Proceedings of the Second International Conference on Machine Learning and Cybernetics, Xi'an, 2-5 November 2003

INTRUSION DETECTION BASED ON HIDDEN MARKOV MODEL

QING-BO YIN, LI-RAN SHEN, RU-BO ZHANG, XUE-YAO LI, HUI-QIANG WANG

College Of Computer Science And Technology, Harbin Engineering University, Harbin 150001 ,China
E-MAIL:s1_123@up369.com

Abstract:

The intrusion detection technologies of the network security are researched, and the technologies of pattern recognition are used to intrusion detection. Intrusion detection rely on a wide variety of observable data to distinguish between legitimate and illegitimate activities. Hidden Markov Model (HMM) has been successfully used in speech recognition and some classification areas. Since Anomaly Intrusion Detection can be treated as a classification problem, some basic ideas have been proposed on using HMM to model normal behavior. The experiments have showed that the method based on HMM is effective to detect anomalistic behaviors.

Keywords:

Hidden markov model; Intrusion detection; Network security

1 Introduction

Intrusion detection play an important role in detecting attacks that exploit the vulnerabilities or flaws in computer networks. An ideal intrusion detection system is the one that has 100% attack detection rate along with 0% false positive rate (the rate of mis-classified normal behavior), requires light load of monitoring, and involves minor calculation or overhead. Current intrusion detection systems, however, are plagued by either high false alarm probability or low attack detection accuracy. There are two general approaches to intrusion detection: misuse detection and anomaly detection. Misuse detection via signature verification compares a user's activities with the known signatures of attackers attempting to penetrate a system. While misuse detection is useful for finding known intrusion types, it cannot detect novel attacks. Unlike misuse detection, anomaly detection identifies activities that deviate from established statistical patterns for users, systems or networks. Machine learning techniques have been used to capture the normal usage patterns and classify the new behavior as either normal or abnormal. In spite of their capability of detecting unknown attacks, anomaly detection systems suffer from high false alarm rate when normal user profiles and system or network behavior vary widely.

In this paper, a new technique has been used for learning program behavior in intrusion detection. Our approach employs HMM to classify each program behavior into either normal or intrusive class that improves the model time and performance by only considering the system calls of privilege programs as time series.

2 Principle and Method

2.1 Principle

Anomaly detection can be combined with signature verification to detect attacks more efficiently. The biggest challenge is to choose features that best characterize the user or system usage patterns so that non-intrusive activities would not be classified as anomalous. One could use, for instance, Unix Shell command lines, login events or system calls as observable to generate profiles of user behavior. Since a user's behavior can change frequently, user profiles have to be updated periodically to include the most recent changes.

More recently, learning program behavior and building program profiles, especially those of privileged programs, has become an alternative method in intrusion detection. In Unix, intruders usually gain super-user status by exploiting privileged programs. A program profile can be generated by monitoring the program execution and capturing the system calls associated with the program. Compared to user behavior profiles, program profiles are more stable over time because the range of program behavior is more limited. Furthermore, it would be more difficult for attackers to perform

0-7803-7865-2/03/$17.00 ©2003 IEEE

intrusive activities without revealing their tracks in the execution logs. Therefore program profiles provide concise and stable tracks for intrusion detection. Now, almost all the research in learning program behavior has used short sequences of system calls as the observable, and generated a large individual database of system call sequences for each program. A program's normal behavior is characterized by its local ordering of system calls, and deviations from their local patterns are regarded as violations of an executing program. It is still a tedious and costly approach because system and application programs are constantly updated, and it is difficult to build profiles for all of them.

Now, a new method is introduced for detecting intrusions based on the temporal behavior of applications. It builds on an existing method of application intrusion detection developed at the University of New Mexico that uses a system call sequence as a signature. Intrusions are detected by comparing the signature of the intrusion and that of the normal application. Analysis shows that the temporal behavior for many applications is relatively stable. This paper discusses the experiments that test the effectiveness of the temporal signature on different applications, alternative intrusions. The results show that by choosing appropriate analysis methods and adjusting the parameters, intrusions are readily detected.

2.2 Hidden Markov Model

A HMM is a doubly stochastic process with an underling stochastic process that is not observable, but can only be observed through another set of stochastic processes that produce the sequence of observed symbols. This is a useful tool to model sequence information. A HMM's states represent some unobservable condition of the system being modeled. In each state, there is a certain probability of producing any of the observable system outputs and a separate probability indicating the likely next states. By having different output probability distributions in each of the states, and allowing the system to change states over time, the model is capable of representing nonstationary sequences.

3 Feature Extract and Intrusion Detection Algorithm

3.1 Feature Extract

The signal series of the training data $x(n), n = 1 \cdots N$ is parted to M sections:

$$x_k(p) = x((M-1)*\lfloor N/M \rfloor + p), \quad p = \lfloor 1 \cdots N/M \rfloor, \quad k = 1 \cdots M \tag{1}$$

Correlation analysis:

$$R_k(\tau) = \sum_{n=-\infty}^{\infty} x_k(n) x_k(n+\tau) \tag{2}$$

3.2 Intrusion Detection Algorithm

This method can be derived using simple "occurrence counting" arguments or using calculation to maximize the auxiliary quantity. In maximum-likelihood criterion, try to maximize the probability of given sequence of observations. Given HMM λ, is the total likelihood of the observations and can be expressed mathematically as

$$ML = P(O/\lambda) \tag{3}$$

Typically, parameter estimation for HMM is performed using standard Baum-Welch algorithm with the maximum-likelihood criterion. The Baum-Welch algorithm for HMM is simple, well defined, and stable. This requires additional two variables: $\xi_t(i,j)$ is defined as the probability with which it stays at state s_t at time t and stays at state s_j at time t+1.

$$\begin{aligned}\xi_t(i,j) &= P(q_t = s_i, q_{t+1} = s_j / O, \lambda) \\ &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O/\lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}\end{aligned} \tag{4}$$

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

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{5}$$

Summing up the two variables over time t respectively, get the probability with which state i transits to state j and the expectation at state i. Given the above variables calculated, a new model $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ can be adjusted using the following equations:

$\bar{\pi}$ = expect frequency(number of times) in state S_i at time (t = 1) (6)
$= \gamma_1(i)$

$$\overline{a_{ij}} = \frac{\text{expect number of transitions from state Si to state Sj}}{\text{expected number of transitions from state Si}} = \sum_{t=1}^{T-1} \xi_t(i,j) \Big/ \sum_{t=1}^{T-1} \gamma_t(i) \tag{7}$$

$$\overline{b_j}(k) = \frac{\text{expected number of times in state j and observing symbol } V_k}{\text{expected number of times in state j}} = \sum_{\substack{t=1 \\ s.t. O_t = v_k}}^{T} \gamma_t(j) \Big/ \sum_{t=1}^{T} \gamma_t(j) \tag{8}$$

If the current model is defined as $\lambda = (A, B, \pi)$, and the reestimated model is defined as $\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\pi})$, it has been proven by Baum and his colleagues that either 1) the initial model λ defines a critical point of the likelihood function, in which case $\overline{\lambda} = \lambda$;or 2) model $\overline{\lambda}$ is more likely than model λ in the sense that $P(O/\overline{\lambda}) > P(O/\lambda)$.

3.3 Anomaly Detection

Anomaly detection matches current behavior against the normal behavior models and calculates the probability with which it is generated out of each model. Forward-backward procedure can be used for this. Forward procedure calculates the probability $P(O/\lambda)$ with which input sequence O is generated out of model λ using forward variables. Forward variable α denote the probability at which a partial sequence $O_1, O_2, ..., O_T$ is observed and stays at state S_i.

$$\alpha_t(i) = P(O_1, O_2, ..., O_t, q_t = S_i / \lambda) \tag{9}$$

According to the above definition, $\alpha_t(i)$ is the probability with which all the symbol V_k in input sequence are observed in order and the final state is i. Summing up $\alpha_t(i)$ for all i yields the value $P(O/\lambda)$. $\alpha_t(i)$ can be calculated by the following procedure.

Step 1. extract feature

Step 2 . initialization:

$$\alpha_1(i) = \pi_i b_i(O_1) \tag{10}$$

step 3. induction:

for t=1 to T-1

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1}) \tag{11}$$

step 4. termination:

$$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{12}$$

4 Experimental Results

The data used in experiments have been obtained from the University of New Mexico. All of these data sets are publicly available and carefully described at http://www.cs.unm.edu/immsec/data-sets.html. Table 1 summarizes the different data sets and program. Intrusions were taken from public advisories posted on the Internet.

A desirable intrusion detection system must show a high intrusion detection rate with a low false-positive error rate. Standard HMM has a fixed number of states, so we must decide the size of model before training. Through the experiments, we find that the model work well when the 4~16 states are used in HMM. Table 2 shows the intrusion detection rate and false-positive error rate.

Table 1. Amount of data available for each program

Program	Intrusions	Normal data available	
	Number of traces	Number of traces	Number of system calls
MIT lpr	1001	2703	2926304
UNM lpr	1001	4298	2027468
named	2	27	9230572
xlock	2	72	16937816
login	9	12	8894
ps	26	24	6144
inetd	31	3	541
stide	105	13726	15618237
sendmail		71760	44500219

Table 2. The anomaly detection rate for each model

State	Attack	Detection rate	F-P error
5	439	100%	0.436%
10	439	100%	1.67%
16	439	100%	1.45%

The results indicate that comparing the signature of the intrusion and that of the normal application is useful to detect intrusion when normal behaviors have been modeled by HMM. The combination of 5 states is seen to be the most effective in this experiment.

5 Conclusions

In this paper, a new method has been proposed for anomaly intrusion detection using the temporal information of the privilege program. The idea comes from the observation that is stable or change slowly in short time. With sufficiently large number of training data, it will show better performance.

The primary experiments show that this method performed well. However just as everything, there are something that are needed to be ulteriorly studied, such as feature extract and deciding the number of the states of HMM and so on. All these work are still doing now. But it is believed that the temporal signature method provides an effective approach to the detecting anomalistic behaviors.

References

[1] W. Lee, S. J. Stolfo, and P. K. Chan. Learning patterns from UNIX process execution traces for intrusion detection. In *AAAI Workshop on AI Approaches to Fraud Detection and Risk Management*, pp.50–56. AAAI Press, July 1997.

[2] L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, Vol.77(2), pp.257–286, 1989.

[3] L. R. Rabiner and B. H. Juang. An introduction to Hidden Markov Models. *IEEE ASSP Magazine*, pp.4–16, Jan-uary 1986.

[4] Warrender. C, Forrest. S.and Pearlmutter. B. Detecting Intrusion Using System Calls: Alternative Data Models. IEEE Symposium on Security and Privacy, May 1999.