WORKLOAD HIDDEN MARKOV MODEL FOR ANOMALY DETECTION

Juan Manuel García
Instituto Tecnológico de Morelia
Morelia, México

Tomás Navarrete
Instituto Tecnológico de Morelia
Morelia, México

Carlos Orozco
FIRA - Banco de México
Morelia, México

Keywords: Intrusion detection, anomaly detection, time series analysis, Markov processes.

Abstract: We present an approach to anomaly detection based on the construction of a Hidden Markov Model trained on processor workload data. Based on processor load measurements, a HMM is constructed as a model of the system normal behavior. Any observed sequence of processor load measurements that is unlikely generated by the HMM is then considered as an anomaly. We test our approach taking real data of a mail server processor load to construct a HMM and then we test it under several experimental conditions including a simulated DoS attacks. We show some evidence suggesting that this method could be successful to detect attacks or misuse that directly affects processor performance.

1 INTRODUCTION

Since the beginning of the intrusion detection study (Denning, 1987) two complementary approaches to detect a possible intruder were established :

1. *Anomaly detection*, where the strategy is to suspect of what is considered an unusual activity for the subject (users, processes, etc.) and carry on further investigation. This approach is particularly effective against novel (i.e. previously unknown) attacks. Its main drawback is the high rate of false positives, because any legitim but new activity can rise an alert.
2. *Signature detection*, where the strategy is to look for some special activity (*signature*) of previously known attacks. Signature based detection systems detects previously known attack in a timely and efficient way. The main issue of this approach is that in order to detect an intrusion this must to be previously detected.

In order to carry on anomaly detection is required to establish what is the normal state of a system. Several approaches have been used to define what is system normality. The state of a computer system can be defined in terms of several measurable variables like processor load, memory usage, processes number, etc. Matter of fact, system administrators usually observe some of this variables in order to detect if something is going wrong. This variables then can be used to define system normality using statistical analysis, so we can obtain an adaptive intrusion detection system.

In this work, we use CPU server load measurements to construct a Hidden Markov Model that reflects the expected variations of server workload. We pretend to find an effective and efficient way to detect attacks (like DoS) that directly degrade the server performance.

2 RELATED WORK

Anomaly detection relies on models of what is considered the ’normal’ behavior of users, systems and applications and interprets deviations of this behavior as evidence of malicious activity (Denning, 1987; Ko et al., 1997; Gosh et al., 1998; Lane and Brodley, 1999). Several techniques to express quantitatively the normal state of a system have been proposed, including analysis of data streams of network traffic (Lee et al., 1999), sequence analysis for operating system calls (Forrest et al., 1996), and data mining of system audit data (Lee and Stolfo, 2000).

Some recent work on anomaly detection applies

Manuel García J., Navarrete T. and Orozco C. (2006).
WORKLOAD HIDDEN MARKOV MODEL FOR ANOMALY DETECTION.
In *Proceedings of the International Conference on Security and Cryptography*, pages 56-60
DOI: 10.5220/0002099700560060

principles and concepts borrowed from biological sciences (Coull et al., 2003; Burgess, 1998; Forrest et al., 1996; Forrest et al., 1997). In particular some approaches are inspired by immunology, as in (Forrest et al., 1996) where system calls are analyzed to infer anomalous behavior. In (Lane and Brodley, 1999) a technique that applies Instance Based Learning (IBL) to temporal sequence of events in order to characterize normal behavior of users, systems and applications was presented. In (Michael and Ghosh, 2002) a finite-state machine was constructed from audit data to monitor statistical deviations from normal program behavior. In (Yin et al., 2004) a Markov chain model of system calls was applied to anomaly detection.

Specially related to our work, are the techniques presented by (Wright et al., 2004) to building Hidden Markov Models profiles for network traffic, using only information that remains invariant after encryption like packet size and arrival time.

Hidden Markov Models (HMMs) are a tool for modeling time series data, and are used for computational molecular biology, data compression, speech recognition, computer vision and other areas of artificial intelligence and pattern recognition. For a general introduction to Hidden Markov Models and its applications see (Ghahramani, 2002; Jordan et al., 1999).

3 MEASUREMENTS

First of all, processor workload of a mail server was registered over several months. Processor load was measured each 120 seconds through SNMP queries (MacFaden et al., 2003; Presuhn, 2002). This time interval was used because was observed that any measurement taken in between was almost invariant.

As an example, measurements taken on January 2005 are showed in figure 1. Each curve in figure correspond to a day of the month, and each point represents a 15 min. average of processor load.

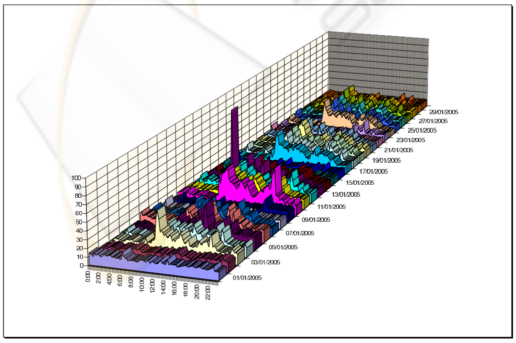

Figure 1: Processor load measured on January 2005.

Strong correlation was observed between measurements taken on the same day of the week, with least activity on weekends and most activity on Monday morning. Monthly averages were calculated for the seven days of the week. In figure 2 are showed the Monday averages where peaks of activity can be observed. We found that those peaks are close related to the organization activity. For example, the peak at 9:00 a.m. showed in fig. 2 correlates to the beginning of weekly activities, when most of the users download the emails that they receive on weekend. Also we can see an activity descent near three o'clock when most of the users take their lunch, and a peak of activity near the checkout time at eight o'clock. On weekends, when nobody is working, the activity measured remains almost flat.

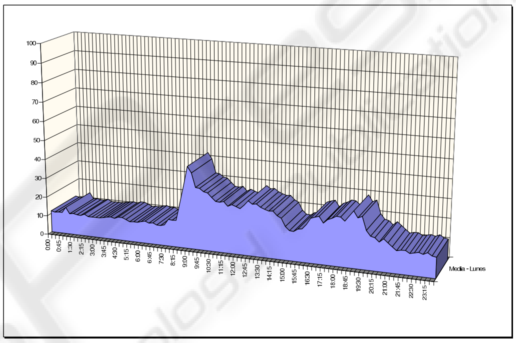

Figure 2: Average processor load on mondays.

4 HMM CONSTRUCTION

Let us denote the observation at time t by the variable Y_t. First, according to the HMM, we assume that the observation at the time t was generated by some process whose state S_t is *hidden* from the observer. A second assumption is that the state S_t is dependent only of the previous state S_{t-1} and the output Y_t only depends on the state S_t. A third assumption is that the hidden state variable is *discrete*: S_t can take on K values denoted by the integers $\{1, \ldots, K\}$. Then, in order to define a HMM, is needed to specify a probability distribution over the initial state $P(S_1)$, the $K \times K$ state transition matrix defining $P(S_t|S_{t-1})$ and output model Π defining $P(Y_t|S_t)$.

Actually we construct a HMM for each day of the week, and we use monthly average sequences of each day of the week as input of the learning algorithm . In the following discussion, every numerical example is referred to the HMM of Monday.

In our case, our sequence of observations Y_t take integer values ranging from 0 to 100, representing the percentage of processor load. To determine K we applied the gradient over the monthly averaged values to obtain critical points that suggest state transitions, and taking the number of critical points we obtain that for Mondays $K = 6$. Then each state transition is related with a major change of the average of processor load and we use this fact to construct the initial probabilities of our model.

We take a Gaussian observation model

$$P(Y_t|S_t) = \frac{1}{\sigma_S\sqrt{2\pi}} e^{-(Y_t^2-\mu_S^2)/(2\sigma_S^2)} \tag{1}$$

where σ_S and μ_S depends on the state S_t.

We associate S_1 with the ”on rest” state of the system, and taking our observations beginning at midnight, S_1 is always the initial state so we take the start state probability $P(S_1)$ as 1 when $S_1 = 0$ and 0 elsewhere.

To estimate the initial transition matrix we estimate probabilities using a kind of Bayesian rule as follows

$$P(S_i|S_j) \approx \frac{f(\mu(S_i)|\mu(S_j))}{f(\mu(S_j))} \tag{2}$$

where $f(\mu(S_i)|\mu(S_j))$ is the observed frequency of transition of the S_j associated average to the S_i related average, and $f(\mu(S_j))$ as the measured frequency of the average associated with the S_j state.

Using this method we estimate a initial state transition matrix as

$$\begin{pmatrix} 0,97 & 0,03 & 0 & 0 & 0 & 0 \\ 0 & 0,88 & 0,12 & 0 & 0 & 0 \\ 0 & 0 & 0,94 & 0,5 & 0 & 0 \\ 0 & 0 & 0 & 0,8 & 0,2 & 0 \\ 0 & 0 & 0 & 0 & 0,92 & 0,08 \\ 0,13 & 0 & 0 & 0 & 0 & 0,87 \end{pmatrix}$$

With this initial state transition matrix, the Baum-Welch algorithm (Ghahramani, 2002) was applied to learn the processor load measurement sequences to obtain the state transition matrix:

$$\begin{pmatrix} 0,977 & 0,023, & 0 & 0 & 0 & 0 \\ 0 & 0,933 & 0,067 & 0 & 0 & 0 \\ 0 & 0 & 0,939 & 0,061 & 0 & 0 \\ 0 & 0 & 0 & 0,89 & 0,11 & 0 \\ 0 & 0 & 0 & 0 & 0,916 & 0,084 \\ 0,199 & 0 & 0 & 0 & 0 & 0,801 \end{pmatrix}$$

As can be observed, the final transition matrix is close to our initial matrix so we can guess that our outlined construction method give us almost a correct HMM.

Finally, as output probability distributions we adjust normal distributions to the observations measured, obtaining for each state the parameters showed at table 1

Table 1: Output normal distributions parameters.

State	μ_S	σ_S
0	11,41	2,47
1	32,86	9,50
2	25,33	4,11
3	16,35	1,82
4	24,00	5,13
5	21,64	6,35

5 EXPERIMENTAL RESULTS

Given a observation sequence $X_1, X_2, \ldots, X_T$ we use the Forward-Backward algorithm to estimate the probability $P(X_{1:T})$ that such sequence could be generated by our HMM. Since the probability $P(Y_{1:T})$ for a typical sequence $Y_1, \ldots, Y_T$ of normal behavior is $\sim 10^{-104}$ we use a the following metric

$$|\log P(X_{1:T}) - \log P(Y_{1:T})| \tag{3}$$

to discriminate between normal and abnormal observations.

To test our model we apply several simulated observation sequences to see if they could be detected as anomalies:

1. *Noise.* We test our detector with a sequence $X_1, X_2, \ldots$ where each X_i is a random number between 0 and 100 following a binomial distribution with different means. With exception of weekend’s models, this sequence was alway rejected.
2. *Catastrophe.* (Burgess et al., 2002) In a valid observation sequence we introduce sudden discontinuous changes with variant intensity. The anomaly was detected if the magnitude measured by (3) was greater than $\sim 2,6$.
3. *DoS attack.* In a similar way, we introduce in a valid observation sequence some measurements indicating a processor overload. This kind of sequences were always detected as an anomaly, except for the following case.
4. *Mimicry attacks* (Wagner and Soto, 2002) To simulate a mimicry attack effect, we introduce some data indicating processor overload on a valid sequence, but in coincidence with normal peaks of activity. As in catastrophe’s case, this simulated attacks were detected if (3) was greater than $\sim 2,6$.

By the time this paper is written, we are testing our model on an real productive environment in order to obtain rates of false positives. A first discovery is that normal administrative tasks (like a patch installation) can rise false alerts. We are also testing with a HMM generated by K-Means algorithm but, until now, whitout obtaining better results that those reported here.

6 CONCLUSIONS AND FUTURE WORK

In this paper we present an application of Hidden Markov Models of processor load behavior for anomaly detection. We show experimental evidence suggesting that this approach can be successful to detect attacks or misuse that directly affects processor performance. As we state in the introduction, system normality can be defined in terms of several variables like processor load, memory usage, etc., and then our method could be more effective if it takes into account not only processor load but another parameters like network traffic (Wright et al., 2004).

We found in our case that processor load is close related with activity cycles of our organization. A more realistic model of what is the normal behavior of a system must take under consideration natural and social cycles of activity.

Finally we agree with (Axelsson, 2000) in the conclusion that intrusion detection is a problem far from been solved.

ACKNOWLEDGEMENTS

The authors wish to thank to FIRA - Banco de México for provide us with experimental data and logistic support.

Also we wish to tank to our anonymous referees for their useful and insightful comments.

REFERENCES

Axelsson, S. (2000). The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3(3):186–205.

Burgess, M. (1998). Computer immunology. In *LISA '98: Proceedings of the 12th Conference on Systems Administration*, pages 283–298, Berkeley, CA, USA. USENIX Association.

Burgess, M., Haugerud, H., Straumsnes, S., and Reitan, T. (2002). Measuring system normality. *ACM Trans. Comput. Syst.*, 20(2):125–160.

Coull, S., Branch, J., Szymanski, B., and Breimer, E. (2003). Intrusion detection: A bioinformatics approach. In *ACSAC '03: Proceedings of the 19th Annual Computer Security Applications Conference*, page 24, Washington, DC, USA. IEEE Computer Society.

Denning, D. E. (1987). An intrusion-detection model. *IEEE Trans. Softw. Eng.*, 13(2):222–232.

Forrest, S., Hofmeyr, S. A., and Somayaji, A. (1997). Computer immunology. *Commun. ACM*, 40(10):88–96.

Forrest, S., Hofmeyr, S. A., Somayaji, A., and Longstaff, T. A. (1996). A sense of self for unix processes. In *SP '96: Proceedings of the 1996 IEEE Symposium on Security and Privacy*, page 120, Washington, DC, USA. IEEE Computer Society.

Ghahramani, Z. (2002). An introduction to hidden markov models and bayesian networks. *Hidden Markov models: applications in computer vision*, pages 9–42.

Gosh, A. K., Wanken, J., and Charron, F. (1998). Detecting anomalous and unknown intrusions against programs. In *ACSAC '98: Proceedings of the 14th Annual Computer Security Applications Conference*, page 259, Washington, DC, USA. IEEE Computer Society.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233.

Ko, C., Ruschitzka, M., and Levitt, K. (1997). Execution monitoring of security-critical programs in distributed systems: a specification-based approach. In *SP '97: Proceedings of the 1997 IEEE Symposium on Security and Privacy*, page 175, Washington, DC, USA. IEEE Computer Society.

Lane, T. and Brodley, C. E. (1999). Temporal sequence learning and data reduction for anomaly detection. *ACM Trans. Inf. Syst. Secur.*, 2(3):295–331.

Lee, W. and Stolfo, S. J. (2000). A framework for constructing features and models for intrusion detection systems. *ACM Trans. Inf. Syst. Secur.*, 3(4):227–261.

Lee, W., Stolfo, S. J., and Mok, K. W. (1999). Mining in a data-flow environment: experience in network intrusion detection. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 114–124, New York, NY, USA. ACM Press.

MacFaden, M., Partain, D., Saperia, J., and Tackabury, W. (2003). *Configuring Networks and Devices with Simple Network Management Protocol (SNMP), RFC3512*. RFC Editor, United States.

Michael, C. C. and Ghosh, A. (2002). Simple, state-based approaches to program-based anomaly detection. *ACM Trans. Inf. Syst. Secur.*, 5(3):203–237.

Presuhn, R. (2002). *Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), RFC 3418*. RFC Editor, United States.

Wagner, D. and Soto, P. (2002). Mimicry attacks on host-based intrusion detection systems. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 255–264, New York, NY, USA. ACM Press.

Wright, C., Monrose, F., and Masson, G. M. (2004). Hmm profiles for network traffic classification. In *VizSEC/DMSEC '04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 9–15, New York, NY, USA. ACM Press.

Yin, Q., Zhang, R., and Li, X. (2004). An new intrusion detection method based on linear prediction. In *InfoSecu '04: Proceedings of the 3rd international conference on Information security*, pages 160–165, New York, NY, USA. ACM Press.