Chapter 15
Stochastic Tools for Network Intrusion Detection

Lu Yu and Richard R. Brooks

Abstract With the rapid development of Internet and the sharp increase of network crime, network security has become very important and received a lot of attention. We model security issues as stochastic systems. This allows us to find weaknesses in existing security systems and propose new solutions. Exploring the vulnerabilities of existing security tools can prevent cyber-attacks from taking advantages of the system weaknesses. We propose a hybrid network security scheme including intrusion detection systems (IDSs) and honeypots scattered throughout the network. This combines the advantages of two security technologies. A honeypot is an activity-based network security system, which could be the logical supplement of the passive detection policies used by IDSs. This integration forces us to balance security performance versus cost by scheduling device activities for the proposed system. By formulating the scheduling problem as a decentralized partially observable Markov decision process (DEC-POMDP), decisions are made in a distributed manner at each device without requiring centralized control. The partially observable Markov decision process (POMDP) is a useful choice for controlling stochastic systems. As a combination of two Markov models, POMDPs combine the strength of hidden Markov Model (HMM) (capturing dynamics that depend on unobserved states) and that of Markov decision process (MDP) (taking the decision aspect into account). Decision-making under uncertainty is used in many parts of business and science. We use here for security tools. We adopt a high-quality approximation solution for finite-space POMDPs with the average cost criterion and their extension to DEC-POMDPs. We show how this tool could be used to design a network security framework.

Keywords POMDP · DEC-POMDP · IDS

L. Yu (✉) · R. R. Brooks
Clemson University, Clemson, SC, USA
e-mail: lyu@g.clemson.edu; rrb@g.clemson.edu

N. S. V. Rao et al. (eds.), *Proceedings of International Symposium on Sensor Networks, Systems and Security*, https://doi.org/10.1007/978-3-319-75683-7_15

15.1 Intrusion Detection System

Intrusion detection systems continuously monitor the computer system or network and generate alarms to inform the system administrator of suspicious events. IDSs are now considered a necessary addition to the security infrastructure of an organization [9]. The objective of intrusion detection is to detect malicious activities and accurately differentiating them from benign activities. According to the Common Intrusion Detection Framework (CIDF) [11], a general IDS architecture has four modules:

- Event-box (E-box) sensors monitor and collect information about the target system.
- Database-box (D-box) stores information from the E-box.
- Analysis-box (A-box) analyzes data stored in D-box and generates alarms if necessary.
- Response-box (R-box) implements countermeasures to thwart malicious intrusions.

The primary classes of detection methodologies include *signature-based detection*, *anomaly-based detection*, and *stateful protocol analysis* [9]. IDSs that employ signature-based detection identify attacks by comparing existing signatures of known attacks with the stored network traffic. When a match is found, IDSs will trigger the corresponding countermeasure to counteract the detected intrusion. Signature-based detection provides accurate detection results for well-specified attacks and effective known countermeasures can be taken. The major drawback of signature-based detection is its inability to detect new, unknown attacks. With new attacks appearing continuously, signature-based detection techniques suffer from high false negative (FN) rates. The anomaly-based detection has the potential to detect new types of attacks by estimating the deviation of observed information from the predefined baseline of "normal." However, there still exist several significant issues regarding anomaly-based detection, including high FP rates, low throughput but high cost, absence of appropriate metrics, etc. [5]. Stateful protocol analysis may provide more accurate detection results than the anomaly-based detection but is much more resource intensive due to the complex analysis and overhead generated from state tracking [9]. Typically, the more an IDS's detection accuracy can be improved from the default configuration, the less efficient it is. As a result, continuous monitoring may cause excessive computation bandwidth, which is undesirable for any computer system and network.

In addition, various intrusion prevention technologies have been implemented in IDSs, such as logging off the unauthorized user, shutting down the system, or reconfiguring the network if possible [16], etc. Despite strengthening the security of the information and communication systems, the intrusion prevention capabilities entail high cost in terms of energy and host resources.

It is not hard to see from the above introduction that, as a popular and effective tool against cyber-attacks by guarding the system's critical information, the resource cost of IDSs must be taken into account. IDS scheduling is needed to balance between security performance and resource consumption.

There are three primary types of IDS, namely, network-based IDS (NIDS), host-based IDS (HIDS), and stack-based IDS (SIDS). We choose HIDS since HIDS can be selectively deployed on critical machines, such as management servers, data servers and administrator consoles, etc.

15.2 Honeypot

Honeypots are needed to supplement IDSs in the proposed security scheme because they complement most other security technologies by taking a proactive stance. A honeypot is a closely monitored computing resource used as a trap to ensnare attackers. As defined by Spitzner in [10], "A honeypot is a security resource whose value lies in being probed, attacked, or compromised." The principal objectives of honeypots are to divert attackers away from the critical resources and study attacker exploits to create signatures for intrusion detection. The attraction of honeypots to attackers mitigates the threat of malicious attacks and thus helps secure the valuable information and important services located on the real targets.

Based on the level of interaction between the honeypots and the attackers, honeypots can generally be divided into the *high-interaction honeypots* and the *low-interaction honeypots*. Typical examples are honeyd [8] and honeynet [10]. Honeyd allows users to set up multiple virtual honeypots with different characteristics and services on a single machine. Honeynet monitors a larger and more diverse network when one honeypot may not be sufficient. It is very complex and expensive to deploy and maintain a high-interaction honeypot because it emulates almost all the activities found in a normal operating system. Deployment and configuration of a low-interaction honeypot is much easier and cheaper since it only simulates some system services. "BitSaucer" [1] proposed by Adachi et al. is a hybrid honeypot composed of both low-interaction and high-interaction capabilities.

Honeypots can also be divided into: *research* and *production honeypots* [7]. The primary function of a research honeypot is to extract the signatures of emerging attacks, which can be used to improve the detection accuracy of IDSs. A thorough understanding of the observed traffic data is time-intensive and requires analysts with comprehensive expertise in almost all network-related fields. Moreover, the deployment of research honeypots provides little benefit in strengthening the system security. Production honeypots are placed within the production network to mitigate risk. Most production honeypots are low-interaction honeypots and capture limited information. Example of production honeypot is Nepenthes [2]. Since our scheme is meant to improve security, we use production honeypots in combination with IDSs.

A honeypot mostly does not deal with false positives like IDS since all the services simulated by a honeypot have no production value. All the traffic that enters and leaves a honeypot is suspicious and should be monitored and analyzed [4]. However, not all attempts to access a honeypot are malicious. For example, a person might mistype the address of a computer and accidentally connect to a deployed honeypot. As a result, uncertainty is also involved in honeypots scheduling.

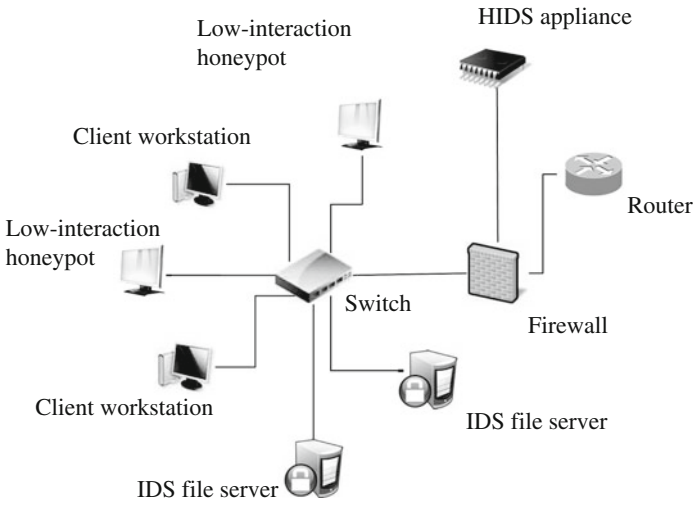

Fig. 15.1 Example distributed hybrid security scheme combining HIDSs with honeypots

15.3 System Model

Both HIDSs and honeypots detect intrusions and can operate at all times. However, intrusion detection and system emulation consume a large amount of energy and other resources, including memory, processor usage, and disk storage. We need to balance security performance and the resource cost by scheduling IDS and honeypot activities. The scheduling problem is modeled as a discrete-time process. In the proposed scheme, more than one HIDS or honeypot can be active during each time period.

We now consider an example of the hybrid system. Assume a local area network (LAN) is equipped with $K - H$ HIDSs and H honeypots. This brings the total number of the security devices to K. Without loss of generality, we also assume the network topology is static and there are more machines than available HIDSs. Inevitably, only some machines have HIDSs installed. An example network is shown in Fig. 15.1.

Suppose each HIDS can operate in three modes: *monitor*, *prevention*, and *sleep*, which are the action spaces for a HIDS. The HIDS is set to *monitor* for intrusion detection and can sleep for energy saving. A preventive action might be taken by switching to *prevention* mode if an unauthorized or malicious activity is identified and consumes more resources. Similarly, the action space of a deployed honeypot can be specified as *monitor*, *analysis*, and *sleep*. Further analysis will be carried out if traffic anomalies are detected. Each HIDS makes decision based on local information, which can be modeled as a partially observable Markov decision process (POMDP), which combines the strength of hidden Markov model (HMM) [6, 15] (capturing dynamics that depend on unobserved states) and that of Markov decision process (MDP) (taking the decision aspect into account).

Let $S^{(k)}(t)$ denote the state of an arbitrary device k (HIDS or honeypot) at time t, we assume $S^{(k)}(t) = \langle X^{(i)}(t), Y^{(k)}(t)\rangle$, where $X^{(k)}(t)$ represents the security condition and $Y^{(k)}(t)$ represents the resource consumption level. For instance, the security state can be simply divided into: "$secure$" and "$compromised$." If we use the notation $\mathscr{X}$ to denote the state space of $X^{(k)}(t)$, then $\mathscr{X} = \{secure, compromised\}$. The state space of $Y^{(k)}(t)$, denoted by $\mathscr{Y}$, includes three consumption levels: $\{low, medium, high\}$. The correspondence between different consumption levels and operating states are:

- *Low*: The device is not chosen, i.e., in the sleep mode.
- *Medium*: The device is working in the monitor mode.
- *High*: The HIDS/honeypot is working in prevention/analysis mode.

Since the IDSs used are HIDSs, each HIDS only monitors the machine it resides on, ignoring the rest of the network. As a decentralized control scheme, the decision to activate a certain security device is based on local observations. To complete the problem, we assume the observation space is identical to the space of security conditions, i.e., $\mathscr{O} = \mathscr{X} = \{$"$secure$", "$compromised$"$\}$. Note that an intrusion alarm does not necessarily mean there is an attack, and vice versa. Intrusion detection can make two types of errors: false positive (FP) and false negative (FN). A large volume of FPs result in lots of time wasted on determining whether an alert is an attack when it is actually benign [3]. FNs result in security holes due to failing to raise alarms when intrusions occur. Since the goal of intrusion detection is to precisely differentiate the intrusions from legitimate behaviors, both errors are significant performance indexes of IDSs and have been embodied in the observation probabilities. For instance, the following observation probability

$$\begin{aligned} Pr\{O^{(k)}(t+1) = \text{"}compromised\text{"} \mid A^{(k)}(t) \\ = \text{"}monitor\text{"}, S^{(k)}(t+1) = \langle secure, medium\rangle\} \end{aligned} \tag{15.1}$$

is equal to the FP rate of device k.

The local state of each host in the network is closely related to the security posture of the entire network. The analysis is divided into two cases:

- Switching between different modes of an HIDS will change the power consumption level of the local machine but have no influence on the attack activities in the LAN.
- The activation of a honeypot will positively impact the network's operation and security by distracting adversaries away from the valuable resources in the LAN and accordingly will mitigate the threat posed to the rest of the network.

It is obvious from the preceding analysis that the choice of each agent may affect the state of the entire network. This makes the decentralized partially observable Markov decision process (DEC-POMDP) a more suitable tool to model the scheduling for the distributed system. Some might argue that a centralized controller can also be adopted in this case. However, a common drawback exists in most centralized controls: it may consume a prohibitive amount of bandwidth and instantaneous

communication between the agents and the controller. In addition, possible security breaches are brought in since the transmitted information may be intercepted by the adversaries. Therefore, the DEC-POMDP is generally more preferable.

15.4 DEC-POMDP Formulation for the Distributed Hybrid Security System Combining IDS and Honeypot

We define the state of the DEC-POMDP formulation at time t, denoted by $S(t)$, as the combination of $S^{(k)}(t)$, $i = 1, 2, \cdots, K$. Notationally,

$$S(t) = [S^{(1)}(t), S^{(2)}(t), \cdots, S^{(K)}(t)]$$

Similarly, we can write the action of time t as

$$A(t) = [A^{(1)}(t), A^{(2)}(t), \cdots, A^{(K)}(t)]$$

We now consider the state transition law. The state $S(t)$ evolves based on a $(|\mathscr{X}| \times |\mathscr{Y}|)$-state Markov decision process. Let W denote the state transition probability function, then

$$W(\overline{s}, \overline{a}, \overline{s}') = Pr\{S(t+1) = \overline{s}' | S(t) = \overline{s}, A(t) = \overline{a}\}$$

where $\overline{s}, \overline{s}' \in \mathscr{S}^K$ and $\overline{a} \in \mathscr{A}^K$.

The observation probabilities of the DEC-POMDP formulation are slightly different from the standard POMDP. As indicated in (15.1), the quantities of the observations probabilities are assigned according to the FP and FN rates of the devices. Consequently, the observation of each agent only depends on the local information. It follows that $V^{(k)}(a, s', o')$, the observation probability of device i, is given by

$$V^{(k)}(a, s', o') = Pr\{O^{(k)}(t+1) = o' \mid A^{(k)}(t) = a, S^{(k)}(t+1) = s'\}$$

Finally, the immediate reward at time t is defined as the sum of each local immediate reward

$$r(S(t), A(t)) = \sum_{k=1}^{K} r(S^{(k)}(t), A^{(k)}(t)) \tag{15.2}$$

The values of the immediate rewards are assigned according to the following rules: A successful detection of an attack results in a large reward; on the contrary, an unnecessary further analysis staged due to a misjudgment will cause a large penalty, so will the misdetection of an attack; furthermore, monitoring is not free and monitoring a secure machine comes at a small penalty.

15.5 NLP-Based Solution of the DEC-POMDP

The scheduling model for the hybrid system is a DEC-POMDP. Thus, we need to augment the POMDP solution method in [13, 14] to situations of multiple controllers. As was mentioned in [12], the solution of a DEC-POMDP consists of a set of policy graphs, one for each agent. Accordingly, the goal is to optimize a set of finite state machines (FSM). We will show in the followings that the extension of the NLP-based solution in [13, 14] to DEC-POMDP is very straightforward. In order to present the algorithm for DED-POMDPs, we make the following assumptions:

- There are K agents in the DEC-POMDP.
- The state space of the DEC-POMDP is denoted by $\mathscr{S}$. Each agent has the same action space $\mathscr{A}$ ("*prevention*" of an IDS corresponds to "*analysis*" of a honeypot) and observation space $\mathscr{O}$.
- Each agent chooses the actions according to a fixed-size FSC. The set of the nodes in the FSC of agent k is denoted by $\mathscr{N}^{(k)}$.
- We use the notation $\overline{n}$ to denote a vector of length K, where $\overline{n}(k) \in \mathscr{N}^{(k)}$. The observation vector $\overline{o}$ and the action vector $\overline{a}$ are defined likewise.
- $x_k(n, a)$ and $y_k(n, o', n')$ are the control variables of the FSC of agent k.

The formal representation of the NLP-based solution of DEC-POMDPs satisfying the above assumptions is:

For variables : $\pi_{\overline{n}s\overline{a}}$ and $g^{(k)}(n_k, {o_k}', {n_k}', {a_k}')$,

$$\text{where} g^{(k)}(n_k, {o_k}', {n_k}', {a_k}') = x_k(n', a')y_k(n, o', n')$$

$$\text{maximize} \sum_{\overline{n}} \sum_{s \in \mathscr{S}} \sum_{\overline{a} \in \mathscr{A}} \pi_{\overline{n}s\overline{a}} \cdot r(s, \overline{a}^K)$$

Subject to :

$$\text{For } \forall s' \in \mathscr{S}, \forall \overline{n} \in \bigwedge, \forall \overline{a}' \in \mathscr{A}^K,$$

$$\pi_{\overline{n}'s'\overline{a}'} = \sum_{\overline{o} \in \mathscr{O}^K} \sum_{s \in \mathscr{S}} \sum_{\overline{a} \in \mathscr{A}^K} \{\pi_{\overline{n}s\overline{a}} \sum_{\overline{o}' \in \mathscr{O}^K} P(s, \overline{a}, s') Q(\overline{a}, s', \overline{o}')$$

$$\prod_k g^{(i)}(n_k, {o_k}', {n_k}', {a_k}')\},$$

$$\forall n_k \in \mathscr{N}^{(k)}, \forall {o_k}' \in \mathscr{O}, \sum_{n_k'} \sum_{{a_k}' \in \mathscr{A}} g^{(k)}(n_k, {o_k}', {n_k}', {a_k}') = 1,$$

$$\text{for } k = 1, 2, \cdots, K$$

The optimal solution to the NLP in (15.3) provides an optimal set of FSCs of the given size. The solution representation owes its availability to one critical factor: each agent behaves independently. That is, all the policy graphs are independent from each other.

15.6 Summary

This chapter starts with the introduction of the two security technologies adopted in the proposed security scheme. We choose HIDS, in combination with honeypot, with the purpose to integrate the advantages of both tools in our system. We formulate the decentralized control of the system as a DEC-POMDP. In the end of the chapter, we show how to extend the FSC-based POMDP algorithm described in [13, 14] to solving DEC-POMDP.

References

1. Adachi, Y., Oyama, Y.: Malware analysis system using process-level virtualization. In: Proceedings of IEEE Symposium on Computers and Communications, pp. 550–556 (2009)
2. Baecher, P., Koetter, M., Dornseif, M., Freiling, F.: The nepenthes platform: An efficient approach to collect malware. In: Proceedings of the 9th International Symposium on Recent Advances in Intrusion Detection, pp. 165–184. Springer, Berlin (2006)
3. Bakar, N., Belaton, B., Samsudin, A.: False positives reduction via intrusion alert quality framework. In: Joint IEEE Malaysia International Conference on Communications and IEEE International Conference on Networks, pp. 547–552 (2005)
4. Baumann, R.: http://security.rbaumann.net/download/honeyd.pdf. Originally published as part of the GCIA practical
5. Garcia-Teodoroa, P., Diaz-Verdejoa, J., Macia-Fernandeza, G., Vazquezb, E.: Anomaly-based network intrusion detection: Techniques, systems and challenges. Comput. Secur. **28**(1–2), 18–28 (2009)
6. Lu, C., Schwier, J.M., Craven, R.M., Yu, L., Brooks, R.R., Griffin, C.: A normalized statistical metric space for hidden Markov models. IEEE Trans. Cybern. **43**(3), 806–819 (2013)
7. Mokube, I., Adams, M.: Honeypots: Concepts, approaches, and challenges. In: ACMSE 2007, Winston-Salem, NC, pp. 321–325 (2007)
8. Provos, N.: In: Proceedings of the 12th USENIX Security Symposium, pp. 1–14 (2004)
9. Scarfone, K., Mell, P.: Guide to Intrusion Detection and Prevention Systems (IDPS). Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD (2007). NIST special publication 800-94
10. Spitzner, L.: Honeypots: Tracking Hackers. 1st edn. Addison-Wesley, Boston, MA (2002)
11. Tung, B.: The common intrusion detection framework (1999). http://gost.isi.edu/cidf/
12. Yu, L.: Stochastic tools for network security: Anonymity protocol analysis and network intrusion detection. Ph.D. thesis, Clemson University (2012). http://tigerprints.clemson.edu/all_dissertations/1061/
13. Yu, L., Brooks, R.: Observable subspace solution for irreducible pomdps with infinite horizon. In: Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research, p. 83. ACM (2011)

14. Yu, L., Brooks, R.R.: Applying pomdp to moving target optimization. In: Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop, p. 49. ACM (2013)
15. Yu, L., Schwier, J.M., Craven, R.M., Brooks, R.R., Griffin, C.: Inferring statistically significant hidden Markov models. IEEE Trans. Knowl. Data Eng. **25**(7), 1548–1558 (2013)
16. Zheng, J., Jamalipour, A.: Wireless Sensor Networks: A Networking Perspective. John Wiley & Sons, Hoboken (2009)