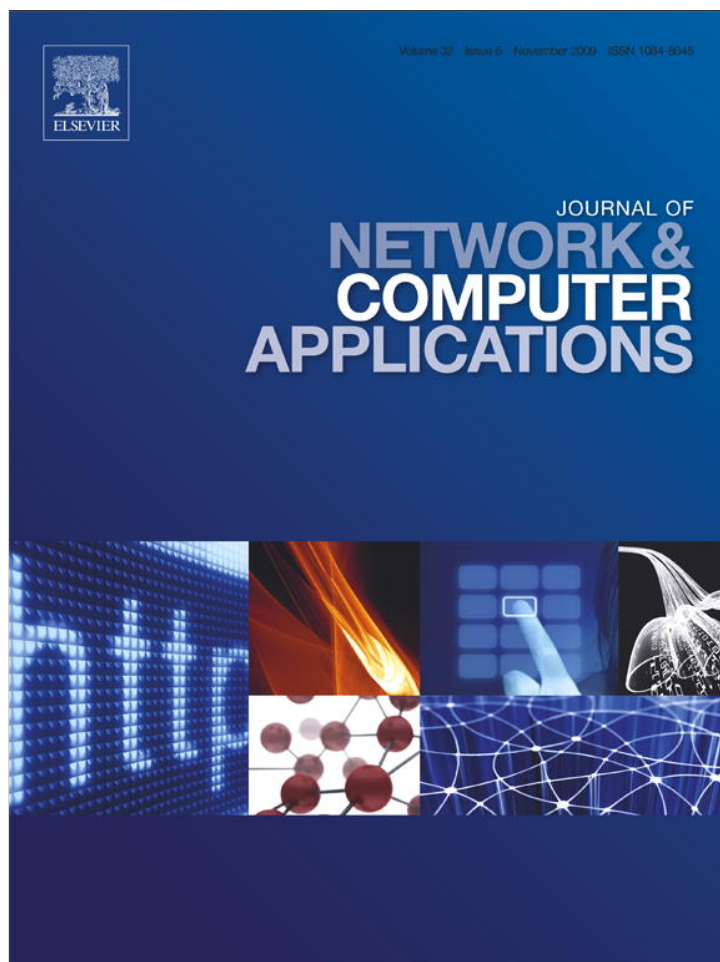


Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.



This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

<http://www.elsevier.com/copyright>



Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

A program-based anomaly intrusion detection scheme using multiple detection engines and fuzzy inference

Xuan Dau Hoang^a, Jiankun Hu^{b,*}, Peter Bertok^b

^a Department of Computer Science, PTIT, Hanoi, Vietnam

^b School of Computer Science and IT, RMIT University, Melbourne 3001, Australia

ARTICLE INFO

Article history:

Received 21 May 2008

Received in revised form

19 March 2009

Accepted 1 May 2009

Keywords:

Anomaly intrusion detection

Fuzzy logic

Program intrusion detection

Hidden Markov model

Multiple detection engines

ABSTRACT

In this paper, a hybrid anomaly intrusion detection scheme using program system calls is proposed. In this scheme, a hidden Markov model (HMM) detection engine and a normal database detection engine have been combined to utilise their respective advantages. A fuzzy-based inference mechanism is used to infer a soft boundary between anomalous and normal behaviour, which is otherwise very difficult to determine when they overlap or are very close. To address the challenging issue of high cost in HMM training, an incremental HMM training with optimal initialization of HMM parameters is suggested. Experimental results show that the proposed fuzzy-based detection scheme can reduce false positive alarms by 48%, compared to the single normal database detection scheme. Our HMM incremental training with the optimal initialization produced a significant improvement in terms of training time and storage as well. The HMM training time was reduced by four times and the memory requirement was also reduced significantly.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

Computer security has become an issue of growing concern, and costs US firms nearly \$67 billion per year (Evers, 2008). Worms, viruses and Trojan horses are the most costly, followed by computer theft, financial fraud and network intrusion, according to the quoted survey. As complete prevention of computer attacks is not possible, intrusion detection systems (IDS) play a very important role in minimizing the damage caused by different computer attacks. Detecting intrusions without prior knowledge of the attack method is most challenging. Anomaly intrusion detection approaches (Abadeh et al., 2007; Anderson et al., 1994, 1995; Bose et al., 2007; Forrest et al., 1996; Gómez et al., 2003; Hautamaki et al., 2004; Hoang et al., 2003a, 2003b; Hoang and Hu, 2004; Hwang et al., 2007; Lee et al., 1999, 2000; Patcha and Park, 2007; Tokhtabayev and Skormin, 2007; Warrender et al., 1999) to name just a few here, seem to be promising, and have attracted considerable attention. The principle of anomaly intrusion detection is to establish a normal profile of the monitored object first and any significant deviation from this normal profile is regarded as an anomaly which could flag an intrusion (Denning, 1987). Anomaly intrusion detection techniques can be generally classified into three categories: statistical detection methods, data-mining-based methods, and machine learning-

based methods (Patcha and Park, 2007). Statistical anomaly detection methods build two profiles: a normal profile during a training phase and the current profile during the detection phase. They monitor activities, such as CPU usage, number of TCP connections, in terms of statistical distribution. During operation these two profiles are compared, and an anomaly is identified if there is a significant difference between them. Smaha et al. proposed modelling the activities statistically by using random Gaussian distribution. An enhanced version of this approach was implemented by intrusion detection expert system (IDES) (Lunt et al., 1992) and next generation intrusion detection expert system (NIDES) (Anderson et al., 1994, 1995). One difficulty with statistical-based anomaly detection methods is determining what a meaningful activity is.

Data-mining-based methods can automate the process of finding meaningful activities and interesting features. They include classification-based intrusion detection, clustering and outlier detection and associate rule discovery (Anderson et al., 1995; Hautamaki et al., 2004; Hoang et al., 2003b; Lee et al., 1999, 2000; Patcha and Park, 2007). Generally, they are computational intensive and produce very high false alarm rates. System call-based sequence analysis is one of the widely used machine learning techniques for anomaly detection. A representative work is the normal-sequence database detection scheme, where a sliding window is used to partition sequences and an intrusion is detected based on the comparison of observed sequences with previously established sequences in the database (Forrest et al., 1996; Warrender et al., 1999). Bayesian networks have also been

* Corresponding author. Tel.: +61 3 99259793; fax: +61 3 9662 1617.
E-mail address: jiankun@cs.rmit.edu.au (J. Hu).

used in anomaly intrusion detection. Such an approach has the advantage of potentially detecting distributed attacks, in which each individual attack session is not suspicious enough to generate an alert. One major drawback of Bayesian networks is that they require a very accurate behavioural model of the monitored subject, which is unrealistic (Patcha and Park, 2007). The hidden Markov model (HMM) is a very powerful machine learning-based tool for anomaly intrusion detection (Davis and Lovell, 2002; Hoang et al., 2003a; Rabiner, 1989). It has been demonstrated that the HMM model performs the best in terms of detection rate and false alarm rate among normal-sequence database schemes, neural networks schemes and data-mining schemes. However, it has high computational costs (Warrender et al., 1999).

The performance of an individual detection engine is rarely satisfactory. In the field of machine learning, it is believed that integration of multiple classifiers can produce better results than individual classifiers. Webb et al. (2005) showed that a uniform weighting of a set of classifiers outperformed any of the individual classifiers; Oliver and Hand (1996) demonstrated that a decision forest can classify better than any individual decision tree. Recently, several attempts have been made to perform anomaly intrusion detection by using multiple classifiers (Analoui et al., 2007; Bose et al., 2007; Cho, 2002; Giacinto and Roli, 2002; Feng et al., 2007; Hoang et al., 2003a; Tsang et al., 2005; Vokorokos et al., 2008; Ye and Xu, 2000). Most of that work is either based on the integration of multiple classifiers operating on different feature sets of networks, such as packet header sets and TCP protocol data sets, or integration of signature-based IDS and anomaly IDS. Hoang et al. (2003a) proposed a multiple-layer approach, where the HMM model and the normal database refer to the same set of system calls. We believe that different detection engines working with the same set of system calls can reveal different aspects of the monitored program. This will provide a more comprehensive understanding of the monitored behaviour, and can subsequently help reduce the false alarm rate. In the context of HMM model scheme and normal-sequence database scheme chosen in this paper, the normal-sequence database scheme is very reliable in making decisions based on frequently observed sequences, but is fairly weak when it comes to infrequently observed system sequences. On the other hand, the HMM model performs well when judging such sequences, due to the generation feature of the HMM model. Based on our previous work (Hoang et al., 2003a), this paper explores effective ways of integrating the HMM model and the normal-sequence database scheme for program-based anomaly intrusion detection. Our major contributions are: (i) a fuzzy framework proposed to integrate HMM anomaly intrusion detection engine and the normal-sequence database anomaly intrusion engine for program-based anomaly intrusion detection. Note: while fuzzy-based algorithms have been traditionally used as detection engines to reduce false alarm rate (Abadeh et al., 2007; Dickerson et al., 2001; Dong et al., 2005; Florez et al., 2002; Gómez and Dasgupta, 2002; Luo et al., 2001), our work is to use fuzzy-based algorithms in integrating the outputs of different detection engines. (ii) To address the challenging issue of high cost in HMM training, an incremental HMM training with optimal initialization of HMM parameters is suggested. (iii) Using the public intrusion system calls database available from the project of Computer Immune Systems at the University of New Mexico (University, 2005), the proposed schemes have been experimentally verified. The experimental results show that the proposed fuzzy-based detection scheme reduced false positive alarms by 48%, compared to the single normal database detection scheme. Our HMM incremental training with the optimal initialization also produced a significant improvement in terms of training time and storage.

The HMM training time was reduced by four times, and the memory requirements also decreased significantly.

The rest of this paper is organized as follows: Section 2 introduces the proposed HMM incremental training scheme with the optimal initialization of HMM parameters. Section 3 describes the proposed fuzzy-based scheme for program anomaly intrusion detection using system calls. Section 4 presents the experimental results and a discussion. Our conclusions and future work are in Section 5.

2. HMM incremental training with initial optimization

2.1. Preliminaries of hidden Markov model

A hidden Markov model is a double embedded stochastic process with two hierarchy levels. The upper level is a Markov process, in which the states are not observable. Observations are made at the lower level and are probabilistic functions of the upper level Markov states. Different Markov states will have different observation functions.

HMMs are very powerful modelling tools although they are computationally expensive (Davis and Lovell, 2002; Gotoh et al., 1998; Hoang et al., 2003a; Hoang and Hu, 2004; Patcha and Park, 2007; Rabiner, 1989; Varghese and Jacob, 2007). HMMs have been widely used in DNA sequence modelling, speech recognition and pattern recognition. For convenience, we use the same HMM notations as in (Hoang et al., 2003a). A HMM has the following elements:

- N : number of states in the model
- M : number of distinct observation symbols per states
- T : length of the observation sequence, i.e. the number of symbols observed
- i_t : state in which we are in at time t
- $V = \{v_1, v_2, \dots, v_M\}$: the discrete set of possible observation symbols
- $\pi = \{\pi_i\}$, $\pi_i = P(i_1 = i)$: the probability of being in state i at $t = 1$
- $A = \{a_{ij}\}$, $a_{ij} = P(i_{t+1} = j, i_t = i)$: the probability of being in state j at time $t+1$ given that we were in state i at time t .
- $B = \{b_j(k)\}$, $b_j(k) = P(v_k \text{ at } t | i_t = j)$: the probability of being observing symbol v_k given that we are state j .
- $O = \{O_1, O_2, \dots, O_t, \dots, O_T\}$: observation sequence; O_t denotes observation symbol observed at time t .

And $\lambda = \{A, B, \pi\}$ will be used as compact notation to denote an HMM.

The Baum–Welch algorithm is the most popular method to estimate HMM parameters from observations. HMM training using the Baum–Welch algorithm is considered as batch training, because it allows only one observation sequence. Given the observation sequence $O = \{O_1, O_2, \dots, O_T\}$, the algorithm estimates the HMM model's parameters $\lambda = \{A, B, \pi\}$, to maximize $P(O|\lambda)$. The Baum–Welch algorithm can be described in brief as follows (Gotoh et al., 1998):

1. Let initial model be λ_0
 2. Compute the new model λ based on λ_0 and observation sequence O
 3. If $\log(P(O|\lambda)) - \log(P(O|\lambda_0)) < \Delta$ go to step 5
 4. Else set $\lambda_0 \leftarrow \lambda$, and go to step 2
 5. Stop
- where Δ is a pre-defined threshold value of the natural logarithm of the probability.

2.2. Optimal initialization of HMM parameters for HMM training

In the HMM training process, the HMM parameters $\lambda(A, B, \pi)$ are adjusted so as to maximize the probability of the observation sequence $O, P(O|\lambda)$. Initialization is the first step in this training process, in which HMM parameters $\lambda(A, B, \pi)$ are assigned to pre-selected initial values. In theory, the values of HMM parameters in the training process should converge to a local maximum of the likelihood function (Rabiner, 1989). Choosing the initial values for HMM parameters so that the local maximum is the global maximum of the likelihood function is a crucial question. Unfortunately, there is no simple or straightforward answer to this question (Pacha and Park, 2007; Rabiner, 1989).

In order to find optimal initial values for HMM parameters, we propose to compute prior probabilities $C = \{c_{ij}\}$, $i = 1, N; j = 1, N$, where N is the number of hidden states, and to use these values to initialize the HMM training parameters. The prior probabilities are defined as the occurrence frequencies of two consecutive observation symbols in the input observation sequence.

To compute prior probabilities, we count the number of occurrences of each pair of consecutive symbols (O_t, O_{t+1}) , $t = 1, \dots, T-1$ in input sequence O . After normalization, prior probabilities C are used to initialize the HMM parameters in the training process. We use C as the initial values for HMM parameters A and B , since C is relatively close to these parameters in probability terms.

The prior probabilities are computed using the following simple algorithm:

Input: Observation sequence $O = \{O_t\}$, $t = 1, T$. Also noted that we selected the number of hidden states equal to the number of distinct observation symbols, or $N = M$.

Output: The matrix of prior probabilities $C = \{c_{ij}\}$, $i = 1, N; j = 1, N$.

1. Set $c_{ij} = 0$; $i = 1, N, j = 1, N$
2. For each pair of consecutive observations O_t and O_{t+1} in sequence O , $t = 1, T-1$:
 - (a) Find the corresponding element of C , c_{ij} : $i O_t, j O_{t+1}$.
 - (b) Increase counter c_{ij} by 1: $c_{ij} c_{ij}+1$.
3. Normalize the C matrix based on the transition probability distribution constraints.

2.3. HMM incremental training

Gotoh et al. (1998) proposed efficient HMM training schemes using incremental ML and MAP estimation algorithms for speech recognition. HMM incremental training has the advantage of faster convergence than that of traditional batch training. However, their algorithms require the subsets of training data be independent. In our training data set, the system calls are related and so the subsets are not independent. Davis and Lovell (2002) proposed a simple method to learn HMM from multiple observation sequences (HMMMOSA). In their approach, first the set of sub-sequences are used to learn a set of sub-models independently. Next, when the learning of all sub-models is complete, the sub-models are merged, by using weights, to produce the final HMM. The subsets do not have to be independent in this method.

In our approach, we modify the HMMMOSA scheme (Davis and Lovell, 2002) to make it incremental. Our new HMM training scheme first divides the long training sequence into a number of subsets of sequences. Next, each subset of data is used to train one sub-model and then the sub-model is incrementally merged into

the final model. The training scheme has the following steps:

1. Divide single observation sequence O into K sub-sequences $\{O_{(1)}, O_{(2)}, \dots, O_{(K)}\}$.
2. Initialize the HMM final model $\lambda \leftarrow \emptyset$ (empty model).
3. Take a sub-sequence $O_{(k)}$ to train sub-model $\lambda_{(k)}$ using HMM batch training algorithm.
4. Incrementally merge $\lambda_{(k)}$ into final model λ .
5. Repeat steps 3 and 4 for all sub-sequences.

The HMM parameters in the incremental merging step of the sub-model $\lambda_{(k)}$ and the final HMM λ are calculated as follows:

$$\bar{a}_{ij} = w_k * a_{ij}^{(k)} + w * a_{ij}$$

$$\bar{b}_{ij} = w_k * b_{ij}^{(k)} + w * b_{ij}$$

$$\bar{\pi}_i = w_k * \pi_i^{(k)} + w * \pi_i$$

where $w_k = 1/P(O_{(k)}|\lambda_{(k)})$, $P(O_{(k)}|\lambda_{(k)})$ is the probability to generate sub-sequence O_k from model λ_k , and $w = 1/P(O_{(1)}, O_{(2)}, O_{(k-1)}|\lambda)$, $P(O_{(1)}, O_{(2)}, O_{(k-1)}|\lambda)$ is the probability to generate sub-sequences $\{O_{(1)}, O_{(2)}, O_{(k-1)}\}$ from model λ .

3. The proposed fuzzy-base program anomaly detection scheme

3.1. The proposed fuzzy-based detection scheme

Fig. 1 shows the proposed fuzzy-based detection scheme that is developed in two stages: (a) training stage and (b) testing stage. In the training stage, the detection model is constructed from the training data, which consists of normal traces of system calls of a program. In the testing stage, the constructed detection model is

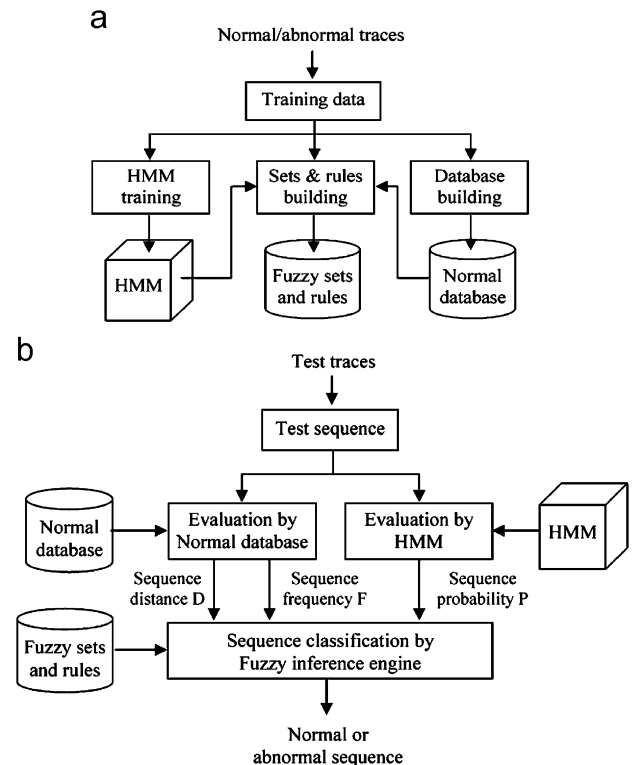


Fig. 1. The proposed fuzzy-based detection scheme: (a) training stage and (b) testing stage.

used to evaluate test traces of system calls in order to find possible intrusions. The two stages of the proposed scheme can be described as follows:

- **Training stage:** A normal database, an HMM model and fuzzy sets are built from the training data.
 - **Normal database:** The database is an ordered list of all unique short sequences of system calls found in the training data. The database is created from normal traces of system calls using the method given in Forrester et al. (1996). Each short sequence in the normal database has k system calls. In addition, the occurrence frequency of each short sequence in the training data is also recorded in the normal database.
 - **HMM model:** The HMM model is trained using normal traces of system calls, based on the HMM incremental training scheme, given in our previous work (Hoang and Hu, 2004).
 - The fuzzy sets are created, as discussed in Section 3.2.
- **Testing stage:** First, short sequences are formed from test traces of system calls using the sliding window method (Forrester et al., 1996). The sequence length is k system calls. Then, each short sequence is evaluated in two steps as follows:
 - Evaluation of the short sequence by the normal database and by the HMM model: In this step, the normal database and the HMM model are used to compute the input parameters for the fuzzy inference engine.
 - Classification of the test sequence by the fuzzy inference engine: In this step, the fuzzy inference engine applies the fuzzy sets and rules to interpret the input parameters in order to produce the output which is the status of the short sequence: normal or abnormal.

3.2. Fuzzy inference for sequence classification

As discussed in Section 3.1, the fuzzy inference engine is used to evaluate each short sequence to find anomalies by combining multiple sequence parameters. Fig. 2 shows the fuzzy inference engine for the classification of short sequences of system calls. The engine takes the sequence's parameters as input, and then applies the fuzzy sets and rules to produce the sequence's status as output. The sequence parameters include a sequence probability P generated by the HMM model, and the sequence distance D and frequency F produced by the normal database.

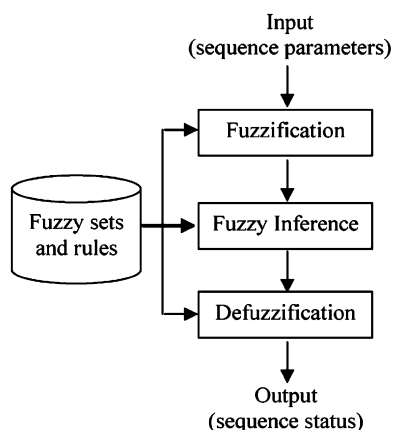


Fig. 2. The fuzzy inference engine for the classification of short sequences of system calls.

3.2.1. Creation of fuzzy sets and rules

As shown in Fig. 2, fuzzy sets and rules are used by the fuzzy inference engine to interpret the input and generate the output.

3.2.1.1. *Creation of fuzzy sets.* We empirically created fuzzy sets to represent the space of each sequence parameter as follows:

- Four fuzzy sets, namely *VeryLow*, *Low*, *High* and *VeryHigh*, are created for the sequence probability P , to represent sequence probabilities.
- Four more fuzzy sets, namely *Zero*, *Small*, *Medium* and *Large*, are created to represent sequence distances, where zero means matching sequences.
- Three fuzzy sets, namely *Low*, *Medium* and *High*, are created to represent sequence frequencies.
- Two fuzzy sets, namely *Normal* and *Abnormal*, are created to represent the space of the output sequence anomaly score parameter. The anomaly score fuzzy sets are used in the defuzzification process to convert the output fuzzy set to an actual anomaly score of the sequence.

Appendix A is a graphical presentation of these fuzzy sets.

3.2.1.2. *Creation of fuzzy rules.* Since the input sequence parameters of the fuzzy rules, which include probability P , distance D and frequency F , are generated by the HMM model and the normal database, our fuzzy rules inherit the assumptions used by the normal database and the HMM-based detection schemes. These assumptions are as follows:

- A sequence, which is produced with a likely probability by the HMM model, is considered to be normal.
- A sequence, which is produced with an unlikely probability by the HMM model, is considered to be abnormal.
- A mismatching sequence is more suspicious than a matching sequence. The larger the distance between a test sequence and normal sequences is, the more likely the test sequence is abnormal.
- A matching sequence with a low occurrence frequency is more suspicious than a sequence with a high occurrence frequency.

Based on the above assumptions, we manually devised a set of 17 fuzzy rules for the sequence classification. An example of such a rule reads "IF probability IS *Low* AND distance IS *Zero* AND frequency IS *Low* THEN the test sequence IS *abnormal*". The full list of fuzzy detection rules used in this paper are given in Appendix B.

3.2.2. Sequence classification using fuzzy reasoning

The fuzzy reasoning process, as shown in Fig. 2, evaluates each sequence of system calls in three phases: fuzzification, fuzzy inference and defuzzification. Fuzzification is the process of transforming crisp input values into linguistic values which usually are fuzzy sets. There are two tasks performed in the fuzzification process: input values are converted to linguistic values that are represented by fuzzy sets, and membership functions are applied to compute the degree of truth for each matched fuzzy set.

Defuzzification is the process of transforming the fuzzy value into a crisp value. In our fuzzy inference engine, the output anomaly score fuzzy set is defuzzified to produce the sequence's anomaly score. There are many defuzzification techniques available, such as the centroid method, max-membership method and weighted average method. We used the max-membership method to compute the crisp output from the output fuzzy set.

In the fuzzy inference phase, all rules in the fuzzy rule-base are applied to input parameters in order to produce an output. For each rule, first, each premise is evaluated, and then all premises connected by an AND are combined by taking the smallest value of their degree of membership as the combination value of rule's truth value. The final output fuzzy set of the fuzzy rule-base is the OR combination of the results of all individual rules that fire. It is noted that the truth value of a rule that fires is non-zero. The output fuzzy set is defuzzified to produce a crisp output value. A sample of fuzzy inference of anomaly score A from probability P and distance D is given in Fig. 11, Appendix A.

4. Experimental results and discussions

4.1. Data set

We used *sendmail* traces of system calls collected in a synthetic environment, as given in University (2005). The format of system call traces and the data collection procedures were discussed in Forrest et al. (1996). The data sets include:

- Normal traces are those collected during the program's normal activity. Normal traces of the *sendmail* program include 2 traces with the total of 1,595,612 system calls.
- Abnormal traces are those that come from a program's abnormal runs generated by known intrusions. In the case of *sendmail* abnormal traces, they consist of 1 trace of *sm5x* intrusion, 1 trace of *sm565a* intrusion, 2 traces of *syslog-local*, and 2 traces of *syslog-remote* intrusion.

4.2. Experimental design

In order to measure the detection rate and the false alarm rate of our fuzzy-based detection model, our experiments were designed as follows:

- *Measurement of the efficiency of the proposed HMM incremental training:* In this test, first we measured the training time of the Baum–Welch HMM batch training using the original training set of 1,000,000 system calls, which had been selected from the *sendmail* data sets. Then, we formed seven incremental training options by dividing the original training set of 1,000,000 system calls into 2, 5, 10, 20, 30, 40 and 50 subsets. For each incremental training option, each subset was used to update the HMM model incrementally, until convergence.
- *Measurement of the false positive rate:* In this test, we used the proposed fuzzy-based detection scheme to classify normal traces of system calls, which had not been used in the

construction of the normal database, the HMM model and the fuzzy sets. Since the normal traces did not contain any intrusions, any reported alarms were considered false positives. This experiment was set up as follows:

- Select the first 1,000,000 system calls of *sendmail* normal traces as the full training set.
- Form 4 training sets which account for 30%, 50%, 80% and 100% of the size of the full training set.
- Construct normal databases and HMM models from these training sets. The chosen values for the sequence length were $k = 5, 11$ and 15 system calls.
- For each training set and on each selected sequence length, construct membership functions to fuzzy sets of three sequence parameters, as discussed in Section 3.2.
- Select three test traces, each with 50,000 system calls from the *sendmail* normal traces, which are not used in the training process, to test for false positive alarms of our scheme, the normal-sequence database scheme (Forrest et al., 1996) and the two-layer scheme (Hoang et al., 2003a). Reported abnormal short sequences were counted for each test trace.
- *Measurement of anomaly signals and the detection rate:* In this test, we use the proposed fuzzy-based scheme to classify abnormal traces of system calls to find possible intrusions. Since the abnormal traces have been collected from the program's abnormal runs with known intrusions, reported alarms in this case can be considered true alarms or detected intrusions. This experiment was designed as follows:
 - Construct a normal database and an HMM model for the *sendmail* program from normal traces of 1,000,000 system calls. We chose the sequence length $k = 11$ to construct the normal database from normal traces, and to form short sequences from abnormal traces for testing.
 - Construct membership functions to fuzzy sets of the three sequence parameters, as discussed in Section 3.2.
 - Use the proposed fuzzy-based detection scheme to evaluate abnormal traces to find abnormal sequences.
 - Use temporally local regions to group individual abnormal sequences to measure the anomaly signals. The selected region length is $r = 20$.

4.3. Experimental results

4.3.1. Reducing HMM training cost

Table 1 shows the training time of the Baum–Welch HMM batch training and the proposed HMM incremental training scheme. The experimental results of the HMM batch training scheme, based on the Baum–Welch algorithm, are given in the

Table 1
Training time of HMM batch training and incremental training on the number of subsets.

Index $i = 0, \dots, 7$	Number of subsets A	Length of each subset B	Total training time (min) C	Average training time per subset (min) $D_i = C_i/A_i$	Normalized time difference between incremental and batch training $E_i = (C_0 - C_i)/C_0$
0	1 ^a	1,000,000	123.92	123.92	0
1	2	500,000	127.72	63.86	-3.07%
2	5	200,000	98.03	19.75	20.89%
3	10	100,000	90.75	9.08	26.77%
4	20	50,000	65.93	4.80	46.79%
5	30 ^b	33,333	60.22	2.01	51.41%
6	40	25,000	48.47	1.21	60.89%
7	50	20,000	55.27	1.11	55.40%

The total length of each training set is 1,000,000 system calls.

^a Batch training: number of subset is 1.

^b There is 1 subset of 33,343 system calls in this incremental option.

first row of the table. Other rows show the experimental results of the proposed HMM incremental training scheme. There are seven incremental training options with the number of training subsets, ranging from 2 to 50. The difference between the training times of HMM batch training and each incremental training option was computed, normalized and given in the last column as a percentage.

In Table 1, it is shown that the proposed incremental training scheme can reduce training time substantially on all incremental training options except the option with 2 subsets. While the training time of the HMM batch training on a single set is 123.92 min, the training time of HMM incremental training on 40 subsets is 48.47 min. This is a 60.89% reduction. The corresponding reductions in the training time of other HMM incremental training options on 5, 10, 20, 30, and 50 subsets are also very good, 20.89%, 26.77%, 46.79%, 51.41%, 55.40%, respectively.

Fig. 3 shows the dependence of the training time on the number of subsets. When the number of subsets increases from 1 (batch mode) to 2, the training time increases slightly, from 123.92 to 127.72 min. Then, the training time decreases significantly, from 127.72 to 48.47 min. This is a reduction of about 2.6 times, when the number of subsets increases from 2 to 40. When the number of subsets increases to 50, the training time increases slightly again to 55.27 min. This means that the proposed HMM incremental training gives best performance on 40 subsets, or 25,000 system calls per subset.

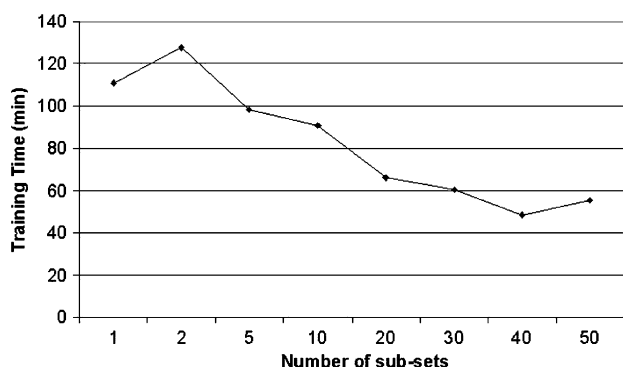


Fig. 3. The dependence of training time on the number of subsets in HMM incremental training. 1 subset indicates HMM batch training. The total length of each training set is 1,000,000 system calls.

Table 2 Overall false positive rate of the normal database scheme, the two-layer detection scheme and the fuzzy-based detection scheme with the short sequence length, $k = 5, 11$ and 15.

Training data sets (% of full data set)	Normal database scheme 0 (%)	Two-layer scheme 0 (%)	Fuzzy-based scheme (%)
<i>Sequence length, $k = 5$; 3 test traces with the total of 149,988 sequences</i>			
30	0.131	0.112	0.067
50	0.099	0.079	0.057
80	0.094	0.069	0.049
100	0.094	0.069	0.049
<i>Sequence length, $k = 11$; 3 test traces with the total of 149,970 sequences</i>			
30	0.194	0.170	0.099
50	0.155	0.115	0.081
80	0.150	0.107	0.077
100	0.147	0.107	0.077
<i>Sequence length, $k = 15$; 3 test traces with the total of 149,958 sequences</i>			
30	0.225	0.164	0.107
50	0.176	0.121	0.091
80	0.174	0.116	0.085
100	0.171	0.116	0.085

When the number of training subsets is too small, the proposed HMM incremental training scheme performs poorer than the HMM batch training scheme. This is because the time is not sufficient to compensate the extra training time caused by the additional overhead of the HMM incremental weighted merging. However, when the number of training subsets is larger, the time saving increases, and this additional overhead becomes insignificant.

4.3.2. False positive rate

Table 2 shows the overall false positive rate for three test traces with a total of 150,000 system calls (each trace consisting of 50,000 system calls), as reported by the normal-sequence database scheme (Forrest et al., 1996), by the two-layer detection scheme (Hoang et al., 2003a) and by the fuzzy-based detection scheme, on different training sets with the sequence length $k = 5, 11$ and 15. The total number of short sequences in the test traces is dependent on the sequence length and is also given in Table 2.

It can be seen from Table 2 that the false positive rate of the fuzzy-based detection scheme is much lower than that of the normal-sequence database scheme (Forrest et al., 1996). For example, the fuzzy-based detection scheme produced 48.23%, 48.89% and 50.96% fewer false positive alarms than the normal database scheme, for the training set of 80% of full set, with sequence lengths of $k = 5, 11$ and 15, respectively.

It is also noted that there is a significant reduction in the false positive rate of the fuzzy-based detection scheme, compared to that of the two-layer detection scheme (Hoang et al., 2003a). For example, the fuzzy-based detection scheme produced 29.81%, 28.13% and 26.44% fewer false positive alarms than the two-layer detection scheme for the training set of 80% of the full set, with sequence lengths of $k = 5, 11$ and 15, respectively (refers to Table 2).

Fig. 4 shows the dependence of the false positive rate on the size of the training sets with the sequence length $k = 11$. When the size of the training set increases, the false positive rate of the normal database scheme (Evers, 2008) and the two-layer scheme (Hoang et al., 2003a) decreases considerably, especially from the training set of 30% of the full set to the set of 50% of the full set. Since the fuzzy-based scheme has already achieved a low false positive rate at the set of 30% of the full set, there is only a small reduction in the false positive rate when the size of the training set increases.

4.3.3. Anomaly signals and the detection rate

Table 3 shows a summary of the detection results of the two-layer scheme and the fuzzy-based scheme for some abnormal traces which were generated by some known intrusions. The detection performance results of the normal database scheme are taken from Table 3 of (Forrest et al., 1996). Similar to the anomaly signal measurement method described in (Hoang et al., 2003a), we measure anomaly signals based on temporally local regions. The anomaly score A of a region is computed as the ratio of the number of detected abnormal short sequences in the region to the length of the region r . The average of anomaly scores are computed over abnormal regions that have the anomaly score A greater than the region score threshold \hat{A} ($A \geq \hat{A}$), where $\hat{A} = 40.0\%$.

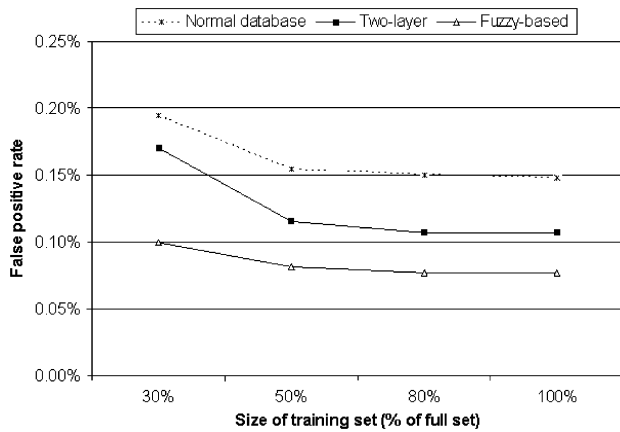


Fig. 4. The relationship between the size of training sets and the false positive rate with $k = 11$.

It can be seen from Table 3 that the fuzzy-based scheme produced significantly better detection results than the two-layer scheme (Hoang et al., 2003a), in terms of the number of detected abnormal regions and the generated anomaly signal level. For the “sm5x” intrusion trace, the rates of detected abnormal regions are 31.58% and 67.11% by the two-layer scheme and fuzzy-based scheme, respectively. Also for this test trace, the fuzzy-based scheme generated an average anomaly score of 72.55%, compared to the average anomaly score of 60.42% produced by the two-layer scheme.

Figs. 5 and 6 show the anomaly signals produced by the two-layer scheme (Hoang et al., 2003a) and the fuzzy-based scheme for syslog-local No. 1 and syslog-remote No. 1 abnormal traces, respectively, with the sequence length $k = 11$. It is noted that anomaly signals are measured based on temporally local regions for both schemes. These figures show that the proposed fuzzy-based scheme generated much stronger anomaly signals than the two-layer scheme (Hoang et al., 2003a.)

4.4. Discussions

Our HMM incremental training with optimal initialization of HMM parameters achieved significant improvement in HMM training time and storage requirement. As compared to HMM batch training, our HMM incremental training scheme reduced the training time by over four times, and decreased the required storage space by K times, where K is the number of training subsets used. This efficiency improvement is significant for practical use.

The proposed fuzzy-based detection scheme generated much fewer false positive alarms than the normal-sequence database scheme (Forrest et al., 1996), as shown in Table 2. For example, the

Table 3

Detection results produced by the normal database scheme, by the two-layer scheme and by the fuzzy-based scheme for some abnormal traces.

Name of test abnormal traces	% detected abnormal sequences by 0	% of detected abnormal regions		Average of scores of abnormal regions	
		Two layer (%)	Fuzzy-based (%)	Two layer (%)	Fuzzy-based (%)
sm565a	0.60	38.46	76.92	68.00	88.00
sm5x	2.70	31.58	67.11	60.42	72.55
syslog-local No. 1	5.10	12.00	60.00	73.33	84.67
syslog-local No. 2	1.70	16.67	60.26	71.54	86.49
syslog-remote No. 1	4.00	28.26	67.39	72.31	86.53
syslog-remote No. 2	5.30	24.68	61.04	74.74	83.40

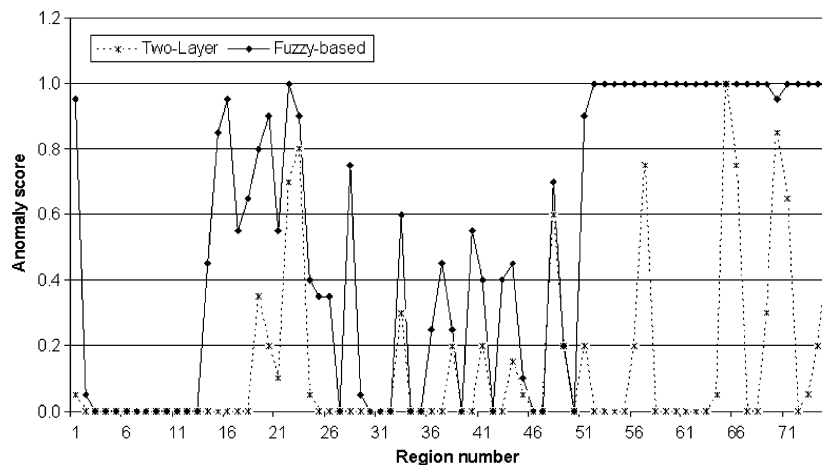


Fig. 5. Anomaly signal generated for syslog-local abnormal trace No. 1 by the two-layer and fuzzy-based schemes.

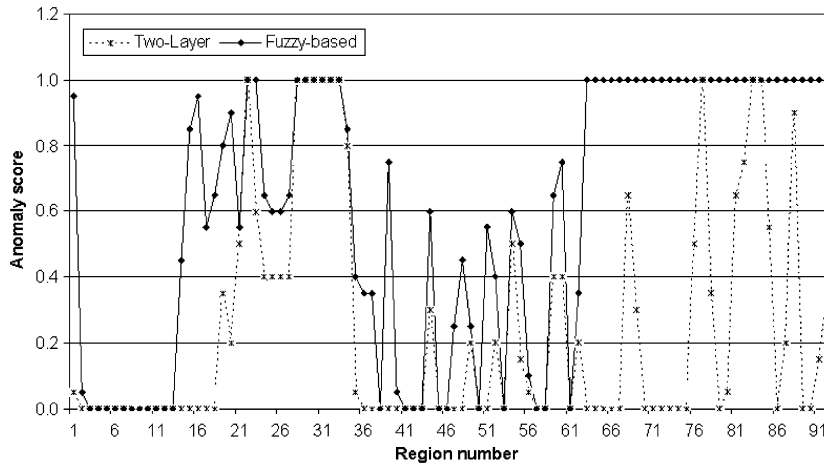


Fig. 6. Anomaly signal generated for *syslog-remote* abnormal trace No. 1 by the two-layer and fuzzy-based schemes.

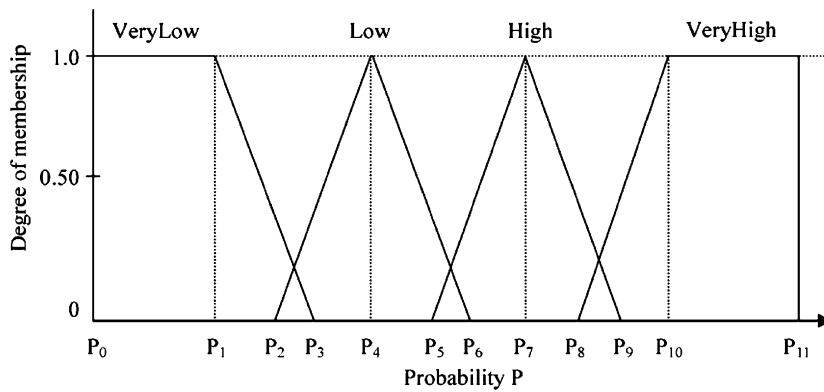


Fig. 7. Fuzzy sets for sequence probabilities *P*.

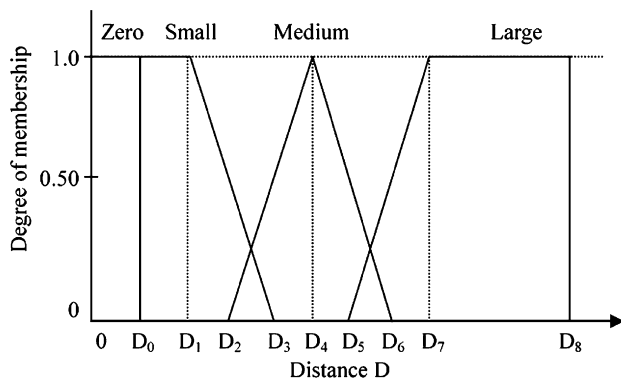


Fig. 8. Fuzzy sets for sequence distance *D*.

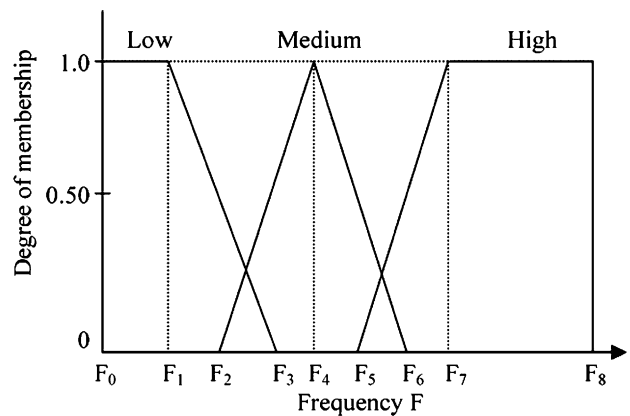


Fig. 9. Fuzzy sets for sequence frequency *F*.

false positive rate of the normal database scheme is 0.174%, as opposed to 0.085% of the proposed scheme, or a reduction of 50.96%, when using the training set of 50% of the full set with $k = 15$.

It is also noted that the proposed detection scheme achieved a much lower false positive rate on small-size training sets than the normal-sequence database scheme (Forrest et al., 1996). On the training set of 30% of the full set, the false positive rate of the proposed detection model is lower than

that of the normal database scheme on the full training set. This means that the proposed detection model requires significantly less training data to achieve a better level of false positive rates than the normal-sequence database scheme (Forrest et al., 1996).

According to experimental results presented in Table 3, our scheme correctly detected all intrusions embedded in all abnormal traces tested. In contrast, the normal database scheme (Forrest et al., 1996) almost completely missed the *sm565a*

intrusion, with only 0.6% of abnormal sequences detected. The same scheme (Forrest et al., 1996) possibly also missed the *syslog-local* intrusion, embedded in *syslog-local* trace No. 2, with just 1.7% of abnormal sequences detected.

The fuzzy inference engine plays an important role in the reduction of false positive alarms and in the increase of the detection rate. The fuzzy inference engine that incorporates multiple sequence information generated by the normal database and by the HMM models, accurately classifies the majority of sequences, and has few false alarms and a high detection rate.

5. Conclusions and future work

In this paper, we presented a fuzzy-based scheme for the integration of HMM anomaly intrusion detection engine and normal-sequence database detection engine for program anomaly intrusion detection using system calls. Instead of using crisp conditions, or fixed thresholds, fuzzy sets are created to represent the space of sequence parameters. A set of fuzzy rules is created, which combine multiple sequence parameters and

determine the sequence status through a fuzzy reasoning process. In order to address the issue of prohibitive computational cost of HMM model training, an incremental HMM training method and an initial optimization scheme have been proposed. Experimental results have shown that the proposed detection scheme reduced false positive alarms by 48% and 28%, compared to the normal-sequence database scheme (Forrest et al., 1996) and the two-layer scheme (Hoang et al., 2003a), respectively. The proposed detection scheme also generated much stronger anomaly signals, compared to the normal-sequence database scheme (Forrest et al., 1996) and the two-layer scheme (Hoang et al., 2003a). The HMM training time was reduced by four times and the memory requirement was also decreased significantly. These improvements have made a good progress towards online and real-time intrusion detection. However, ongoing effort is needed before anomaly IDS technology can be deployed for real-life online intrusion detection. In a most recent work, a data pre-processing scheme has been proposed to reduce the load of training (Hu et al., 2009). Our future work is to investigate how to integrate these approaches together and conduct real-life data test.

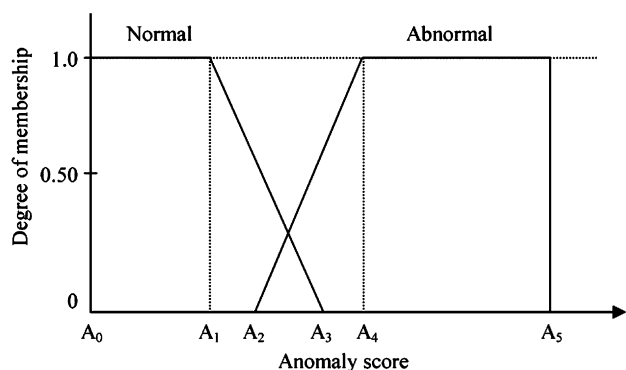


Fig. 10. Fuzzy sets of sequence anomaly score A.

Acknowledgement

The authors acknowledge the financial support from the ARC (Australia Research Council) Linkage Grant with Project ID LP0455324, ARC Discovery Grant with Project ID DP0985838, and National Foundation for Science and Technology Development (NAFOSTED) of Vietnam.

Appendix A. Fuzzy sets

Fuzzy sets for sequence probabilities P , sequence distance D , sequence frequency F , sequence anomaly score A and calculation of sequence anomaly score using fuzzy inference are given in Figs. 7–11.

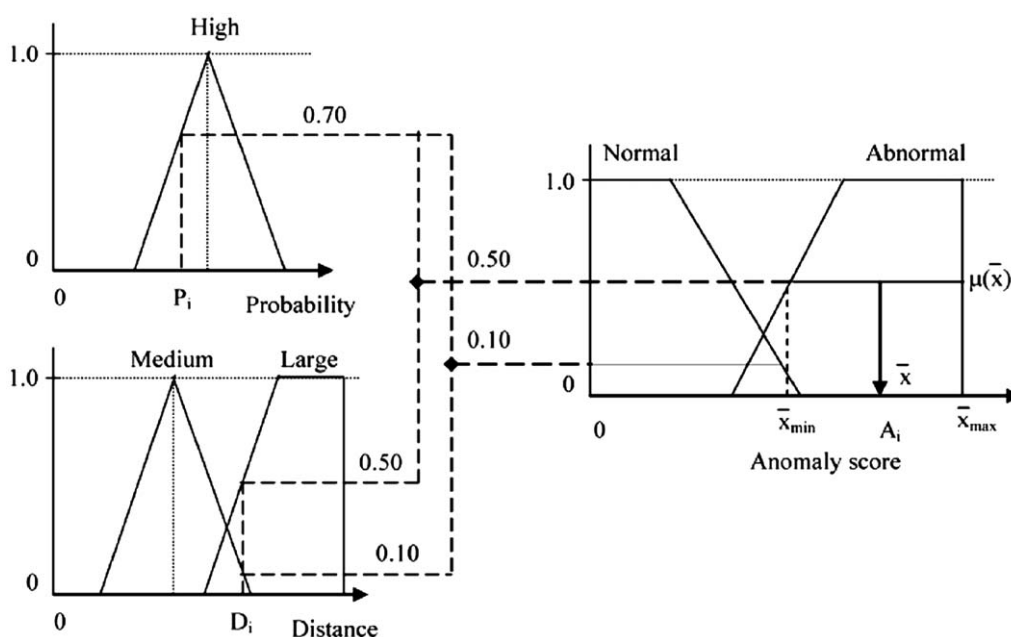


Fig. 11. Calculation of sequence anomaly score using fuzzy inference.

Appendix B. Fuzzy rules

- (1) IF probability IS VeryLow AND distance IS Large THEN the test sequence IS abnormal.
- (2) IF probability IS VeryLow AND distance IS Medium THEN the test sequence IS abnormal.
- (3) IF probability IS VeryLow AND distance IS Small THEN the test sequence IS abnormal.
- (4) IF probability IS VeryLow AND distance IS Zero AND frequency IS Low THEN the test sequence IS abnormal.
- (5) IF probability IS VeryLow AND distance IS Zero AND frequency IS Medium THEN the test sequence IS abnormal.
- (6) IF probability IS VeryLow AND distance IS Zero AND frequency IS High THEN the test sequence IS normal.
- (7) IF probability IS Low AND distance IS Large THEN the test sequence IS abnormal.
- (8) IF probability IS Low AND distance IS Medium THEN the test sequence IS abnormal.
- (9) IF probability IS Low AND distance IS Small THEN the test sequence IS abnormal.
- (10) IF probability IS Low AND distance IS Zero AND frequency IS Low THEN the test sequence IS abnormal.
- (11) IF probability IS Low AND distance IS Zero AND frequency IS Medium THEN the test sequence IS normal.
- (12) IF probability IS Low AND distance IS Zero AND frequency IS High THEN the test sequence IS normal.
- (13) IF probability IS High AND distance IS Large THEN the test sequence IS abnormal.
- (14) IF probability IS High AND distance IS Medium THEN the test sequence IS normal.
- (15) IF probability IS High AND distance IS Small THEN the test sequence IS normal.
- (16) IF probability IS High AND distance IS Zero THEN the test sequence IS normal.
- (17) IF probability IS VeryHigh THEN the test sequence is normal.

References

- Abadeh MS, Habibi J, Lucas C. Intrusion detection using a fuzzy genetics-based learning algorithm. *Journal of Network and Computer Application* 2007; 30:414–28.
- Analoui M, Bidgoli MB, Rezvani, HM. Hierarchical classifier combination and its application in networks intrusion detection. In: 7th IEEE international conference on data mining workshops, 28–31 October 2007. p. 533–8.
- Anderson D, Frivold T, Tamaru A, Valdes A. Next generation intrusion detection expert system (NIDES). Software user's manual, beta-update release. Computer Science Laboratory, SRI International, Menlo Park, CA, USA, Technical Report SRI-CSL-95-0, May 1994.
- Anderson D, Lunt TF, Javitz H, Tamaru A, Valdes A. Detecting unusual program behaviour using the statistical component of the next-generation intrusion detection expert system (NIDES). Computer Science Laboratory, SRI International, Menlo Park, CA, USA, SRI-CSL-95-06, May 1995.
- Bose S, Bharathimurugan S, Kannan A. Multi-layer integrated anomaly intrusion detection system for mobile ad hoc networks. In: IEEE ICSCN 2007, MIT Campus, India, February 22–24, 2007. p. 360–5.
- Cho S. Incorporating soft computing techniques into a probabilistic intrusion detection system. *IEEE Transactions on Systems, Man, and Cybernetics* 2002;32(2).
- Davis RIA, Lovell BC. Improved estimation of hidden Markov model parameters from multiple observation sequences. In: International Conference on Pattern Recognition, Quebec City, Canada, August 2002. p. 168–71.
- Denning DE. An intrusion-detection model. *IEEE Transactions in Software Engineering* 1987; 13:222–32.
- Dickerson J, Juslin J, Koukousoula O. Fuzzy intrusion detection. In: Proceedings of the North American Fuzzy Information Processing society, Vancouver, Canada, July 25, 2001. p. 1506–10.
- Dong SK, Nguyen HN, Park JS. Genetic algorithm to improve SVM based network intrusion detection system. *Advanced information AINA 2005*. In: 19th international conference on networking and applications 2005, vol. 2. p. 155–8.
- Evers J. FBI: computer crime costs US firms \$67bn. <<http://news.zdnet.co.uk/security/0,100000189,39248195,00.htm>>. Retrieved on 15 April, 2008.
- Feng C, Peng J, Qiao H, Rozenblit JW. Alert fusion for a computer host based intrusion detection system. In: The 4th annual IEEE international conference and workshops on the, engineering of computer-based systems, 26–29 March 2007. p. 433–40.
- Florez G, Bridges S, Vaughn R. An improved algorithm for fuzzy data mining for intrusion detection. In: Annual meeting of the North American on fuzzy information processing society, June 27–29, 2002. p. 457–62.
- Forrest S, Hofmeyr S, Somayaji A, Longstaff T. A sense of self for Unix processes. In: Proceedings of the IEEE symposium on computer security and privacy, 1996.
- Giacinto G, Roli F. Intrusion detection in computer networks by multiple classifier systems. In: Proceedings of the 16th international conference on pattern recognition, vol. 2, August 11–15, 2002. p. 390–3.
- Gómez J, Dasgupta D. Evolving fuzzy classifiers for intrusion detection. In: The 3rd annual IEEE workshop on information assurance, New Orleans, Louisiana, USA, June 17–19, 2002.
- Gómez J, González F, Dasgupta D. An immuno-fuzzy approach to anomaly detection. In: IEEE international conference on fuzzy systems, vol. 2, May 25–28, 2003. p. 1219–24.
- Gotoh Y, Hochberg MM, Silverman HF. Efficient training algorithms for HMMs using incremental estimation. In: *IEEE transactions on speech and audio processing*, vol. 6 (6), 1998. p. 539–48.
- Hautamaki V, Karkkainen I, Franti P. Outlier detection using k-nearest neighbour graph. In: Proceedings of the 17th international conference on pattern recognition, Los Alamitos, CA, USA, 2004. p. 430–3.
- Hoang X, Hu J, Bertok P. A multi-layer model for anomaly intrusion detection using program sequences of system calls. In: Proceedings of IEEE international conference on network, Sydney, Australia, September 2003a. p. 531–6.
- Hoang X, Hu J, Bertok P. Intrusion detection based on data mining. In: The 15th international conference on enterprise information systems, Angers, France, vol. 3, 2003b. p. 341–6.
- Hoang X, Hu J. An efficient hidden Markov model training scheme for anomaly intrusion detection of server applications based on system calls. In: Proceedings of the IEEE international conference on network, November 2004, vol. 2. p. 470–4.
- Hu J, Yu X, Qiu D, Chen HH. A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection. *IEEE Network* 2009;23(1).
- Hwang K, Cai M, Chen Y, Qin M. Hybrid intrusion detection with weighted signature generation over anomalous Internet episodes. *IEEE Transactions on Dependable and Secure Computing* 2007;4(1):41–55.
- Lee W, Stolfo SJ, Mok KW. A data mining framework for building intrusion detection models. In: Proceedings of the IEEE symposium on security and privacy, Oakland, CA, 1999. p. 120–32.
- Lee W, Nimbalkar RA, Yee KK, Patil SB, Desai PH, Tran TT, Stolfo SJ. A data mining and CIDF based approach for detecting novel and distributed intrusions. In: Proceedings of the third international workshop on Recent Advances in Intrusion Detection (RAID 2000), Toulouse, France, 2000. p. 49–65.
- Lunt TF, Tamaru A, Gilham F, Jagannatham R, Jalali C, Neumann PG, Javitz HS, Valdes A, Garvey TD. A real-time intrusion detection expert system (IDES). Computer Science Laboratory, SRI International, Menlo Park, CA, USA, Final Technical Report, February 1992.
- Luo J, Bridges S, Vaughn R. Fuzzy frequent episodes for real-time intrusion detection. In: IEEE international conference on fuzzy systems, Melbourne, Australia, December 2–5, 2001.
- Oliver JJ, Hand D. Averaging over decision trees. *Journal of Classification* 1996; 13:281–97.
- Patcha A, Park JM. An overview of anomaly detection techniques: existing solutions and latest technological trends. *Computer Networks* 2007;51: 3448–70.
- Rabiner L. A tutorial on hidden Markov model and selected applications in speech recognition. *Proceedings of the IEEE* 1989;77(2).
- Tokhtabayev AG, Skormin VA. Non-stationary Markov models and anomaly propagation analysis in IDS. In: The 3rd international symposium on information assurance and security, 2007. p. 203–8.
- Tsang CH, Kwong S, Wang H. Anomaly intrusion detection using multi-objective genetic fuzzy system and agent-based evolution computation framework. In: Proceedings of the 5th IEEE International Conference on Data Mining (ICDM'05), 2005.
- University of New Mexico's Computer Immune Systems Project: <<http://www.cs.unm.edu/~immsec/systemcalls.htm>>, Retrieved on 2005.
- Varghese SM, Jacob KP. Process profiling using frequencies of system calls. In: The 2nd international conference on availability, reliability and security, 2007. p. 473–9.
- Vokorokos L, Chovanec M, Latka O, Kleinova A. Security of distributed intrusion detection system based on multisensor fusion. *SAMI 2008*. In: 6th international symposium on applied machine intelligence and informatics, 2008. p. 19–24.
- Warrender C, Forrest S, Pearlmutter B. Detecting intrusions using system calls: alternative data models. In: Proceedings of the IEEE symposium on security and privacy, Oakland, CA, USA, 1999. p. 133–45.
- Webb GI, Boughton J, Wang Z. Not so naive Bayes: aggregating one-dependence estimators. *Machine Learning* 2005;58(1):5–24.
- Ye N, Xu M. Information fusion for intrusion detection. In: Proceedings of the 3rd international conference on information fusion, vol. 2, 2000. p. THB3/17–THB3/20.