Evaluations for an Immunity-Based Anomaly Detection with Dynamic Updating of Profiles

Takeshi Okamoto[1] and Yoshiteru Ishida[2]

[1] *Dept. of Information Network and Communication, Kanagawa Institute of Technology, 1030 Shimo-ogino, Atsugi, Kanagawa 243-0292, Japan*
[2] *Dept. of Knowledge-Based Information Engineering, Toyohashi University of Technology, Tempaku, Toyohashi , Aichi 441-8580, Japan*

Abstract: This paper presents evaluations of an immunity-based anomaly detection method with dynamic updating of profiles. Our experiments showed that the updating of both self and nonself profiles markedly decreased both the false alarm and missed alarm rates in masquerader detection. In computer worm detection, all the *random-scanning worms* and simulated *metaserver worm* examined were detected. The detection accuracy of the simulated *passive worm* was markedly improved.

Keywords: Immunity-based system, anomaly detection, computer worm, ROC, adaptation.

I. INTRODUCTION

Many anomaly detection methods [1] are restricted to the reference of a single user profile. One drawback of these methods is that many false alarms arise when valid users carry out new operations that they have never performed previously. To improve their detection accuracy, we have proposed a new immunity-based anomaly detection method with multiple agents based on the specificity and diversity of the immune system [2]. Our approach makes use of multiple profiles rather than a single profile, which leads to an improvement in detection accuracy.

In addition, we incorporated a new framework of dynamically updating of profiles with test sequences (i.e., not training sequences) into our immunity-based anomaly detection method. The updating of both self and nonself profiles markedly decreased both the false alarm and missed alarm rates in internal masquerader detection [3]. However, the detection accuracy of external masqueraders and computer worms was not evaluated.

In this paper, we made a slight change in profile construction to escape assignment overflow. In experiments, we evaluated the extent to which profile updating improved the detection accuracy of external masqueraders and computer worms.

II. RELATED WORKS

Artificial immune systems for computer security can be divided roughly into three types [4]: hybrid approaches combined with multiple conventional detection methods [5], approaches inspired by the mechanism of negative selection in the thymus [6], and approaches motivated by the danger theory [7].

Our system is related to those using the second approach. The difference in intrusion detection between our method and those reported previously is the reference information used for detection. Previous systems referred only to nonself information, while our method refers to both self and nonself information. This reference to self information contributes to a reduction in false alarms.

III. IMMUNITY-BASED ANOMALY DETECTION

1. Definitions of "self" and "nonself"

The heart of the immune system is the ability to distinguish between "self" (i.e., the body's own molecules, cells, and tissues) and "nonself" (i.e., foreign substances, such as viruses and bacteria). Similarly, operation sequences executed by a user on his/her own account are defined as "self," and all other sequences are defined as "nonself." For example, if one user executes commands on his/her own account, the command sequence is "self." If another user executes commands on someone else's account, the command sequence is "nonself." Such a user is defined as a masquerader or an intruder, regardless of whether the user's actions are malicious.

In an immunity-based anomaly detection system, operation sequences for each user in the training data belong absolutely to "self." The operation sequences are used to construct a profile for each user. The profile yields the probability that the operation sequence belongs to "self." Based on this probability, the system classifies the operation sequence as either belonging or not belonging to "self."

2. Generation of agents

An immune cell has a unique receptor with high affinity for specific antigens. Similarly, our immunity-based system generates a user-specific agent for every user, i.e., every account. An agent has a unique profile, representing the probability that the operation sequence is performed by the original user. The probability is expressed by a score, which is derived from the detection method, i.e., HMM, IPAM, the Bayes 1-step Markov method, etc. We chose the HMM method, because previous studies indicated that it performs well [2]. The parameters of the HMM are given by $\lambda = [\pi, A, B]$ and V, where π is the initial state distribution, A is the state-transition probability distribution, B is the operation probability distribution, and V is the operation table that assigns a unique number to each operation in training data composed of operation sequences obtained previously from each original user. The size of the operation table V is limited to M_{max} to avoid assignment overflow, where M_{max} is specified by a system administrator. The parameters π, A, B are estimated from the training data, as described [2]. The parameter V is determined by operations in training data.

The agent can compute the likelihood $P(O|\lambda)$ of the sequence O with the profile λ. The likelihood $P(O|\lambda)$ represents the probability that the sequence O was performed by the original user corresponding to the agent, i.e., the profile λ. The agent would compute a high likelihood, i.e., a high score, only for the sequences of the original user corresponding to the agent.

3. Adaptive discrimination of self and nonself

Our immunity-based system has a user-specific agent for every account. Each agent monitors operations on its own account until the length of the operation sequence reaches L, where L is specified by a system administrator. The agent of the account on which the length of the sequence reaches L is activated.

The activated agent shares the sequence with all the other agents. All agents compute their own score of the sequence. The activated agent computes the effective threshold, $Min + (Max - Min) \times Th$, where Min is the minimum score of all scores, Max is the maximum score of all scores, and Th is the percentage difference between Min and Max. Th is specified by a system administrator. The activated agent compares its own score, X, with the effective threshold, Y. If $X \geq Y$, the activated agent classifies the sequence as normal, i.e., self. Otherwise, the agent classifies the sequence as abnormal, i.e., nonself. Exceptionally, provided that X is equal to the computational minimum value of $P(O|\lambda)$, the sequence is regarded as abnormal. Conversely, the sequence is regarded as normal if X is equal to the computational maximum value of $P(O|\lambda)$.

If the activated agent classifies the sequence as self, it updates its own profile (i.e., a self profile). If not, the agent that computed the maximum score of all agents updates its own profile (i.e., a nonself profile) and the activated agent raises an alarm to a system administrator. These profiles are newly estimated from the sequence just examined and all the sequences trained previously. Note that the size of the operation table V is limited to M_{max} specified by a system administrator. Once the size of V reaches M_{max}, the least frequently used operation is replaced with a new one.

Finally, the activated agent returns to a normal state and continues to monitor operations on its own account.

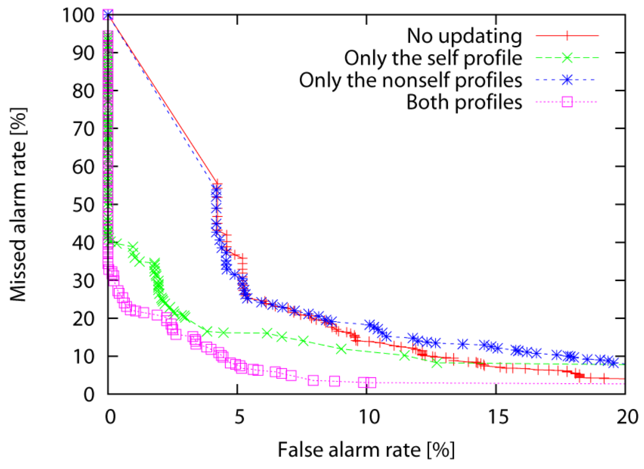
IV. EXPERIMENTS AND DISCUSSIONS

1. Masquerader detection

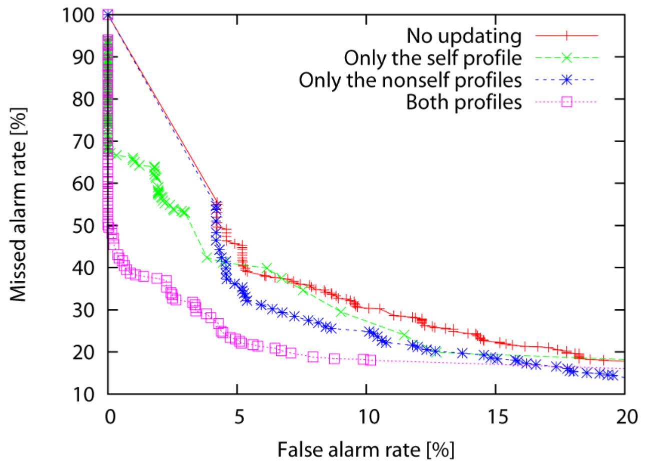
As experimental data, we used network traffic captured from 12 users for about one month. These data are identical to those used in our previous study [8]. This experiment uses web traffic extracted from the data, as web traffic accounts for the majority of network traffic. The web traffic of each user contained more than 3,000 requests. The first 500 requests for each user are training data to allow construction of a profile. The next 1,000 requests are test data to evaluate the detection accuracy. The test for the sequence is performed every 100 requests. All the profiles that are to be updated are updated synchronously by incrementing the sequence number. Anomalous behavior is simulated by testing one user's request sequence against another user's profile.

The number of hidden states of the HMM is set to 1 due to the lowest computational cost and the best accuracy of other states [8]. The HMM parameter is equal to $\lambda = [1, 1, B]$, where B is equal to a relative frequency distribution of operations in training data.

The metrics of detection accuracy are based on the false alarm rate, i.e., false positive rate, and missed alarm rate, i.e., false negative rate. In general, there is a trade-off between the false alarm rate and the missed alarm rate. The relationship between these rates can be described visually by a receiver operating characteristic (ROC) curve, which is a parametric curve generated by



(a) Internal masquerader detection



(b) External masquerader detection

Fig. 1. ROC curves of internal and external masquerader detection for the method without updating any profiles, with updating only the self profiles, with updating only the nonself profiles, and with updating both profiles.

varying the threshold Th from 0% to 100%, and computing these rates at each threshold Th. In this experiment, each ROC curve consists of 101 points for simplicity. Each point corresponds to one threshold from $Th = 0$ to $Th = 1.0$. In addition, the area under the ROC curve (AUC) is computed as a scalar measure for ROC analysis. The AUC enables quantitative comparison of multiple ROC curves. In this experiment, the AUC may be much larger than the exact AUC due to thinning out plots.

We evaluated the detection accuracy of the profile updating. Figure 1 shows ROC curves of internal and external masquerader detection for the method without updating any profiles, with updating only the self profiles, with updating only the nonself profiles, and with updating both profiles. In each ROC curve, 6 internal users were chosen randomly from among the total of 12 users, with all the others being the external users. Each point on the ROC curve is an average over 100 combinations. The statistics of the AUCs for 100 combinations of internal and external users were examined and the statistical significance of differences was analyzed by ANOVA with Dunnett's test for multiple comparisons.

In Figure 1(a), the method with updating both profiles indicated the best detection accuracy of all curves. The mean AUC of the method with updating both profiles was 0.026, which was significantly lower ($P < 0.001$) than that of the method without updating any profiles (0.072). This statistical significance seemed to be dependent mainly on the updating of the self profiles, as the updating of only the self profiles decreased the false alarm rate, whereas the updating of only the nonself profiles slightly increased the missed alarm rate. It should be noted that the updating of both profiles achieved a missed alarm rate of 34.37% without false alarms at the threshold 43.43%.

Similar to Figure 1(a), Figure 1(b) indicated that the method with updating both profiles outperformed all the other methods. The mean AUC of the method with updating both profiles was 0.108, significantly lower ($P < 0.001$) than that of the method without updating any profiles (0.150). This statistical significance is due to the updating of both the self and nonself profiles. The updating of the self profiles decreased the number of false alarms, and the updating of the nonself profiles decreased the number of missed alarms. It should be noted that the updating of both profiles achieved a missed alarm rate of 50.13% without false alarms at the threshold 43.43%.

2. Worm detection

Computer worms are divided into five types of target discovery: *random-scanning*, *hit-list*, *metaserver*, *topological*, and *passive worms* [9].

A. Random-scanning worms

We evaluated four random-scanning worms in the wild: `CodeRedv2`, `CodeRedII`, `Slammer`, and `Blaster`. These worms attempt to infect randomly selected computers. As with our previous study [8], there were no missed alarms without false alarms on any of the accounts for all the worms examined in all methods.

B. Hit-list worms

A *hit-list worm* attempts to infect computers of target lists pre-generated by an attacker or its author.

The hit-list includes IP addresses of vulnerable servers or always-connected IP addresses. All the methods seem to detect these worms if the hit-list does not include IP addresses of the operation table. Otherwise, for example, an attacker can randomly insert many IP addresses of popular websites into the hit-list at the expense of high-speed spreading. In that case, our method could not detect these worms because IP addresses of popular websites are likely to coincide with those of the operation table.

C. Metaserver worms

A *metaserver worm* obtains a target list from a metaserver that keeps a list of active servers and attempts to infect computers on these lists. The `Santy` worm is a *metaserver worm*, which attempts to propagate to IP addresses in the search results provided by Google^{TM} (`www.google.com`).

All the methods detected the simulated *metaserver worm* [8] because these worms have difficulty guessing IP addresses on the operation table.

D. Topological worms

A *topological worm* obtains a target list from the devices of the infected computer. For example, the worm obtains targets from peer-to-peer software in an infected computer and attempts to infect all peers. This worm may escape all methods, because the traffic pattern of this worm may appear normal and the peers may be included in the operation table. An alternative method would be needed to prevent *topological worms* from spreading.

E. Passive worms

The *passive worm*, which either waits for target computers to visit or follows user's requests into target computers, is more difficult for an anomaly detection system to detect, because its behavior is similar to that of the user.

The evaluation results of a simulated *passive worm* [8] were almost the same as those in Fig. 1. The mean AUC of the method with updating both profiles was 0.029, which was significantly lower ($P < 0.001$) than that of the method without updating any profiles (0.121). The mean difference between the method with updating both profiles and without updating any profiles was larger than that of the internal masquerader detection. Briefly, profile updating markedly improved the detection accuracy of the simulated *passive worm*.

V. CONCLUSIONS

We have made a change in our immunity-based anomaly detection method to escape assignment overflow in the operation table, and we evaluated and discussed the extent to which profile updating improves the detection accuracy of external masqueraders and computer worms.

Experimentally, we showed that the updating of both profiles markedly decreased both the false alarm rate and the missed alarm rate in masquerader detection. In worm detection, all the *random-scanning worms* and the simulated *metaserver worm* examined were detected. The detection accuracy of the simulated *passive worm* was markedly improved. Detection of *topological worms* may require an alternative method to investigate inbound traffic in order to detect exploit codes and/or shellcodes.

ACKNOWLEDGEMENTS

This work was supported by Grant-in-Aid for Scientific Research (B) 19700072, 2009.

REFERENCES

[1] Schonlau M, DuMouchel W, Ju W, et al. (2001), Computer intrusion: Detecting masquerades, Statistical Science 16(1):58-74.
[2] Okamoto T, Ishida Y (2009), An Immunity-Based Anomaly Detection System with Sensor Agents, Sensors 9(11):9175-95.
[3] Okamoto T, Ishida Y (2008), Dynamic Updating of Profiles for an Immunity-Based Anomaly Detection System. LNAI 5179, pp. 456-64.
[4] Kim J, Bentley P, Aickelin U, et al. (2007), Immune system approaches to intrusion detection - a review, Natural computing 6(4):413-66.
[5] Kephart J (1994), A biologically inspired immune system for computers. Artificial Life IV, pp. 130-39.
[6] Forrest S, Hofmeyr S, Somayaji A, et al. (1996), A sense of self for unix processes. Proc. of the 1996 IEEE Symposium on Security and Privacy, pp. 120-28.
[7] Aickelin U, Cayzer S (2002), The danger theory and its application to artificial immune systems. Proc. of the 1st Int. Conf. on Artificial Immune Systems, pp. 141-48.
[8] Okamoto T, Ishida Y (2006), Towards an immunity-based anomaly detection system for network traffic. LNAI 4252, pp. 123-30.
[9] Weaver N, Paxson V, Staniford S, et al. (2003), A taxonomy of computer worms. Proc. of the 2003 ACM workshop on Rapid malcode, pp. 11-18.