Journal on Internet of Things
DOI: 10.32604/jiot.2020.010035
Article

Tech Science Press

KNEMAG: Key Node Estimation Mechanism Based on Attack Graph for IoT Security

Bichen Che[1], Long Liu[2,*] and Huali Zhang[1]

[1]School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China
[2]College of New Media, Beijing Institute of Graphic Communication, Beijing, China
[*]Corresponding Author: Long Liu. Email: longliu52@126.com
Received: 20 May 2020; Accepted: 15 August 2020

Abstract: With the rapid development and widespread application of the IoT, the at-tacks against IoT vulnerabilities have become more complex and diverse. Most of the previous research focused on node vulnerability and its risk analysis. There is little information available about the importance of the location of the node in the system. Therefore, an estimation mechanism is proposed to assess the key node of the IoT system. The estimation of the key node includes two parts: one is the utilization relationship between nodes, and the other is the impact on the system after the node is conquered. We use the node importance value and the node risk value to quantify these two parts. First, the node importance value is calculated by considering the attack path that pass through the node and the probability that the attacker will abandon the attack. Second, in addition to node vulnerabilities and the consequences of being attacked, two quantitative indicators are proposed to comprehensively assess the impact of nodes on the system security, and the node risk value is calculated based on the grey correlation analysis method. Third, the key node in the IoT system could be obtained by integrating the node importance value and risk value. Finally, the simulation experiment result shows that the presented method could find the key node of the system quickly and accurately.

Keywords: IoT; vulnerability assessment; key node

1 Introduction

IoT is another revolutionary development of the information industry following the computer, Internet and mobile communications, bringing a lot of convenience to people's daily life. With a large quantity of smart terminals accessing the Internet, various types of attacks are gradually moving from the Internet to the IoT [1]. The attacks method against IoT vulnerabilities is more complicated and diversified, and its scope and scale are gradually expanded [2]. According to CVE vulnerability database statistics, the number of IoT vulnerabilities that have been published has reached 9,367 so far [3]. There were 2,035 growth holes in 2018, an increase of 41.2% compared with last year, of which the high-risk vulnerabilities accounted for 42.57% [4]. Therefore, for the purpose of improving the security of IoT systems, it is critical to analyze the risk of IoT nodes and find the most important nodes to be protected.

There are many researches on the risk assessment of IoT systems, but most of the previous articles focus on the harm caused by node vulnerabilities to the system and the asset value of the nodes, and little attention has been devoted to the utilization relationship between the IoT nodes. The evaluation of IoT key nodes mainly consists of two parts. One is the importance value of the system nodes. It can be analyzed by the location of the node in the system, the dependencies between nodes, and the difficulty of the node being overcome. The other is the risk value. It can be evaluated by the value of the node's assets, the degree of impact on the system after the node is attacked, the exposure level of the node vulnerability,

and the degree of repair.

Therefore, in this paper, we propose a mechanism to evaluate the key node of the IoT system by combining the node importance and risk value. Firstly, we establish an attack graph model to analyze and subjectively show the possible attack paths and the dependencies between nodes. Then, the calculation algorithm of the node importance value is discussed by considering the number of all attack paths that may pass through the node and the possibility that the attacker may give up continuing to attack the target. The more nodes that are connected to other nodes, the more important it is. Also, in addition to node vulnerabilities and the consequences of being attacked, we add two influencing factors based on CVSS that affect the security risk of the IoT system, and then calculate the node risk value through the grey correlation analysis method. Finally, the key value of each node is obtained by integrating the importance and risk value. Comparing the numerical values, the most critical node in the IoT system can be obtained, which provides a certain reference value for the security personnel to take accurate and effective defense measures.

2 2 Related Work

2.1 IoT System Node Analysis Method

The vulnerability research of the IoT system is a complex and broad research topic, involving multiple disciplines such as formal taxonomy, statistics, data mining, risk management, visualization, and mathematical modeling [5]. Shi [6] proposed the structure and classification model of IoT device identification, which helped us understand the IoT system and its application. There are many nodes in the IoT system, and the attack paths are intertwined and complicated. In order to improve the security of the system more effectively, the security personnel need to evaluate each node of the IoT system to find the node with the greatest threat to the system security, so they could focus on deploying defense measures at the node. The mainstream analysis methods include the attack graph and the attack tree. They are quantitative or semi-quantitative models for the assets, threats and vulnerabilities of the system based on the research of the components and security issues of the system [7].

The attack tree uses a tree structure to describe the attack that the attacker may launch against the system. The root node of the tree represents the attack target, the child node represents the child target before reaching the total target, and the leaf node represents the specific attack method for each step. Eric [7] and Patel [8] used the attack tree to perform the vulnerability analysis and security assessment on the nodes. However, when the attack tree is applied in a specific instance, its structure will become huge and complex. A complete attack tree is likely to include hundreds of leaf nodes, which is not applicable for the large-scale IoT system.

The attack graph describes the attack scenario in which the attacker uses the system vulnerabilities and the dependency between the vulnerabilities to invade the network. It consists of the topology structure and configuration information of the network, and enumerates the possible attack paths. The methods for security assessment of the IoT system using the attack graph include the Bayesian network [9], the Markov chain [10], the Petri net [11], and the game theory [12]. Because the attack graph can intuitively display the attack path and exploit vulnerabilities, it is more conducive to evaluate key nodes. In recent years, the technology for automatically generating attack graphs has gradually matured. Therefore, we select the attack graph to evaluate the risk of the IoT nodes.

2.2 Vulnerability Assessment Based on Attack Graph

The relationship of IoT system nodes are complex, so attackers often use multiple isolated and less-affected node vulnerabilities to conduct multi-step attacks when launching attacks. Using this attack method not only reduces the probability of an attacker being discovered, but also increases the probability of successfully attacking the target. Therefore, the researchers built the attack map to more clearly analyze the risk that the system might have. The attack graph is a tool used to describe how an attacker exploits the connectivity between hosts and the dependencies between vulnerabilities to attack a target host [13]. It comprehensively analyzes multiple network configurations and vulnerability information, 7-

enumerates all possible attack paths, and intuitively describes the relationship between vulnerabilities in the target network, the relationship between vulnerabilities and network security configurations, and potential threats. The attack graph not only portrays the effects of a single weakness, but also the hazards that may be caused by the interdependence of weaknesses. Swiler et al. [14] first proposed the attack graph model, then the attack graph technology has been developed, up to now, the attack graph has been automatically constructed and generated.

The attack graph consists of nodes and directed edges. At present, the difficulty of vulnerability research based on the attack graph mainly focuses on the node probability calculation and risk quantification of the attack graph. Ou et al. [15] first proposed two challenges in calculating the probability of attack graph nodes: the repetitive calculation of cyclic node, which was probability caused by the existence of ring path, and the problem of probability miscalculation, which was caused by correlation between vulnerability utilization. Some people suggested that the nodes in the loop path could be deleted to solve the problem of loop node probability repetition, but this method might lose some important loop-free attack paths. In order to solve this problem, Ye et al. [16] combined the attack graph with the general security vulnerability assessment system, removed the unreachable path in the attack graph, and proposed the concept and calculation method of the maximum reachable probability. Dai et al. [17] proposed a node fuzzy risk assessment method based on risk attack graph, which used the maximum flow-based path planning method to correlate isolated vulnerabilities. Chen et al. [18] defined the acyclic attack path with the path length not greater than the constant n for the problem of the ring attack path in the attack graph. The forward search method and the depth-first search strategy were used to find the effective attack path. In addition, they set up a collection of intermediate nodes to prevent loop paths. However, these algorithms are very complicated. When dealing with a system with a relatively large number of nodes, there will be disadvantages such as slow running time and consumption of system resources.

In order to make better use of the attack graph for risk assessment, Alhomidi et al. [19] proposed a network vulnerability assessment method based on attack graphs, which used genetic algorithms to determine the risk of the attack path and gave the risk assessment value for the given network. Moreover, they also proposed a population-based approach to explain large-scale attack maps and find the most important paths. Swiler et al. [20] proposed a method for analyzing the vulnerability of the IoT from internal and external, which could analyze the risk of specific network assets and check the probability of possible attack paths. But these studies regard each node as an independent individual, without considering the association between the nodes. Therefore, Sawilla et al. [21] used PageRank algorithm to calculate the relative value of network nodes on the generated attack graph, and provided guidance for protecting the most important nodes in the IoT system. However, the initial weights of each node were the same, and the possibility of the attacker abandoning the attack was not considered. In order to solve the above problems, we proposed KNEMAG, a key node estimation mechanism based on attack graph for IoT security.

3 The Algorithm for Calculating the Node Importance Value

3.1 Attack Graph Construction

In the IoT system, the nodes are closely related. The attacker often chooses to start the attack from the terminal with weak protection capability, and then gradually obtains the operation authority of the node at the previous level through the operation of lifting the rights [22]. As a result, the attacker has the ability to control the IoT system, steal confidential information, or use a large-scale IoT devices for DDoS attacks. Therefore, when judging the importance of the IoT node, it should not only analyze the property of the node, but also comprehensively consider it with the attack path.

In order to present the vulnerability and relevance of nodes in the IoT system more comprehensively, objectively and clearly, we combine the attack graph with the node importance algorithm proposed in this paper to establish an attack model based on node importance. The attack graph is an effective method to reflect the logical relationship between nodes and to visually describe the attack path. The nodes represent

the loopholes that the attacker can exploit, and the edges represent the utilization relationship between the loopholes. In the traditional attack graph analysis, each node is treated equally, but in practical applications, different permissions and vulnerabilities will play different roles in the success of an attack. Therefore, each node should not be treated the same, and nodes with higher importance value should be concentrated more and prioritized.

The proposed algorithm calculates node importance value based on the link structure between nodes. Its main idea is that the nodes in the attack graph that are dependent on by many nodes or some crucial nodes are more important than nodes that are dependent on a small number of nodes or some unimportant nodes. In summary, the greater the number of other nodes that depend on node v, the higher the importance value of the node. The larger the value of the edge, the more important node v is.

In the attack graph, linking node A to nodes B and C means that node A diverts its own importance value to B and C with a certain weight. In the IoT system, the transfer ratio between nodes should not be evenly because the number of attack paths passing through each node is different. Nodes that exist in more attack paths should have more importance values. In addition, the connection relationship between two vulnerability nodes should also be considered. We divide the connection relationships between nodes into three types: physical dependencies, communication dependencies, and data dependencies. Physical dependencies mean that nodes are connected by physical lines. Communication dependency refers to the connection relationship established between nodes through a communication protocol. Data dependency means that a node needs data provided by another node while the network is running.

Based on the above influence factors, combined with the characteristics of the IoT system, we define the attack graph model $M = (A, E, S, N)$:

- A: Vulnerability node set: A collection of penetration behaviors that an attacker could exploit with a single vulnerability, $A = \{a_1, a_2, \ldots, a_n\}$, where a_i represents the *i-th* attack behavior, and n is the total number of all vulnerability nodes in the IoT system. Define $a_i = \{CVE, host\}$, CVE is the CVE number of the node vulnerability, and $host$ is the host name of the node, e.g., $a_i = \{CVE-2018-11456, user1\}$.
- E: directed edge set: The directed edge set of the next vulnerability node that the attacker points to after tapping the node, $E = \{e_{12}, e_{23}, \ldots, e_{uv}\}$, where e_{uv} indicates the directed edge from u points to v. The edge value is the transfer ratio, which means the proportion of one node passing its importance value to the next node.
- S: Dependency weight: The connection relationship between two nodes connected by a directed edge, including physical dependencies, communication dependencies, and data dependencies. According to the strength of the dependencies, the weights of 4, 2, and 1 are assigned in turn. When there are multiple dependencies between nodes, the weights are superimposed.
- *N: Number of nodes: The sum of the number of all nodes in the IoT system.*

The following figure is an example of the attack model, including the attacker attacker0, the exploited intermediate host user1 and the target host user2. The attacker's goal is to obtain the root privileges of user2. As can be seen from the figure, the attacker can reach the user2 through 4 attack paths. First, use the ftp protocol vulnerability on user1 to obtain the trust of user1. Then use the ftp protocol vulnerability on user2 to obtain the trust of use2. The second is to use the buffer overflow vulnerability on user1 to obtain the access rights of user1, and then uses the plaintext transmission vulnerability on user2 to obtain the root permission of user2. The third is to use the ftp protocol vulnerability on user1 to obtain the trust of user1, and then use the plaintext transmission vulnerability on user2 to obtain the root permission of user2. The fourth is to use the ftp protocol vulnerability on user2 to directly obtain the trust of user2, thereby obtaining root privileges.

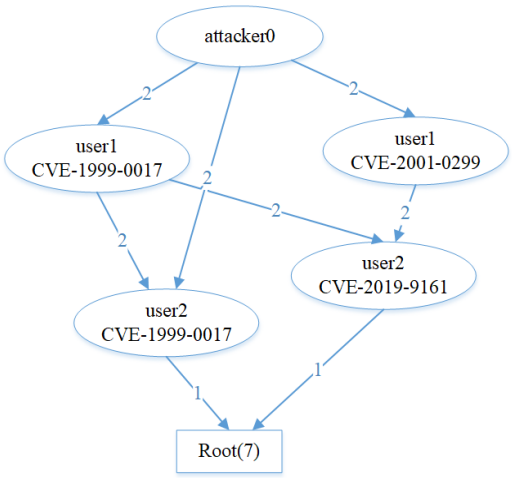

Figure 1: Example of an attack graph

3.2 Node Importance Calculation Based on PageRank

The PageRank algorithm can efficiently identify the most influential pages in a search engine index. Because the link relationship between web pages is similar to the utilization relationship between IoT nodes, we introduce the PageRank algorithm to identify the key node in the IoT system. However, it could not be directly applied to the IoT system [23–25] due to the complex relationship between nodes in the IoT system. Therefore, we made the following improvements to the algorithm under the consideration of the IoT characteristics:

a. Consider the number of attack paths that pass through the node

In the PageRank algorithm, the node distributes the PR values evenly to all nodes pointed to. But, in the IoT system, there are differences between the pointed nodes, e.g., some nodes pass more attack paths and some nodes store processed private data. This type of node has a greater impact on the system when it is overcome, so its importance value should be higher, that is, the node's PR value distribution should be non-average. We adopt a method of assigning the PR value of a node according to the ratio of the number of attack paths passing through the next node, so as to measure the importance of the node more accurately.

b. Consider the dependencies between nodes

Unlike the web page, the connection methods between the IoT nodes are different, so we add the connection weight when calculating the edge value in order to reflect the relationship between the nodes more intuitively. We divide the dependencies between nodes into three categories: physical dependencies, communication dependencies, and data dependencies. Physical dependencies mean that nodes are connected by physical lines. Communication dependency refers to the connection relationship established between nodes through a communication protocol. Data dependency means that a node needs data provided by another node while the network is running.

Physical dependency represents the dependencies of individual nodes on physical connections. If the nodes u and v are directly linked by physical lines, then u and v have physical dependencies. Communication dependency means that two nodes establish a connection relationship through direct message communication. For node u, v, if node v needs to send a message to node u to maintain system operation, then u is considered to be dependent on v. Data dependency stands for the dependencies established by two nodes through data transmission. For node u, v, if node u needs node v to provide data to it during system operation, then u is said to depend on v. According to the strength of the dependency, we assign the weight of 4, 2, 1 to the physical dependence, communication dependency and data dependence. When there are multiple dependencies between nodes, they are superimposed.

c. Add the probability δ of abandoning an attack.

The parameter δ in the original PageRank algorithm represents the probability that people will give up browsing the web pages. Google sets it to an experience value of 0.15, but this empirical value should not be applied to the IoT system. We propose to use the probability of abandoning the attack as the value of the δ parameter. The new δ represents the probability that the attacker chooses to give up after analyzing the possibility that the node can reach the target. When the probability of reaching the target is higher, the probability of abandoning the attack is smaller. When calculating δ, first calculate the cumulative probability that the node may reach the target, and then subtract the cumulative probability by 1 to get the probability that the attacker will give up the attack.

3.2.1 The Algorithm for Calculating the Number of Attack Paths

We propose the algorithm *Countpath* to calculate the number of attack paths that pass through the specific node. The main idea of this algorithm is to find all possible attack paths from the initial node to the target node and determine whether the specified node exists in each attack path. The algorithm *Sumpath* is used to find all attack paths. Its main idea is to start from the target node *v* and use the forward search and depth-first search strategy to iteratively calculate the effective attack path of all the parent nodes that can reach *v*. In order to ensure that the calculated attack path does not contain a circle, the node track set trace is introduced to store the searched node name. If the node is already in the trace before the iteration, it means that this iteration will generate a circled attack path, so the iteration will terminate. If the node is not in the trace, the node is then added into the trace collection.

Record all valid attack path sets of the target node as $Path(v)$. Let *a* be the parent node of the target node *v*, then the *n* effective attack paths that can reach node *v* are composed of node *v* and $(n-1)$ valid paths combinations of reachable node *a*.

Algorithm 1 Calculate the number of attack paths through the node

Main Procedure *Countpath*

Input: Specified node *u*

Output: Number of attack paths through the node

Sumpath(u)

// $Path(v_1),\ldots,Path(v_n)$ is the set of all attack paths

For each $Path(v_j)$

If $u \in Path(v_j)$

$Num \leftarrow Num+1$

Return Num

Subroutine *Sumpath(x)*

// $a_1, a_2, \ldots, a_n$ is the parent node connected to node *x*

For each a_i

If $a_i \notin trace$

Sumpath(a_i*)*

$Path(v_j) \leftarrow Path(v_j) \cup \{a_i\}$

$trace \leftarrow trace \cup \{a_i\}$

Else break

Return $Path(v_1),\ldots,Path(v_n)$

3.2.2 The Algorithm for Calculating the Probability of Abandoning the Attack

The probability of abandoning an attack means the probability that an attacker will abandon the continued backward attack from the node, which can be obtained by cumulative probability. The cumulative probability represents the probability that an attacker will start attacking at node *t* and reach the target node *v* through all paths that can reach it. If a_j is the parent of S_i, the cumulative probability $P_d(S_i)$ can be calculated by Eq. (1):

$$P_d(S_i) = P(S_i \mid a_j) \times P_d(a_j). \tag{1}$$

where $P(S_i \mid a_j)$ is the conditional probability distribution function of S_i. Since the child node and the parent node have an OR relationship, the probability of the state node can be derived from Eq. (2):

$$P_d(S_i) = 1 - \prod [1 - P_d(a_j) \times P(v)]. \tag{2}$$

where $P(v)$ is the probability that the node is successfully utilized, which is related to the vulnerability's own attributes.

The algorithm of calculating δ is given below. It mainly calculated by calling the node cumulative probability function *SumProb*. The algorithm of calculating the cumulative probability is to first find all the paths from the specified node to the target node, and then calculate the cumulative probability of successfully reaching the target based on the vulnerability node on the path.

Algorithm 2 Calculate the probability δ of abandoning an attack

Procedure *SumProb*

Input: Specified node *t*, the probability of successful utilization of each node $P(v)$

Output: The probability δ of abandoning an attack

Sumpath(t)

// Find all paths from the specified node to the target node

For each $P(v_m)$

For each $S_{m1} \in Path(v_m)$

$P_d(S_{m1}) \leftarrow 1$

$Temp \leftarrow 1$

Stack.push(S_i)

//Push the nodes $S_{m1},\ldots,S_{mn}$ in the path onto the stack in turn

while(!Stack.empty()){

$S_{mi} \leftarrow$ Stack.pop() //Get the target node S_{mn} first

for each S_{mi}

$Temp \leftarrow Temp \times P_d(S_{m(i-1)})P(v_i)$

$P_d(S_{mi}) \leftarrow 1 - [1 - Temp]$

// Calculate the probability of reaching the target node in the path

Return $P_d(S_{mn})$

$P_{sum} \leftarrow (S_{1n}) + P_d(S_{2n}) + \cdots + P_d(S_{mn})$

$\delta \leftarrow 1 - P_{sum}$

3.2.2 The Algorithm for Calculating the Node Importance Value

The algorithm for calculating the importance value of the node mainly includes the following processes:

a. In the initial stage: The attack graph is constructed according to the vulnerability of the IoT node and the utilization relationship between them. Initialize the dependency weight S according to the dependency relationship between the nodes, and set the node importance value of the initial node to 1.

b. Calculate the number of attack paths $R_1, R_2, \ldots, R_i$ of all the next-level nodes pointed to by the node, and the importance weight of the next node can be calculated by $E = \frac{R_i}{R_1 + R_2 + \cdots + R_i}$. Then, the ratio of the node importance value to the next node is $E \times S$.

c. With each round of calculations, the node's current importance value will be continuously updated. In this paper, the power iteration method is used to obtain the node importance value. The main idea is that each node assigns its current importance value to each node pointed to by the node according to the method in Step b, and sums all importance values that are passed to the node. In addition, considering the possibility of the attacker giving up the attack, a new node importance value can be obtained. When each node obtains the updated importance value, a round of importance value update calculation is completed.

The formula for calculating the node importance value is as shown in Eq. (3):

$$PR(u) = \delta \times \sum_{v \in I_{in}} PR(v) \times E \times S + \frac{1-\delta}{N}. \tag{3}$$

where PR represents the importance value of the node, E represents the weight of the dependencies between the nodes, S represents the ratio between each node pointed to by the node, δ represents the probability of the attacker giving up the attack, and N represents the total number of nodes.

d. After several rounds of iterative calculations, the node importance value tends to be stable. When $|P_{n+1} - P_n| < \varepsilon$ is satisfied, the iteration ends and the importance values of all nodes are obtained.

The main calculation process of the node importance value is shown in Fig. 2.

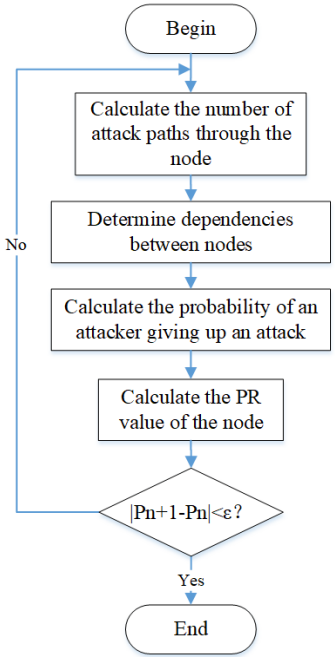

Figure 2: The node importance calculation algorithm flow chart

4 The Node Risk Value Calculation Based on Grey Correlation Method

4.1 Quantification of Four Risk Indicators

Judging the criticality of the node in the IoT system depends not only on the im-portance of the node in the network, but also on the risk value of the node. In this paper, based on the CVSS standard, we propose four kinds of indicators to quantify the risk value of the IoT system nodes, and obtain the node risk value by analyzing the vulnerability, the consequences of the attack, the value of the asset, and the time measurement.

The four types of risk indicators are shown in Fig. 3.

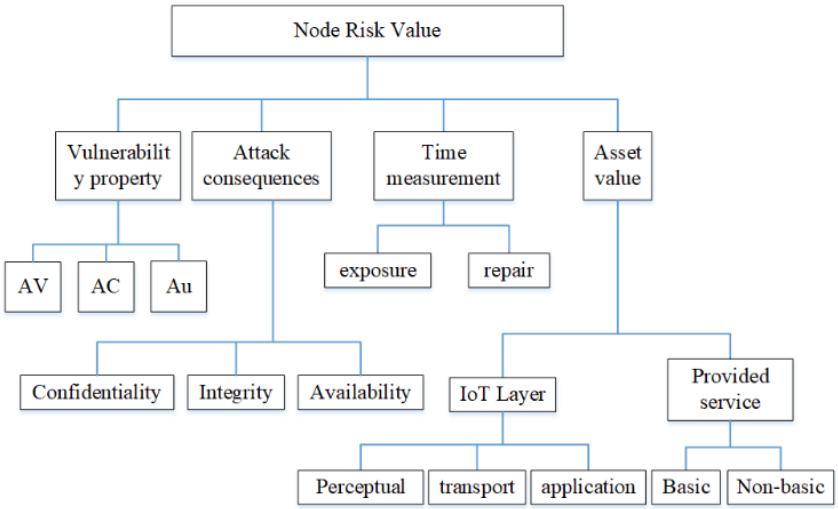

Figure 3: Risk indicators

4.1.1 The Property of The Vulnerability V

The vulnerability is different because they are successfully exploited by attackers with various difficulty factors. According to the Availability of the CVSS standard, we consider the vulnerability's own attributes from the way the vulnerability is exploited, the difficulty of exploiting the vulnerability, and the authentication method that is required to exploit the vulnerability.

AV (Access Vector): The access vector, which represents the way in which vulnerabilities may be exploited, including local (L, Local), neighboring networks (A, Adjacent Network), and networks (N, Network). The simpler the vulnerability is exploited, the lower the access vector value and the higher the risk of vulnerability. For example, the vulnerability exploited remotely is more likely to be successfully exploited than the vulnerability exploited locally, and the risk is greater.

AC (Access Complexity): Access complexity, indicating how easy it is to exploit the vulnerability. The values of access complexity include H (High), M (Medium), and L (Low). The less vulnerable the vulnerability is, the higher the complexity of the vulnerability access and the higher the risk of the vulnerability.

Table 1: Value Assignment of Availability in CVSS v3.0

Indicator Name	Metric	Possible Value	Numerical Value
Availability	Access Vector	L/A/N	0.395/0.646/1
	Access Complexity	H/M/L	0.35/0.61/0.71
	Authentication	M/S/N	0.45/0.56/0.704

Au (Authentication): Authentication, which means the authentication method that an attacker needs to be used when exploiting a vulnerability. The value of the authentication includes M (Multiple

Authentication), S (Single Authentication), and N (None). The simpler the authentication required to exploit a vulnerability, the higher the authentication value and the higher the risk of vulnerability.

According to the CVSS standard, the formula for calculating the vulnerability's own properties is as shown in Eq. (4):

$$V=2\times AV\times AC\times AU. \tag{4}$$

The value assignment of availability in CVSS v3.0 is summarized in Tab. 1.

4.1.2 The Consequences of The Attack I

The consequences of an attack represent the impact that an attacker may have on the system after being intercepted by a readable, writable, and empowered operation. It is measured by three indicators: Confidentiality, Integrity and Availability. They respectively represent the impact on the confidentiality, integrity, and availability of the target system after the vulnerability is successfully exploited. The value of each index includes N (None), P (Partial Impact), and C (Complete Impact). The requirements for these three indicators are different in different systems, for example, the banking system will have higher requirements for confidentiality. The greater the impact on the target system after the vulnerability is exploited, the higher the risk value of the node.

Table 2: Value Assignment of Influence in CVSS v3.0

Indicator Name	Metric	Possible Value	Numerical Value
Influence	Confidentiality	N/P/C	0/0.275/0.66
	Integrity	N/P/C	0/0.275/0.66
	Availability	N/P/C	0/0.275/0.66

According to the CVSS standard, the formula for calculating the consequences of a node after being compromised is as shown in Eq. (5):

$$I(a_i)=10.41\times[1-(1-C_i)\times(1-I_i)\times(1-A_i)]. \tag{5}$$

Tab. 2 gives the value assignment of influence in CVSS v3.0.

4.1.3 Time Measurement T

Time is one of the most important factors that affect the attacker's attack. The longer the vulnerability is discovered, the more mature the attack on the vulnerability, and the more complete the vulnerability will be fixed. This section evaluates the node vulnerabilities by selecting the representative indicators in the time metric of the CVSS scoring standard, the degree of node exposure and the degree of node repair.

Node exposure level (X): The extent to which the current vulnerability is known and used by the attacker since the vulnerability information was released. If the vulnerability has matured and exploitable attack tools on the Internet, the attack threshold will be greatly reduced. The higher the exposure of a node vulnerability, the larger the value of X, the higher the risk of that node. T The degree of node exposure and the degree of node repair are a contradiction. As time goes by, the attacker will be more familiar with the vulnerability, but at the same time the node will be better repaired. According to the exposure degree of the node, it is divided into the following three situations: The vulnerability is widely known and there are mature attack tools. The node is highly exposed and is quantified as 6; the vulnerability is known by some people and there are not many attack tools that can be used. The node is exposed to a general extent, which is quantified as 3; The vulnerability is basically unknown and there are no attack tools that can be exploited. The node is exposed to a low level and is quantized to 1.

Node repair degree (Y): The extent to which the current vulnerability is fixed by security personnel since the vulnerability information was release. According to the study of the half-life of the vulnerability, about 50% of the most popular high-risk vulnerabilities will have an impact for a year or so, and one year

later, these vulnerabilities will be replaced by new high-risk vulnerabilities, because the past node vulnerabilities have been completely fixed. The higher the degree of repair of the node vulnerability, the larger the value of Y, the lower the risk of the node. In this paper, we expand the gap between patched and unpatched, because the risk of 0day vulnerability is greater in the actual attack and defense process. Therefore, we divide the degree of repair of nodes into the following three cases: the node vulnerability has been fixed and there are mature vulnerability repair methods. The node is highly repaired, and its Y value is quantized to 10; The node vulnerability has not been completely fixed, and there are some ways to fix it. The node is generally repaired, and its Y value is quantified to 5; The node vulnerability has not been fixed and there is no corresponding fix solution. The degree of repair of the node is very low, and its Y value is quantized to 1.

The formula for calculating the node time metric is given by Eq. (6):

$$T = \frac{X}{Y}. \tag{6}$$

4.1.4 Node Asset Value C

The asset quantification of the node is closely related to the IoT system level of the node and the service type it provided.

The IoT system level (L): Nodes transmit different types of data at different levels of the IoT system, producing different values. The IoT system is mainly divided into three levels, the perceptual layer of perceiving data, the network layer of transmitting data, and the application layer of applying data. The perceptual layer contains many sensors, generating a large amount of data related to user privacy. But this data is scattered and messy and its usability is small. Since its value is not high, its *L* value is quantized to 0.25. The work of the network layer includes communication between nodes, and storing, querying, analyzing, and mining data of the perceptual layer. The attacker can attack the node by tampering with the communication data. The value of the processed data is important, so the L value of the node at this level is quantized to 1; The application layer uses the processed data to provide specific services to the user. When the layer node is compromised, it will affect the user. The data availability is medium and has some value, so its L value is quantized to 0.5.

The provided service type (S): The services provided by nodes can be divided into two categories: Basic services and non-based services. Basic services stand foe the services used to support other services, such as database services. Non-basic services represent the general application services provided, e.g., the word processing services. When the node providing the basic service is overcome, the system will be difficult to operate normally, so it has higher value. We quantify the services provided by the nodes based on the type and number of services running on the nodes. The basic service is 0.5 and the non-base is 0.1.

According to the CVSS standard, the formula for calculating the asset value of a node is as shown in Eq. (7):

$$C = [T + (10 - T) \times S] \times L. \tag{7}$$

4.2 The Node Risk Value Calculation Based on Grey Correlation

In the previous section, we present four categories of indicators for assessing the risk of IoT nodes and quantify them separately. We use the gray evaluation method to normalize the quantized values of the four indicators by calculating the gray correlation degree of each indicator to obtain the comprehensive risk value of the node. This method has the advantages of simple calculation, no need for a large number of samples, and distribution laws.

The normalized risk analysis steps based on the grey correlation analysis method are as follows:

a. Determine the reference sequence. The optimal value of each indicator is selected from *n* evaluation nodes to form a reference sequence x_0 :

$$x_0 = (x_{01}, x_{02}, \ldots, x_{0n}). \tag{8}$$

b. Determine the difference between the two levels. The absolute difference sequence Δ_{ij} of the evaluation object sequence x_i and the reference sequence x_0 is

$$\Delta_{ij} = \left| x_{ij} - x_{0j} \right|. \tag{9}$$

Based on the above, the maximum difference and the minimum difference sequence can be obtained as follows:

$$\Delta_{\max} = \max_{1 \le i \le n} \max_{1 \le j \le m} (\Delta_{ij}). \tag{10}$$

$$\Delta_{\min} = \min_{1 \le i \le n} \min_{1 \le j \le m} (\Delta_{ij}). \tag{11}$$

where $i = 1, 2, \ldots, n, j = 1, 2, \ldots, m.$

c. Determine the correlation coefficient. The correlation coefficient η_{ij} of the evaluation object sequence x_i and the reference sequence x_0 is:

$$\eta_{ij} = \frac{\Delta_{\min} + \alpha \times \Delta_{\max}}{\Delta_{ij} + \alpha \times \Delta_{\max}}. \tag{12}$$

where α is the control coefficient, generally satisfying $\alpha \in (0,1)$, which can attenuate the problem of distortion of the correlation coefficient caused by the maximum difference $\Delta_{\min}$.

d. Determine the degree of relevance. The relationship r_{0i} of each evaluation object sequence with the reference sequence x_0 is

$$r_{0i} = \frac{1}{m} \sum_{k=1}^{m} \eta_{ij}. \tag{13}$$

5 Case Analysis

5.1 Experimental Environment Topology

The network structure of the IoT system is huge and complex, including a variety of subsystems and network domains such as the terminal device layer, the communication transport layer, and the application layer. Therefore, in this paper, we take a simplified industrial internet of things network as an example to construct an attack graph, calculate node importance and risk value, and finally find the key node in the system. The simplified structure of an industrial internet of things network is shown in Fig. 4.

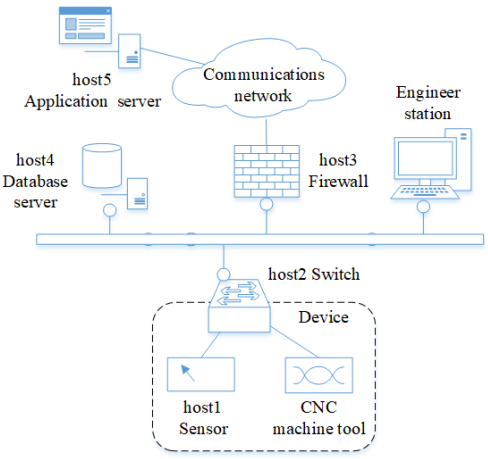

Figure 4: Simplified industrial Internet of Things network structure

The industrial internet of things network is mainly composed of sensors, CNC machine tools, switches, database servers, firewalls, engineer station and application servers. There are five hosts with vulnerability nodes, and one or more vulnerabilities may exist on one node, as shown in Tab. 3.

Table 3: ICS vulnerability node

Node	CVE numbers	V(AV/AC/Au)	I(C/I/A)	C(L/S)	T(X/Y)
Sensor	CVE-2010-5305	N/L/N	P/P/P	0.25/0.1	5/1
Switch	CVE-2018-4833	A/L/N	P/P/P	1/0.5	5/5
Application Server	CVE-2019-3941	N/L/N	N/P/P	0.5/0.5	1/5
Database Server	CVE-1999-0017	N/L/N	P/P/P	1/0.5	10/1
Database Server	CVE-2012-2122	N/H/N	P/P/P	1/0.5	5/1
Firewall	CVE-2018-11456	N/L/N	P/N/N	1/0.1	10/10
Application Server	CVE-2000-0964	N/L/N	C/C/C	0.5/0.5	10/1

5.2 Experimental Environment Topology

According to the vulnerability of each host in the system, the attack graph can be scanned, as shown in Fig. 5:

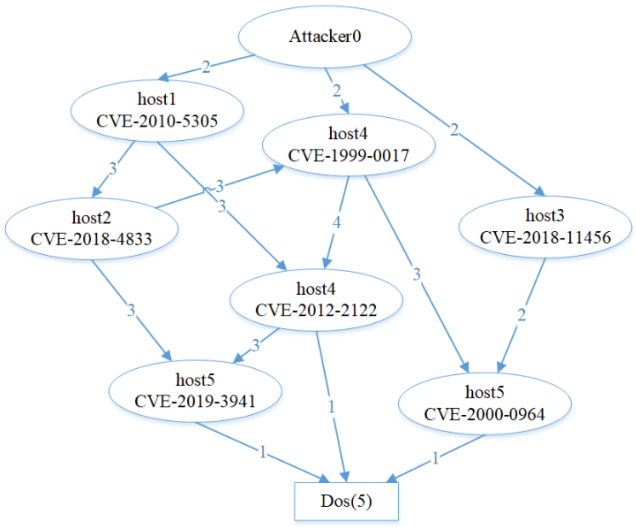

Figure 5: Attack Graph

From the attack graph, we can see that there are 8 possible attack paths in the attack graph:

a. Attacker0→ CVE-2010-5305→ CVE-2018-4833→ CVE-2019-3941→ Dos(5)

b. Attacker0→CVE-2010-5305→ CVE-2012-2122→ CVE-2019-3941→ Dos(5)

c. Attacker0→CVE-2010-5305→ CVE-2012-2122→ Dos(5)

d. Attacker0→CVE-2010-5305→CVE-2018-4833→ CVE-1999-0017→ CVE-2012-2122→ Dos(5)

e. Attacker0→CVE-2010-5305→CVE-2018-4833→ CVE-1999-0017→ CVE-2000-0964→ Dos(5)

f. Attacker0→ CVE-1999-0017→ CVE-2012-2122→ Dos(5)

g. Attacker0→ CVE-1999-0017→ CVE-2012-2122→ CVE-2019-3941→ Dos(5)

h. Attacker0→ CVE-1999-0017→ CVE-2000-0964→ Dos(5)

i. Attacker0→CVE-2018-11456→CVE-2000-0964→ Dos(5)

By using the algorithm proposed in this paper, the important values of each node can be obtained, as shown in Tab. 4.

Table 4: ICS vulnerability node

Node	Number of attack paths through the node	Importance value
Node1	5	0.2406
Node 2	3	0.2898
Node 3	3	0.6080
Node 4	5	0.3638
Node 5	5	0.4664
Node 6	1	0.2406
Node 7	3	0.2134

5.2.1 Effectiveness Analysis

The node importance value obtained by the algorithm proposed in this paper is com-pared with the PageRank algorithm, as shown in the line chart of Fig. 6. It can be seen from the figure that the proposed algorithm is consistent with the node importance trend of the PageRank algorithm, which indicates that the algorithm is feasible. Moreover, the proposed algorithm is more accurate and comprehensive after increasing the assessment of the possible attack paths and the probability of attack abandonment.

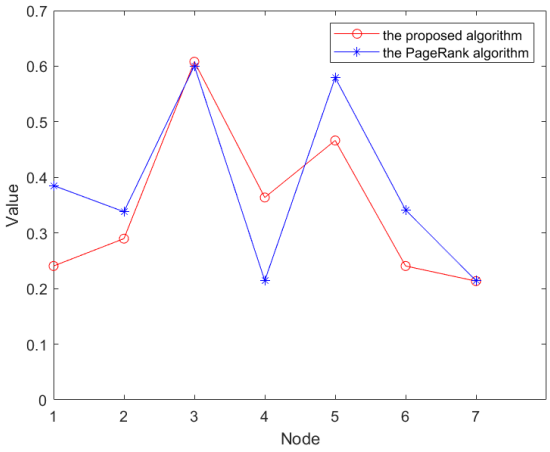

Figure 6: Comparison of the proposed algorithm with the PageRank algorithm

5.2.2 Algorithm Complexity Analysis

After ensuring the validity of the algorithm, we also need to evaluate the efficiency of the algorithm. If the proposed algorithm is too complex, consumes too much resources, or takes too long to calculate the results, the availability of the proposed algorithm will be greatly reduced.

According to the estimation method in literature [17], we set the number of nodes in the attack graph is n, the number of attack paths is R, the maximum depth of the path in the attack graph is L, and the maximum number of parent nodes of each node is M. For each node in the attack graph, we

assume $a_1, a_2, \ldots, a_M$ is all its parent nodes, so the algorithm needs to call M sub-functions at most and the number of iterations is up to L times. Therefore, the time complexity of the proposed path search algorithm can be defined as $O((n+R)\times M^L)$, and the time complexity of BFS and DFS algorithm in literature [20] is both $O(n^2)+O(R\times\log(R))+O(R)=O(n^2)$. When the number of nodes is small, the DFS algorithm of [20] will get the results faster, but when the system scale is increased, the algorithm of this paper is less complex and consumes less system resources.

In order to verify the performance of the path search algorithm more accurately, we set the number of nodes is 20, 40, 60, 80, 100, a comparison chart can be obtained as shown in Fig. 7. It can be seen from the figure that when the node of the system is below 40, the literature [17] takes less time, and the complexity of the two algorithms is similar. When the number of system nodes increases, the proposed algorithm has less complexity and better performance. Overall, the path search algorithm proposed in this paper is more efficient than the literature [17].

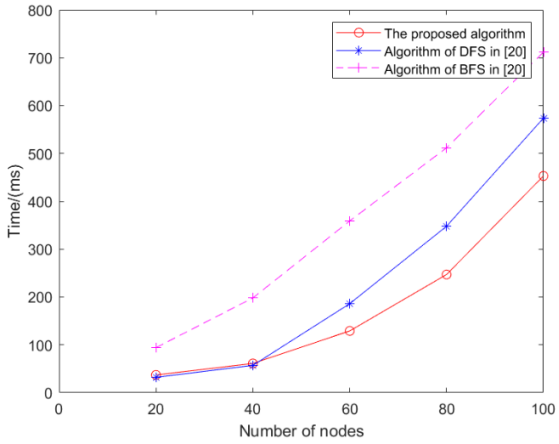

Figure 7: The algorithm complexity of the proposed algorithm, the algorithm DFS and BFS in [17]

5.3 Risk Value Analysis

According to the four risk indicators, the risk quantized values of the seven nodes are calculated separately, the vulnerability of the node, the impact on the system after the attack, the value of the node asset and the time measurement are shown in Tab. 5:

Table 5: Quantification of risk indicators

Node	Vulnerability Property	Attack consequences	Asset value	Time measurement
Node1	0.9997	6.4427	1.375	5
Node 2	0.6458	6.4427	5.5	1
Node 3	0.9997	4.9385	2.5625	0.25
Node 4	0.9997	6.4427	10	10
Node 5	0.4928	6.4427	7.5	5
Node 6	0.9997	2.8628	1.9	1
Node 7	0.9997	10.0009	5	10

In order to comprehensively consider the impact of four indicators on system security, we perform normalization analysis on each node according to the gray correlation method, perform normalization analysis on each node.

The relationship r_{0i} of each evaluation object sequence with the reference sequence x_0 is:

$$r_{01} = \frac{1}{4}\sum_{k=1}^{m} \eta_{0j} = 0.5745$$

$$r_{02} = 0.4397, r_{03} = 0.5285, r_{04} = 0.8752$$

$$r_{05} = 0.4902, r_{06} = 0.5080, r_{07} = 0.8658$$

5.4 Key Value Analysis

After calculating the node importance and risk value respectively, we multiply the two indicators according to the formula $K_i = \mu r_i \times \lambda I_i$, μ and λ are the weights, which have different values in IoT systems with different emphasis. The industrial internet of things system in this case has same requirements for node position and node risk, so we take both λ and μ as 1. The key values of each node can be calculated, as shown in Tab. 6.

Table 6: Quantification of Risk Indicators

Node	1	2	3	4	5	6	7
Value	0.1382	0.1274	03213	0.3184	0.2286	0.1222	0.1848

Comparing the key values of each node, it can be seen that node 3 is the critical node in the industrial internet of things IoT system. This is because the location of node 3 in the system is the most important and its asset value is relatively high.

6 Conclusion

Most of the research on IoT node risk is based on the analysis of the node's asset value or its own vulnerability. It does not combine the overall system and the attacker to comprehensively analyze the utilization relationship between the vulnerabilities.

In view of these deficiencies, we propose a method to evaluate the key node of the IoT system by combining the node importance value and the node risk value. First, in the algorithm for calculating the node important values, the attack paths and the probability of abandoning the attack are considered, and an attack graph model is proposed to more clearly calculate the utilization and dependencies between nodes. Second, based on the IoT characteristics and the CVSS standard, apart from the node vulnerability and the consequences of the attack, two additional indicators for evaluating the node risk value and their corresponding quantified values are proposed to make it better applicable to the IoT system. We use the gray correlation method to calculate the combined effects of the four indicators. Third, the importance and risk value are multiplied to get the key value of the IoT node. The node with the highest key value is the most critical node in the IoT system. At last, the application case shows that the proposed method is feasible, objective and comprehensive. This method could provide a reference for security personnel to set protection strategies to some extent.

Although the method proposed in this paper can comprehensively judge the key node, there still contains some other problems unsolved. For example, the judgment of the key node lacks more objective and reliable performance indicators, and we need to simulate attackers to launch a large number of attacks for estimation. Therefore, the future work will focus on performance assessment in dynamic environment and the behavior prediction of attackers.

Acknowledgement: This work is supported by the National Key R&D Program of China (2017YFB0802703), Major Scientific and Technological Special Project of Guizhou Province (20183001), Open Foundation of Guizhou Provincial Key VOLUME XX, 2019 Laboratory of Public Big Data (2018BDKFJJ014), Open Foundation of Guizhou Provincial Key Laboratory of Public Big Data

(2018BDKFJJ019) and Open Foundation of Guizhou Provincial Key Laboratory of Public Big Data (2018BDKFJJ022).

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] J. Su, Z. Sheng, D. Hong and V. Leung, "An efficient sub-frame based tag identification algorithm for UHF RFID," in *Proc. IEEE Int. Conf. on Communications*, pp. 1–6, 2016.

[2] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018.

[3] CERT/CC, "CNCERT Internet Security Threat Report," 2019, [Online]. Available: https://www.cert.org.cn/publish/main/upload/File/cnvd201904.pdf

[4] CERT/CC, "A review of China's Internet network security situation in 2018," 2019, [Online]. Available: https://www.cert.org.cn/publish/main/upload/File/2018situation.pdf

[5] Z. Sun, B. Luo, S. Luo and H. Zhu, "Security model of Internet of Things based on hierarchy," *Computer Engineering*, vol. 37, no. 10, pp. 1–7, 2011.

[6] C. T. Shi, "A novel ensemble learning algorithm based on DS evidence theory for IoT security," *Computers, Materials & Continua*, vol. 57, no. 3, pp. 635–652, 2018.

[7] B. Eric and J. Lowe, "The myths and facts behind cyber security risks for industrial control systems," in *Proc. of the VDE Kongress*, vol. 116, pp. 213–218, 2004.

[8] S. C. Patel and Y. Yu, "Analysis of SCADA security models," *International Management Review*, vol. 3, no. 2, 68, 2007.

[9] M. Frigault, L. Wang, S. Jajodia and A. Singhal, "Measuring the overall network security by combining CVSS scores based on attack graphs and bayesian networks," *Network Security Metrics*, Springer, pp. 1–23, 2017.

[10] P. Holgado, V. Villagra and L. Vazquez, "Real-time multistep attack prediction based on hidden markov models," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2017.

[11] T. M. Chen, J. C. Sanchez-Aarnoutse and J. Buford, "Petri net modeling of cyber-physical attacks on smart grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 741–749, 2011.

[12] K. Durkota, V. Lisý, C. Kiekintveld, B. Bošanský and M. Pěchouček, "Case studies of network defense with attack graph games," *IEEE Intelligent Systems*, vol. 31, no. 5, pp. 24–30, 2016.

[13] Z. Qu, S. Chen, S. Ji, S. Ma and X. Wang, "Anti-noise bidirectional quantum steganography protocol with large payload," *International Journal of Theoretical Physics*, vol. 57, no. 6, pp. 1903–1927, 2018.

[14] L. P. Swiler, C. Phillips, D. Ellis and S. Chakerian, "Computer-attack graph generation tool," in *Proc. DARPA Information Survivability Conf. and Exposition II*, DISCEX'01, Anaheim, CA, USA, pp. 307–321, 2011.

[15] X. Ou and A. Singhal, *Quantitative Security Risk Assessment of Enterprise Networks*. Springer, New York, 2012, pp. 13–23.

[16] Y. Ye, X. Xu, Y. Jia and Z. Qi, "An attack graph-based probabilistic computing approach of network security", *Chinese Journal of Computers*, vol. 33, no. 10, pp. 1987–1996, 2010.

[17] F. Dai, Y. Hu, K. Zheng and B. Wu, "Exploring risk flow attack graph for security risk assessment," *IET Information Security*, vol. 9, no. 6, pp. 344–353, 2015.

[18] F. Chen, D. Liu, Y. Zhang and J. Su, "A scalable approach to analyzing network security using compact attack graphs," *Journal of Networks*, vol. 5, no. 5, pp. 90–94, 2010.

[19] M. Alhomidi and M. Reed, "Risk assessment and analysis through population-based attack graph modelling," in *World Cong. on Internet Security*, London, UK, pp. 19–24, 2013.

[20] L. P. Swiler and C. Phillips, "A graph-based system for network-vulnerability analysis," in *Proc. of the 1998 Workshop on New Security Paradigms*, Charlottesville, Virginia, USA, pp. 71–79, 1998.

[21] S. Sawilla and X. Ou, "Googling attack graphs," *Defence R&D Canada-Ottaw*, pp. 12–15, 2007.

[22] L. Kou, Y. Q. Shi, L. G. Zhang, D. Liu and Q. Yang, "A Lightweight three-factor user authentication protocol for the information perception of IoT," *Computers, Materials & Continua*, vol. 58, no. 2, pp. 545–565, 2019.

[23] J. Su, G. Wen and D. Hong, "A new RFID anti-collision algorithm based on the Q-ary search scheme," *Chinese Journal of Electronics*, vol. 24, no. 4, pp. 679–683, 2015.

[24] H. Chen, K. Liu, C. Ma, Y. Han and J. Su, "A novel time-aware frame adjustment strategy for RFID anti-collision," *Computers, Materials & Continua*, vol. 57, no. 2, pp. 195–204, 2018.

[25] J. Su, X. Zhao, D. Hong, Z. Luo and H. Chen, "Q-value fine-grained adjustment based RFID anti-collision algorithm," *IEICE Transactions on Communications*, vol. E99.B, no. 7, pp. 1593–1598, 2016.