Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 32 (2014) 947 – 952

Procedia Computer Science

The 4th International Symposium on Frontiers in Ambient and Mobile Systems (FAMS-2014)

Policy Misuse Detection in Communication Networks with Hidden Markov Models

Umut Tosun*

Baskent University Department of Computer Engineering, Engineering Faculty Baglica Campus, Ankara 06530, Turkey

Abstract

With the recent advances in computer networking applications, Intrusion Detection Systems (*IDS*) are widely used to detect the malicious connections in computer networks. *IDS* provide a high level security between organizations while preventing misuses and intrusions in data communication through internet or any other network. Adherence to network usage policies is crucial since a system or network administrator needs to be informed whether the information is compromised, if the resources are appropriately used or if an attacker exploits a comprised service. Server flow authentication via protocol detection analyzes penetrations to a communication network. Generally, port numbers in the packet headers are used to detect the protocols. However, it is easy to re-map port numbers via proxies and changing the port number via compromised host services. Using port numbers may be misleading for a system administrator to understand the natural flow of communications through network. It is also difficult to understand the user behavior when the traffic is encrypted since there is only packet level information to be considered. In this paper, we present a novel approach via Hidden Markov Models to detect user behavior in network traffic. We perform the detection process on timing measures of packets. The results are promising and we obtained classification accuracies between %70 and %100.

Keywords: Policy Misuse, Hidden Markov Models

1. Introduction

Server flow authentication is important to understand if a service is compromised or not in a system. In this paper, we try to solve the server flow authentication problem. The question is to detect the *TCP* application protocol information of a stream of data flowing in a system. In this way, the system administrator can analyze the resource utilization of the system, compromised services and unauthorized access to resources. The classical way is to inspect the source destination port mappings of client and server. Some of the well-known protocols like *DNS* uses port 53 and http uses port 80. Port label and traffic type relation is a common asset for intrusion detection[1],[2]. Different attack

*Umut Tosun. Tel.: +0-090-312-2466661/2099 ; fax: +0-090-312-2466660.
E-mail address: utosun@baskent.edu.tr

doi:10.1016/j.procs.2014.05.516

scenarios may eliminate the usage of port numbers for protocol detection. As an example, proxies can be used to bypass network policies[4].

Proxy converts the port numbers of the garbage packets to valid port numbers. Thus, the system policies are invaded. Similarly, compromised services place a back door for the intruder to access control over the system resources. As an example *ssh* port can be compromised and the compromised service can redirect the packet to another port in case of connection failure. The attacker knows this property and arranges the functionality of the binary in order to use it both with *ssh* and other service. There are also some client installed programs working as a back door in the system. These are installed via peer to peer programs, viruses or trojans. It is difficult to understand the traffic if user installed applications are working as trojans or worms. In all of these scenarios, port numbers fail to detect the existence of a threat. The identification of the protocols might clarify the authentication of the server flows for all of these scenarios. Several studies were proposed for the detection of unauthorized services. File system integrity and intrusion evidences are checked by methods like *Tripware*[6] and *Chkrootkit*[5]. These tools suffer in terms of configuration difficulties. Moreover, the systems that these tools are working might be compromised. In that case, the reports of these tools are doubtful. Malicious software such as user installed malware may lead an unauthorized access to a computer and hide programs, processes, files etc. from the operating system. Some third party audit software might also be used to detect these misleading situation. However, there is still a risk that the server flows are not analyzed correctly. A Hidden Markov Model (*HMM*) is a Markov process with some hidden states. It is the simplest form of a dynamic Bayesian Network[7],[8],[9],[10],[11]. In traditional discrete and continuous Markov chains, all the states and state transition values are observable. However, in a hidden markov model, the state is not observable but the output is observable. There is a hidden model behind the visible layer. The model is hidden even if some of the parameters of the model are visible. *HMM*s are widely used especially in speech recognition[21], gesture recognition[23], recognition of hand written texts[22] and bio informatics[24].

In this paper, we propose a new model based on *HMM*s to understand the anomalies in the network traffic and to detect whether there is a compromised service in the system or not. Section 2 presents the related work on the intrusion detection systems. Section 3 proposes our *HMM* for server flow authentication. Section 4 discusses our experimental results and finally on Section 5 we provide our concluding remarks and future work.

2. Related Work

Intrusion Detection is suggested as an approach to prevent unauthorized access to a system[12]. Most of the intrusion detection systems are not such dynamic to overwhelm misuses and anomalies. A system, program or person who tries to get unauthorized access to some system resources or who tries to break down system functionality is called an intruder[13]. Intrusion Detection Systems(*IDS*) try to detect the attempts to break down the system integrity and privacy, anomalies and unauthorized access to system resources[14]. *IDS* provides reports on system activity to help system administrators to understand abnormal situations.

Even though the genetic algorithms are mostly used in optimization problems[15], they also have application areas in intrusion detection. Crosbie and Spafford[16] are the frontiers of genetic programming based approaches in intrusion detection. In their approach, they train different agents for each parameter to observe. Finally, these agents communicate with each other to detect intrusions. This approach has the problem of time consumption due to the improper initialization of agents. Goyal and Kumar[17] proposed a *GA* which can successfully classify *smurf* attack with high detection rates. *HMM*s[18] are used to detect misuses and anomalies[20],[27]. *HMM* provides great success however it is computationally expensive. *HMM* is used in speech, text and image recognition[21],[22],[23] and bio informatics[24]. Complex internet attacks[25],[26] are shown to be detected by non-probabilistic machine learning algorithms. Our approach differs from these applications in a way that it is one of the first approaches analyzing server flows to detect protocol information. In our approach, we try to detect protocol level information from time stamps of packets. There are some previous approaches to detect anomalies from server flows by decision trees[27].

If the server flows are encrypted, payload information of the packets are not sufficient. Packet header and traffic characteristics are processed to create a feature set. Several observations are made by previous works[28]. As an example, they claim that *http* packets contain *psh* flags rarely while *telnet* packets contain more often. Our approach is more different. We suggest to extract protocol information from information like packet sizes and time to transmit a packet.

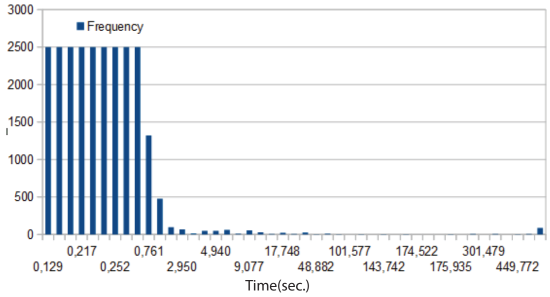

Fig. 1. Histogram for Domain Protocol Frequencies

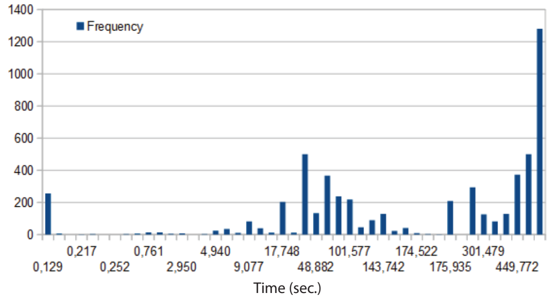

Fig. 2. Histogram for login Protocol Frequencies

3. Hidden Markov Models for Server Flow Authentication

In this section, we explain our proposed *HMM* model and its implementation over training and test data. Wide Area Network Traffic Dataset of the Lawrence Berkeley Laboratory[31] is used in our experiments.

Algorithm 1 Clustering Algorithm Used To Train Hidden Markov Model

```
Read training data
centroidListALL = {}
MeanValues = {}
for each protocol $p_i$ in the list do
    centroidList $\longleftarrow$ k-means_algorithm($p_i$)
    centroidListALL $\longleftarrow$ centroidListALL $\cup$ centroidList
    calculate mean value $m_i$ for value distribution of $p_i$
    MeanValues $\longleftarrow$ $m_i$
end for
Train HMM with MeanValues and centroidListALL.
```

We performed our experiments specifically on two type of features. These are the packet transmission duration and packet size. We first modelled a training data consisting of smtp, telnet, nntp, domain and login records. We used the *k-means* algorithm to cluster data and for each of the n protocols, k centroid values are found as stated in Algorithm 10.

Our System consists of 5 Markov Models for each protocol. Each of the protocols are represented by two states. These are the *small* packet and *large* packet states. Each state has n × k inner states representing a packet to be in each cluster. We categorized the packets as *small* or *large* with respect to their packet sizes. If a packet's size is larger than

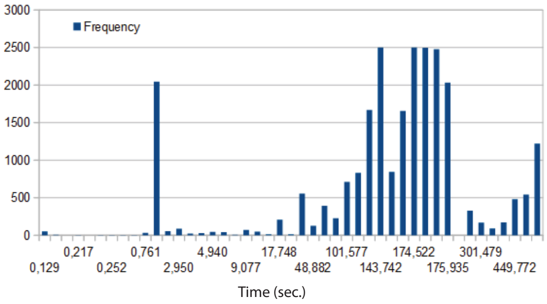

Fig. 3. Histogram for telnet Protocol Frequencies

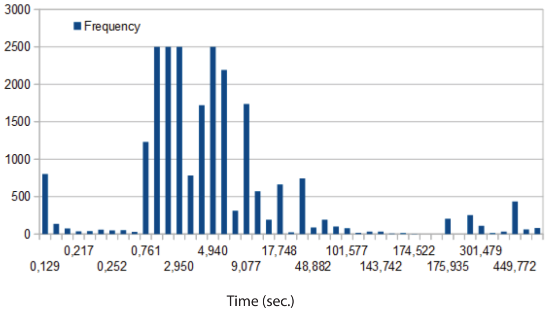

Fig. 4. Histogram for smtp Protocol Frequencies

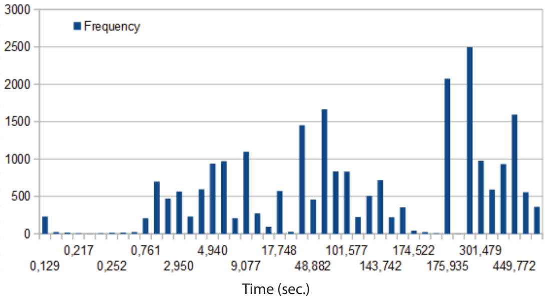

Fig. 5. Histogram for NNTP Protocol Frequencies

the mean of all training packets for a protocol, it is stated as *large*. Otherwise, it is a *small* packet. Figure 6 shows how each *HMM* is constructed.

4. Experiments and Results

In our experiments, we trained 5 Hidden Markov Models for each protocol. We used 25000 *telnet*, *smtp*, *nntp*, *domain* records and 5000 *login* records. We used two parameters: packet size and packet duration. We first analyzed the training data. We used k-means clustering and divided each training set to 9 clusters. For 5 protocols, 9 × 5 = 45 clusters are selected. We separated each group of packets as *small* and *large* packets. The states of the model represent whether a packet is small or large. Finally, 5 *HMM*s with two states and 45 clusters exist. Figure 1, Figure 2, Figure 3,

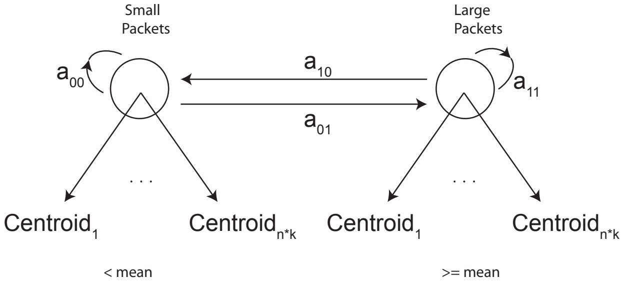

Fig. 6. Hidden Markov Model of Our Approach.

Figure 4 and Figure 5 show histograms for respective protocol data. We trained our *HMM* model with sequences of 100 packets. We provide an iteration number of 100 for Baum Welch Learner in our tests. We trained the *HMM* with test data first. Initial probabilities of the states are assumed to be equal. After having trained the *HMM*s the classification part of our algorithm looks at the packet sizes and the corresponding packet duration probabilities for each interval of respective protocols. If there are more than one protocol satisfying the criteria, the protocol with the highest probability corresponding to the duration interval is selected.

Table 1. Confusion Matrix

	Domain	Login	Smtp	Nntp	Telnet
Domain	24134	0	2	1071	0
Login	121	51771	468	12387	5613
Smtp	3589	677	92706	22567	12344
Nntp	866	8546	1275	60643	16662
Telnet	64	478	125	655	3490

Table 2. Detection Rates

Protocol	Total Records	Detected	Correctly Detected(%)
Domain	25207	24134	95.75
Login	70360	51771	73.6
Smtp	131883	92706	70.3
Nntp	87922	60643	69
Telnet	4812	3490	72.5

We used duration and packet size parameters to detect protocols. These parameters were not successful to detect protocols exactly because some protocols dominate others in specific regions. Moreover, they can be transmitted for a long time interval overlapping with other protocols. Table 1 shows the confusion matrix and Table 2 shows the detection rates. The results are promising and an overall detection rate over %70 is provided.

5. Conclusion

Our approach worked in high accuracy for the detection of protocols. Most of the time, training *HMM*s may be problematic since there is not always enough information. Rare data problems may arise and *HMM*s may be trained unbalanced. We tried to train the models equally except login protocol. User behaviour needs to be modelled with more parameters than duration and packet size. Clustering the data in more than two dimensions may cause memorization of the cases for *HMM*. Our approach focuses on network based intrusion detection. Another alternative may be to detect intrusions computer based. Different threads may be used to detect ip based packet flows. This approach needs support by system administrator because users may change ip numbers if shut down happens in a period of time and restart later. System administrator may assign static ip numbers to each computer to pursue this case. Our approach is only restricted to protocol detection via information level data. Multistage attack patterns can also be detected with *HMM*s and multiple training methods can be developed for paralelizing *HMM* training procedure.

References

1. P . A. Porras and P . G. Neumann. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In Proc. 20th NIST-NCSC National In-formation Systems Security Conference, pages 353 365, 1997. URL http://citeseer.nj.nec.com/porras97emerald.html.
2. M. Roesch and C. Green. Snort - The Open Source Network Intrusion Detection System. URL http:// www.snort.org/.
3. iNetPrivacy Software Inc. Antifirewall. URL http: http://inetprivacy.com/
4. G. H. Kim and G. Spafford. The Design and Implementation of Tripwire: A File System In-tegrity Checker. In ACM Conference on Com-puter and Communications Security, pages 1829, 1994. URL citeseer.nj.nec.com/article/ kim94design.html.
5. N. Murilo and K. Steding-Jessen. chkrootkit: A T ool that Locally Checks for Signs of a Rootkit. URL http://www.chkrootkit.org/.
6. G. H. Kim and G. Spafford. The Design and Implementation of Tripwire: A File System In-tegrity Checker. In ACM Conference on Com-puter and Communications Security, pages 1829, 1994. URL citeseer.nj.nec.com/article/ kim94design.html.
7. Baum, L. E.; Petrie, T. (1966). "Statistical Inference for Probabilistic Functions of Finite State Markov Chains". The Annals of Mathematical Statistics 37 (6): 15541563. doi:10.1214/aoms/1177699147. Retrieved 28 November 2011. 2.
8. Baum, L. E.; Eagon, J. A. (1967). "An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology". Bulletin of the American Mathematical Society 73 (3): 360. doi:10.1090/S0002-9904-1967-11751-8. edit 3.
9. Baum, L. E.; Sell, G. R. (1968). "Growth transformations for functions on manifolds". Pacific Journal of Mathematics 27 (2): 211227. Retrieved 28 November 2011. 4.
10. Baum, L. E.; Petrie, T.; Soules, G.; Weiss, N. (1970). "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains". The Annals of Mathematical Statistics 41: 164. doi:10.1214/aoms/1177697196. edit 5.
11. Baum, L.E. (1972). "An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process". Inequalities 3: 18.
12. M. Botha, R. Solms, "Utilizing Neural Networks For Effective Intrusion Detection", ISSA, 2004.
13. R. Graham, "FAQ: Network Intrusion Detection Systems". March 21, 2000.
14. D. Zamboni, "Using Internal Sensors For Computer Intrusion Detection". Center for Education and Research in Information Assurance and Security,Purdue University. August 2001.
15. U. Tosun, A. Cosar, T. Dokeroglu, "A Robust Island Parallel Genetic Algorithm for the Quadratic Assignment Problem", International Journal of Production Research, vol 51, pg 4117, 2013.
16. M. Crosbie, E. Spafford, "Applying Genetic Programming to Intrusion Detection", Proceedings of the AAAI Fall Symposium, 1995.
17. Anup Goyal, Chetan Kumar, "GA-NIDS: A Genetic Algorithm based Network Intrusion Detection System", 2008.
18. Rabiner, L.R., Juang, B.H.: An Introduction to Hidden Markov Models. In: IEEE ASSP Magazine, 1986
19. Rabiner L. R. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition IEEE 1989
20. Warrender, C., Forrest S., and Pearlmutter, B.: Detecting Intrusions Using System Calls: Alternative Data Models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy. (1999)
21. Huang, X.D., Ariki, Y., Jack, M.A.: Hidden Markov Models for Speech Recognition, Edinburgh University Press (1990)
22. Wen, Y.: Text Mining Using HMM and PPM. Masters thesis. Department of Computer Science, University of Waikato (2001)
23. Bunke, H., Caelli, T. (eds.): Hidden Markov Models: Applications in Computer Vision. World Scientific, Series in Machine Perception and Artificial Intelligence, Vol. 45. (2001)
24. Boys, R.J., Henderson, D.A., Wilkinson, D.J.: Detecting Homogeneous Segments in DNA Sequences by Using Hidden Markov Models. Applied Statistics 49(2) (2000) 269
25. Green, J., Marchette, D., Northcutt, S., and Ralph, B.: Analysis Techniques for Detecting Coordinated Attacks and Probes. In: Proceedings of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring (1999)
26. Cuppens, F.: Managing Alerts in a Multi-intrusion Detection Environment. In: 17th Annual Computer Security Applications Conference. New Orleans, Louisiana (2001)
27. Ourston Dirk, Matzner Sara, Stump William Applications of Hidden Markov Models to Detecting Multi-Stage Network Attacks Proceedings of the 36th Hawaii International Conference on System Sciences
28. Early P. James, Brodley E. Carla, Rosenberg Catherine Behavioral Authentication of Server Flows Proceedings of the 19th International Conference onSystem Sciences
29. Duda O. Richard, Hart E. Peter, Stork G. David Pattern Classification Wiley, ISBN:0-471-05669-3
30. Jahmm Library http://www.run.montefiore.ulg.ac.be/ francois/software/jahmm/
31. http://ita.ee.lbl.gov/html/contrib/LBL-CONN-7.html