JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 39, 167-181 (2023)
DOI: 10.6688/JISE.202301_39(1).0010

Analysis of *SQL* Injection Based on Petri Net in Wireless Network

YI-CHUAN WANG[1,2], GUI-LING ZHANG[1] AND YA-LING ZHANG[1,+]
[1]*School of Computer Science and Engineering*
[2]*Shaanxi Key Laboratory for Network Computing and Security*
Xi'an University of Technology
Xi'an, 710000 P.R. China
E-mail: {chuan; ylzhang}@xaut.edu.cn; 2167721368@qq.com

SQL injection is becoming more frequent and is urgent to be studied the defense mechanism. In this paper, a fine-grained Petri Net (*PN*) model is proposed to describe the process of *SQL* Boolean injection and *SQL* time blind injection. The results show that the speed of *PN* model simulating *SQL* injection can be scaled, such as 10 times the speed. It is conducive to faster-than-real-time simulation and prediction of unknown vulnerabilities when the injection speed is fast. It is good for fine-grained analysis of the attack and establish a patch model when the injection speed is slow. The modeling and simulation in this paper provide a theoretical guarantee for the occurrence mechanism of vulnerabilities and the discovery of unknown vulnerabilities.

Keywords: *SQL* injection, wireless network, network security, petri net, fine-grained modeling

1. INTRODUCTION

With the rapid development of wireless network, security events occur frequently in wireless networks. At present, a variety of low-cost radio technologies are used to realize wireless communication. The easy access of these new radio technologies brings huge security problems.

SQL injection attack [1] is one of the most harmful attacks in wireless networks. With the increasing development of the network and the increasing complexity of network attack methods, *SQL* injection vulnerability is exploited by attackers to obtain sensitive information related to the database. An approach to identify and mitigate *SQL* injection attacks was proposed by W. H. Another [2]. Communication devices are connected to each other through cables, and nodes without direct connection cannot access the network for illegal activities in wired network. However, due to the broadcast characteristics of wireless network and the openness of interface, both authorized users and illegal users can access it. The open environment makes wireless networks more vulnerable to malicious attacks. It includes passive eavesdropping for data interception and active interference for legitimate transmission. Therefore, *SQL* injection attacks are more likely to occur in wireless networks, leading to the disclosure of sensitive information. This paper aims to study the *SQL* injection attack caused by the inherent openness of wireless network, and design an effective defense mechanism to improve the security of wireless network. For the serious *SQL* injection attacks in wireless networks, existing studies use random forest or machine learning strategy modeling to detect *SQL* injection attacks, but cannot predict unknown attacks,

Received October 1, 2021; revised November 14, 2021 & February 27, 2022; accepted March 9, 2022.
Communicated by Xiaohong Jiang.
[+] Corresponding author.

and the description granularity is still too macro. In recent years, the research idea of combining network attack and defense with Petri Net has been gradually proposed [3-5], which is more micro compared with the traditional modeling idea. Therefore, based on the micro fine-grained advantage of Petri Net in describing the state, the *SQL* injection process is modeled in this paper in wireless networks. The effectiveness of *PN* model of *SQL* injection attack is proved by qualitative analysis and quantitative analysis in wireless networks. Furthermore, according to the conflict in *PN* model, the patch model corresponding to *SQL* injection is established to prevent *SQL* injection attack and maintain the security of user sensitive information.

The remainder of this paper is organized as follows. Section 2 introduces the related work of detecting *SQL* injection attacks in wireless network. Section 3 introduces the modeling of *SQL* injection based on *PN* in wireless network. Section 4 makes a fine-grained analysis of the *PN* model of *SQL* injection and establishes the patch model. Section 5 simulates the *PN* model and makes qualitative and quantitative analysis. Section 6 summarizes the relevant research of this paper and puts forward the future work.

2. RELATED WORK

This paper mainly models and analyzes the micro process of network attack in wireless network. In the following related work, we will classify and elaborate the existing research work on attack behavior modeling in wireless networks and *SQL* injection attack modeling.

A. Existing research work on modeling attacks in wireless networks

Wireless network is a network that can realize the interconnection of various communication devices without wiring. This concept was proposed by Y. Zou [6]. It generally adopts *OSI* protocol architecture, including application layer, transmission layer, network layer, *MAC* layer and physical layer. Different layers support different protocols and show different security vulnerabilities. Wireless networks are similar to wired networks, and the main difference is the physical layer and *MAC* layer. The security threats and vulnerabilities related to these protocols are protected on each layer.

The existing research on wireless network attack modeling and analysis mainly includes three aspects: using random forest to establish model, using machine learning strategy and establishing *SEIRD* model. T. Le *et al.* [7] proposed Random Forest model to detect attacks in wireless network. The model can only detect known attacks, but cannot predict unknown attacks. This paper proposes a method of using Petri Net to establish attack model, which can not only detect known attacks, but also predict unknown attacks. E. M. D. L. Pinto *et al.* [8] propose a new machine learning based strategy to detect spoofing attacks in wireless sensor networks (*WSNs*). Compared with other methods, the performance of detecting attacks in this paper is improved by about 10%, but no measures are taken to prevent attacks. The method of using Petri Net to establish attack model proposed in this paper establishes the defense patch model while establishing the attack model, which provides a higher guarantee for network security. S. R. Biswal *et al.* [9] proposed a Susceptible-Exposed-Infectious-Recovered-Dead (*SEIRD*) model to understand the dynamics of malware propagation in WSN. The *SEIRD* model focuses on early detection of malicious signals in the network and corresponding application security mechanisms to eliminate them. Compared with the traditional model, it provides an improved security mechanism, but it cannot realize faster-than-real-time simulation and predict unknown malicious signals. However, the *PN* model established in this paper can not only provide the defense model of the attack but also predict the unknown attack. The paper has high innovation and research value compared with the existing relevant research in recent years. However, due to the problems of transmission delay in wireless networks, there are some challenges in the modeling and analysis of this paper. This paper also analyzes the impact of time delay on transmission efficiency.

B. Existing research work on modeling and analyzing *SQL* injection

SQL injection mainly attacks data-driven applications by inserting some malicious *SQL* statements to obtain sensitive information of databases and users, and try to access legitimate websites. Y. Zou *et al.* [10] mentioned that *SQL* injection is an *HTTP* attack in the application layer of *OSI* layered protocol architecture. This section mainly discusses the modeling of *SQL* injection attack based on pattern neural network, the existing research work of *ATAR* model and *RESNET* of *SQL* injection attack, and analyzes the defects of the existing *SQL* injection attack model, which leads to a new idea of modeling *SQL* attack behavior in wireless network. M. B. A *et al.* [11] proposed a pattern based neural network model to detect *SQL* injection attacks efficiently, which has higher accuracy than similar models. However, the patch model is not established, and the effectiveness of the *SQL* injection attack detection model are not proved in wireless network. H. Gao *et al.* [12] proposed an *ATTAR* model to detect the syntax and behavior characteristics of *SQL* injection attacks by using language pattern and access behavior mining. However, the description of attack process in *ATAR* model is not micro enough and the corresponding patch model is not established. Sangeeta *et al.* [13] proposed a *RESNET* model to automatically learn and detect *SQL* injection attacks, which can effectively identify different types of *SQL* injection attacks compared with similar models. However, the effectiveness of *SQL* injection in wireless network has not been proved. Based on the defects of the existing *SQL* injection attack model, the *PN* model of *SQL* injection attack based on Petri Net pays more attention to describing the micro process of the attack. Furthermore, this paper establishes an *SQL* injection patch model based on Petri Net to ensure the security of the network. Finally, it verifies the availability and effectiveness of *PN* model in the wireless network, and can predict unknown attacks.

C. The *PN* model modeling strategy proposed in this paper

PN is composed of ***Place***, ***Transition***, and direct arc. *PN* is used to describe states and changes. The circle represents $\mathcal{P}$ and is used to describe the state. The square represents $\mathcal{T}$ to describe the operation. *PN* is defined as follows.

Definition 1: A triple $PN = (\mathcal{P}, \mathcal{T}, \mathcal{F})$ is called a Petri Net if it satisfies the following conditions.

(1) $\mathcal{P}$ is the finite set of ***Place*** and $\mathcal{T}$ is the finite set of ***Transition***.
(2) Among them, $\mathcal{P} \neq \phi$, $\mathcal{T} \neq \phi$, and $\mathcal{P} \cap \mathcal{T} \neq \phi$.
(3) $\mathcal{F} = (\mathcal{P} \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P})$ represents the flow relationship of *PN*.

The paper study the process of *SQL* injection using *PN* and analyze its advantages in wireless network. It solves the macro problems of the traditional model, verifies the availability of the model in the wireless network, makes up for the limitations of modeling in wired network and establishes a patch model to defend and predict unknown vulnerabilities.

3. MODELING *SQL* INJECTION IN WIRELESS NETWORK

The main attack methods of a complete *SQL* injection attack are *SQL* Boolean injection and *SQL* time blind injection. *SQL* Boolean injection determines whether there is an injection point in the website according to whether the return value of the page is true or false. If there are injection points in the website, we can further use *SQL* time blind injection to construct time blind injection function, and then judge the length of database name and other sensitive information to carry out power raising attack. In order to analyze the process of *SQL* injection attack in wireless networks, *PN* models are established in this study for two *SQL* injection methods to find the attacks and vulnerabilities in wireless networks.

3.1 *PN* Modeling Strategy

The mapping relationship during *PN* modeling is shown in Table 1.

Table 1. Mapping relationship between basic elements of *PN* model.

PN model	*SQL* injection attack
$\mathcal{P}$	State
$\mathcal{T}$	Action
$\mathcal{F}$	State transform

In the research of formal modeling in this paper, $\mathcal{P}$ is used to represent various system states or vulnerability states in *PN* model, such as legal of input statement, *SQL* injection, illegal state of input statement, *etc.* $\mathcal{T}$ is used to represent the operation of *SQL* injection vulnerability, such as *SQL* injection attack, executing illegal *URL* statement, injection attack success, *etc.* $\mathcal{T}$ has the property of describing micro execution operation. Therefore, in a specific attack and defense, those related to state are represented by $\mathcal{P}$ and operations are represented by $\mathcal{T}$. The token value can be set according to the vulnerability exploitation in $\mathcal{P}$, indicating the number of resources the system has or the number of attempts to penetrate in the wireless network.

3.2 *PN* Modeling of *SQL* Boolean Injection Attack

According to *PN* modeling strategy, the attack process is divided into $\mathcal{P}$ set and $\mathcal{T}$ set to establish the *PN* model of *SQL* Boolean injection. $\mathcal{P}$ mainly include illegal construction state, Boolean injection state, *etc.* They are represented by circles in the model, in which the security state and normal state are the legitimate states of the system, while the illegal splicing statement and injected state are caused by the illegal operation of the malicious attacker. $\mathcal{T}$ mainly include constructing illegal statements and Boolean injection operations, which are represented by squares. A specific token value can be set for the initial $\mathcal{P}$ to indicate the number of resources owned by the system or the number of attempts to inject.

The model of *SQL* Boolean injection is shown in Fig. 1. The description of $\mathcal{P}$ and $\mathcal{T}$ in the *SQL* Boolean injection model is shown in Tables 2 and 3.

Table 2. Description of $\mathcal{P}$.

$\mathcal{P}$	Status
P_0	Normal
P_1	Legal splicing
P_2	Splicing statement
P_3	Illegal splicing
P_4	Splicing completion
P_5	Legal
P_6	Illegal
P_7	Injection
P_8	Security
P_9	Vulnerability

Table 3. Description of $\mathcal{T}$.

$\mathcal{T}$	Operation
T_0	Boolean injection start
T_1	Construct legal statements
T_2	Construct illegal statement
T_3	Send attack code
T_4	Failed
T_5	Succeed
T_{12}	System reset

In Fig. 1, P_3 is an illegal state, which mainly carries out illegal *URL* splicing for vulnerable statements in the database. T_3 is an operation to start injection attack. P_9 is an insecurity state successfully injected, indicating that the website has been successfully attacked. In the whole model, the states like P_0, P_5 and P_8 are secure. These states are allowed in the normal operation of the website. While the states like P_3 and P_9 are illegal. These states are prohibited in the normal operation of the website. Set the token value of 50 in P_0 to indicate that the model is injected 50 times in the wireless network. Analyze the *SQL* injection and verify the effectiveness and correctness of the method and model in wireless network.

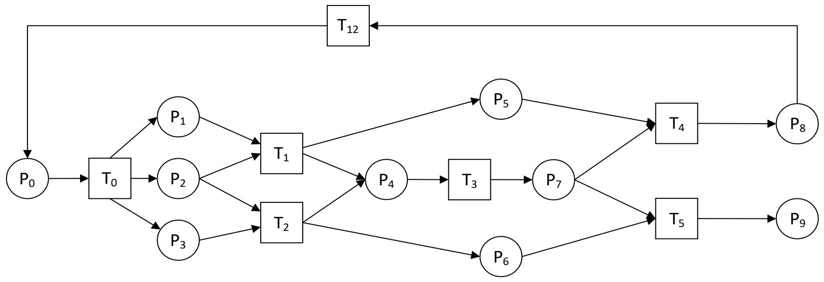

Fig. 1. *SQL* Boolean injection model diagram.

3.3 *PN* Modeling of *SQL* Time Blind Injection Attack

The *SQL* time blind injection is established. The main $\mathcal{P}$ include insecurity state, illegally constructed *URL* statement state. Among them, security state is the state of legal existence of the system. The main $\mathcal{T}$ are illegal construction of statements and *SQL* time blind attacks. The token value can be set to indicate the number of resources owned by the system website or the number of attempts to inject.

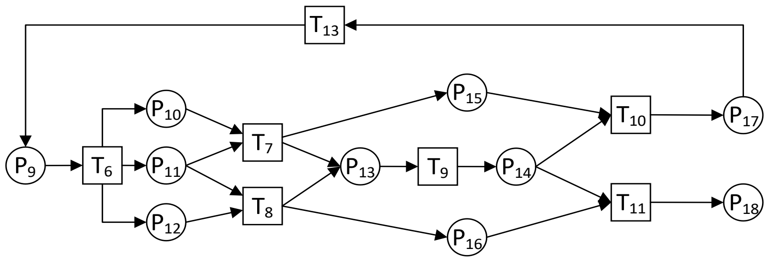

Fig. 2. Diagram of *SQL* time blind injection attack model in wireless network.

The model of *SQL* time blind injection attack is shown in Fig. 2. The description of $\mathcal{P}$ and $\mathcal{T}$ is shown in Tables 4 and 5. P_{12} is an illegal state according to the *PN* model. The execution time of statements in the database is delayed by constructing the *sleep ()* function, resulting in illegal users getting the sensitive information of the database. T_9 is an operation that illegally starts *SQL* time blind injection attack, which is a crucial step in *SQL* time blind injection. P_{18} is an insecure state successfully injected by *SQL* time blind injection attack, indicating that the website has been successfully attacked. In the whole model, the states like P_{15} and P_{17} are the normal states allowed in the website, and the states like P_{12} and P_{18} are the illegal states. Set the token value of 50 in P_9 to indicate that the model simulates *SQL* time blind injection for 50 times. Analyze the *SQL* time blind injection and verify the effectiveness and correctness of the method and model. We can find out whether there is a risk in the *PN* model through whether there is a collision when the model is running, and locate the location of the risk through the unsafe state to prepare for the establishment of the patch model below.

Table 4. Description of $\mathcal{P}$.

$\mathcal{P}$	Status
P_9	Insecure
P_{10}	Legally constructed
P_{11}	Construct statement
P_{12}	Illegally constructed
P_{13}	Construction completes
P_{14}	Blind
P_{15}	Legal
P_{16}	Illegal
P_{17}	Security
P_{18}	Vulnerability

Table 5. Description of $\mathcal{T}$.

$\mathcal{T}$	Operation
T_6	Time blind
T_7	Legally constructed
T_8	Illegally constructed
T_9	Time blind
T_{10}	Injection failed
T_{11}	Injected successfully

Wireless network does not need to be physically connected through cables, which reduces the cost but increases the network delay. Therefore, it will cause time delay in the process of *SQL* time blind injection. The formula for calculating the blind injection time of *SQL* time is shown in Eq. (1):

$$T = T_s + T_1 + T_d, \tag{1}$$

where T represents the time spent on *SQL* time blind injection attack. T_s represents the time spent on executing blind injection function. T_l represents the time spent on page loading. T_d represents the network transmission delay. It is also the biggest difference factor affecting the attack time between wired and wireless networks. The calculation method of *SQL* time blind injection time is the same in wireless network and wired network, and Eq. (1) is adopted for both.

We can get Eqs. (2) and (3) as follows,

$$T - T_s = \delta \tag{2}$$

$$T - T_s = T_l + T_d \tag{3}$$

where δ represents the difference between the total time spent on blind injection attack. Eq. (4) can be derived as follows:

$$\delta = T_1 + T_d \tag{4}$$

where the confidence interval of δ is $\delta \in (\delta_0, \delta_m)$ and δ_0 represents the minimum value of δ and δ_m represents the maximum value of δ.

The probability of success of *SQL* time blind injection attack is recorded as P. The probability of success of the itch attack is recorded as P_i. The *SQL* time blind injection attack injects different times according to different settings of blind injection function parameters. The probability formula of successful injection is shown in Eq. (5):

$$P = \prod_{1}^{n} P_i \cdot \tag{5}$$

Here, the *PN* model of *SQL* time blind injection is established, and the formal analysis of blind injection attack time is helpful to explore the success of blind injection attack and its influencing factors in wireless network.

4. ANALYSIS OF *SQL* INJECTION MODEL

This section makes a code level analysis of the model established above. The descriptions of $\mathcal{P}$ and $\mathcal{T}$ are deeply analyzed to explore the corresponding relationship between each node in the model and *SQL* injection statements. Furthermore, the corresponding patch model is established for the two methods of *SQL* injection to prevent attacks.

4.1 Comprehensive Model Diagram of *SQL* Injection Attack

Once *SQL* Boolean injection finds the injection point, *SQL* time blind injection can be used to further mine the sensitive information related to the database. Based on the relationship between the two injection methods and the above modeling strategy, a comprehensive model is established in wireless network as shown in Fig. 3.

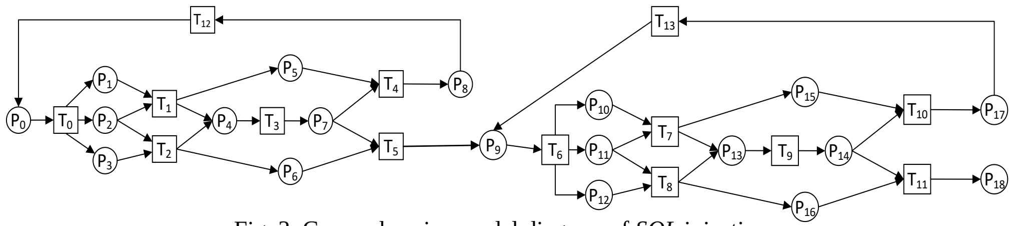

Fig. 3. Comprehensive model diagram of *SQL* injection.

In Fig. 3, the descriptions of $\mathcal{P}$ and $\mathcal{T}$ is the same as above. The construction code of the injection point in the website obtained by *SQL* Boolean injection is shown in Table 6. The simulation experiment uses the open source SQLi-labs shooting range practice experiment platform provided by the cloud performance website. The website is "https://www.yunyansec.com/#/experiment/security/". For the injection point, the *SQL* time blind injection is further used to construct the blind injection function, and the database length information is obtained through the background database processing. The codes are shown in Table 7.

Table 6. *SQL* Boolean injection attack code.

Type of attack	Attack Code
SQL Boolean injection attack	http://af5fded8.yunyansec.com/Less-8/?id=1

Table 7. *SQL* time blind injection attack code.

Type of attack	Attack Code
SQL time blind injection attack	http://af5fded8.yunyansec.com/Less-9/?id=1'and if (length (database ())>8, sleep (10), 0) --+

In Fig. 3, the left side of P_9 is the modeling of *SQL* Boolean injection and the right part is *SQL* Boolean blind injection. The *SQL* injection attack model in wireless network comes from the real attack and is simulated in the target aircraft. Therefore, the modeling in this paper has a strong fit with *SQL* injection code and actual attack and defense.

According to the attack code of *SQL* time blind injection, we can know that the blind injection function is the main factor leading to the success of the attack. When time blind injection is carried out in wireless networks, it can be divided into three cases according to different injection times returned and injection results. They are the success of the attack when the attack time is extended according to the parameter rules of the blind note function, the failure of the attack caused by the irregular extension of the attack time, and the success of the attack. As shown in in Eq. (1), the total time after successful attack is T. the reliability of *TCP* protocol in data transmission is affected by checksum, serial number, timeout retransmission mechanism, acknowledgement response mechanism, flow control, congestion control and other factors, considering T_d influence of *TCP*, the process of reliable data transmission of *TCP* protocol is analyzed.

Suppose that the time parameters of the blind injection function are $Ts_1 = 10$, $Ts_2 = 20$ and $Ts_3 = 5$ respectively, and the total time after the *SQL* blind injection attack is T_1, T_2 and T_3 respectively. According to in Eq. (1). If $(T_1 - Ts_1) / Ts_1 = (T_2 - Ts_2) / Ts_2 = (T_3 - Ts_3)$

$/ Ts_3$. This indicates that δ is relatively stable, SQL time blind injection in wireless network succeeded. If $(T_1 - Ts_1) / Ts_1 \neq (T_2 - Ts_2) / Ts_2 \neq (T_3 - Ts_3) / Ts_3$ and there is no rule in the change of δ. It can be preliminarily determined as two cases.

Case 1: The user mechanism may be honeypot or blind box mechanism. At this time, although the time shown by T is not much different from the time it should take for the attack to succeed, it is possible that the delay caused by the wireless network delay is too large, resulting in the increase of T_d in Eq. (1) and the total time. Because the performance of wireless network in transmission is good and bad, the total attack time increases, which is not the success of time blind injection. By increasing the number of injections, we can judge whether the increase of time is caused by network transmission delay. According to Eq. (5), assuming that the probability of successful TS_1 attack is P_1, assuming that the probability of successful TS_2 attack is P_2, assuming that the probability of successful TS_3 attack is P_3, then the probability of success of each *SQL* time blind injection attack is $P_1 \cdot P_2 \cdot P_3$. With the increase of injection times, if the change trend of total time is still the same, the more accurate the probability of attack success is, the smaller the probability of being cheated is.

Case 2: The time blind injection is successful. Calculate the confidence interval range for δ from $\delta \in (\delta_0, \delta_m)$ above. The confidence level is assumed to be Q. E_{Td} is the mean of transmission delay. n represents the number of injection attempts. The value range of confidence level is $Q \in (0, 1)$, and the specific value is determined according to the mean value and trust level. The specific confidence interval is: $(E_{Td} - \frac{1}{\sqrt{n}} z_{(1-Q)}, E_{Td} + \frac{1}{\sqrt{n}} z_{(1-Q)})$. If δ is within the confidence interval, the test is successful, δ changes within the normal range, and *SQL* time blind injection is successful. If δ is not within the confidence interval, the test fails and δ does not change within the normal range. Therefore, it is the increase of attack time caused by wireless network delay rather than blind injection success. When the delay increases all the time in the wireless network, it may lead to timeout re-transmission. If the delay is large enough in the wireless network, it may lead to the failure of the attacked target machine.

4.2 *SQL* Boolean Injection Attack Defense Patch Model

In this section, the *SQL* Boolean injection *PN* patch model is established in the wireless networks. A vulnerability is a defect in a website or system. An attacker can exploit the vulnerability to attack the website. A patch is a program that fixes a defect. For the *SQL* Boolean injection and time blind injection vulnerabilities in network, some researchers have proposed a patch method. This section aims to patch the security risks of the attack model established above and establish a patch model. The *SQL* injection vulnerability is due to the lack of strict legitimacy verification when executing statements in the database. Therefore, the design idea of *SQL* Boolean injection patch model is to add hash verification when executing *URL* statements in the background database to judge whether the hash corresponding to the *URL* is in the hash library. The patch model is shown in Fig. 4. Compared with the attack model, the description of adding $\mathcal{P}$ and $\mathcal{T}$ in the patch model is shown in Tables 8 and 9.

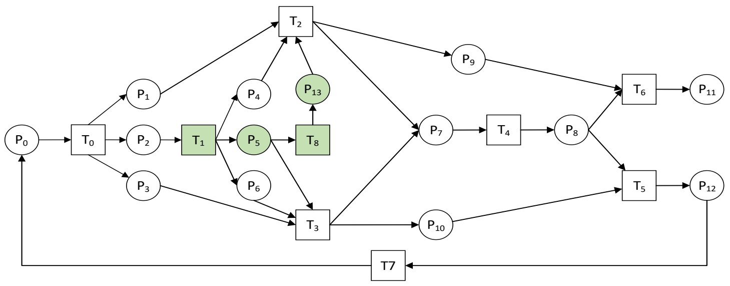

Fig. 4. *SQL* Boolean injection patch model diagram.

Table 8. Description of $\mathcal{P}$.

$\mathcal{P}$	Status
P_5	Hash match
P_{13}	Exception hash match

Table 9. Description of $\mathcal{T}$.

$\mathcal{T}$	Operation
T_1	Hash consistency matching
T_8	Detect abnormal hash match

T_1, T_8, P_5 and P_{13} are added to the patch model based on the *SQL* injection attack model. T_1 indicates hash verification, and P_5 indicates that the system is in hash matching state. If the hash matching fails, the system will prevent the execution of illegal construction statements and return to the 404. It is conducive to protecting the legitimate information of users. Set the token value to 50 in P_0, indicating that the model attempts to simulate *SQL* Boolean injection 50 times to verify the effectiveness of hash verification method and the correctness of patch model.

T_8 and P_{13} detect and process the matched exception hash. The storage capacity of P_{13} is 0. When an abnormal hash flows to this path, the token value cannot be stored in P_{13}, resulting in model collision and packet discarding, and resources cannot flow into subsequent repositories. Illegal injections are blocked in the patch model, and legitimate attempts will enter P_{12}. Therefore, the patch model has a significant defense effect.

4.3 *SQL* Time Blind Injection Patch Model

This section establishes the patch model of *SQL* time blind injection. When the constructed *URL* is correct, the blind injection function *sleep ()* will work. The patch model adds filter blind injection function mechanism and is shown in Fig. 5 and the description of adding $\mathcal{P}$ and $\mathcal{T}$ is shown in Tables 10 and 11.

Table 10. Description of $\mathcal{P}$.

$\mathcal{P}$	Status
P_{10}	Filtering
P_{11}	Capacity Zero

Table 11. Description of $\mathcal{T}$.

$\mathcal{T}$	Operation
T_7	Matching database
T_8	Detecting anomalies

T_7 and T_8 and the P_{10} and P_{11} are added restrictions based on the time blind injection attack model. T_7 and T_8 represent the execution of the statements corresponding to the *URL*, and detect the functions leading to blind injection such as exception filtering *sleep ()* function, so as to avoid the extension of network running time caused by illegal injection. The capacity of P_{11} is 0. Once collision is detected, the token will not flow after P_{11}.

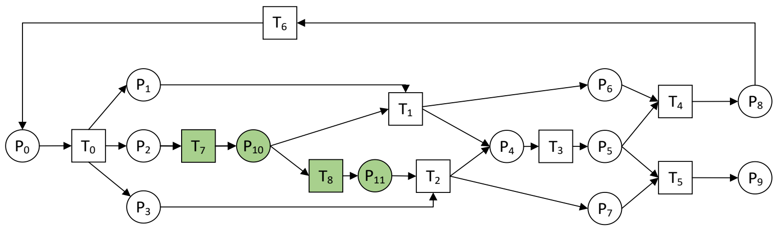

Fig. 5. The diagram of *SQL* time blind injection patch model in wireless network.

This paper studies the *PN* model of *SQL* injection attack, and the patch model is established according to its risk location. At the same time, the *PN* model is established for the existing patch code, and it is found that it is consistent with the patch model established in the paper. In the future research, for the attack without successfully establishing the patch, we will study the *PN* structure of the attack, find the risk location, establish the patch model, and finally get the patch code.

5. EXPERIMENTAL ANALYSIS

5.1 Experimental Environment

The implementation environment configuration is shown in Table 12.

Table 12. The environment configuration.

Environment	Edition
CPU	i5-5200U CPU 64bit operating system
Tina tools	V3.6.0
Hackbar	V2.3.1

5.2 Simulate *SQL* Injection Attack

The purpose of this study is to explore the efficiency of *SQL* injection attack and to analyze the attack process in fine granularity in wireless network. Taking Boolean injection as an example, the time spent on 50 injections is analyzed. The time spent is shown in Table 13.

As shown in Table13, the *PN* model established in this paper has advantages in injection time. The model simulation injection time increases with the increase of firing duration from 1 to 10, and the injection speed decreases, which is conducive to fine-grained description of the injection process and details. When the firing duration is 10, the injection time increases, but it is still shorter than the target injection time. The *PN* model can quickly predict attacks in a short time, and fine-grained analyze the essence of attacks and establish a patch model in a long time. The modeling method has advantages, which can quickly find vulnerabilities after the attack, predict unknown vulnerabilities and avoid various risks.

Table 13. Key time comparison table of *SQL* Boolean injection.

SQL Model	*SQL* injection attack	*PN* model	
		firing duration=1	firing duration=10
Total Time(s)	30.38	1.60	2.74

Taking Boolean injection as an example, the total time spent on 50 injections is analyzed. The time spent in reproducing the *SQL* time blind injection attack in the shooting range and simulating the running time by using *Boolean* injection *PN* model is shown in Table 14 in the wireless network:

Table 14. Key time comparison table of *SQL* Boolean injection in wireless network.

SQL Model	*SQL* injection attack	*PN* model	
		firing duration=1	firing duration=10
Total time(s)	32.35	3.63	4.78

According to the Table 14, the time of *SQL* injection is slightly longer than that in the normal *SQL* injection. The effectiveness of the model in analyzing the penetration process and detecting vulnerabilities will not be affected. It is proved that the *SQL* injection model established by *PN* has strong availability in the wireless network and plays an important role in ensuring network security in the wireless network.

Normal *SQL* injection analysis using *PN* and *SQL* injection analysis in wireless network. The injection time of 10 times, 20 times, 30 times, 40 times and 50 times are counted respectively. The firing duration of *PN* model is 1 and 10 respectively. Compare the running time with the generated histogram. The time of 10, 20, 30, 30, 40 and 50 simulated *SQL* injections under normal injection and wireless network is shown in Figs. 7 and 8.

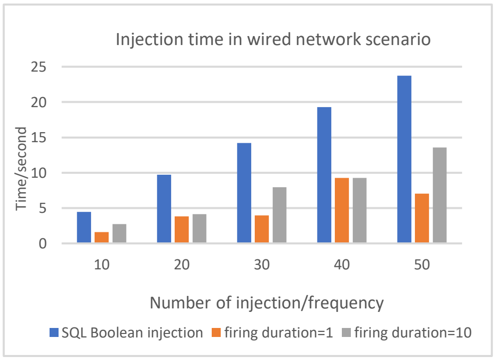

Fig. 7. Time of Boolean injection.

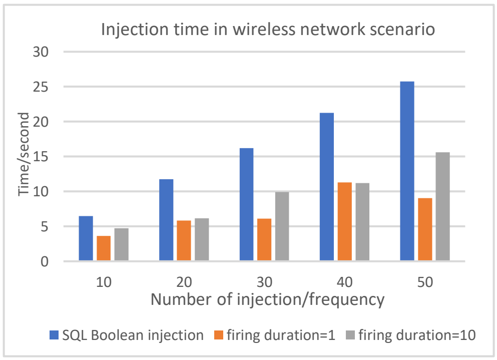

Fig. 8. Time of Boolean injection in wireless.

As can be seen from Figs. 7 and 8, the *SQL* injection time increases with the increase of injection attempts. Throughout the whole experimental process, *SQL* injection takes longer time than *PN* model simulation injection. In the wireless network, *PN* has the same advantages as above. When the firing duration is large, it is conducive to fine-grained analysis of the penetration process and establish the patch model. When the firing duration is small, it is conducive to faster-than-real-time simulation and prediction of unknown vulnerabilities quickly. The experimental results show that using *PN* model to simulate *SQL* injection attack in wireless network is about 2 seconds longer than using *PN* direct injection, but the advantages of faster-than-real-time simulation and *PN* model to simulate injection of fine-grained features are not affected. Therefore, the *PN* model established in this paper is effective in wireless network.

The *PN* model of *SQL* injection attack established is qualitatively analyzed to prove the availability and effectiveness of the model established in this paper in the wireless network in the wireless network. We use √ to denote the model has the functions listed in the table and × to denote it does not as show in Table 15.

Table 15. Qualitative analysis table.

SQL model	Code level analysis	Patched model	Automatic operation	clear state	Prediction unknown	Applicable wireless network
M. B. A *et al.* [11]	×	×	×	√	×	×
H. Gao *et al.* [12]	×	×	×	×	×	×
Sangeeta *et al.* [13]	×	×	√	×	×	×
Petri model	√	√	√	√	√	√

In Table 15, A fine-grained qualitative analysis is made for several typical *SQL* injection models. The *SQL* injection attack model proposed by M. B. A *et al.* [11] has high accuracy, but there is no patch model, and whether it is suitable for wireless network is unknown. The *ATTAR* model proposed by H. Gao *et al.* [12] does not analyze the attack process in fine-grained from the perspective of code, and it is also not suitable for wireless network. The *RESNET* model proposed by Sangeeta *et al.* [13] has the advantage of automatically detecting *SQL* injection attacks, but it does not propose a defense model. On the contrary, this model aims to describe the attack process from a micro perspective and establish the corresponding patch model to ensure the security of the network. More importantly, it verifies the availability and effectiveness of *PN* model in wireless network, and can predict unknown attacks. The details of *SQL* injection are extended to make up for the shortcomings of existing methods.

6. CONCLUSIONS

In this paper, *SQL* injection is modeled by mapping to $\mathcal{P}$ and $\mathcal{T}$ of Petri Net The specific location of security risk can be judged with the *PN* model. The results show that the speed of *PN* model simulating *SQL* injection can be scaled, such as 10 times the speed. It is conducive to faster-than-real-time simulation and rapid prediction of unknown vulnerabilities when the injection speed is fast. It is conducive to fine-grained analysis of the attack process, establish a patch model and protect network security when the injection speed is slow. In the future, we plan to add the probability to the *PN* model to predict the probability of unknown vulnerabilities. Furthermore, we establish the patch model for the existing new vulnerabilities and reverse the patch code.

ACKNOWLEDGMENTS

This research work is supported by the National Natural Science Founds of China (62072368, U20B2050), Key Research and Development Program of Shaanxi Province (2020GY-039, 2021ZDLGY05-09, 2022GY-040).

REFERENCES

1. Q. Li, F. Wang, J. Wang, and W. Li, "LSTM-based SQL injection detection method for intelligent transportation system," in *Proceedings of IEEE Transactions on Vehicular Technology*, Vol. 68, 2019, pp. 4182-4191.
2. W. H. Rankothge, M. Randeniya, and V. Samaranayaka, "Identification and mitigation tool for SQL injection attacks (SQLIA)," in *Proceedings of IEEE 15th International Conference on Industrial and Information Systems*, 2020, pp. 591-595.
3. M. A. Shahriar *et al.*, "Modelling attacks in blockchain systems using petri nets," in *Proceedings of IEEE 19th International Conference on Trust*, *Security and Privacy in Computing and Communications*, 2020, pp. 1069-1078.
4. L. Wang, Z. Zhu, Z. Wang, and D. Meng, "Colored petri net based cache side channel vulnerability evaluation," *IEEE Access*, Vol. 7, 2019, pp. 169825-169843.
5. S. Talukder, I. I. Sakib, F. Hossen, Z. R. Talukder, and S. Hossain, "Attacks and defenses in mobile IP: Modeling with stochastic game petri net," in *Proceedings of International Conference on Current Trends in Computer*, *Electrical*, *Electronics and Communication*, 2017, pp. 18-23.
6. Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A survey on wireless security: Technical challenges, recent advances, and future trends," in *Proceedings of the IEEE*, Vol. 104, 2016, pp. 1727-1765.
7. T. Le, T. Park, D. Cho, and H. Kim, "An effective classification for DoS attacks in wireless sensor networks," in *Proceedings of the 10th International Conference on Ubiquitous and Future Networks*, 2018, pp. 689-692.
8. E. M. D. L. Pinto, R. Lachowski, M. E. Pellenz, M. C. Penna, and R. D. Souza, "A machine learning approach for detecting spoofing attacks in wireless sensor networks," in *Proceedings of IEEE 32nd International Conference on Advanced Information Networking and Applications*, 2018, pp. 752-758.
9. S. R. Biswal and S. K. Swain, "Model for study of malware propagation dynamics in wireless sensor network," in *Proceedings of the 3rd International Conference on Trends in Electronics and Informatics*, 2019, pp. 647-653.
10. Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A survey on wireless security: Technical challenges, recent advances, and future trends," in *Proceedings of the IEEE*, Vol. 104, 2016, pp. 1727-1765.
11. M. B. A and M. Arock, "Efficient detection of SQL injection attack (SQLIA) using pattern-based neural network model," in *Proceedings of* International *Conference on Computing*, *Communication*, *and Intelligent Systems*, 2021, pp. 343-347.

12. H. Gao, J. Zhu, L. Liu, J. Xu, Y. Wu, and A. Liu, "Detecting SQL injection attacks using grammar pattern recognition and access behavior mining," in *Proceedings of IEEE International Conference on Energy Internet*, 2019, pp. 493-498.
13. Sangeeta, S. Nagasundari, and P. B. Honnavali, "SQL injection attack detection using ResNet," in *Proceedings of the 10th International Conference on Computing, Communication and Networking Technologies*, 2019, pp. 1-7.

Yi-Chuan Wang (王一川) received the Ph.D. degree in Computer System Architecture from Xidian University, Xi'an, China, in 2014. He is currently an Associate Professor with the Shaanxi Key Laboratory of Network Computing and Security Technology, Xi'an University of Technology, Xi'an, China. His research interests include networks security and system vulnerability analysis. He is a member of ACM and CCF.

Gui-Ling Zhang (张贵玲) is studying for a master's degree in Xi'an University of technology. Her main research focuses on the correlation analysis of network system vulnerabilities in network power prevention.

Ya-Ling Zhang (张亚玲) received her Ph.D. degree in Mechanism Electron Engineering from Xi'an University of Technology in 2008. She is a Professor at Xi'an University of Technology. Her current research interests include cryptography and network security.