An Intrusion Intention Analysis Algorithm Based on Attack Graph

Zhen Zhu[1*],Guofei Chai[2]
[1]Equipment and Training Management Center (Information Center), Quzhou College of Technology, Quzhou 324000, China
[2]College of Electrical and Information Engineering, Quzhou University, Quzhou 324000, China
[*]Email: zhuzhen_qzct@126.com

Received: January 11, 2021. Revised: June 28, 2021. Accepted: July 16, 2021. Published: July 20, 2021.

Abstract—The discovery of intrusion intention is one of the challenging tasks faced by network security managers. To detect intrusion detections, this paper presents a domain-device attack graph, and collects and analyzes the underlying data of the network topology. On this basis, the attack graph Map was quantified by the Bayesian theory. The minimum weight spanning tree (Min-WFS) algorithm was adopted to automatically recognize the calculation cost of key devices in the network topology, providing an important basis for network maintenance. Experimental results show that the intrusion intentions can be effectively identified with the aid of the quantified domain-device attack graph Map, and this identification method is easy to implement.

Keywords—network security; intrusion intention; attack graph; recognition algorithm

I. INTRODUCTION

The recognition of intrusion intention refers to the effective identification of the means and goals of the attacker by analyzing the massive alarm data fed back from the underlying intrusion detection system in the Internet environment. It is essentially a scientific analysis of the intrusion process [1]. As an emerging focus of network security, the rapid and effective identification of intrusion intention provides network managers with an important basis for security management, laying the basis for the early detection and prevention of network security threats, as well as the analysis on network security situation [2].

Computer scientists have initially identified intrusion intention with the aid of artificial intelligence. For instance, Qi and Xu [3] combined security analysis and attack defense model into an intrusion detection model based on attack graph; using offensive and defense game technology, the proposed model facilitates the intelligent decision-making for the analysis on network security situation. Aiming to detect unknown malicious mobile agents, Bagga et al. [4] proposed an architecture for intrusion prevention system based on adaptive attack graph; inspired by biological immune system, their architecture effectively prevents man-in-the-middle (MITM) attack, masquerade attack, replay attack, denial of service (DoS), and unauthorized access attack, through the Boyer–Moore string search algorithm of k-nearest neighbors (k-NN) classifier, and the N-gram feature analysis of mobile agents; experimental results show that the architecture towers over the relevant schemes in timeliness, security, and accuracy, and applies to network security defense in mobile agent environment.

Chamotra et al. [5] proposed a highly interactive honeypot baselining structure to overcome the difficulty in preventing attacks and destruction of network sensors; By attack graph modeling, this structure discovers key intrusion intentions, and realizes early prevention of attacks and destruction of network sensors. Singh and De [6] developed a multi-layer perceptual genetic algorithm that fuses attack graph technology to effectively protect the network from distributed DoS: firstly, the features of the incoming data packets are analyzed, quantified, and combined; then, the risky hosts are identified in the network, and maintained to prevent the distributed DoS. Subbulakshmi [7] presented an integrated detection and defense mechanism to solve the series of problems caused by distributed DoS on the network; under the mechanism, the attack graph model of the network is generated by machine learning algorithms like neural network (NN), self-organizing mapping and enhanced support vector machine (SVM); the real Internet Protocol (IP) address of the attacker is recognized by computing the entropy of each node in the model, thereby preventing the attack.

Breier and Branišová [8] noted that the network security vulnerabilities could be identified from the system log, and created an intrusion intention detection method based on data mining; Under the framework of Apache Hadoop, the proposed method supports distributed storage and processing of data, and achieve forecast and blocking of intrusion intentions by mining and computing the data on known vulnerability features. Lee and Kim [9] defined and described all possible threats to broadcast services on the Internet, and constructed a security vulnerability scoring system for these threats based on general information technology; the proposed system can establish the system attack graph by assigning weights to different vulnerabilities, and make accurate forecast of intrusion intentions. Hu et al. [10] designed a prediction scheme of intrusion intention with batch attack graphs: first, a stacked autoencoder network is introduced to generate a two-layer attack graph model; then, an overall prediction route for intrusion intentions is generated by a set, and used to maintain network security.

Considering the security issues of fifth generation (5G) networks, Rupprecht et al. [11] provided a strategy for identifying intrusion intentions related to mobile network: Based on the goals, recommended defense measures, potential causes, and root causes, the strategy classifies and plots the known attacks, derives the potential intrusion route through casual analysis, and blocks the intrusion intentions by maintaining the equipment on the route. To maintain the safety of the Internet of things (IoT), Bajpai et al. [12] developed an approach to recognize and detect the intrusion of IoT devices: various scanning techniques are adopted to pinpoint the vulnerabilities of IoT devices in the network, set up attack graph models, and detect the intrusion intentions; the network security is enhanced by maintaining the core devices. In addition, the authors discussed the strengths and defects of the approach, and demonstrated the actual maintenance results. To prevent network intrusion, Nicho [13] proposed a cyclic intrusion intention recognition model, involving such phases as planning, execution, checking, and action. The model quantifies the route of network intrusion. Experimental results show that the cyclic model could effectively determine the primary route of network intrusion, providing a guidance for the identification of intrusion intentions and the maintenance of network equipment.

Since multi-server authentication is prone to network intrusion, Irshad et al. [14] put forward a detection method of multi-service intrusion intention for multimedia service providers; compared with traditional intrusion intention detection methods, their method has certain advantages in the prediction of network intrusion on multi-server authentication. Based on Chebyshev chaotic map attack graph, Chatterjee et al. [15] proposed an identification scheme for multi-server intrusion intention, which searches for the intrusion intentions in each server of the network through Chebyshev chaotic mapping, biometric verification, and attack graph iteration. This scheme is easier to deploy and maintain than other schemes. Kfoury et al. [16] set up an intrusion intention detection system based on self-organizing mapping NN. In this system, the attack routes are divided into three categories; a complete attack graph is formed by modeling the corresponding attack data, and used to eliminate the network vulnerabilities.

Phan and Park [17] proposed an effective scheme for network intrusion detection in the software-defined networking (SDN)-based cloud: with the help of SVM and self-organizing mapping, the scheme models the intrusion intentions, and detects the maximum intrusion risk in the network by IP filtering; experimental results show that the scheme is an effective and innovative way to detect intrusion intentions. In view of the diversity and complexity of network intrusions, Noor et al. [18] invented a novel recognition framework for intrusion intention based on machine learning. Under the framework, the threats extracted from known threat sources are associated with relevant detection mechanisms, producing a semantic attack graph; then, the graph is quantified into the probability relationship between nodes, and the intrusion route is optimized iteratively through machine learning and continuous training; in this way, the security of the entire network is evaluated and maintained. To safeguard the wireless network, Ostad-Sharif et al. [19] proposed an intrusion detection method for wireless network; their method constructs an attack graph by formal technology, and detects the intrusion risks in wireless network in a comprehensive manner.

For the security of vehicle network, Mishra et al. [20] established a two-way authentication framework based on the chaotic mapping. The potential intrusion intentions are detected through simulated attacks, aiming to make the communication safe, efficient, and anonymous. Simulation results demonstrate the high detection efficiency and accuracy of the framework. Based on IEEE 1815.1, Kwon et al. [21] presented an intrusion intention detection system for the security of cyber-physical system (CPS) in the power industry: the bidirectional recurrent neural network (RNN) is adopted to build the attack graph, and the grid security is assured through predictive analysis; experimental results show that the proposed system can successfully detect five types of CPS malware behavior (CMB) attacks, and three types of false data injection (FDI) and disabling reassembly (DR) attacks. Drawing on theories of machine deep learning, Jeong et al. [22] organized an artificial intelligence analysis model for intrusion intentions: the representative datasets on attack graphs and network equipment vulnerabilities are employed to quantify and train the model, using autoencoders and convolutional neural network (CNN); the trained model could accurately detect the intrusion intentions.

In the light of the features of cloud computing networks, Harikrishna and Amuthan [23] came up with a network intrusion prevention scheme based on convolutional recursively enhanced self-organizing mapping and software-defined network, which organizes network attack graph and detects intrusion intentions by vector quantization. Compared with the existing intrusion intention systems, this scheme boasts a high prediction accuracy and low false alarm rate. Sengupta et al. [24] designed an intrusion intention identification system for the industrial IoT security against various attack threats, provided the specific solutions of the system, and summarized several open directions for future research on the detection of intrusion intention. Maniyath and Thanikaiselvan [25] introduced he chaotic encryption algorithm to intrusion prevention, and created a chaotic encryption defense mechanism based on the attack graph, which can effectively prevent the illegal intrusion of network core resources.

The above research literature mainly focuses on the specific detection of network intrusion, ignoring the identification of intrusion attempts. Drawing on the above research, this paper further presents a domain-device attack graph model, and relies on the Bayesian probability theory to investigate the automatic identification of intrusion intentions

II. Automatic Intrusion Intention Identification Model

A. Structure design

The intrusion intention is defined as the real intention of the attacker in invading the network. It could be measured by the information value that the attacker wishes to acquire, and the degree of damage to the network service. To realize this intention, the attacker needs to breach the network first. Therefore, the intrusion intention could be reconstructed by modeling, analyzing, and computing the network intrusion, and collecting the traces of intrusion. Through the reconstruction, it is possible to detect the route and harms of intrusion, and enable network managers to predict, evaluate, and block the intrusion, resulting in the overall improvement of network security. Based on the various information required for intrusion simulation and the steps and methods of intention reconstruction, an automatic identification structure is designed as shown in Figure 1.

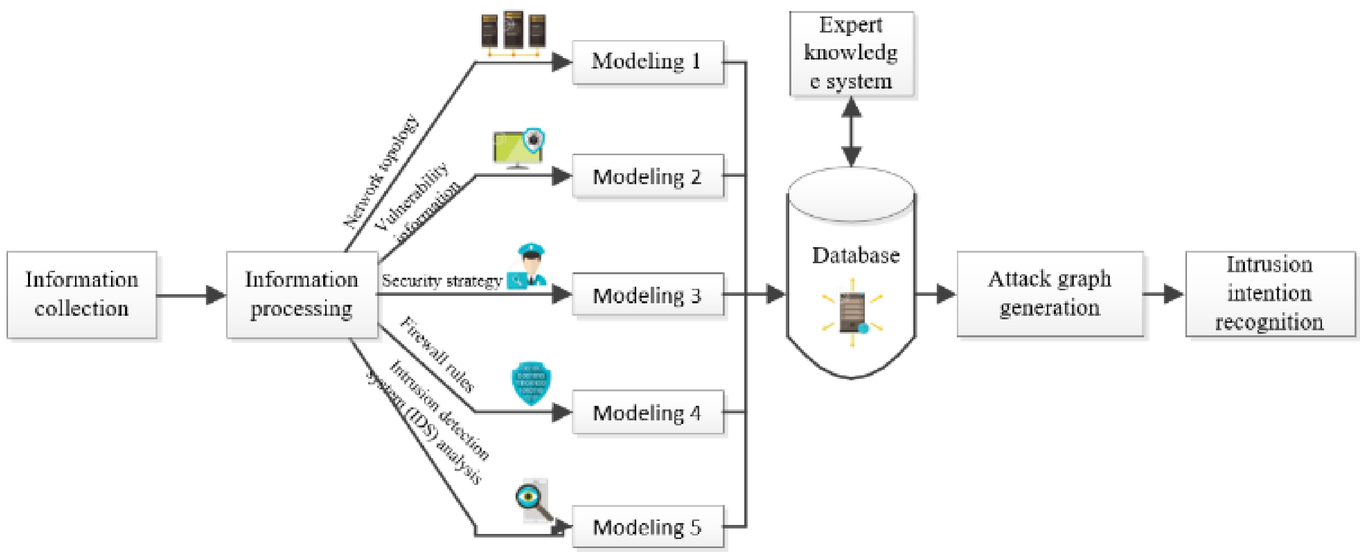

Fig. 1 The structure of intrusion intention recognition

B. Definitions and constraints

The domain-device attack graph was modeled under the following formal constraints:

Let H be the set of vulnerabilities for network devices. Then, any vulnerability h is the set H can be defined as a triple (h_{id}, h_p, h_c), where $h_{id}$is the serial number of the vulnerability in the Common Vulnerabilities and Exposures (CVE); $h_p$is the set of preconditions for the attacker to successfully exploit the vulnerability; $h_c$is the set of consequences after the attacker exploits the vulnerability.

Let Ebe the set of network devices in the device layer. Then, any network device *e* in the set E can be defined as a triple (*H*, *o*, *NetE*), where H is the set of vulnerabilities in the network device; *o*is the set of open ports of the device; *NetE*is the set of network devices linked to this device.

Let D be the set of domains in the network. Then, any domain *d*in the set D can be defined as a pair (*E*, *NetD*), where E is the set of network devices in the domain; *NetD*is the set of domains linked to this domain.

Let N be the set of node devices. Then, any device n in the set N can be defined as a triple (n_{id}, *D*, *E*), where $n_{id}$is the serial number of the node device; D is the set of domains for the node device; E is the set of devices covering the node device.

In a common intrusion event, the attacker firstly scans all vulnerabilities of a node device n in the network topology, creating a set h of vulnerabilities. Then, the attacker breaches into a device e in a domain d, and completely controls the device e from low authority to high authority. Next, the attacker breaches into another device *e′* in the domain d via the *NetE* of device e. In this way, the entire domain d is controlled by the attacker. Finally, the attacker breaches into every other domain *d′* via the NetD of domain d, thereby realizing its intrusion intention. Hence, the domain-device attack graph can be modeled as shown in Figure 2.

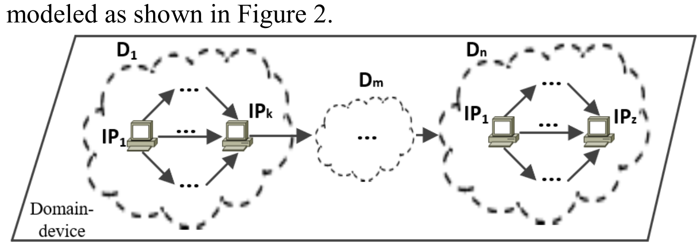

Fig. 2 The domain-device attack graph

Definition 1. Domain-device attack graph *Map*

The domain-device attack graph is a directed graph formalized as a pair ($M_D(N_d, L_d)$, $M_E(N_e, L_e)$), where $M_D(N_d$,

L_d)is the attack graph at the domain layer; $M_E(N_e, L_e)$is the attack graph at the device layer; $N_q$is the set of node devices; $L_q$is the set of links between node devices; $q$is $d$or e .

In Definition 1, if there exists alinkl_{ij} that makes the node device n_i point to node device n_j, then the attacker successfully uses device $n_i$to breach node device n_j, where $l_{ij}\in L_q$, $n_i\in N_q$, $n_j\in N_q$, $n_i\neq n_j$, and $q=d$or e.

Definition 2. Intrusion route R

In the domain-device attack graph *Map*, if there exists a set N' of node devices, such that the attacker can realize its intrusion intention from the initial node device $n_0$along the devices in set N', then the link formed by the node devices in the set N' and the links between the devices is an intrusion route of the domain-device attack graph, and denoted as r. In the *Map*, all the intrusion routes form a set R.

Definition 3. Set K_{minH} of minimum weight points

In the domain-device attack graph *Map*, the set of node devices is denoted as $M(N)$. Let $K_i$be a nonempty subset that excludes the initial node device $n_0$and the target node device n_n, and satisfies $K_i\subset M(N)$. If any intrusion route $r$in set Rpasses all node devices in K_i, then the set $K_i$can be called the set K_{minH} of minimum weight points.

To design the generation strategy of K_{minH}, the following formal constraints were put forward:

Let A_{rank}be the rank of the set $M(N)$of node devise, i.e., $A_{rank}=|M(N)|$; Let D_{rank}be the rank of the set R of intrusion routes, i.e., $D_{rank}=|R|$ and $D_{rank} = C_{A_{rank}-2}^{1} + C_{A_{rank}-2}^{2} + \cdots + C_{A_{rank}-2}^{A_{rank}-2}$; Let $r_i$be an arbitrary intrusion route that satisfies $r_i\in R$; Let $N_i$be the set of node devices along the intrusion route r_i. In $r_i$and N_i,i = 1, 2, …, D_{rank}.

C. Generation strategy for the attack graph

The generation algorithm for the domain-device attack graph can be implemented in the following steps:

Step 1. Initialize the variables related to the strategy, and define *Pow*as the authority variable.

Step 2. Remove a device from the set E of network devices in the device layer, and store it in variable e.

Step 3. Set the authority variable *Pow*corresponding to e as *Null*, calculate the rank of the vulnerability set $H_e$of network device e, and assign the result to variable *Num*.

Step 4. Set the loop variable i=1.

Step 5. Set the loop variable $j=i+1$.

Step 6. If the consequence $h_c$of the successful intrusion of vulnerability $h_i$exactly meets the precondition h_p, for the intrusion of vulnerability h_j, i.e., the relationship $h_j\times h_p \subseteq h_i\times h_c$, then assign *Guest*or *Admin*to the authority variable *Pow*corresponding to e, and jump to Step 9; otherwise, go to Step 7.

Step 7. Set variable $j=i+1$, judge whether j equals variable *Num*; if not, jump to Step 6; otherwise, go to Step 8.

Step 8. Set variable $i=i+1$, judge whether i equals *Num*-1; if not, jump to Step 5; otherwise go to

Step 9. Remove the next device from the set E of network devices in the device layer, and store it in variable e.

Step 10. If set E<>Null, jump to Step 3; otherwise, go to Step 11.

Step 11. Restore the set E of network devices in the device layer, and set *Num* as the rank of set E.

Step 12. Set the loop variable i=1.

Step 13. Set the loop variable $j=i+1$.

Step 14. If the authority of device e_i is not *Null*, i.e., the *Pow* corresponding to device e_i is *Guest*or *Admin*, go to Step 14; otherwise, jump to Step 17.

Step 15. If device $e_i$and device $e_j$can be linked via port o, and if device $e_j$can be intruded via device e_i to obtain the authority *Guest*or *Admin* of device e_j, then go to Step 16; otherwise, jump to Step 17.

Step 16. Add devices $e_i$and e_j to the set$M_E(N_e)$ of node devices in the attack graph at the device layer, and add the link $e_i\rightarrow e_j$to the set $M_E(N_e, L_e)$ of links in that graph.

Step 17. Set $j=j+1$, and judge whether j equals variable *Num*; if not, jump to Step 13; otherwise, jump to Step 18.

Step 18. Set $i=i+1$, and judge whether i equals variable *Num*-1; if not, jump to Step 13; otherwise, jump to Step 19.

Step 19. Add the attack graph $M_E(N_e, L_e)$ at the device layer to the attack graph *Map*.

Step 20. Set the loop variable i=1.

Step 21. Set the loop variable $j=i+1$.

Step 22. Let $d_1$be the protection domain of device e_i, and $d_2$be that of devicee_j.

Step 23. If the two domains are different, i.e., d_1<>d_2, if device $e_i$has been breached, go to Step 24; otherwise, jump to Step 26.

Step 24. Add the protection domain of device e_i to the attack graph $M_D(N_d)$ at the domain layer.

Step 25. If device $e_j$can be intruded via device e_i to elevate the authority of device e_j to *Guest*or *Admin* of device e_j, then add the protection domain of device $e_j$to the set of domains $M_D(N_d)$in the attack graph at the domain layer, and add the link $d(e_i)\rightarrow d(e_j)$to the set of links $M_D(L_d)$in that graph; otherwise, go to Step 26.

Step 26. Set $j=j+1$, and judge whether j equals variable *Num*; if not, jump to Step 22; otherwise, go to Step 27.

Step 27. Set $i=i+1$, and judge whether i equals variable *Num*-1; if not, jump to Step 21; otherwise, go to Step 28.

Step 28. Add the attack graph $M_D(N_d, L_d)$ at the domain layer to the attack graph *Map*.

After analysis, the time complexity of generation algorithm for the domain-device attack graph was obtained as $O(|E|\times|H_e|^2)$.

III. Quantification and Response of Intrusion Intention

A. Quantification of intrusion intention

In the domain-device attack graph, the success or failure of the intrusion into each node depends on the attributes of the vulnerabilities. Here, three vulnerability attributes are defined: easiness *d_deg*, privacy *p_deg*, and return rate *r_deg*.

Depending on the actual operation of the complex network, the probability of successful manipulation of vulnerability *h*can be defined as:

$$\rho(h) = \xi_1 d\_deg + \xi_2 p\_deg + \xi_3 r\_deg \quad (1)$$

where, ξ_1, ξ_2, and $\xi_3$weights. Their values can be assigned by network managers according to the actual situation (Table 1).

Table 1. The attributes and values of vulnerabilities

Attribute	Degree	Value
ξ1	Easy	0.9
	Moderate	0.6
	Difficult	0.3
ξ2	Small	0.6
	Medium	0.8
	Large	0.9
ξ3	Low	0.5
	Medium	0.7
	High	0.9

On the attack graph at the device layer, if there exist j routes passing through k node devices that allow the attacker to realize the intrusion intention *i*. Then, the probability for the intrusion intention *i* to be realized can be described as:

$$\rho(i) = 1 - \prod_j [1 - \prod_k \rho(\rho_k)] \quad (2)$$

By Bayesian formula, the relative probability that the intrusion intention *i* can be realized via route t can be obtained as:

$$\rho(e_t \mid i) = \frac{\rho(i \mid e_t) \times \rho(e_t)}{\rho(i)} \quad (3)$$

where, $t = 1, 2, \ldots, j$.

If the relative probability of a route is relatively large, then the attacker is very likely to realize its intrusion intention along this route. Hence, the network managers should focus on protecting the node devices on this route.

B. Min-WFS algorithm

The set K_{minH} of minimum weight points of domain-device attack graph *Map*was generated by the minimum weight spanning tree (Min-WFS) algorithm:

Step 1. Initialize the variables related to the strategy, and set the flag variable *Flag*to true.

Step 2. Define variable *i* as the number of elements in the set R.

Step 3. Set the flag variable *Flag*to true.

Step 4. Define variable *j* as the number of elements in the set M(N).

Step 5. Judge whether the intersection between sets $K_i$and $N_j$is empty; if yes, there exist a route that does not pass the nodes in set K_i, set the flag variable *Flag*to false, and go to Step 6; otherwise, go to Step 6.

Step 6. Set *j*=*j*-1, and judge whether *j* is zero; if yes, go to Step 7; otherwise, jump to Step 5.

Step 7. Judge whether the flag variable *Flag*is true; if yes, add set $K_i$to set *K*.

Step 8. Set *i*=*i*-1, and judge whether *i* is zero; if yes, go to Step 9; otherwise, jump to Step 3.

Step 9. Find the minimum element in set K, and add it to the set $K_{minH}$of minimum weight points.

Through analysis, the time complexity of *Min-WFS*algorithm was obtained as $O(D_{rank} \times A_{rank})$.

C. Response to intrusion intention based on K_{minH}

In the attack graph *Map*, the intrusion intention is mainly curbed by cutting off the intrusion routes. Hence, it is an economical method to respond to the intention based on K_{minH}, which can be obtained by Min-WFS algorithm.

Let $e_i$be an arbitrary node device in set K. Then, the cost $Cost(e_i)$ of maintaining this device covers labor cost, software and hardware cost, and other costs.

Under the aforementioned assumptions, the cost of the optimal maintenance measure for network managers to block intrusion intention equals the sum of the maintenance costs of every node device in the K_{minH}:

$$Sum_{Cost} = \sum_{i=1}^{|K_{min\,H}|} Cost(e_i) \quad (4)$$

IV. Simulation and Results Analysis

A. Simulation environment and vulnerability test

To test the proposed detection algorithm for intrusion intention based on domain-device attack graph, the research team designed a simulation environment, which consists of four domains and the Internet. The four domains were named: D_1, D_2, D_3, and *DMZ*. The access policies of each domain are as follows:

Network devices $E_4$and E_9 in $D_1$and D_2can access the database server in D_3; Network devices $E_4$and E_9 in $D_4$and D_7can access each other; the devices in the same domain can access each other; intranet devices can exchange data with the Internet via *DMZ*, while other inter-domain accesses are banned. Figure 3 illustrates the simulation environment.

The X-Scan software was adopted to scan the vulnerabilities of each network device in Figure 3. The information on the domain-device system and vulnerabilities thus obtained are listed in Table 2.

Table 2. The domain-device system and vulnerabilities

Domain number	Device number	System configuration	Vulnerability
DMZ	E1	Windows Server 2003	CVE-2004-0575
		Titan FTP6.0.3	CVE-2008-0702
	E2	Windows Server 2003	CVE-2002-0364
		IIS 5.0 Web	CVE-2006-2379
	E3	Check Point VPN-1 Server 4.1	CVE-2004-0040
D1	E4	Windows Server 2000	CVE-2007-0038
D2	E7	Windows XP	CVE-2006-2370

	E8	Windows XP	CVE-2003-0252
D3	E10	Windows XP	CVE-2004-13062
		SQL Server	CVE-2004-0893
			CVE-2003-0004

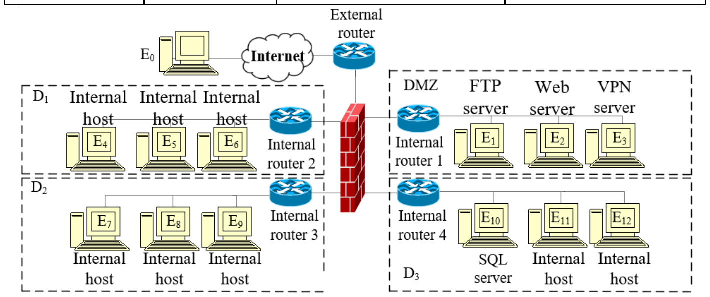

Fig. 3 The simulation environment

B. Analysis and recognition of intrusion intention

The SQL server E_{10}, which contains a massive amount of sensitive data, is the primary target of most intrusion intentions, and thus in need of special protection. Let i be the intrusion intention on network device E_{10}. Then, the device-device attack graph (Figure 4) was plotted by the generation algorithm for the domain-device attack graph.

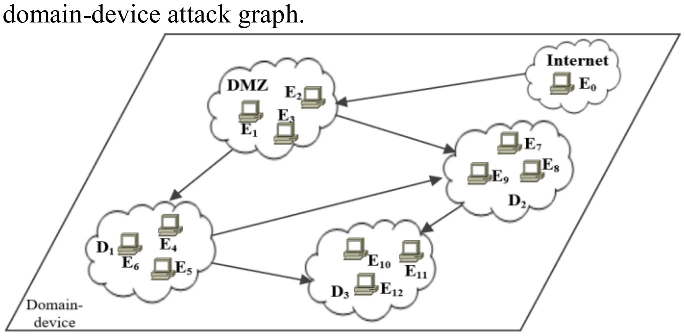

Fig. 4 The device-device attack graph for intrusion intention i

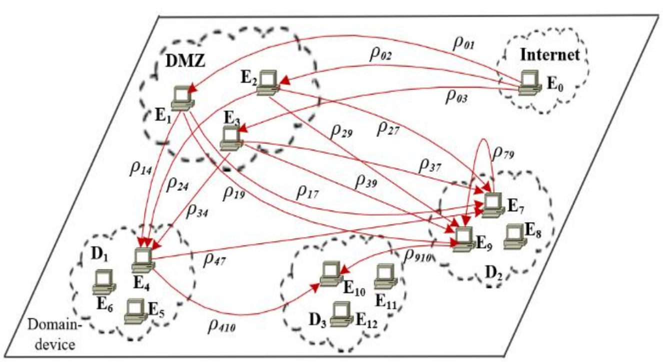

Fig. 5 The device-device intrusion routes for intrusion intention i

By formula (1), the probabilities for devices E_1, E_2, E_3, E_4, E_7, E_9, and E_{10}in Figure 4 to be intruded could be obtained as 0.3, 0.2, 0.5, 0.6, 0.7, 0.3, and 0.5, respectively. That is, ρ_{01}=0.3, ρ_{02}=0.2, ρ_{03}=0.5, $\rho_{14}=\rho_{24}=\rho_{34}$=0.6, $\rho_{17}=\rho_{27}=\rho_{37}=\rho_{47}$=0.7, $\rho_{19}=\rho_{29}=\rho_{39}=\rho_{79}$=0.3, and $\rho_{410}=\rho_{910}$= 0.5. Then, Figure 4 was quantified to obtain the device-domain intrusion routes (Figure 5). In total, network device E_{10}could be intruded by 12 different routes. The distribution of these routes is described in Table 3.

Since there are 12 intrusion routes in the domain-device attack graph *Map*, the probability for each route to be used by the attacker is 1/12, i.e., $\rho(r)$=1/12≈0.083. By formulas (2) and (3), the probability $\rho(i)$and relative probability $\rho(r|i)$ for the attacker to realize the intrusion intention i via each route were obtained by formulas (2) and (3). Figure 6 presents the distribution of the two probabilities.

Table 3.The intrusion routes for intrusion intention i

Route number	Route details	Route number	Route details
1	E0→E1→E4→E10	7	E0→E2→E7→E9→E10
2	E0→E1→E4→E7→E9→E10	8	E0→E2→E9→E10
3	E0→E1→E7→E9→E10	9	E0→E3→E4→E10
4	E0→E1→E9→E10	10	E0→E3→E4→E7→E9→E10
5	E0→E2→E4→E10	11	E0→E3→E7→E9→E10
6	E0→E2→E4→E7→E9→E10	12	E0→E3→E→E10

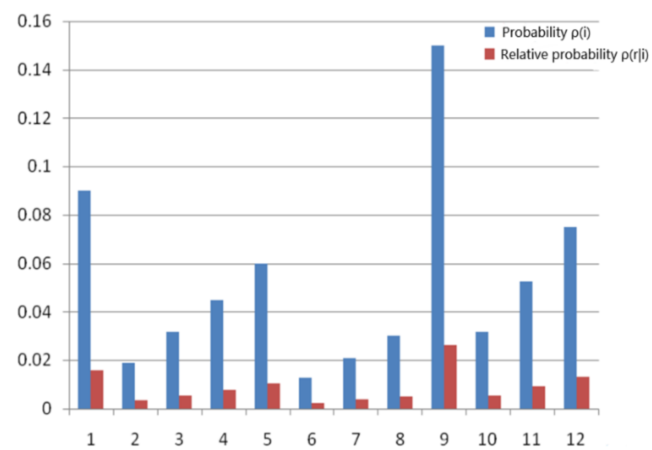

Fig. 6 The probability and relative probability of realizing intrusion intention i along each route

As shown in Figure 6, the probability maximized at $\rho(r_9|i)$=0.0262, indicating that the intrusion intention is most likely to be realized along the route $E_0 \rightarrow E_3 \rightarrow E_4 \rightarrow E_{10}$. By the Min-WFS algorithm, the set of minimum weight points for each route was determined as (E_4, E_9). To effectively curb the intrusion intention i on network device E_{10}, the network managers need to step up the protection of devices $E_4$and $E_9$by timely downloading related patches, and restricting the access of some users to E_{10}. By formula (4), the cost of these efforts is *Cost*(E_6)+*Cost*(E_9). Through the maintenance of E_6 and E_9, network managers can prevent the occurrence of network intrusion with the maximum probability, and achieve the goal of low-cost maintenance of network security.

V. Conclusions

As a hot topic in network security management, the identification of intrusion intention is an important means to analyze and assess the situation of network intrusion, providing the basis for managers to effectively determine network vulnerabilities and prevent network intrusions. As a result, many network security experts are striving to develop robust identification technologies for intrusion intention. Drawing on the previous results [10], the research team presented an automatic analysis, detection, and response method for intrusion intentions based on domain-device attack graph. Bayesian probability analysis was introduced into the attack graph to quantify each intrusion route in the graph, and then determine the set of minimum weight points. The network devices in the set should be maintained carefully by network managers, thereby curbing the realization of intrusion intention. To provide the data basis for efficient and simple management of intranet, the research team will further refine the attack graph into domain-device-vulnerability attack graph, and improve the accuracy of intrusion intention identification by quantifying the routes at the vulnerability layer.

ACKNOWLEDGMENT

The Project Supported by Zhejiang Provincial Natural Science Foundation of China (LQ1 7F030005), Guiding Project of Science and Technology Plan in Quzhou.

References

[1] R. Trifonov,S. Manolov,G. Tsochev,G. Pavlova, “Automation of Cyber Security Incident Handling through Artificial Intelligence Methods”, WSEAS Transactions on Computers, vol. 18, no. 35, pp. 274-280, 2019.
[2] A. Andreatos,N. Chatzipantou, “Using Nagios on a Raspberry Pi to Monitor a Network with Emphasis on Security”, WSEAS Transactions on Computers, vol.19, no. 31, pp. 262-267, 2020.
[3] F. Qi, H.L. Xu, “Research on network defense graph model in network security”, International Journal of Security and Its Applications, vol. 10, no. 11, pp. 23-32, 2016.
[4] P. Bagga, R. Hans, V. Sharma, “A biological immune system (BIS) inspired mobile agent platform (MAP) security architecture”, Expert Systems with Applications, vol. 72, no. 4, pp. 269-282, 2017.
[5] S. Chamotra, R.K. Sehgal, R.S. Misra, “Honeypot baselining for zero day attack detection”, International Journal of Information Security and Privacy (IJISP), vol. 11, no. 1, pp. 63-74, 2005.
[6] K.J. Singh, T. De, “MLP-GA based algorithm to detect application layer DDoS attack”, Journal of information security and applications, vol. 36, no. 11, pp. 145-153, 2017.
[7] T. Subbulakshmi, “A learning-based hybrid framework for detection and defence of DDoS attacks”, International Journal of Internet Protocol Technology, vol. 10, no. 1, pp. 51-60, 2017.
[8] J. Breier, J. Branišová, “A dynamic rule creation based anomaly detection method for identifying security breaches in log records”, Wireless Personal Communications, vol. 94, no. 3, pp. 497-511, 2017.
[9] J.H. Lee, S.J. Kim, “Analysis and security evaluation of security threat on broadcasting service”, Wireless Personal Communications, vol. 95, no. 4, pp. 4149-4169, 2017.
[10] F. Hu, J.Y. Wang, X.F. Xu, C.J. Pu, T. Peng, “Batch image encryption using generated deep features based on stacked autoencoder network”, Mathematical Problems in Engineering, pp. 3675459, 2017.
[11] D. Rupprecht, A. Dabrowski, T, Holz, E. Weippl, C. Popper, “On security research towards future mobile network generations”, IEEE Communications Surveys & Tutorials, vol. 20, no. 3, pp. 2518-2542, 2018.
[12] P. Bajpai, A.K. Sood, R.J. Enbody, “The art of mapping IoT devices in networks”, Network Security, no. 4, pp. 8-15, 2018.
[13] M. Nicho, “A process model for implementing information systems security governance”, Information and computer security, vol. 26, no. 1, pp. 10-38, 2018.
[14] A. Irshad, M. Sher, S.A. Chaudhry, Q. Xie, S. Kumari, F. Wu, “An improved and secure chaotic map based authenticated key agreement in multi-server architecture”, Multimedia Tools and Applications, vol. 77, no. 1, pp. 1167-1204, 2018.
[15] S. Chatterjee, S. Roy, A.K. Das, S. Chattopadhyay, N, Kumar, A.V. Vasilakos, “Secure biometric-based authentication scheme using Chebyshev chaotic map for multi-server environment”, IEEE Transactions on Dependable and Secure Computing, vol. 15, no. 5, pp. 824-839, 2016.
[16] E. Kfoury, J. Saab, P. Younes, R. Achkar, “A self organizing map intrusion detection system for rpl protocol attacks”, International Journal of Interdisciplinary Telecommunications and Networking (IJITN), vol. 11, no. 1, pp. 30-43, 2019.
[17] T.V. Phan, M. Park, “Efficient distributed denial-of-service attack defense in SDN-based cloud”, IEEE Access, no. 7, pp. 18701-18714, 2019.
[18] U. Noor, Z. Anwar, A.W. Malik, S. Khan, S. Saleem, “A machine learning framework for investigating data breaches based on semantic analysis of adversary’s attack patterns in threat intelligence repositories”, Future Generation Computer Systems, vol. 95, no. 6, pp. 467-487, 2019.
[19] A.Ostad-Sharif, A. Babamohammadi, D. Abbsinezhad-Mood, M. Nikoghadam, “Efficient privacy-preserving authentication scheme for roaming consumer in global mobility networks”. International Journal of Communication Systems, vol. 32, no. 5, pp. e3904, 2019.
[20] D. Mishra, V. Kumar, D. Dharminder, S. Rana, “SFVCC: Chaotic map-based security framework for vehicular cloud computing”, IET Intelligent Transport Systems, Vol. 14, No. 4, pp.241-249, 2020.
[21] S. Kwon, H. Yoo, T. Shon, “IEEE 1815.1-Based power system security with bidirectional rnn-based network anomalous attack detection for cyber-physical system”, IEEE Access, no. 8, pp. 77572-77586, 2020.
[22] J. Jeong, S. Kwon, M.P. Hong, J. Kwak, T. Shon, “Adversarial attack-based security vulnerability verification using deep learning library for multimedia

video surveillance", Multimedia Tools and Applications, pp. 1-15, 2019.

[23] P. Harikrishna, A. Amuthan, "SDN-based DDoS attack mitigation scheme using convolution recursively enhanced self organizing maps", Sadhana - Academy Proceedings in Engineering Sciences, vol. 45, no. 1, pp. 104, 2020.

[24] J. Sengupta, S. Ruj, S.D. Bit, "A Comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT", Journal of Network and Computer Applications, vol. 149, no. 1, pp. 102481, 2020.

[25] S.R. Maniyath, V. Thanikaiselvan, "An efficient image encryption using deep neural network and chaotic map", Microprocessors and Microsystems, no. 77, pp. 103134, 2020.

Zhen Zhu, male, was born in southeastern China's Zhejiang Province in June 1984. He holds a bachelor's degree, and now works as an engineering in network and information security.

Guofei Chai, male, was born in southeastern China's Zhejiang Province in August 1986. He holds a doctor's degree, and now works as a lecturer on multi-agent systems.