A Novel Game Machine Learning Method for Calculating Optimal Response for Edge Server

Rui Zhang, Hui Xia(✉), Ju-fu Cui, Yi-zhe Li, Shu-shu Shao, and Hang Ren

College of Computer Science and Technology, Qingdao University, Qingdao 266100, China
xiahui@qdu.edu.cn

Abstract. Mobile edge computing extends traditional cloud services to the edge of the network and enables edge server to handle network requests with low latency requirements. However, the edge server is closer to the terminal device with relatively limited storage capacity and computing capacity, and is more vulnerable to the invasion of attackers. To solve this problem, we proposed a game machine learning method to determine the optimal response of edge server to attackers, so as to defend against attackers. First, we used Hidden Markov Model to fit the behavior model of the attacker; secondly, due to the payoff of edge server is closely related to the attacker's behavior model, we used the gradient ascent method to maximize the payoff of edge server; finally, the optimal response of edge server was determined. Detailed experimental results showed that the new scheme can improve the payoff of the edge server and defend against attackers.

Keywords: Mobile edge computing · Hidden Markov Model · Edge server · Optimal response

1 Introduction

Mobile edge computing technology [1] enables the network service environment and cloud computing technology to combine at the edge of the network, improves the computing and storage capacity of the edge network, and reduces the network operation and service delivery delay. The network structure of this technology is three layers [2], i.e., edge device layer, edge server layer and cloud server layer. The edge device layer usually deploys some low-level electronic equipment, which runs in the physical world to complete tasks such as sensing, driving, and control. The edge server layer consists of several sub-layers, which are composed

Supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61872205, the Shandong Provincial Natural Science Foundation under Grant No. ZR2019MF018, and the Source Innovation Program of Qingdao under Grant No. 18-2-2-56-jch.

of different edge servers. The cloud server layer includes the cloud server and the data processing center.

Mobile edge computing enables edge server to handle network requests with low latency requirements. However, the edge server is closer to the terminal equipment of the Internet of Everything. The openness and heterogeneity of the terminal equipment, as well as the relatively limited computing and storage resources, have greatly increased the difficulty of the edge server protection, which has led to widespread cyber threats in edge servers. For example, an attacker could tamper with communication data packets, inject spurious pressure measurements to trick the decision-maker, delay the action of control valve and cause equipment damage in the scenario of a smart manufacturing plant. Without appropriate safety precautions, not only the production process may be interrupted, but also the lives of workers will be threatened to a great extent. In mobile edge computing, the uav's operating system is attacked, which will generate simulated global positioning system signals, mislead the uav system components, and drive them to the target area for capture. Therefore, it is a prerequisite and necessary condition for the further development of mobile edge computing technology to guarantee the security of edge server and enhance its ability to resist various security threats.

At present, the security protection technology in the edge computing environment mainly includes four aspects: intrusion detection, access control, defense strategy and key management. Intrusion detection is mainly used to monitor and detect the abnormal data on the host side or network side. Zhou et al. [3] proposed a general IDS framework for fog computing and developed a cloud and fog hybrid intrusion detection scheme. Chaabouni et al. [4] reviewed existing NIDS implementation tools and data sets, as well as free and open source network sniffing software. However, such intrusion detection scheme did not give full play to the characteristics of mobile edge technology and failed to effectively utilize its advantages. Controlling the access of malware to the edge computing environment can effectively defend against attackers. Yu et al. [5] proposed a universal framework of functional encryption suitable for fog computing access control, which not only provided privacy and fine-grained access control in fog computing, but also ensured the security of fog computing under channel attack. Yang et al. [6] proposed an intelligent IoT medical big data storage system with adaptive access control capability. However, there are some problems in the access control scheme of mobile edge computing technology, such as heavy computation, complex model and difficulty in rewriting parameters. Zheng et al. [7] reviewed existing defense strategies for moving targets. Huang et al. [8] proposed a dynamic game framework to simulate the long-term interaction between stealth attackers and active defenders. However, the traditional security defense strategies do not take into account the random distribution of attackers or the overall network cost. Key management is an encryption technique for communication in edge computing networks. Anzani et al. [9] proposed an improved scheme for hybrid symmetric design based on the hybrid key predistribution method of symmetric design, which improved connectivity and durability. Bitansky et al.

[10] showed how to use secret key function encryption to obtain exponentially valid undistinguished obfuscation. However, traditional key management schemes have poor scalability and lack of lightweight implementation methods, so they are not suitable for edge computing networks with features such as resource sharing, scalability, and virtualization.

Inspired by the above scheme, this paper proposed a novel game machine learning method to determine the optimal response for the edge server to the attacker. The contribution of this paper is as follows:

(1) This paper used Hidden Markov Model to predict the attacker's observed action sequence in next T period.

(2) Based on the results of the first step, this paper used the gradient ascent method to maximize the payoff of edge server for determine its optimal response to the attacker.

2 Preface

This section describes the theory of Markov process.

2.1 Markov Process

Markov process is a kind of random process proposed by the Russian mathematician A.A. Markov in 1907, in which, given the current state, its future evolution does not depend on its past evolution. In the real world, the Brownian motion of papers in liquids, the number of people infected with infectious diseases, the number of people waiting at stations, the changes in the number of animals in the forest, etc., can all be regarded as Markov processes. A Markov process refers to the transition of each state in the process only depends on the previous n states. The First-order Markov can be described as follows,

$$\begin{aligned} &\Pr(X_{n+1} = x | X_1 = x_1, X_2 = x_2, ..., X_n = x_n) \\ &= \Pr(X_{n+1} = x | X_n = x_n) \end{aligned} \tag{1}$$

Similarly, the m-order Markov can be defined as follows,

$$\begin{aligned} &\Pr(X_n = x_n | X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}, ..., X_1 = x_1) \\ &= \Pr(X_n = x_n | X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}, ..., X_{n-m} = x_{n-m}) \end{aligned} \tag{2}$$

As can be seen from the above, the Markov model can be represented as a triple, $(S, \prod, A)$, where S is a set of states, $\prod$ is the probability distribution of the initial state, and A is the state transition probability. In practical applications, the Markov process is not sufficient to solve the existing problems. Therefore, the Hidden Markov Model is proposed. Hidden Markov model is a quintuple $\{N, M, n, A, B\}$, where N is the number of hidden states; M is the number of observable states, its value can be obtained from the training set; $n = \{n_i\}$ is the probability of the initial state, that is, the probability of each hidden state occurring in the initial state, $A = \{a_{ij}\}^{N*N}$ is the transfer matrix of the hidden state, which refers to the probability of transition from the first state to the second state, $B = \{b_{ij}\}^{N*M}$ is the confusion matrix, which refers to given the initial state s the probability of the occurrence of an observation. Each probability in a state transfer matrix and a confusion matrix is time-independent, that is, as the system evolves, these matrices do not change over time.

3 Defense Strategy of Edge Server

This section describes the interaction process between the edge server and the attacker, and how to determine the optimal response of the edge server.

3.1 Two-Players Security Game

When the attackers attack the edge server, the attacker hopes to obtain the highest reward at the lowest cost, while the edge server hopes to defend attackers at the lowest cost, thus, the interaction between the attacker and the edge server can be modeled as a two-players security game. Assume that the attacker has two strategies: the non-attack strategy (i.e., *NA*) refers that the attacker doesn't launch an attack to the edge server; the attack strategy (i.e., *A*) refers that the attacker launches an attack to the edge server. Similarly, the edge server also has two strategies: the defense strategy (i.e., *D*) refers that the edge server would defense attackers; the non-defense strategy (i.e., *ND*) refers that the edge server would not defend the attacker. For attackers and edge servers, the payoff matrix is shown in Table 1.

Table 1. Payoff matrix

Payoff CE\AT	A	NA
D	$p - c_D, u - c_A$	$-c_D, 0$
ND	$-r, u - c_A + r$	$0, 0$

Where *AT* refers to the attacker, *CE* refers to the edge server, r means the additional payoff obtained by the attacker when the attacker initiates the attack and the edge server does not defend, c_A is the cost of the attacker, c_D is the cost of the edge server to defend the attacker's attack, u is the payoff from the attacker attacking the edge server. When the attacker plays the attack strategy A and the edge server plays the defense strategy D, the edge server will obtain the payoff p at the cost of c_D, and the attacker will gain the payoff u at the cost of c_A; When the attacker plays the attack strategy A and the computing center plays strategy *ND*, the payoff of the edge server is -r, the attacker will get the payoff u and the additional payoff r at the cost of c_A. When the attacker plays the attack strategy *NA* and the edge server plays the defense strategy D, the payoff of the edge server is -c_D, and the attacker's payoff is 0; When the attacker plays the attack strategy *NA* and the edge server plays the non-defense strategy *ND*, the payoff of the edge server and the attacker are 0.

Assuming $p - c_D > 0, u - c_A > 0$, from Table 1, the strategy profile (*attacker D*, *attacker A*) is a pure strategy *Nash equilibrium* of this game. However, in practical applications, this cannot be achieved because there is strong assumption: the information between the attacker and the edge server is known to each other, that is, the game is a perfect information game. This assumption cannot be realized when the edge server defends the attacker's intrusion, because the attack launched by the attacker is irregular and undirected, i.e. the edge computing is not certain that the attacker will be able to launch an attack properly, nor is it certain that the attack will be sufficient to launch an attack on it. To solve this problem, this paper proposes a game machine learning method to determine the edge server defense strategy.

3.2 Edge Computing and Attacker's Payoff

In order to determine the optimal defense strategy for edge servers, this paper defines the payoffs of the edge server as,

$$Payoff = \sum_{i \in attacker} \varphi_i(g, d) - \sum_{i \in attacker} c_{D_i} - \sum_{i \in attacker} r_i \tag{3}$$

In order to clearly represent every element of the payoffs of the edge server, this paper gives the most primitive form of the payoff of edge server. Where i is the number of attacks by an attacker, *attackers* is the set of attackers, $\varphi_i(g, d, s)$ is the edge server's payoff when it defends attackers, r_i is the cost of edge server when it does not take a defense strategy while the attacker attacks, g is the behavior model of the attacker, and d is the state of the edge server (the strategy of the edge server). Similarity, the payoff of attackers can be defined as,

$$Utility = \sum_{i \in attacker} u_i - \sum_{i \in attacker} c_{A_i} + \sum_{i \in attacker} r_i \tag{4}$$

Where u_i is the payoff obtained by the attacker's successful attack, c_{A_i} is the cost of the attacker launching the attack, and r_i is the additional payoff obtained by the attacker. From (4), the payoff of the attacker is related to the payoff obtained by each attack and the cost of launching the attack. From (3), the payoff of the edge server is related to its own state, the attacker's behavior model, and the attacker's historical data. Therefore, in order to determine the optimal strategy of the edge server, we need to evaluate the attacker's behavior model. Before determining the optimal strategy of the edge server, we first define the optimal response of the edge server as follows,

Optimal response: a strategy which can maximize the payoff of the edge server calls the optimal response of the edge server to an attacker.

From the definition of optimal response, this paper can transform the process of solving the optimal strategy of the edge server into the process of optimizing the calculation center's payoff.

3.3 The Edge Server's Optimal Response

Assume that the attacker's next attack is only related to its current state. An attacker launches an attack, there may be multiple states, but the attacker has only two kinds of behaviors. The attacker's hidden state set can be defined as $S = \{s_1, s_2, \cdots s_n\}$, and the attacker's observable behavior set is $A = \{a_1, a_2\}$. The attacker's hidden state transition matrix N and observable action transition matrix M can be obtained,

$$N = \begin{pmatrix} p_{s_1,s_1} & \cdots & p_{s_1,s_n} \\ \vdots & \cdots & \vdots \\ p_{s_n,s_1} & \cdots & p_{s_n,s_n} \end{pmatrix} \tag{5}$$

$$M = \begin{pmatrix} p_{s_1,a_1} & p_{s_1,a_2} \\ \cdots & \cdots \\ p_{s_n,a_1} & p_{s_n,a_2} \end{pmatrix} \tag{6}$$

Where $p_{s_i,s_j} = P(s_{t+1} = s_j | s_t = s_i)$, $p_{s_i,a_k} = P(a_t = a_k | s_t = s_i)$, the probability distribution of the original state is $\Pi = [\pi(i)]_n, \pi(i) = P(s_1 = s_i)$. The observable action a_{t+1} can be generated according to the observable action transfer distribution p_{s_i,s_j} of the attacker's hidden state s_t, and then the hidden state s_{t+1} can be generated according to the state transfer distribution p_{s_i,a_k} of the hidden state s_t. After T rounds of iteration, we can the attacker's observation sequence $a = \{a_1, \cdots, a_T\}$.

With the help of Hidden Markov model, the attacker's observation sequence in the next T cycles can be obtained. We can maximize the payoff of the edge server with the gradient ascent method to determine the optimal response of the edge server. That is,

$$\max_a \arg \{Pay = \sum_{z=t}^{T} \sum_{i \in attacker} (\varphi_{i,z}(g, a) - c_{D_{i,z}} - r_{i,z})\} \tag{7}$$

4 Experimental Stimulation

This paper uses anaconda integrated development tool to verify the game machine learning method. Firstly, this paper uses the Hidden Markov Model to predict the observable behavior sequence of attackers in 15 cycles under different initial states, and then based on this result determine the optimal response strategy of edge server. Secondly, based on the prediction results of Hidden Markov Model, this paper compares the changing trend of the edge server and the attacker's payoff in different initial states. Finally, to verify the efficiency of the proposed scheme, this paper compares and analyzes the *OUR* scheme with the *Random* scheme (randomly taking strategy), *ALL-D* scheme (always taking the defensive strategy) and *ALL-ND* (always taking the defensive strategy) to verify the *OUR* scheme can improve earnings at the edge of the computing center and defense the invasion of the attacker.

The parameter setting of the experiment is shown in Table 2. The hidden state transition matrix of the and the observable behavior state transition matrix of the attacker in Hidden Markov Model are shown in N and M.

Table 2. Parameters setting

Parameter	n	p	u	c_D	c_A	r	a_0	s_0	T
Value	3	0.6	0.3	0.2	0.2	0.5	0.5	0.5	0.5

$$N = \begin{pmatrix} 0.2 & 0.3 & 0.5 \\ 0.5 & 0.2 & 0.3 \\ 0.3 & 0.5 & 0.2 \end{pmatrix}, M = \begin{pmatrix} 0.5 & 0.5 \\ 0.4 & 0.6 \\ 0.7 & 0.3 \end{pmatrix}$$

4.1 The Prediction of Observable Behavior of Attackers

Table 3 predicts the observable behavior sequence of attackers in the next 15 interaction cycles based on Hidden Markov Model. In Table 3, AT is the attacker and CE is the edge server. Assuming that the initial state of the attacker is s_1, the observable action a_2 can be generated according to the observable action transfer distribution p_{s_i,s_j} of the attacker's hidden state s_1, and then the hidden state s_3 can be generated according to the state transfer distribution of the hidden state s_1. After 15 rounds of iteration, the observation sequence of the attacker can be obtained as $a = \{A, NA \cdots, A, NA\}$. Since the observable behavior of the attacker in the next 15 interaction cycles has been determined, the edge server can determine its own optimal response according to the result, maximize its own payoffs, and then resist the attack of the attacker.

4.2 Comparison of the Payoff of Attacker and Edge Server

Based on the prediction results of the Hidden Markov Model on the observable behavior sequence of attackers in the next 15 cycles, Fig. 1 compares the changing trends of the edge server and the attacker's payoff in the initial state of s_1 and s_2. It can be seen from Fig. 1(a) that the payoff of edge server is higher than that of attacker, which is determined by the payoff matrix of two players. According to the payoff parameters set in this paper, when the attacker plays strategy A, the optimal response of the edge server is strategy D, and the payoff of the edge server and the attacker is 0.4 and 0.1 respectively. When the attackers play strategy NA, the optimal response of the edge server is strategy NA, the payoff of the edge server and the attacker are 0.

From Fig. 1(a), the edge of computing center and the attacker's payoff is decreased in the second interaction cycle. The former yields decreased from 0.4 to 0, which yields decreased from 0.1 to 0, this is because the edge of computing center and the attackers are adjusted the strategy, that is, the edge server plays

Table 3. The observable behavior of attackers

Number	State	Action	Strategy (AT)	Strategy (CE)
0	s_1	a_2	*A*	*D*
1	s_3	a_1	*NA*	*ND*
2	s_2	a_2	*A*	*D*
3	s_1	a_1	*NA*	*ND*
4	s_3	a_1	*NA*	*ND*
5	s_2	a_2	*A*	*D*
6	s_1	a_2	*A*	*D*
7	s_3	a_1	*NA*	*ND*
8	s_2	a_2	*A*	*D*
9	s_1	a_2	*A*	*D*
10	s_3	a_1	*NA*	*ND*
11	s_2	a_2	*A*	*D*
12	s_1	a_1	*NA*	*ND*
13	s_3	a_2	*A*	*D*
14	s_2	a_2	*A*	*D*
15	s_1	a_1	*NA*	*ND*

the strategy *D* and the attacker plays the strategy *A* in the first interaction cycle, but the computing center and the attacker's strategy adjustment for *ND* and *NA* respectively in the second interaction cycle. Similarly, we can know that the reason for the change of payoff curve of edge server and attacker. Comparing Fig. 1(a) and Fig. 1(b), it can be seen that the edge server can determine the optimal response even if the attacker's initial state is different.

4.3 Comparison Between Our Scheme and the Other Three Schemes

In order to verify the efficiency of the proposed scheme, Fig. 2 compares the payoff variation trend of the edge server in *OUR* scheme, *Random* scheme, *ALL-D* scheme and *ALL-ND* scheme in different initial states. According to Fig. 2(a), the results of *OUR* scheme are the best, followed by *ALL-D*, and the worst effect of *ALL-ND*. This is because the OUR scheme uses the prediction results of Hidden Markov Model to determine the optimal response of edge server for the attacker's strategy.

Therefore, *OUR* scheme has the highest payoff and the best effect. The reason for the worst effect of *ALL-ND* is that no matter what strategy the attacker takes, the edge server always plays the strategy *ND*, so the edge server in this scheme has the lowest payoff and the worst effect. Similarly, it can be concluded that the effect of *Random* and *ALL-D* is lower than that of *OUR* scheme. Contrast Fig. 2(a) and Fig. 2(b), the attacker in any initial state, the payoffs of the edge server are all the highest in the *OUR* scheme, this is because this paper uses Hidden Markov Model to predict the attacker in the observable behavior of the future interaction cycle, the edge server based on the prediction results defends against attackers.

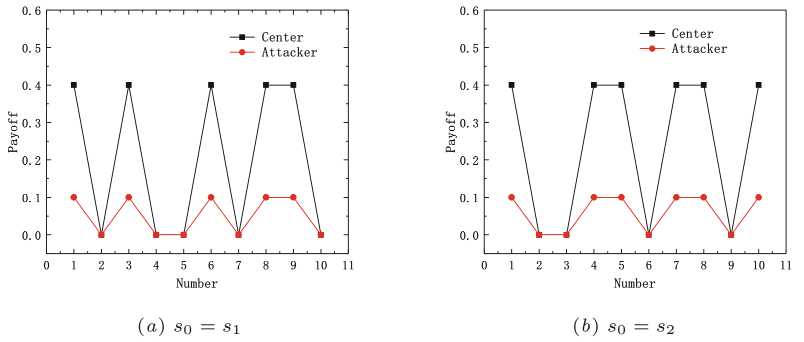

Fig. 1. Comparison of edge server and attacker's payoff.

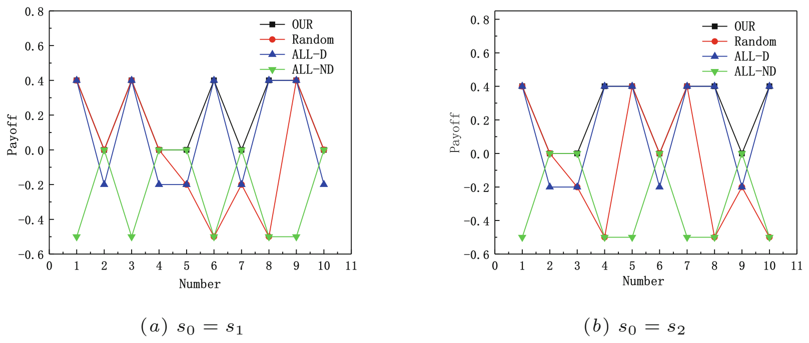

Fig. 2. Payoff comparison of edge server.

5 Conclusion

The key to promote the application of mobile edge computing technology is to improve the ability of edge server to resist attackers. In this paper, a game machine learning method is proposed to solve this problem. In the scheme, Hidden Markov Model is used to fit the behavior model of the attacker, and the gradient ascending method is used to maximize the benefits of the edge server, so as to determine the optimal response of the edge server to attacker. Detailed experimental results verify the effectiveness of the proposed scheme.

References

1. Han, Y., Wang, X., Leung, V., Niyato, D.: Convergence of edge computing and deep learning: a comprehensive survey. arXiv preprint arXiv:1907.08349 (2019)
2. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.: A survey on mobile edge computing: the communication perspective. IEEE Commun. Surv. Tutor. **19**(4), 2322–2358 (2017)
3. Zhou, X., Xing, L.: Sample selected extreme learning machine based intrusion detection in fog computing and MEC. Wirel. Commun. Mob. Comput. **2018**, 1–10 (2018)
4. Chaabouni, N., Mosbah, M., Zemmari, A., Sauvignac, C., Faruki, P.: Network intrusion detection for IoT security based on learning techniques. IEEE Commun. Surv. Tutor. **21**(3), 2671–2701 (2019)
5. Yu, Z., Man, H., Xu, Q.: Towards leakage-resilient fine-grained access control in fog computing. Future Gener. Comput. Syst. **78**(1), 763–777 (2018)
6. Yang, Y., Zheng, X., Guo, W., Liu, X., Chang, V.: Privacy-preserving smart IoT-based healthcare big data storage and self-adaptive access control system. Inf. Sci. **479**, 567–592 (2019)
7. Zheng, J., Namin, A.: A survey on the moving target defense strategies: an architectural perspective. J. Comput. Sci. Technol. **34**(1), 207–233 (2019)
8. Huang, L., Zhu, Q.: A dynamic games approach to proactive defense strategies against advanced persistent threats in cyber-physical systems. Comput. Secur. **89**, 101660 (2020)
9. Anzani, M., Haj Seyyed Javadi, H., Modirir, V.: Key-management scheme for wireless sensor networks based on merging blocks of symmetric design. Wireless Netw. **24**(8), 2867–2879 (2017). https://doi.org/10.1007/s11276-017-1509-y
10. Bitansky, N., Nishimaki, R., Passelègue, A., Wichs, D.: From cryptomania to obfustopia through secret-key functional encryption. J. Cryptol. **33**(2), 357–405 (2019). https://doi.org/10.1007/s00145-019-09337-9