Incorporating Soft Computing Techniques Into a Probabilistic Intrusion Detection System

Sung-Bae Cho, *Member, IEEE*

***Abstract*—There are a lot of industrial applications that can be solved competitively by hard computing, while still requiring the tolerance for imprecision and uncertainty that can be exploited by soft computing. This paper presents a novel intrusion detection system (IDS) that models normal behaviors with hidden Markov models (HMM) and attempts to detect intrusions by noting significant deviations from the models. Among several soft computing techniques neural network and fuzzy logic are incorporated into the system to achieve robustness and flexibility. Self-organizing map (SOM) determines the optimal measures of audit data and reduces them into appropriate size for efficient modeling by HMM. Based on several models with different measures, fuzzy logic makes the final decision of whether current behavior is abnormal or not. Experimental results with some real audit data show that the proposed fusion produces a viable intrusion detection system. Fuzzy rules that utilize the models based on the measures of system call, file access, and the combination of them produce more reliable performance.**

***Index Terms*—Anomaly detection, fuzzy logic, hidden Markov model, intrusion detection system, self-organizing map, soft computing.**

I. Introduction

RECENT proliferation of computer communication infrastructure has opened the era of data processing based on computers and raised concerns about computer security. A recent report says that threats to data system have increased 250% during last five years, resulting in $100 billion worth of losses. In particular, attacks to the financial, military, and energy sectors are increasing, requiring detection of illegal intrusions in advance to prevent the damage to the critical communication infrastructure. Large web sites adopt intrusion detection system (IDS) to protect them from various attacks, and there are several products based on diverse intrusion detection techniques.

Intrusion detection is to find attacks exploiting illegal uses or misuses. Basically, an IDS analyzes information about users' behaviors from various sources such as audit trail, system table, and network usage data. The intrusion detection techniques can be divided into two groups according to the type of information they use: misuse detection and anomaly detection. Misuse detection uses knowledge about attacks, whereas anomaly detection uses normal behaviors. Misuse detection attempts to model attacks on a system as specific patterns, and systematically scans the system for occurrences of the patterns [1]–[3]. It has the advantage that known attacks can be detected reliably with low false-positive error, and economically because it just requires scanning known attack patterns. However, it suffers from the inherent shortcoming that it cannot detect novel attacks against systems.

To remedy the problem of detecting novel attacks, anomaly detection attempts to detect intrusions by noting significant deviation from normal behaviors [4]–[6]. One clear drawback is that nonintrusive behavior falling outside the normal range may be labeled as an intrusion, resulting in high false-positive error. It also has to analyze large amount of data to model normal behaviors. In spite of the potential drawbacks, anomaly detection has more potential because it has the ability to detect unknown and novel attacks against computer systems.

In this paper, we exploit soft computing techniques to develop a novel IDS based on anomaly detection using hidden Markov model (HMM). Normal behavior models are required to determine whether current behavior is an anomaly or not. This paper uses HMM as a modeling tool and tests two different modeling strategies: single model without user discrimination and separate models for individual users. Among the principal constituents of soft computing, we have incorporated neural network and fuzzy logic into the IDS to address two issues to implement an innovative anomaly detection system. Unlike traditional hard computing whose prime desiderata are precision, certainty, and rigor, soft computing aims to mimic the ability of the human mind to effectively employ modes of reasoning that are approximate rather than exact [7].

First of all, we should decide the target activities to monitor for the system. Because there is a large amount of raw audit data to analyze, it is required to effectively reduce the data and extract information from them. In this paper, we extract the measures of audit data from Sun Microsystem's Basic Security Module (BSM) [8], and reduce them using self-organizing map (SOM) that is a neural network to be able to cluster input patterns into fixed vectors topologically. The reduced data can be used to model normal behaviors for intrusion detection. To select the optimal measures of a BSM event, we evaluate the effect of the measures by investigating the performance to detect intrusions.

We should also be able to appropriately determine whether current behavior is normal or not. In this paper, we implement three models based on HMM with different measures. The problem is now to decide if the present behavior is abnormal from the three information sources that are inherently imprecise and uncertain. Fuzzy logic is used to make the final decision from the three sources of information to improve the performance and reliability of the system.

The rest of this paper is organized as follows. In Section II, we give a brief overview of the relevant works and background.

Manuscript received April 8, 2001; revised May 6, 2002. This work was supported in part by the Ministry of Information and Communication of Korea.

The author is with the Department of Computer Science, Yonsei University, Seoul 120-749, Korea (e-mail: sbcho@cs.yonsei.ac.kr).

Publisher Item Identifier 10.1109/TSMCC.2002.801356.

The overall design composed of preprocessing and intrusion detection is given in Section III. Experimental results are shown in Section IV.

II. Related Works

Anomaly detection requires reference model, modeling technique, and recognition technique that determines whether current activity deviates from normal behavior or not. The reference model requires collecting audit data from user and system activities. An IDS extracts several measures from audit data, and reduces them in order to profile user's normal behaviors. After normal behavior modeling, IDS can determine whether current behavior is normal or not. Typically selected are such measures that are related to the system bugs exploited by invaders for a long time or to important system resources.

BSM collects data whenever an event occurs, makes a record and appends it to the tail of audit data. Therefore, audit data consist of a series of audit records as time goes on. However, because these data are too huge to process in real time, we have to extract useful information from BSM. There is no exact answer to the question which measures are optimal for detecting intrusion. Although much information is available to detect intrusion, it cannot guarantee high detection rate because important information may be hidden.

Extracting effective measure is very important since it determines the performance of IDS. Measures widely used in IDS include CPU time, I/O access time, file access, system call, network activity, login/logout, etc. To select the optimal measures in IDS, we have extracted several statistics from the BSM audit data based on the well-known attacks and investigate the effect of each measure by testing the performance of it. Table I summarizes the measures used in this paper.

Drawback of misuse detection techniques that they can detect known attacks only has stimulated much research on anomaly detection techniques to cope with novel attacks. The temporal sequence of event under consideration is evaluated how far it deviates from the normal model with various techniques [9]. The most widely used technique is statistics [10]–[12]. User or system behavior is measured by a number of variables over time and modeled with average and standard deviation. It determines whether current behavior is deviated from the normal behavior more than the threshold based on the standard deviation. Expert system models normal behaviors as a set of rules [13]. To detect an intrusion, it matches current behaviors against constructed rules.

There are several other techniques. In [14], a pattern-matching technique evaluates the similarity to event sequence vector, after normal behavior model is built based on user activities. References [15] and [16] model the usage patterns for each program. Reference [17] models and predicts user's behavior at higher level by user intention identification. Reference [4] uses computer immunology. There are also several works that evaluate collections of modeling and matching techniques. Reference [15] compares pattern matching, BP neural network and Elman neural network, and [16] compares frequency, data mining, and HMM. Results in [15] and [16] show that exploiting temporal sequence information of event leads to better performance. Temporal sequence information is considered as a significant factor in [14].

TABLE I
Measures Extracted From the BSM to Detect Intrusions

Category	Measures
System call	ID, return value, return status
Process	ID, IPC ID, IPC permission
	exit value, exit status
File access	access mode, path, file system
	file name, argument length

III. Intrusion Detection System

Our intrusion detection system is composed of preprocessing module, which is in charge of data filtering and data reduction, and anomaly detection module, which is responsible for the normal behavior modeling and anomaly detection, as shown in Fig. 1. Normal behaviors are modeled into profile database through model learning.

A. Preprocessing With SOM

Usually, audit data composed of all events may amount to hundreds of megabytes per day, which cannot be easily dealt with in real time. Many measures can be useful in statistical analysis, but they need to be transformed into lower dimensional data for modeling techniques like HMM. Reducing the size of measures means that $x_1, x_2, \ldots, x_n$ are converted into $x'_1, x'_2, \ldots, x'_k$ where k is much smaller than n. In this paper, the measure size is reduced based on the locality of user action. The ranges of the measures are observed and the measures mainly used in the system are stored to find the value of measure for current action. As a result of mapping of the measures, we can reduce the data.

In order to select a set of features that are invariant with respect to some basic transformation groups of the input measures, we can consider several vector quantization and clustering techniques such as k-means algorithm and SOM. Unlike batch algorithms, SOM has several merits that organizes the map automatically according to the input pattern and preserves the topology of input space. We choose the SOM to exploit the characteristics of topology preservation to trace the sequence of observation symbols later in HMM.

SOM is an unsupervised learning neural network using Euclidean distance to compute the distance between input and reference vectors [18]. Learning algorithm of SOM is as follows. Here, $i(x)$ is the best matching neuron to input vector, λ_i means neighborhood function, and η is learning rate.

1) Initialize the reference vectors $(w_j(n))(n = 0)$.
2) Compute the distance and select the minimum value $i(x) = \arg\min \|x(n) - w_j(n)\|$.
3) Update the reference vectors $w_j(n+1) = w_j(n) + \eta(n)\lambda_{i(x)}(n, j)(x(n) - w_j(n))$.
4) Repeat from 2 to 3 until the stop condition satisfies.

Several measures can be extracted from one audit record of BSM that includes an event and the information is normalized

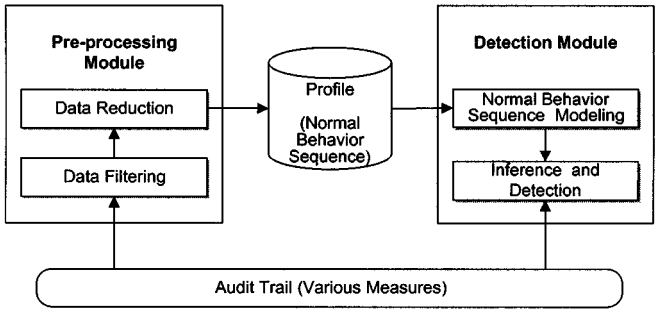

Fig. 1. Overview of intrusion detection system.

for the input of SOM. The normalization depends on the measures, and basically rescales them between 0 and 1. After training the map, the weight vectors of the map try to specify the centers of clusters covering the input space and the point density function of these centers tend to approximate the probability density function of the original input space. From the trained SOM, we can get one representative value (the index of the best matching neuron) with the inputted event composed of many measures: one record is converted into one representative. The reduced information can be used for modeling techniques like HMM.

B. Behavior Modeling With HMM

Several techniques, such as rule-based systems, statistical methods, and neural networks, have been used to model the normal behaviors for intrusion detection system. Neural networks seem to be a good candidate for our problem, but they usually lack the modeling capability of temporal sequences and some researchers attempt to remedy it by utilizing recurrent networks. In terms of modeling a large number of temporal sequences, HMM must be the alternative, because it has been widely used for speech recognition [19].

An HMM is a doubly stochastic process with an underlying stochastic process that is not observable, but can only be observed through another set of stochastic processes that produce the sequence of observed symbols [19]. This model can be thought of a graph with N nodes called state and edges representing transitions between those states. Each state node contains initial state distribution and observation probability at which a given symbol is to be observed. An edge maintains a transition probability with which a state transition from one state to another state is made.

Given an input sequence $O = O_1, O_2, \ldots, O_T$, HMM can model it with its own probability parameters using Markov process though state transition process cannot be seen outside. Once a model is built, the probability with which a given sequence is generated from the model can be evaluated. A model λ is described as $\lambda = (A, B, \pi)$ using its characteristic parameters, as follows:

- T length of the observation sequence;
- N number of states in the model;
- M number of observation symbols;
- Q $\{q_1, q_2, \ldots, q_N\}$, states;
- V $\{v_1, v_2, \ldots, v_M\}$, discrete set of possible symbol observations;
- A $\{a_{ij}|a_{ij} = \Pr(q_j \text{ at } t+1)\}$, state transition probability distribution;
- B $\{b_j(k)|b_j(k) = \Pr(v_k \text{ at } t|q_j \text{ at } t)\}$, observation symbol probability distribution;
- π $\{\pi_i|\pi_i = \Pr(q_i \text{ at } t = 1)\}$, initial state distribution.

The probability with which the sequence is generated from the model can be calculated by summing the probabilities for all the possible state transitions. In practice, more efficient method, known as forward–backward procedure, is used.

1) Anomaly Recognition: Anomaly recognition matches current behavior against the normal behavior models and calculates the probability with which it is generated out of each model. Forward–backward procedure or Viterbi algorithm can be used for this purpose [19]. Each probability is passed to the determination module that decides whether it is normal or not with a threshold.

Forward–backward procedure calculates the probability $\Pr(O|\lambda)$, with which input sequence O is generated with model λ using forward and backward variables. Forward variable α denotes the probability at which a partial sequence $O_1, O_2, \ldots, O_T$ is observed and stays at state q_i

$$\alpha_t(i) = \Pr(O_1, O_2, \ldots, O_t, i_t = q_i|\lambda). \tag{1}$$

According to this definition, $\alpha_t(i)$ is the probability with which all the symbols in input sequence are observed in order and the final state is i. Summing up $\alpha_t(i)$ for all i yields the value $\Pr(O|\lambda)$. Backward variable $\beta_t(i)$ is defined as follows. It can be calculated using the similar process to that for α

$$\beta_t(i) = \Pr(O_{t+1}, O_{t+2}, \ldots, O_T, i_t = q_i|\lambda). \tag{2}$$

2) Normal Behavior Modeling: Determining HMM parameters is to adjust $\lambda = (A, B, \pi)$ to maximize the probability $\Pr(O|\lambda)$. Because no analytic solution is known for it, an iterative method called Baum–Welch reestimation is used [19]. This requires two more variables: $\xi_t(i, j)$ is defined as the probability with which it stays at state q_i at time t and stays at state q_j at time $t+1$:

$$\begin{aligned}\xi_t(i, j) &= \Pr(i_t = q_i, i_{t+1} = q_j|O, \lambda) \\ &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\Pr(O|\lambda)}.\end{aligned} \tag{3}$$

$\gamma_t(i)$ is the probability with which it stays at state q_i at time t:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j). \tag{4}$$

Summing up the two variables over time t, respectively, we can get the probability with which state i transits to state j and the expectation that it stays at state i. Given the previous variables calculated, a new model $\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\pi})$ can be adjusted using the equations shown in (5), at the bottom of the next page.

After $\overline{\lambda}$ is adjusted from sequence O, $\Pr(O|\overline{\lambda})$ is compared against $\Pr(O|\lambda)$. If $\Pr(O|\lambda)$ is greater than $\Pr(O|\overline{\lambda})$, it implies that a critical point in likelihood has been reached, thereby finishing the reestimation. Otherwise, $\Pr(O|\lambda)$ is substituted by $\Pr(O|\overline{\lambda})$ and reestimation continues.

C. Multiple Models Fusion by Fuzzy Logic

The HMM can provide a good probabilistic representation of temporal sequences, but the modeling performance must be dependent on the characteristics of input distributions. The BSM that collects audit data for our IDS provides with many different input symbols as measures according to the usage patterns to the system. The measures are neither statistically independent nor unimodally distributed, and an HMM of a finite size does not often load a particular modeling and generalizes poorly.

The basic idea of the multiple models is to develop n independently trained models with different measures, and to decide if the current input sequence is abnormal by obtaining and fusing the outputs from the different models. There are many ways for the information fusion such as neural networks and parameterized classifiers, but in this problem, fuzzy logic must be better than other methods, because the outputs of the models are uncertain and imprecise, and human experts can have some intuition or knowledge on the characteristics of the models [20]. Fig. 2 illustrates the schematic diagram of the proposed fusion scheme.

A fuzzy inference system provides a computing framework based on the concepts of fuzzy sets, fuzzy if–then rules, and fuzzy reasoning [20]. The basic structure consists of a fuzzy rule-base, reasoning mechanism, and defuzzification mechanism. A fuzzy rule base is a set of fuzzy rules that are expressed as follows.

- Rule 1: If (x_1 is A_1^1) and (x_2 is A_2^1) and … and (x_n is A_n^1), then y is B^1.
- Rule 2: If (x_1 is A_1^2) and (x_2 is A_2^2) and … and (x_n is A_n^2), then y is B^2 … .
- Rule m: If (x_1 is A_1^m) and (x_2 is A_2^m) and … and (x_n is A_n^m), then y is B^m.

Here, x_j ($1 \le j \le n$) are input variables, y is an output variable, and A_j^i and B^i ($1 \le i \le m$) are fuzzy sets which are characterized by membership functions. In this paper, the numbers of input and output variables are three and one, respectively.

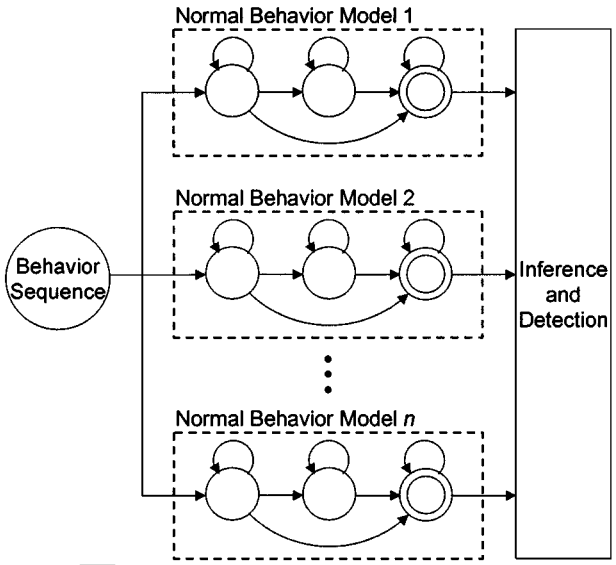

Fig. 2. Scheme of fusing multiple HMM models.

In order to facilitate the design of the detection module, we have to select fuzzy sets for input and output parameters. The optimal selection can be performed with more investigation, but it is beyond the scope of this paper. We just adopt the following fuzzy sets for them.

- Input: HMM evaluation values ($M1$, $M2$, $M3$) from different models [0, $-\infty$).
  Fuzzy set: $I = \{\text{Low, Middle, High}\}$.
- Output: Normality (S) [0, 1].
  Fuzzy set: $O = \{\text{Normal, Abnormal}\}$.

The exact partitioning of input/output spaces depends on membership functions, and triangular shapes specify the membership function in this paper. For fuzzy inference, we use correlation minimum method, which truncates the consequent fuzzy region at the truth of the premise, and the centroid

$$
\begin{aligned}
\overline{\pi}_i &= \text{expected frequency (number of times) in state } S_i \text{ at time } (t=1) \\
&= \gamma_1(i) \\
\overline{a}_{ij} &= \frac{\text{expected number of transistions from state } S_i \text{ to state } S_j}{\text{expected number of transistions from state } S_i} \\
&= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \\
\overline{b}_j(k) &= \frac{\text{expected number of times in state } j \text{ and observing symbol } \nu_k}{\text{expected number of times in state } j} \\
&= \frac{\sum_{\substack{t=1 \\ s.t. O_t=\nu_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}.
\end{aligned} \tag{5}
$$

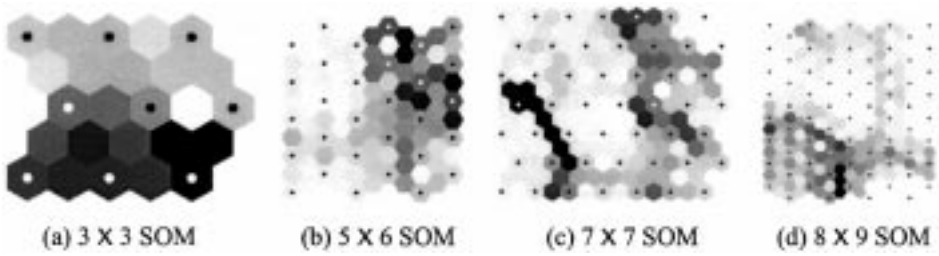

Fig. 3. Variation of distance in reference vectors with map size.

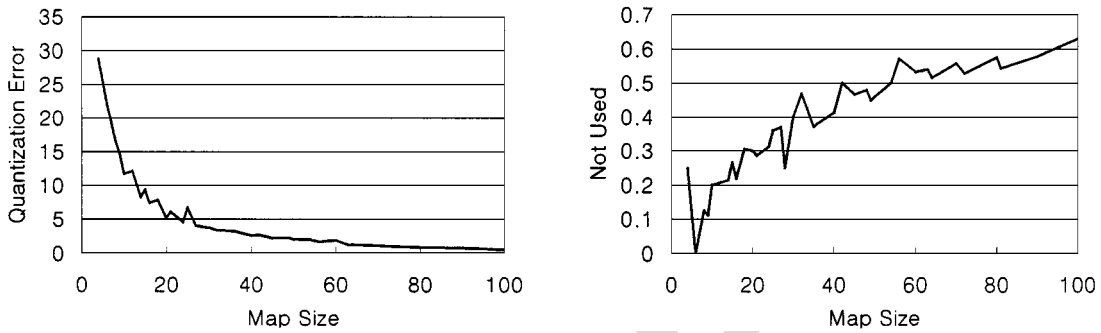

Fig. 4. Variation of quantization error and map usage according to the map size.

defuzzification method is adopted to yield the expected value of the solution fuzzy region [20]. The fuzzy rules must be devised according to the models in consideration.

IV. Experimental Results

We have used data obtained from three graduate students for one week. In order to collect the realistic data, we did not give any restriction on the time period, but they mainly used text editor, compiler, and their own programs. Total 60 000 data have been collected and among them 30 000 are used for training and another 30 000 are for test. Seventeen cases of u2r (user-to-root) intrusion, one of the most typical intrusions, are included in each user's test data set. An unprivileged user acquires super user privilege by exploiting a subtle temporary file creation and race condition bug.

In anomaly detection, it is natural that false-positive error is higher than misuse detection. If false-positive error is high, the IDS may not be practical, because users may be warned from the intrusion detection system, though their action is not illegal. Therefore, we should regard false-positive error as critical error. Actually, we investigate Receiver Operating Characteristics (ROC) curve [15] in the experiment that presents the variation according to the change of false-positive error.

A. Preprocessing

In general, large SOM is better than small one, but it is not certain when the locality of input pattern is high. In the preprocessing step, we have attempted to find the optimal map size of SOM, as well as the optimal measures to improve the performance of the IDS based on HMM by investigating the change of ROC curve for each measure.

1) Effect of Map Size: Applying SOM to real problems requires us to consider the map size. There is no answer to which map size of SOM is the best for intrusion detection. In order to solve this problem, we consider two factors: quantization error and map usage. The map size of low quantization error and high map usage is desirable for SOM.

Fig. 3 shows the variation of reference vectors with the different map sizes of SOM. In the figure, dark area means that one reference vector is distant from the other. The larger the map size becomes the closer the distance among reference vectors is, which also indicates that the locality cannot disappear in spite of large map size of SOM. Also, as can be seen in Fig. 4, the larger the map size the smaller the quantization error, but the probability of usage gets lower. Also, the gradient of error curve is lax when the map size is larger than 25. We can also see that the probability of usage is above 50% when the map size is smaller than 50. Therefore, we can select the useful map size between 25 and 50.

2) Usefulness of Measures: In this paper, we extract three measures from the data related to system call (ID, return value, and return status), process (ID, IPC ID, IPC permission, exit value, and exit status) and file access (access mode, path, argument length, and file system). We have also taken into account the combinations of the three measures, resulting in seven sets of different measures in total.

First of all, we reduce their size based on the locality, convert them into a representative via SOM, and make a sequence so that it can be used for modeling normal behaviors with HMM. Several map sizes such as 5×5, 6×6 and 7×7 are considered for the experiment. The HMM used has ten states and a different number of symbols according to the map size of SOM. The number of states is determined by a thorough investigation with experiments.

Fig. 5 shows the ROC curves with respect to the map size of SOM. In case of map size of 25 (5×5), such measures as system call and combination of process and file access are useful and

file access can be helpful to detect the intrusion well in the map size of 36 (6×6). As can be seen in the 7×7 map, the larger map does not guarantee the performance improvement, because it happens to be worse than the smaller map. In sum, we have obtained the following three sets of measures with different map sizes of SOM.

- The measure based on system call (5×5 SOM).
- The measure based on file access (6×6 SOM).
- Combination of the measures based on system call and file access (6×6 SOM).

B. Modeling and Intrusion Detection

1) Effectiveness of HMM for Intrusion Detection: At first, single HMM is used to model all the users. Optimal HMM parameters are obtained through experiments: the length of input sequence is 30 and the number of states is ten. Fig. 6 shows the change of the sequence evaluation values over time when an intrusion occurs. An attack was started at time 11 and ended at time 32. The sequence evaluation value around time 11 gets lower radically due to the abnormal behavior and this low state lasts by around time 32, which implies the system has a potential to effectively detect the intrusion.

2) Evaluation with Different Modeling Strategies: This experiment is conducted with separate model for each user. Table II shows the intrusion detection rate and false-positive error rate for each model. Here, the threshold is determined with the training data. Separate modeling has produced lower overall false-positive error rates than single modeling. We have got low evaluation values when a user's behavior is evaluated with other users' models as shown in Table III. This shows the proposed technique is effective in modeling user's normal behaviors and can also be used for detecting intrusions from sniffed user accounts that is also very crucial attack for IDS.

3) Improvement with Models Fusion: From the experimental results on the usefulness of measures, we have developed three HMM models based on the measure of system call, the measure of file access, and the combination of them, and attempted to combine them using the fuzzy inference system in Section III. We have devised the following seven rules for intrusion detection based on the in-depth investigation on the characteristics of the three models and attack patterns.

- Rule 1: if ($M1$ is Low) then (S is Abnormal).
- Rule 2: if ($M2$ is Low) then (S is Abnormal).
- Rule 3: if ($M1$ is High) and ($M2$ is Middle) then (S is Normal).
- Rule 4: if ($M1$ is Low) and ($M2$ is Low) and ($M3$ is Low) then (S is Abnormal).
- Rule 5: if ($M1$ is Middle) and ($M2$ is Middle) and ($M3$ is Middle) then (S is Normal).
- Rule 6: if ($M1$ is High) then (S is Normal).
- Rule 7: if ($M1$ is Middle) and ($M2$ is High) then (S is Normal).

We have used the best result from each model because the threshold for each model may differ from each other. The false-positive error rate has enhanced compared to its primitive models as shown in Table IV. This result indicates that the fuzzy logic provides a model for modes of reasoning which are

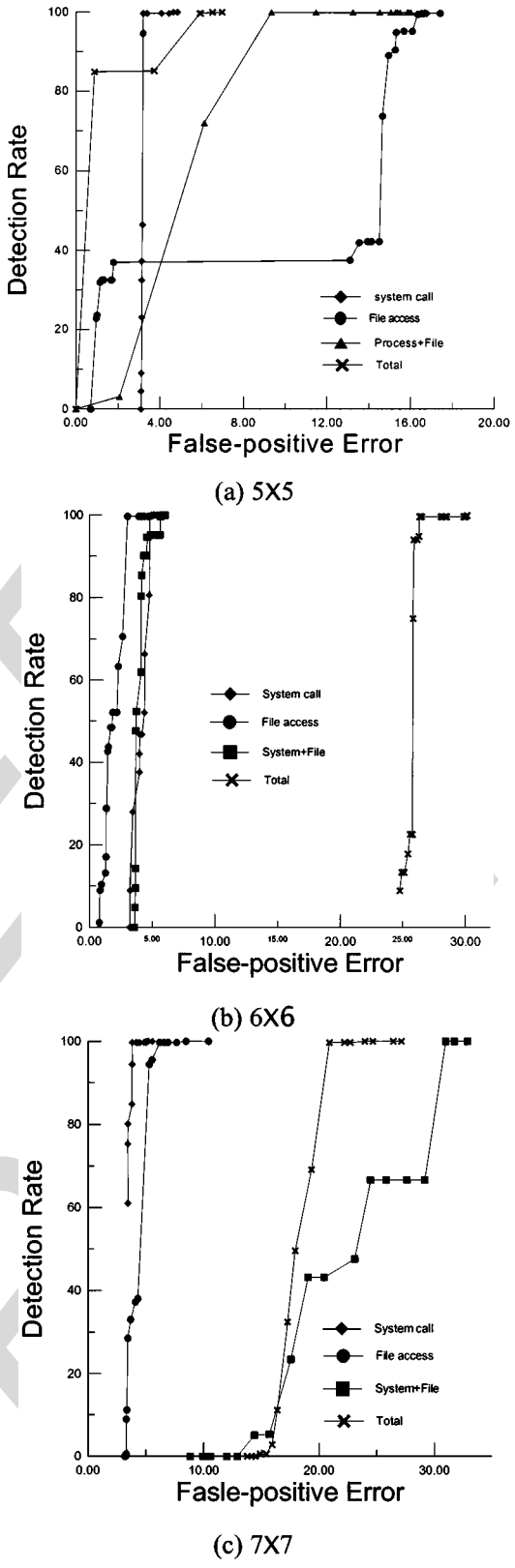

Fig. 5. Measure effect of IDS with the change of map size.

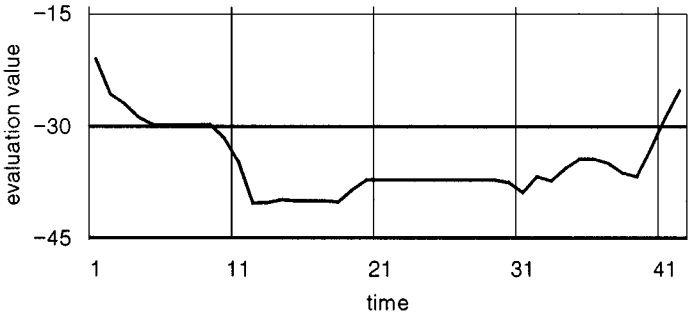

Fig. 6. Change of sequence evaluation values when an intrusion occurs.

TABLE II
THE ANOMALY DETECTION RATE FOR SEPARATE USER MODELING

	single modeling	individual user modeling		
		User 1	User 2	User 3
mean (S.D)	-14.74 (10.42)	-15.54 (5.88)	-4.72 (8.42)	-14.25 (12.34)
threshold	-32.45	-26.21	-31.35	-36.92
detection rate	100%	100%	100%	100%
false-positive error rate	2.95%	4.31%	1.73%	1.96%

TABLE III
THE NUMBER OF SEQUENCES WHOSE EVALUATION VALUES ARE LESS THAN −50 WHEN DIFFERENT USER'S MODEL IS USED

		evaluated user		
		User 1	User 2	User 3
referenced user model	User 1	0	366	55
	User 2	566	315	336
	User 3	475	346	0

approximate rather than exact from imprecise and uncertain information sources.

V. CONCLUDING REMARKS

In this paper, we have proposed a novel intrusion detection system that reduces raw audit data using SOM, models user's normal behaviors using HMM and detects anomalies by combining several models with fuzzy logic. Experimental results indicate that this system has effectively detected the intrusions. It can be concluded that provided with the complete normal behaviors, the fusion of soft computing like neural network and fuzzy logic and hard computing like HMM can be used as an effective intrusion detection technique. For further work, we have to devise more sophisticated techniques of integrating soft computing and hard computing, including an automatic rule acquisition for the fuzzy inference system, as well as evaluate our system with more realistic data sets.

TABLE IV
THE PERFORMANCE OF MULTIPLE MODELS FUSION BY FUZZY LOGIC

	single modeling	fuzzy logic fusion
detection rate	100 %	100 %
false-positive error rate	2.95 %	1.18 %

REFERENCES

[1] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: A rule-based intrusion detection system," *IEEE Trans. Softw. Eng.*, vol. 21, pp. 181–199, Mar. 1995.
[2] W. Lee, S. Stolfo, and P. K. Chan, "Learning patterns from Unix process execution traces for intrusion detection," in *Proc. AAAI97 Workshop on AI Methods in Fraud and Risk Management*, 1997.
[3] G. Vigna and R. A. Kemmerer, "Netstat: A network-based intrusion detection approach," in *Proc. NISSC'98*, Oct. 1998, pp. 338–347.
[4] S. Forrest, S. A. Hofmeyr, and A. Somayaji, "Computer immunology," *Commun. ACM*, vol. 40, no. 10, pp. 88–96, Oct. 1997.
[5] T. F. Lunt, "A survey of intrusion detection techniques," *Comput. Security*, vol. 12, no. 4, pp. 405–418, June 1993.
[6] P. A. Porras and P. G. Neumann, "Emerald: Event monitoring enabling responses to anomalous live disturbances," in *Proc. 20th NISSC*, Oct. 1997, pp. 353–365.
[7] L. A. Zadeh, "Fuzzy logic, neural networks, and soft computing," *Commun. ACM*, vol. 37, no. 3, pp. 77–84, Mar. 1994.
[8] Sunsoft, "Solaris 2.5 Sunshield basic security module guide,", 1995.
[9] H. Deba, M. Dacier, and A. Wespi, "Toward a taxonomy of intrusion-detection systems," *Comput. Networks*, vol. 31, pp. 805–822, 1999.
[10] P. Helman and G. Liepins, "Statistical foundations of audit trail analysis for the detection of computer misuse," *IEEE Trans. Softw. Eng.*, vol. 19, pp. 886–901, Sept. 1993.
[11] P. Helman, G. Liepins, and W. Richards, "Foundations of intrusion detection," in *Proc. 1993 IEEE Symp. Research in Security and Privacy*, Oakland, CA, May 1993, pp. 16–28.
[12] H. Javitz *et al.*, "Next generation intrusion detection expert system (NIDES)—1. statistical algorithms rationale—2. rationale for proposed resolver," SRI International, March, Tech. Rep. A016-Rationales, 1993.
[13] H. Vaccaro and G. Liepins, "Detection of anomalous computer session activity," in *Proc. 1989 IEEE Symp. Research in Security and Privacy*, 1989, pp. 280–289.
[14] T. Lane and C. E. Brodley, "Temporal sequence learning and data reduction for anomaly detection," in *Proc. ACCS'98*, 1997, pp. 150–158.
[15] A. K. Ghosh, A. Schwartzbard, and M. Schatz, "Learning program behavior profiles for intrusion detection," in *Proc. Workshop on Intrusion Detection and Network Monitoring*, Santa Clara, CA, April 1999, pp. 51–62.
[16] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: Alternative data models," in *Proc. IEEE Symp. Security and Privacy*, May 1999, pp. 133–145.
[17] T. Spyrou and J. Darzentas, "Intention modeling: Approximating computer user intentions for detection and prediction of intrusions," in *Information Systems Security*. London, U.K.: Chapman & Hall, May 1996, pp. 39–335.
[18] T. Kohonen, *Self-Organizing Maps*. Berlin, Germany: Springer, 1995.
[19] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, pp. 257–286, Feb. 1989.
[20] J. S. R. Jang, "Fuzzy inference system," in *Neuro-Fuzzy and Soft Computing*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

Sung-Bae Cho (M'98) received the B.S. degree in computer science from Yonsei University, Seoul, Korea, in 1988 and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Taejeon, Korea, in 1990 and 1993, respectively.

He worked as a Member of the Research Staff at the Center for Artificial Intelligence Research, KAIST, from 1991 to 1993. He was an Invited Researcher of the Human Information Processing Research Laboratories at the Advanced Telecommunications Research (ATR) Institute, Kyoto, Japan, from 1993 to 1995, and a Visiting Scholar at University of New South Wales, Canberra, Australia, in 1998. Since 1995, he has been an Associate Professor, Department of Computer Science, Yonsei University. His research interests include neural networks, pattern recognition, intelligent man–machine interfaces, evolutionary computation, and artificial life.

Dr. Cho was awarded outstanding paper prizes from the IEEE Korea Section in 1989 and 1992, and another from the Korea Information Science Society in 1990. He was also the recipient of the Richard E. Merwin prize from the IEEE Computer Society in 1993. He was listed in *Who's Who in Pattern Recognition* by the International Association for Pattern Recognition in 1994, and received the best paper awards at the International Conference on Soft Computing in 1996 and 1998. He also received the best paper award at the World Automation Congress in 1998, and was listed in *Marquis Who's Who in Science and Engineering* in 2000 and in *Marquis Who's Who in the World* in 2001. He is a member of the Korea Information Science Society, INNS, the IEEE Computer Society, and the IEEE Systems, Man, and Cybernetics Society.