Effective Intrusion Type Identification with Edit Distance for HMM-Based Anomaly Detection System

Ja-Min Koo and Sung-Bae Cho

Dept. of Computer Science, Yonsei University,
Shinchon-dong, Seodaemoon-ku, Seoul 120-749, Korea
{icicle, sbcho}@sclab.yonsei.ac.kr

Abstract. As computer security becomes important, various system security mechanisms have been developed. Especially anomaly detection using hidden Markov model has been actively exploited. However, it can only detect abnormal behaviors under predefined threshold, and it cannot identify the type of intrusions. This paper aims to identify the type of intrusions by analyzing the state sequences using Viterbi algorithm and calculating the distance between the standard state sequence of each intrusion type and the current state sequence. Because the state sequences are not always extracted consistently due to environmental factors, edit distance is utilized to measure the distance effectively. Experimental results with buffer overflow attacks show that it identifies the type of intrusions well with inconsistent state sequences.

1 Introduction

As the computer environment changes rapidly, the security mechanism plays an important role. Intrusion means the behavior which impairs the computer system by anomalous way [1] and it damages to the integrity, secrecy and availability of information [2]. To protect the computer system from the intrusions, a variety of security mechanisms such as firewalls have been developed. Intrusion detection system (IDS) is the representative among them [3]. Intrusion detection techniques are divided into two groups according to the type of data they use: misuse detection and anomaly detection. The former uses the knowledge about attacks and the latter does normal behaviors. However, most of IDSs have some technical inertia such as inability of detecting the cause and the path of intrusion because they have been developed to improve the detection rates. In case of misuse detection, it detects known intrusions very well because it is based on the known intrusion information, but it has vulnerability to transformed or novel intrusions. On the other hand, in case of anomaly detection, it can warn the intrusions but it cannot identify the type of intrusions. Moreover, it is very difficult to detect intrusions because it is possible to penetrate the system in other ways even if intrusions are of identical types. Accordingly, it is hard to take actions for each intrusion type in anomaly detection system.

To solve this problem, we analyzed the state sequences of each intrusion type using Viterbi algorithm in HMM-based intrusion detection system, and identified the type of intrusions by comparing the similarity using Euclidean distance [4]. However, since sequences are not always consistently extracted due to environmental factors, it is difficult to identify the type of intrusions even though identical intrusions are

S.K. Pal et al. (Eds.): PReMI 2005, LNCS 3776, pp. 222 – 228, 2005.

attempted. In this paper, we propose a method to identify the type of intrusions using Viterbi algorithm with edit distance, and verify the usefulness by comparing with the results of other distance measures.

2 Related Works

There are several techniques for IDSs based on anomaly detection such as expert system, statistics, neural network and hidden Markov model (HMM) as shown in Table 1.

Hidden Markov Model (HMM): HMM proposed by C. Warender in New Mexico University is good at modeling and estimating with event sequences which are unknown. It performs better to model the system than any others [5]. However, it takes long time to train and detect intrusions. To overcome such disadvantages, we can extract the events of changing the privilege before and after them, and we are able to model the normal behavior with them. By using this method, we can reduce the time to model and maintain good performance [6].

Table 1. Summary of the representative anomaly detection system (ES: Expert System, NN: Neural Network, ST: Statistics, HMM: hidden Markov model)

Organization	Name	Period	Technique			
			ES	NN	ST	HMM
AT&T	Computer Watch	1987-1990	X			
UC Davis	NSM	1989-1995			X	
	GrIDS	1995-	X			
SRI International	IDES	1983-1992			X	
	NIDES	1992-1995			X	
	EMERALD	1996-			X	
CS Telecom	HyperView	1990-1995		X	X	
New Mexico Univ.	C. Warender, et. al [5]	1999			X	X
Yonsei Univ.	YSCIDS [6]	1999-				X

3 Proposed Method

An IDS based on HMM collects and abridges normal auditing data, and it makes normal behavior models for a specific system, and then it detects intrusions with auditing data to detect the intrusions from it. Finally, to identify the type of intrusions, we analyze the state sequences of the system call events using the Viterbi algorithm [4]. Figure 1 shows the overall structure of the proposed method.

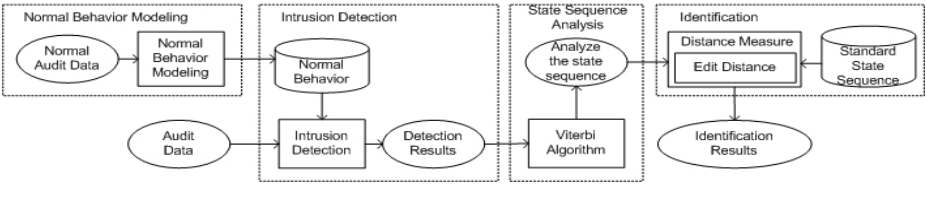


Fig. 1. Overall structure of the proposed method

We measure the similarity between the state sequences extracted and the standard state sequences of each intrusion type using edit distance. Edit distance is based on dynamic programming, and the result of edit distance is the minimal cost of sequence of operations that are used to compare strings x and y.

- $\delta(\varepsilon, a)$: inserting the letter a into string ε
- $\delta(a, \varepsilon)$: deleting the letter a from string ε
- $\delta(a, b)$: replacing a by b, for $a \neq b$

A matrix $C_{0\ldots|x|,0\ldots|y|}$ is filled, where $C_{i,j}$ represents the minimum number of operations needed to match $x_{1\ldots i}$ to $y_{1\ldots j}$. This is computed as follows for $C_{|x|,|y|} = ed(x,y)$ at the end

$$C_{i,0} = i \quad C_{0,j} = j \quad C_{i,j} = \text{if } (x_i = y_j) \text{ then } C_{i-1,j-1} \text{ else } 1 + \min(C_{i-1,j}, C_{i,j-1}, C_{i-1,j-1})$$

The rationale of the above formula is as follows. First, $C_{0,j}$ represents the edit distance between a string of length i or j and the empty string. Clearly, i and j deletions are needed for the nonempty string. For two non-empty strings of length i and j, we assume inductively that all the edit distances between shorter strings have already been computed, and try to convert $x_{1\ldots x}$ into $y_{1\ldots j}$.

Dynamic programming algorithm must fill the matrix in such a way that the upper, left and upper-left neighbors of a cell are computed prior to computing that cell. This is easily achieved by either a row-wise left-to-right traversal or a column-wise top-to-bottom traversal.

In our problem, we use the edit distance to measure the similarity of intrusion types. When intrusions occur, it calculates the distances between the standard state sequences and the current state sequence. Afterward, input state sequence is identified with the corresponding intrusion type of which the distance is the smallest. However, this algorithm has a weakness in case of input sequences as shown in Table 2.

Table 2. An example of the edit distance's in malfunction

A	1	2	3	4	5
B	1	2	3	2	4
C	1	2	3	5	5
Input	1	3	3	5	

Assuming that the input sequences are given as Table 2, we can recognize that one of input state sequences is deleted and substituted with all classes, A, B and C, and the results of edit distance are all 2. Therefore, we cannot identify the type using normal edit distance. Therefore, we use a modified edit distance with weighting facility to identify the type of intrusions well, and the formula is as follows.

$$C_{0,0} = 0$$

$$C_{i,j} = \text{if } x_i = y_j$$

$$\min(C_{i-1,j-1})$$

else

$$\min(w_{\delta(a,b)} \times C_{i-1,j-1} \times abs(a,b), w_{\delta(a,\varepsilon)} \times C_{i-1,j} \times abs(a,\varepsilon), w_{\delta(\varepsilon,a)} \times C_{i-1,j} \times abs(\varepsilon,a))$$

where $w_{\delta(\varepsilon,a)} = 1$; $w_{\delta(a,\varepsilon)} = 2$; $w_{\delta(a,b)} = 3$

4 Experiments

We have collected normal behaviors from six users for 2 weeks using Solaris 7 operating system. They have mainly used text editor, compiler and program of their own writing. In total 13 megabytes (160,448 events are recorded) of BSM audit data are used. Among many intrusions, buffer overflow gets root privileges by abusing systems' vulnerability. We try to penetrate 30 times of each intrusion type. There are 3 kinds of intrusion types called OpenView xlock Heap Overflow, Lpset –r Buffer Overflow Vulnerability and Kcms_sparc Configuration Overflow. The first is used to overflow the buffer by inputting the environmental value which is larger than 1024 bytes. At that time when the buffer is overflow, we can obtain the root privileges by inputting arbitrary commands. The second is the package for setting the printer. Local user can obtain the root privileges with this package operating with long commands with –r flag value. The third overflow is in an sprintf() call that occurs when kcms_sparc configuration is called with -o -S flag value which is larger than 1024 bytes.

In this paper, we use the edit distance for identifying the type of intrusions. To verity the usefulness, we use other 3 kinds of distance measures for comparing the performance: Euclidean distance, Hamming distance and Longest Common Subsequence (LCS).

(1) Euclidean Distance
When intrusions are occurred, the similarity can be compared between the standard state sequence for every intrusion type and the state sequence of current intrusion using Euclidean distance. The formula is as follows.

$$ed = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2}$$

Assuming the analyzed state sequence of current intrusion to x_i, and the state sequence of every intrusion type to y_j, we calculate the Euclidean distance *ed*. The smaller the value is, the higher the similarity is. Hence, the intrusion type that we have found has the least value of *ed*.

(2) Hamming Distance
Hamming distance is used to compare the similarity between two strings whose lengths are the same. The formula is as follows.

$$hd(x, y) = \sum_{i=0}^{n-1} hd(x_i, y_i) \qquad hd(x_i, y_i) = \begin{cases} 0 & x_i = y_i \\ 1 & x_i \neq y_i \end{cases} \quad (iff\ .\ |x| = |y| = k)$$

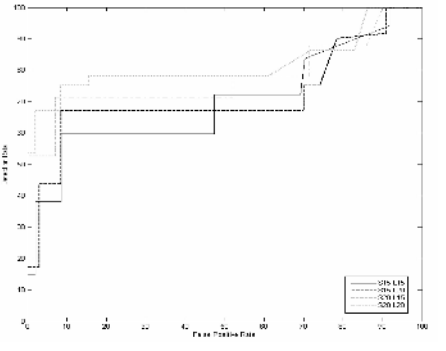
(3) Longest Common Sequence (LCS)
It is the way to calculate the longest common sequence between two sequences: the more common the sequences are, the higher the similarity is. The formula is as follows.

$$C[i, j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ C[i-1, j-1] + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max(C[i, j-1], C[i-1, j]) & \text{if } i, j > 0 \text{ and } x_i \neq y_j \end{cases}$$

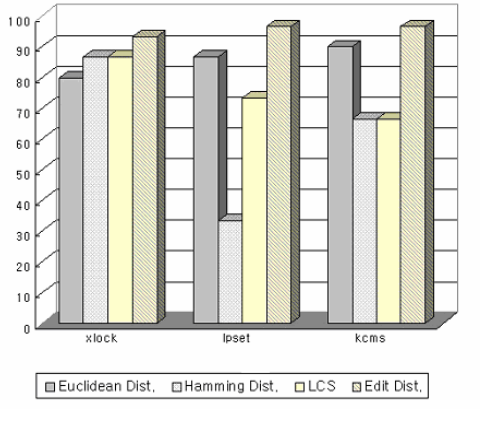
At first, we conduct the experiments of HMM with varying the number of states from 5 to 30, and observation lengths of 10 and 15. If the number of states is set by small value, the state sequences of all intrusion types are extracted consistently because of the possible state transition paths limitation. Therefore, we have difficulties in identifying the type of intrusions with the number of states less then 20. Figure 2(a) shows the performance of the HMM-based IDS with the number of states 15 and 20, and observation lengths 10 and 15.

We have set the experimental environment with the number of states 20 and the number of observation lengths 10 for identifying the type of intrusions, and compared the performance using various distance measures. Figure 2(b) shows the performance among 4 kinds of distance measures.

We can observe that the performance of using edit distance is better than those of the others as shown in Figure 2(b). Especially, since LCS and Hamming distance only



(a)



(b)

Fig. 2. (a) Performance of HMM-based IDS (X-axis: False Positive Rate Y-axis: Detection Rate) (b) Results of the success rates of each distance measure (X-axis: Intrusion Type Y-axis: Accuracy of Identification)

compare the state sequence one by one and check up whether the sequences are identical or not, the performance is lower than any other when trying to identify the lpset intrusion type.

Table 3. Examples of the input state sequence

	1	2	3	4	5	6	7	8	9	10	Type
Case 1	0	2	4	6	8	8	10	10	12	14	
Case 2	0	2	4	6	8	8	10	12	14	16	KCMS
Case 3	0	2	3	5	7	9	11	13	17	18	

For example, assume that the standard state sequences are given as Table 3 and input state sequence is like case 1 of Table 3. When we try to identify the type of intrusions with Euclidean distance, input state sequence is identified as kcms: *D*(xlock)=6.7082, *D*(lpset)=6.6332 and *D*(kcms)=5.6596, but with Hamming distance the input state sequence is identified as xlock, because of *HD*(xlock)=6, *HD*(lpset)=5 and *HD*(kcms)=5. In general, the performance of Hamming distance is lower than that of Euclidean distance.

In addition, with LCS the type of intrusions is identified as specific class which has the largest common subsequences. If the standard state sequences are like Table 3, and the input state sequence is like case 2 of Table 3, it cannot identify the type of intrusions correctly since the xlock and lpset results with LCS are the same: LCS(xlock)=LCS(lpset)=8.

On the other hand, in the case 3 of Table 3, the input state sequence is identified as xlock with Euclidean distance: 2.8284. However, if we use the edit distance to identify the type of intrusion, we can identify the input state sequence as kcms: *E.D*(xlock)=24 and *E.D*(kcms)=21. We can reduce the error rate using the edit distance which performs proper operations.

5 Concluding Remarks

In this paper, we have proposed a method to identify the type of intrusions in the anomaly detection system based on HMM. The proposed method calculates the edit distance to compare the similarity between the standard state sequences modeled and the current state sequence obtained by using Viterbi algorithm when intrusion occurs. Experiments are performed in the intrusion detection system based on HMM in 100% intrusion detection rates. We change the number of states from 5 to 30 and the length of observation symbols from 10 to 20 in the experiments. As a result, since the possible state transition paths are of small number, the system identifies all the types of intrusions when the number of states is more than 20. We have conducted additional experiments with other 3 distance measures, LCS, Hamming distance and Euclidean distance to verify the usefulness. The experiments indicate that it is very difficult to identify the type of intrusions by using LCS and Hamming distance, while the edit distance produces good performance with proper operations.

References

1. D. Denning, “An intrusion-detection model,” *IEEE Trans. on Software Engineering,* vol. 13, no. 2, pp. 212-232, Feb. 1987.
2. B. Balajinath and S. V. Raghavan, “Intrusion detection through learning behavior model,” *Computer Communications,* vol. 24, pp. 1202-1212, Jul. 2001.
3. H. S. Vaccaro and G. E. Liepins, “Detection of anomalous computer session activity,” In *Proc. of IEEE Symp. on Research Security and Privacy,* pp. 280-289, 1989.
4. J.-M. Koo and S.-B. Cho, “Viterbi algorithm for intrusion type identification in anomaly detection system,” *Lecture Notes in Computer Science 2908,* pp. 97-110, Dec. 2003.
5. C. Warrender, S. Forrest and B. Pearlmutter, “Detecting intrusion using calls: Alternative data models,” *In Proc. of IEEE Symp. on Security and Privacy,* pp. 133-145, May 1999.
6. S.-B. Cho and H.-J. Park, “Efficient anomaly detection by modeling privilege flows using hidden Markov model,” *Computers & Security,* vol. 22, no. 1, pp. 45-55, 2003.