Welch algorithm is used to optimize the model parameters in the maximum likelihood estimation criterion State recognition.

Baum-Welch Algorithm. In this paper, the system call sequence is modelled by HMM problem three, that is, the HMM parameter training problem, which is a functional extremum problem, so there is no optimal solution. Baum-Welch algorithm (Baum, 1972) takes recursive thought, which can make $P(o|\lambda)$ local maximum. Baum-Welch algorithm is an iterative hill-climbing, it can only find the local optimal solution. This algorithm is a special case of EM (Maximization Expectation) algorithm. Using this algorithm to train HMM, different initial parameters can produce different training results, generally think the initial value of π and A had little effect on the results of the training, under certain conditions can be selected randomly. But the initial value of B has a great influence on the training results. Under the condition of the given model λ and observation sequence O, the transition probability from i to j is defined as $\xi_t(i,j)$

$$\xi_t(i,j) = P(s_t = i, s_{t+1} = j | X, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \tag{2}$$

$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$, the probability at time t in the state of S_i.

$\sum_{t=1}^{T-1} \gamma_t(i)$, the number of times of the whole process from the state S_i turn out expectations.

$\sum_{t=1}^{T-1} \xi_t(i,j)$, the expected number of times from S_i jump to S_j.

Thus derived the Revaluation formula in Baum-Welch algorithm:

$$\hat{\pi}_i = \gamma_t(i) \tag{3}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \varepsilon_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{4}$$

$$\hat{b}_{ik} = \frac{\sum_{\{t|o(t)=k,1 \le t \le T\}} \gamma_t(i)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{5}$$

Forward Algorithm. The forward algorithm, in the context of a hidden Markov model, is used to calculate a 'belief state': the probability of a state at a certain time, given the history of evidence. The process is also known as filtering. Define the,and the forward variables can be calculated as follows:

(1) *Initialization*

$$a_1(i) = \pi_i b_i, 1 \le i \le N \tag{6}$$

(2) *Induction*

$$\alpha_t(j) = \left[\sum_{i=1}^{n} \alpha_{t-1}(i)\alpha_{ij}\right] b_j(o_t), 2 \le t \le T, 1 \le j \le N \tag{7}$$

(3) *Calculation results*

$$P(O|\lambda) = \sum_{i=1}^{n} \alpha_T(i) \tag{8}$$

Backward Algorithm. The backward variables (Local probability) is the fact that a known HMM λ and time t in a hidden state S_i, the probability of the local observation sequence from the t+1 time to the end time. Same with the forward algorithm, we can move forward (so called backward algorithm) to recursively compute the backward variables. Similarly, the backward variable can be defined $\beta_t(t) = p(o_{t+1} \ldots o_T | q_t = s_t, \lambda)$, and the backward variables can be calculated as follows:

(1) *Initialization:* Make the t = T time all the state of the backward variable is 1:

$$\beta_t(i) = 1, 1 \le i \le N \tag{9}$$

(2) *Induction:* Recursive calculation the backward variable of each time point, t = T–1, T–2, …, 1

$$\beta_t(i)=\left[\sum_{j=1}^{n}\alpha_{ij}b_j(o_{t+1})\beta_{t+1}(j)\right], t = T - 1...1, 1 \leq i \leq N \tag{10}$$

(3) *Calculation results*:

$$P(O|\lambda)=\sum_{i=1}^{n}\pi_i\beta_1(i) \tag{11}$$

3 State Modeling and Detection

Abnormal detection technology through the establishment of the system or the user's normal behavior model, and the deviation degree of the actual behavior of the monitored object and the normal behavior model is the basis of the judgment of the abnormal. System call is the event log that occurs in the user space and the kernel space, its local model not only has a strong regularity, but also has obvious difference in normal behavior and abnormal behavior. So, local pattern of system calls (system calls short sequences) can be used as an effective characterization program behavior characteristics. Therefore, in the anomaly detection, often use the software running system call state to judge whether there is abnormal program.

Carefully observe the system call sequences generated during the operation of the software, and we can find that some sub sequences appear repeatedly in a fixed pattern. This article thus infer such calls may represent a fixed sequence of operations or actions. Based on this judgment, this article to obtain the system call sequence mode with a strong overall pattern from the software normal operation, extract the intrinsic mode of the call sequence, deduced the standard state of the normal operation of the program through a state identification model, and the system call sequence is converted to the state sequence through the HMM model.

However, when a large number of system calls sequences modelled by HMM model, the learning process in which computing resources and training time cost is too expensive, so that training efficiency and detection accuracy of the model is difficult to be taken into account. In order to solve the problem of high cost of computing the hidden Markov model in the massive system call data, this paper presents a new method for HMM-based anomaly detection, a hidden Markov model is established to describe the normal behavior profile of the program in the short sequence of system calls. Since the improved training algorithm, only use a short sequence of mutually different characteristics as training samples, which can avoid the repeated computation of a large number of identical short sequences, thus greatly reducing the training time and computational cost.

3.1 Normal Behavior Modeling of Program Based on HMM

Since the extraction to the system calls of the intrinsic sequence pattern has the relative integrity and the local regularity, therefore, compared with the system call execution trace, the intrinsic sequence patterns can be more stable to characterize the behavior of the program. The hidden Markov model is a statistical tool for the analysis of the transfer of events, can be used to extract the local sequence of the system to more abstract levels, which can be more stable and accurate description of the program's normal behavior patterns.

In this paper, an improved learning algorithm for hidden Markov models is proposed, only calculate the different intrinsic sequence patterns, and the frequency of the local sequence gives different weight. The learning efficiency of the model can be improved effectively by avoiding the repeated computation of a large number of identical sequences.

The process of state identification is mainly used for the training and evaluation of HMM. Firstly, the standard sequential pattern training is used to train the HMM model, then assess the state generated according to established HMM. The HMM-based state recognition step as follows:

First, abstract the existing problem to the problem of generating system call sequence on the state level. Second, extract the vector description of observation status, Mapping of the feature space to the observation of the structure state space, to satisfy the condition of hidden Markov model. Third, input standard samples (system call intrinsic sequence mode) to learn, and constructed hidden Markov model. Fourth, enter the system call sequence to be determined, according to the assessment issue Hidden Markov Models, finding the maximum possible state.

The establishment of the model to complete the first to third step, state assessment section to complete the fourth step.

Model Building. Use the five tuple $\lambda = \{X, O, A, B, \pi\}$ to describe a hidden Markov model, call sequence herein observation sequence corresponding to the system, the state corresponding to the hidden state model. That is, the problem is abstracted as a sequence of system calls generated by a sequence of states. The specific correspondence is shown in Table 2.

Table 2. Relationship of parameters of Hmm

Parameter	Meaning	Relationship
X	The number of Hidden state	The number of different status
O	The number of observations	Different system call number +1
A	State transition probability matrix	The probability of transfer to another state
B	observation probability matrix	The distribution function of the corresponding state of the corresponding system call
π	The initial state probability distribution in the space	The state in which the probability of the initial time

The number of Hidden state are the number of states on a standard derived, the number of observations are system call sequence number plus one, the more calls that are not standard calling sequence sets. It indicates that the call is not within the expected range, shall appear abnormal leeway. In consideration of the abnormal factor, so every time observations must belong to the observation space. A is state transition probability matrix, as shown in Eq. (12), shows the transition probabilities state between state. B is the conditional probability of the system call sequence between states, as shown in Eq. (13), B_{ij} means that when a state is, the probability of observing a sequence of system calls.π is the initial state distribution, as shown in Eq. (14), the initial state is the M state in any of a probability.

$$A = \begin{bmatrix} a_{00} & \cdots & a_{0(N-1)} \\ \vdots & \ddots & \vdots \\ a_{(N-1)0} & \cdots & a_{(N-1)(N-1)} \end{bmatrix} \tag{12}$$

$$B = \begin{bmatrix} b_{00} & \cdots & b_{0(M-1)} \\ \vdots & \ddots & \vdots \\ b_{(M-1)0} & \cdots & b_{(M-1)(M-1)} \end{bmatrix} \tag{13}$$

$$\pi = [\pi_0, \pi_1, \ldots, \pi_{M-1},] \tag{14}$$

HMM is built with the classic Baum-Welch algorithm, this paper establishes a model for each normal state in the state space, a total of N models were established. Input the system call sequence of the state and the initial state distribution matrix, it can be calculated according to the state corresponding HMM Baum-Welch algorithm.

State Evaluation

HMM evaluation problem is that for a given model, find the probability of an observation sequence $p(o|\pi)$, the greater the probability that the observed value is closer to the state represented by the model. In this paper, we determine the state of the system call sequence. The problem with the classic forward backward algorithm (algorithm Forward-backward), the specific practice process as shown in Fig. 3.

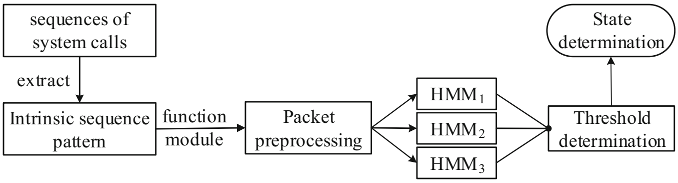

Fig. 3. Process of HMM state recognition

First, the system call sequence is divided into n groups (n usually takes 7 or 8). Each set of sequences should be evaluated based on the established model, and find the maximum value. If the maximum value is greater than the threshold value, this sequence is sentenced to the maximum value corresponding to the HMM state representative. If less than the threshold, then shows that there is not a suspect in the expected state. The division of the sequence will be detected in accordance with the state before and after the decision to make minor adjustments in order to obtain the most accurate determination results. Threshold selection is obtained under the condition of a large number of experiments. After establishing HMM for each standard state, we can use these models to evaluate a large number of system calls to find the corresponding threshold value δ_h.

3.2 Abnormal Behavior Detection

After the completion of the HMM training based on the normal program behavior, in experiment, with each process as the object of study, detecting whether a process is normal. Thus, given a set of data that contains the system call and function call chain, first, they are grouped according to the number of processes, then the system call and function call chain of each process are detected as a group. In the actual anomaly detection, the length of the L sliding window is used to divide the data, and the step length is 1

HMM parameter model $\lambda = (\pi, A, B)$ based on normal data training, for a given function call chain sequence $X = \{o_{t-L+1} \ldots o_t\}$, the function call sequence $Y = \{q_{t-L+1}, \ldots, q_t\}$ corresponding chain system can calculate the probability of its appearance by the following formula:

$$P(X|\lambda) = \pi_{q_{t-L+1}} b_{q_{t-L+1}}(O_{t-L+1}) \prod_{i=t-L+1}^{t-1} a_{q_i q_{i+1}} b_{q_{i+1}}(O_{i+1}) \quad (15)$$

In addition, this paper defines the abnormal degree δ of the software's abnormal behavior as follows:

$$\delta = \frac{N_{ns}}{N_{ts}} \quad (16)$$

Among it: N_{ns} is mismatched short sequence number, defined as the output probability that less than the initial set threshold number, N_{ts} is the total number of short sequences in the test process. The abnormal degree obtained in the experiments experiment. If the exception is greater than the threshold value, it is considered that the process of generating this test sequence may be abnormal, otherwise it is considered normal.

4 Experimental Results

The experimental data are obtained by tracking the sendmail and lpr privilege processes on Ubuntu 14.04, normal data is obtained by simulating the user's normal behavior, Table 3. shows the experimental data. The normal execution sequence as a training

Table 3. Constitute of the data source

Process name	Normal track number	Normal number of system calls	Invasion stitch number	Intrusion system call number
sendmail	2489	1833278	143	7465
lpr	3532	3748577	1001	154162

data set, select the part of the normal execution sequence and execution sequence as intrusion test data.

Existing research results show that the length of the system 6–7 calls short sequence can better characterize the behavior characteristics of the program [13]. Choose a sliding window size of k = 6, based on the training data set to generate a sequence. The training data sets obtained from the sendmail process and lpr process are processed according to the pretreatment method proposed in this paper. The number of intrinsic sequence pattern sets obtained is shown in Table 4, we can see in the table that data preprocessing makes the amount of data greatly reduced.

Table 4. Sequential pattern extraction

Process name	Normal track number	Normal number of system calls	Intrinsic sequence pattern
sendmail	2489	1833278	497
lpr	3532	3748577	722

In the establishment of hidden Markov models, the first to determine the state of the model and the number of observations. Select the size of the state space N = 50, the size of the observation space for the number of system calls M = 256. In order to make the results more general, the initial assignment does not use a priori knowledge, the initial estimate of the state transition matrix A and the output matrix B are assigned in a random way. The structure of the model is connected with the structure of the whole topology. The improved packet preprocessing HMM learning algorithm is proposed in this paper, which is compared with the classic system call training model.

Two methods used in the different parameters and model training time as shown in Table 5, the training time in this paper includes data preprocessing and model learning. It can be seen that the method proposed in this paper is about 5 times less than the traditional method, and the convergence accuracy is 2 orders of magnitude higher than

Table 5. The modeling time and signature size

Method	Process Name	Number of training	Convergence precision	Training time/s
Tradition	sendmail	2489	e^{-7}	6743.3
	lpr	3532	e^{-7}	7987.1
HMM	sendmail	497	e^{-9}	713.4
	lpr	772	e^{-9}	896.5

that of the traditional method. The training time is only about 10% of the traditional method. The efficiency of state identification based on HMM model proposed in this paper is obviously higher than that of the traditional method.

When HMM model training is completed, 50 normal sequences of sendmail and 30 abnormal sequences, 100 normal sequences of lpr and 50 abnormal sequences were used to detect the abnormal behavior. The experimental data are shown in the following Table 6.

Table 6. Abnormal sequence detection results

Method	Process name	False alarm rate/%	False negative rate/%
Tradition	sendmail	4.11	0
	lpr	6.53	0
HMM	sendmail	1.73	0
	lpr	2.95	0

In this experiment, we use the HMM modeling method to detect the abnormal behavior of the system call sequences which is better than the traditional data mining abnormal behavior detection rate of false alarm rate. Obviously, the method can be kept in a low false alarm rate, effective and accurate detection of attacks against the system.

5 Conclusion

In this paper, a method for detecting abnormal behavior of program behavior based on Hidden Markov model is proposed. The normal behavior model is established by using the extracted feature sequence and frequency information from system call execution trace, and to improve the HMM parameter estimation method. At the time of detection, the abnormal behavior of the process is monitored in real time according to the matching degree of the short sequence and the normal behavior model. Using this method can significantly reduce the computational overhead and training time of the model.

Thus, it solves the problem that the training cost of hidden Markov model is too high and the training efficiency and detection performance of the model are difficult to balance. And compared with the classical HMM method, the training time consumption and false alarm rate of the proposed method is much smaller. So it has better real-time performance and practicability, and it can be used as a real time and effective method to detect abnormal behavior of software.

Acknowledgment. This work was supported by National Natural Science Foundation of China (No. U1536122) and the Fundamental Research Funds for the Central Universities (2014ZD03-03).

References

1. Forrest, S., Hofmeyr, S.A., Somayaji, A.: A sense of self for unix processes. In: Proceedings of the IEEE Symposium on Security and Privacy, pp. 120–128 (1996)
2. Lee, W., Stolfo, S.J.: Data mining approaches for intrusion detection. In: Usenix security (1998)
3. Wespi, A., Dacier, M., Debar, H.: Intrusion detection using variable-length audit trail patterns. In: International Workshop on Recent Advances in Intrusion Detection, pp. 110–129. Springer, Heidelberg, October 2000
4. Tao, F., Yin, Z.Y., Fu, J.M.: Software behavior model based on system calls. Comput. Sci. **37**(4), 151–157 (2010)
5. Fu, J.M., Tao, F., Wang, D.: Software behavior model based on system objects. Ruanjian Xuebao/J. Soft. **22**(11), 2716–2728 (2011)
6. Wagner, D., Soto, P.: Mimicry attacks on host-based intrusion detection systems. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, pp. 255–264 (2002)
7. Yap, R.H.: Improving host-based ids with argument abstraction to prevent mimicry attacks. In: International Workshop on Recent Advances in Intrusion Detection. Springer, Heidelberg, pp. 146–164 (2005)
8. Warrender, C., Forrest, S., Pearlmutter, B.: Detecting intrusions using system calls: alternative data models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 133–145 (1999)
9. Zhou, X., Peng, Q., Wang, J.: Intrusion detection method based on two-layer HMM. Appl. Res. Comput. **25**(3) (2008)
10. Chebrolu, S., Abraham, A., Thomas, J.P.: Feature deduction and ensemble design of intrusion detection systems. Comput. Secur. **24**(4), 295–307 (2005)
11. Shuxia, W.: Network intrusion detection method research under big data environment. Bull. Sci. Technol. **8**, 76 (2015)
12. Huang, J.Y., Liao, I.E., Chung, Y.F., Chen, K.T.: Shielding wireless sensor network using Markovian intrusion detection system with attack pattern mining. Inf. Sci. **231**, 32–44 (2013)
13. Lee, W., Stolfo, S. J., Mok, K.W.: A data mining framework for building intrusion detection models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 120–132 (1999)