AAG: A Model for Attack Behavior Judgment in CTF-style Cyber Security Training

Zimian Liu, Han Qiu, Junhu Zhu and Ziyi Zeng
PLA Strategic Support Force Information Engineering University
Zhengzhou, Henan Province, China
mmian1314@163.com

Abstract—**Capture the Flag (CTF) competitions become popular to engage people with the world of cybersecurity. In CTF-style cybersecurity training, judging trainees' attack behavior in terms of attack time, methods, and consequences is the key to know trainee ability and thus to provide more targeted training. Previous work focuses on consequences and lacks judgment on attack methods, which cannot provide enough information for trainers to assess trainee ability. To solve this problem, we propose a model called Attack Action Graph (AAG) to help judge trainees' behavior. By creating AAG, we characterize predictable and unpredictable attack methods and establish an association between attack methods and consequences. Then we update AAG based on feature detection technology to judge attack behavior. In the experiment based on a typical enterprise network, AAG helps us to achieve real-time judgment of all predicted attack behavior. We find that only 5% of the trainees have unpredicted attack behavior. After extracting the characteristics of unpredicted attack behavior and doing offline judgment, we find new attack methods and cheating behavior.**

Keywords-CTF-style cybersecurity training; attack behavior judgment; Attack Action Gragh

I. INTRODUCTION

Cyber attacks are rapidly increasing, more and more organizations conduct cybersecurity training to improve their members' skills of cyber defense[1]. These years, Capture the Flag (CTF) competitions become more and more popular[2]. CTF competitions can improve trainees' defense and attack skills and CTF-style cybersecurity training is a good way to help trainees do better in CTF competitions.

In CTF-style training, trainers assess trainee ability according to the flags they submit[9]. However, even if both two trainees get the same flag, their attack methods may be different, so trainers cannot know what attack methods trainees have already mastered. And if both two trainees cannot get the same flag, their actual abilities may also differ. Therefore, apart from how many flags a trainee gets, judging his attack behavior including both methods and consequences is beneficial to the trainers to assess the trainee's ability better. In some training, trainers usually ask trainees to write reports about their attack process, so they can know how trainees get flags[2]. This method takes too much time to do the judgment. What's more, it's a challenge for trainers to judge whether the content in the report is true.

There is no such work to judge attack behavior in CTF-style training, including both methods and consequences, in an efficient way. We can learn from other related work as follows.

To judge attack behavior, the key is to model attack behavior reasonably. At present, attack graph is a good model to describe attacks[3], which can well describe a series of attack paths to perform security analysis and evaluation on the target network. Attack tree[4] can reflect the various stages of the overall attack process. However, the above models can not describe the dynamic changes of the attack process, so they can not reflect the completion of the attack consequences.

In addition to the behavior model, we also need to identify attack methods automatically to judge behavior in an efficient way. Network behavior identification technology is mainly divided into two categories: behavior-based methods and feature-based methods[5]. Behavior-based methods[6] can identify known and unknown attacks, but it needs a large amount of data for training. The feature-based methods identify attack behavior by finding characteristics[5]. This method can be applied to the automatic identification of attack behaviors better, but they cannot identify the unexpected attack methods.

In this paper, we propose a model to help trainers judge trainees' attack behavior in CTF-style cybersecurity training, called Attack Action Graph (AAG). Then we propose an automatic judgment method based on AAG and implement an automatic behavior judgment system. Finally, we verify the effectiveness of our system by conducting a CTF-style cybersecurity training experiment of a typical enterprise network.

The rest of the paper is organized as follows. In the second section, we give the definition of our model—Attack Action Graph (AAG), and then propose an attack behavior judgment method based on the AAG. In the third section, we design and implement the attack behavior judgment system based on AAG, and then do an experiment to verify the effectiveness of our system. In the fourth section, we summarize our work.

II. ATTACK BEHAVIOR JUDGMENT BASED ON ATTACK ACTION GRAGH

In this section, we propose a model to describe attack behavior, called Attack Action Graph (AAG). Then, we give a behavior judgment method based on the AAG.

978-1-7281-0945-9/19/$31.00©2019 IEEE

A. Definition of Attack Action Gragh (AAG)

1) Definitions of attack scenario

Most cybersecurity training is based on specific attack scenarios preset by trainers. According to scenarios, trainers set several missions for trainees to complete the training. We define the elements in an attack scenario as follows,

a) Definition 1 (Node): A Node N is an entity in an attack scenario, such as a host or a server.

b) Definition 2 (Attack Mission): An Attack Mission M is a mission that trainees need to complete on a particular node. One of its attributes "stage of attack mission" indicates which stage of the entire attack process the mission is in.

c) Definition 3 (Attack Scenario Topology): Attack Scenario Topology (AST) is an undirected graph that describes the network connection relationship of nodes in the attack scenario.

2) Definitions of attack Action

Based on the attack scenario topology (AST), trainers can obtain possible attack methods according to the vulnerability information of the nodes. Usually, an attack method contains several attack actions, which are defined as follows,

a) Definition 4 (Meta Attack Action): A Meta Attack Action a indicates an attack that cannot be subdivided, such as network/vulnerability scan, SQL injection, and so on.

b) Definition 5 (Attack Vector)

$$v_n = [a_1, a_2, \cdots, (a_i^1, a_i^2, \cdots), \cdots]$$

Attack Vector v_{n} is an ordered set of meta attack actions of a specific attack method to complete the attack mission M_{n}. a_i indicates the i-th meta attack action of the attack method. If several meta attack actions can be performed in parallel, we use a_i^j to indicate the j-th meta attack action of the i-th step of the attack method.

c) Definition 6 (Attack Path)

$$p =< CM, PM, V >$$

An attack path p is a set of all possible attack vectors to an attack mission, where,

- CM (Current Mission): CM is the current mission which the attack vectors aim at.
- PM (Prepositive Mission): PM is a set of pre-missions of the current attack mission.
- V (Vectors): V is a set of attack vectors to complete CM.

3) Definition of Attack Action Gragh (AAG)

$$AAG =< M, P, S >$$

The Attack Action Graph is a directed graph with different states. Where,

a) M (Missions): M is a set of attack missions. It includes an initial empty mission M_0, as well as all attack missions in the training.There are two kinds of relationships between attack missions: pre-relationship and parallel relationship. An attack mission can only be executed after its pre-missions are completed. Attack missions that have a parallel relationship can be executed simultaneously.

b) P (Paths):P is a set of attack paths which are the edges of the AAG. The attack paths connect the attack missions and can indicate the relationships between attack missions.

c) S (State): S is the state of the AAG. It has three values, 0, 1, and 2. They respectively represent that AAG is before training, during training, and after training.

B. Behavior Judgment Based on AAG

We can use AAG to judge trainees' attack behavior in three steps: create AAG before the training, update AAG during the training, and review the AAG after the training. The flow chart of the behavior judgment method is shown in Fig 1.

Figure 1. Behavior judgment method based on AAG.

1) Create AAG before The Training

Before the training begins, trainers should create an AAG according to the attack scenario, and the state of the AAG is S=0. AAG can be created in four steps: create an attack scenario, set attack missions, design attack paths, and finally, generate AAG.

a) Create an attack scenario

According to different training purposes, trainers can create different attack scenarios. This step includes selecting nodes with specific vulnerability, creating network connections between nodes, and using AST to describe the attack scenario.

b) Set attack missions

According to the AST built in step *a*, trainers divide the entire training into multiple attack missions and determine the relationships between attack missions. Trainers can divide the

stages of the attack process by setting the stage of the attack mission, such as scanning, breaking, infiltration, and so on.

c) Design attack paths

For each attack mission, trainers set one or more feasible attack methods based on the information of the pre-attack missions and the vulnerabilities of the node.

d) Generate AAG

On the AST built in step *a*, trainers add attack paths among attack missions in the node. When all attack paths are added, the AAG is generated.

2) Update AAG during The Training

During the training, trainers use data collected from the attack scenario to update the AAG. The state of the AAG is S=1.The update of the AAG consists of two parts. One part is to update the completion of the attack missions by verifying flags submitted by trainees. The other part is to update the execution of predictable attack actions. We use a feature-based attack behavior detection method to analyze the traffic and log data generated in the scenario to determine whether a meta attack action is completed.

3) Review The AAG after The Training

After the training, trainers review each trainee's updated AAG, and the state of the AAG is S=2.

In most cases, trainees' attack methods are predictable. In this condition, the final update result of the AAG can be directly used as the judgment result of attack behavior. In rare cases, a trainee's AAG shows that an attack mission is completed, but it doesn't match the corresponding attack path successfully. At this time, trainers need to read the report submitted by the trainee to judge his actual attack behavior. If trainers find that the trainee takes an unexpected attack method, they should add a new attack path to the trainee's AAG.

III. System Implementation And Verification

In this section, we design and implement an automatic behavior judgment system based on AAG. To verify the effectiveness of our system, we do our experiment based on a typical enterprise network described in[10].

A. Framework Design And Implementation

The judgment system shown in Fig 2 is mainly composed of two major modules: a data processing module and a behavior judgment module. The data processing module collects data from the scenario, then stores and pre-processes the data for the use of behavior judgment. The behavior judgment module is to identify trainees' attack actions from the data and updates AAG. It also provides the function for trainers to review trainees' reports and modify AAG.

In the data processing module, the data collector is deployed in the scenario. We use a log collection framework ELK[7] to collect logs and a traffic collection tool Snort[8] to collect traffic data from the scenario. Raw data is sent to the scenario database and data processor at the same time. In the data processor, raw data is processed to facilitate the identification of the attack behavior.

In behavior judgment engine, the automatic identification engine identifies trainees' behavior according to the behavior characteristics in the behavior feature database and then updates AAG. After the training, trainers can review the reports in the report database to judge trainees' behavior offline, and then manually modify the AAG.

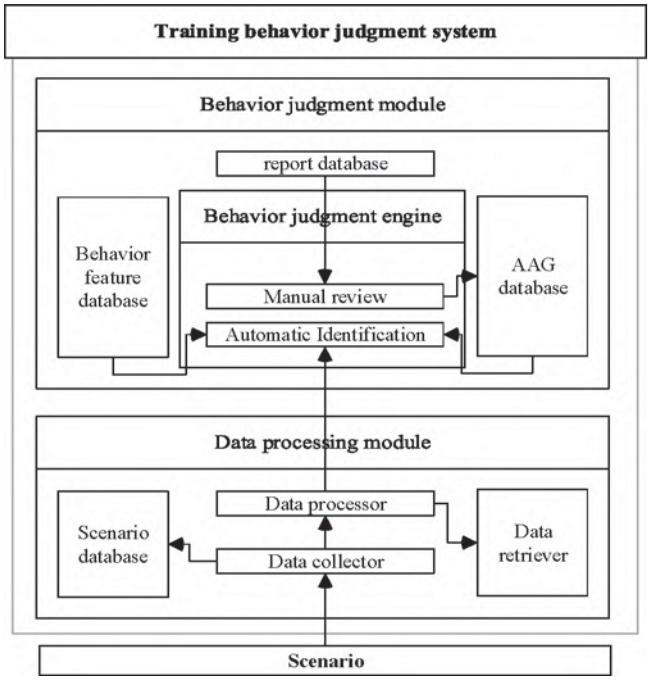

Figure 2. Framework of attack behavior judgment system.

B. Experimental Design And Verification

1) Training Scenario

We use CTF-style offline attack training based on a typical enterprise network in [10] to verify the effectiveness of our judgment system. 60 students participate in this training. We conduct this training on a network security training system based on the vSphere virtualization platform. During the training, we collect traffic and log data from each trainee's scenario through out-band transmission and in-band transmission. The network topology of the scenario is shown in Fig 3.

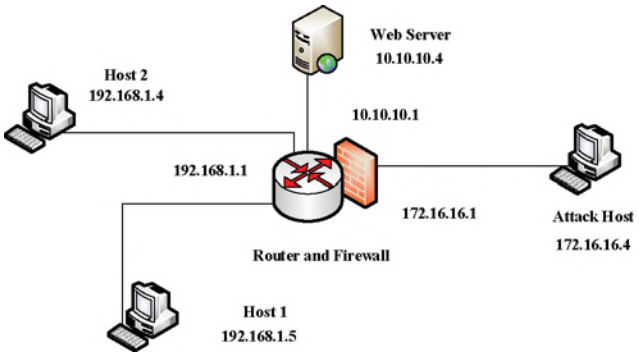

Figure 3. Scenario network topology.

The information of all the nodes in the scenario is shown in TABLE I.

TABLE I. HOST INFORMATION

Node Name	Node Seq	OS Type	Vulnerability
Attack Host	N_0	Windows7	NULL
Web Server	N_1	Windows Server 2003	CGI
Router	N_2	Ubuntu 14.04	SSH weak password
Host1	N_3	Windows7	NULL
Host2	N_4	Windows XP SP3	MS08-067

2) Buiding Initial AAG

Based on the scenario in [10], we divided the entire training into four missions, and the descriptions of each mission are shown in TABLE II.

TABLE II. ATTACK MISSIONS

Mission Seq	Description	Node	Stage
M_1	Get control of the web server	N_1	Target breakthrough
M_2	Get control of the router	N_2	Target breakthrough
M_3	Get control of host1	N_3	Intranet penetration
M_4	Get control of host2	N_4	Intranet penetration

According to the method given in [10], we created possible attack paths shown in TABLE III.

TABLE III. ATTACK PATHS

Attack Path Seq	CM	PM	V
p_1	M_1	$\{M_0\}$	$\{v_1\}$
p_2	M_2	$\{M_0\}$	$\{v_2\}$
p_3	M_3	$\{M_1, M_2\}$	$\{v_3\}$
p_4	M_4	$\{M_2\}$	$\{v_4\}$

The meta attack actions contained in the attack vectors are shown in TABLE IV.

TABLE IV. ATTACK VECTOR

Attack Vector Seq	Meta Attack Action Seq	Description of Meta Attack Action Seq	Stage	Attack Vector Seq	Meta Attack Action Seq	Description of Meta Attack Action Seq	Stage
v_1	a_1	Network, service, and vulnerability scanning	Information collecting	v_3	a_1	Network, service, and vulnerability scanning	Information collecting
	a_2	SQL injection	Target breakthrough		a_2	Firewall bypass	Intranet penetration
					a_3	Middleman attack	Intranet penetration
v_2	a_1	Network, service, and vulnerability scanning	Information collecting	v_4	a_1	Network, service, and vulnerability scanning	Information collecting
	a_2	Weak password attack	Target breakthrough		a_2	Firewall bypass	Intranet penetration
					a_3	MS08-067 exploit	Intranet penetration

Based on the above information, we generated the initial AAG of this training and show it in Fig 4.

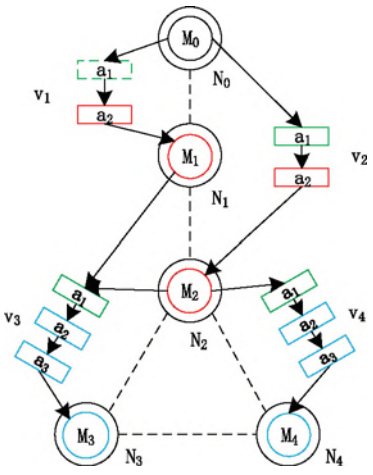

Figure 4. Initial AAG.

3) Experimental results analysis

After the training, we calculated the completion of attack actions and attack missions of all students by checking their updated AAG. Then we read their reports and verified the identification result. The automatic identification and verification results are shown in TABLE V. It shows that 54 students got all the flags, and among them, 51 students' AAG were automatically updated successfully and perfectly matched the initial AAG. Only 5% of the trainees have unpredicted attack behavior.

This result means that those 51 students referred to the textbooks and used the attack methods contained in the AAG preset by trainers, and all of them completed the whole training successfully. The other 3 students' AAG have abnormal matching conditions. We checked their AAG shown in Fig 5.(1) and found that they did not use the MS08-067 vulnerability but successfully obtained control of host 2. Then we read their reports to see how they got the flag. Two of them said they had used the EternalBlue vulnerability to obtain the control of host2 in their reports. We extracted the characteristics of this attack

action and verified it through the traffic data stored in the scenario database. We verified that they did take the correct method to complete the attack mission. Therefore, we manually add this attack path to their AAG. The new AAG is shown in Fig 5.(2).

TABLE V. ATTACK BEHAVIOR IDENTIFICATION RESULTS

Mission Seq	Mission Completion Number	Attack Vector Seq	Meta Attack Action Seq	Identification Number	Actual Completion Number	Automatic Identification Rate
M_1	60	v_1	a_1	60	60	100%
			a_2	60	60	100%
M_2	56	v_2	a_1	60	60	100%
			a_2	56	56	100%
M_3	54	v_3	a_1	56	56	100%
			a_2	55	55	100%
			a_3	54	54	100%
M_4	54	v_4	a_1	56	56	100%
			a_2	54	54	100%
			a_3	51	54	94.44%

Only 1 student did not mention any attack method to complete this mission in his report, so we judged that he hadn't completed this mission and had cheated in this training. This student later admitted that he had obtained the username and password of the host2 directly from others, and then got the FLAG.

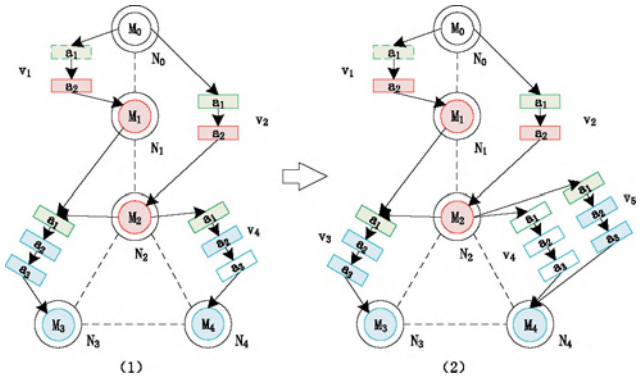

Figure 5. Add a new path to AAG.

AAG can tell trainers more information about trainees' attack behavior. For example, if a student didn't complete a mission, which stage of the attack process he was stuck in? In this training, 56 students completed mission M_1 and M_2, they all found host1 for mission M_3, but only 54 of them complete M_3. From AAG we can know that one of the students who failed to complete M_3 didn't complete meta action a_2, so he was stuck in the firewall. The other student, however, he successfully passed through the firewall and just failed to use the vulnerability to attack the host.

IV. CONCLUSION

In view of the shortcomings of most CTF training in assessing the ability of trainees, we propose a model called Attack Action Graph (AAG) to help judge attack behavior. Then we propose a training behavior judgment method based on AAG and implement an automatic judgment system. We use CTF-style training to verify the effectiveness of our judgment system. The experiment shows that AAG has three benefits for judging attack behavior: (1) It can give trainers more information, such as the attack methods and attack stages, to distinguish trainees' attack ability. (2) It can save trainers time for effective judgment by identify predictable attack methods automatically. (3) It also help trainers identify unexpected attack methods and cheating behavior.

REFERENCES

[1] Patriciu, V., & Furtuna, A. C. (2009). Guide for Designing Cyber Security Exercises 2 . The Need for a Uniform Structure. *Proceedings of the 8th WSEAS International Conference on E-Activities and Information Security and Privacy*, 172–177.

[2] Mirkovic J , Dark M , Du W , et al. Evaluating Cybersecurity Education Interventions: Three Case Studies[J]. IEEE Security & Privacy, 2015, 13(3):63-69.

[3] Kotenko I , Chechulin A . A Cyber Attack Modeling and Impact Assessment framework[C]// International Conference on Cyber Conflict. IEEE, 2013.

[4] Jijun L . An Attack Tree Approach for Network Intrusion Modeling[J]. Computer Engineering & Applications, 2003.

[5] Mitchell R , Chen I R . A Survey of Intrusion Detection Techniques for Cyber-Physical Systems[J]. Computers & Security, 2014, 12(4):405-418.

[6] Lauf A P , Peters R A , Robinson W H . Embedded Intelligent Intrusion Detection: A Behavior-Based Approach[C]// International Conference on Advanced Information Networking & Applications Workshops. IEEE, 2007.

[7] Sachdeva G S. The ELK Stack in Production[M]// Practical ELK Stack. 2017.

[8] Olanrewaju R F, Zahir K A K, Asnawi A L, et al. Modelling of intelligent intrusion detection system: making a case for snort[C]// Wireless Sensors. 2018.

[9] 邱菡, 朱俊虎, 李玉峰, et al. 基于三段式课堂的大学实验课程教学设计与实践[J]. 计算机教育, 2017(11):150-153.

[10] 王清贤,朱俊虎,邱菡. 网络安全实验教程[M]. 电子工业出版社, 2016.