IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 17, 2018, accepted February 9, 2018, date of publication February 13, 2018, date of current version March 12, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2805690

A Vulnerability Assessment Method in Industrial Internet of Things Based on Attack Graph and Maximum Flow

HUAN WANG[1,2], ZHANFANG CHEN[1], JIANPING ZHAO[1,2], XIAOQIANG DI[1,2], AND DAN LIU[1]

[1]Department of Computer Science and Technology, Changchun University of Science and Technology, Changchun130022, China
[2]Jilin Province Key Laboratory of Network and Information Security, Changchun 130022, China

Corresponding author: Jianping Zhao (471745553@qq.com)

This work was supported in part by the Key Science and Technology Project of Jilin Province under Grant 20160204019GX and in part by the National Defense Scientific and Technological Innovation Special Zone Project of China under Grant 17-163-11-ZT-003-020-01.

ABSTRACT To solve the low attack path quantification degree and complex path finding in the industrial Internet of Things, a vulnerability assessment method based on attack graph and maximum flow is proposed. The method takes into account the factors influencing the attack behavior and relationship between network nodes. The attack risk is calculated by common vulnerability scoring system, which increases the attack path quantification degree. The maximum loss flow describes the attack path, evaluates the network vulnerability by maximum loss flow and loss saturation and represents the vulnerability relevance. Avoiding the repeat calculation and obtaining the potential key vulnerability path fast, the augmented road algorithm is used to find optimal attack path within global path. The result shows that the method is feasible and can evaluate the vulnerability and risk path objectively.

INDEX TERMS Vulnerability assessment, attack graph, maximum flow, industrial Internet of Things.

I. INTRODUCTION

With the development of industrial internet of things technology, more and more internet of things have been applied to real life. In industrial internet of things, it is of great significance to ensure the secrecy and integrity of the transmission. Due to the lack of data protection and high-level attack defense, the nodes in the internet of things are easy to be captured by attackers [1]–[3]. In this attack, attackers capture the network node to obtain the key, and crack the data in the network. The privacy, security, and reliability of the network are destroyed [4]–[7].

In recent years, many experts and scholars at home and abroad have studied the model of network vulnerability assessment, and have achieved some results. In literature [8], built the relationship between vulnerabilities by using the idea of Bayesian network separation, proposed a vulnerability assessment method based on Bayesian network. By adding separate nodes between related nodes, the original relationship is decomposed into the relationship between the correlation node and the separation node, and the conditional probability is used to solve the probability calculation error caused by the node correlation. In literature [9], a vulnerability analysis model based on attack graph has proposed. The model takes the security probability attack graph as the probability of attack graph, and solves the problem of cyclic path probability repetition calculation by the maximum reachable probability and the delete unreachable path. Based on the Common Vulnerability Scoring System (CVSS) [10], the safety probability calculation is carried out. The model can adapt to the vulnerability assessment of large-scale network, and the time complexity is small. However, the effect of the method on the network security probability is not explained by the correlation between the permeability and the correlation. In literature [11] and [12], through vulnerability analysis, the safety and reinforcement of network system is proposed according to the idea of repair set. Through the vulnerability and node relationship, construct the initial condition repair set, repair the initial condition by computing the disjunctive normal form of Boolean expression and obtain the best security policy by optimization theory. This method simply considers the relationship between host nodes, but does not consider vulnerability itself and its utilization relation. In addition, the complexity of this method is exponentiation, and it is not suitable for large-scale network scenarios.

In literature [13], a network vulnerability assessment system based on MulVAL and attack graph is proposed. The system improves the PageRank [14] algorithm and proposes an Asset Rank algorithm that can analyze the large scale attack graph. The algorithms are classified according to the importance of the nodes, and different colors are used to mark different levels, so it is convenient for the security personnel to deal with the problem of vulnerability. Although the system has achieved vulnerability analysis and access control, it lacks consideration for the vulnerability of the weakest level division and vulnerability utilization of the host. In literature [15], a vulnerability assessment model and method based on attack graph is proposed. The model weights and disassembles the attack and initial conditions, converts the problem of attack and initial condition to the minimum S-T cut set problem, obtains the key vulnerability by seeking path. The time complexity is O (M+N). The model can achieve vulnerability assessment and security reinforcement, but this method does not consider the problem of repeating strategy in the process of solving the best set. In literature [16], a vulnerability assessment model based on network centrality is proposed. On the basis of CVSS, the model quantifies attack cost of attackers, and analyzes the attack path by quantitative result. Based on the combination of the number of nodes and the connectivity of the node, the critical vulnerability analysis is used to evaluate the vulnerability of the network. But the complexity of the model is too high in the process of solving the key path.

An attack graph can not only describe the attacker's attack scene, but also display the dependency between the vulnerabilities and describe the entire path of the attack. Network vulnerability analysis based on attack graph model is no longer simply considering an isolated vulnerability, but relates the vulnerability of network, and comprehensive analysis and evaluation. Considering the vulnerability and the dependency between them, it can influence the network, and show the attackers' possible attack paths in the form of graph, which is convenient for network administrators to carry out targeted security reinforcement and improve network security level. In this paper, attack graph technology is applied to network vulnerability analysis, and attack graph based evaluation scheme is set up to analyze and evaluate vulnerability from the perspective of network.

The contributions of this paper can be rounded as follows:

1. A vulnerability assessment model based on attack graph is proposed for assessing the vulnerability of industrial internet of things.

2. A vulnerability quantification method based on the largest loss stream is proposed. The mechanism of maximum loss flow is used to characterize the attack path in the network, and the network vulnerability is evaluated according to the maximum loss and the loss of saturation. In the calculation process, when the node size is greater than 300, the time consumption growth rate is significantly less than the breadth-first search method.

3. A search algorithm based on augmented path global optimal attack path is proposed. Avoiding the repeat calculation and obtaining the potential key vulnerability path fast. A better search solution can be obtained when compared with random augmentations and augmentations based on greedy strategies.

The remainder of this paper is organized as follows: Vulnerability analysis model based on attack graph is presented in Section II. Vulnerability quantification based on CVSS is described in Section III. Section IV explains vulnerability attack graph generation process. Optimal attack path determination is described in detail in Section V, and a series of experiments is presented in Section VI. The conclusion is drawn from the research results in Section VII.

II. VULNERABILITY ANALYSIS MODEL BASED ON ATTACK GRAPH

A. VULNERABILITY ASSESSMENT FRAMEWORK

An attack graph is an application of graph theory in the field of network security, which can be used to represent an attack or path, and the node represents the state or property of the host [17]. The attack graph can be divided into state attack graph and attribute attack graph according to the difference of node content. Because the state attack graph will produce ''explosion'' phenomenon due to the number of states of the state. For this reason, this paper uses attribute attack graph to analyze network vulnerability, and proposes a network vulnerability analysis model based on attribute attack graph. The assessment framework is shown in Fig.1.

In Fig.1, the evaluation framework based on attack graph vulnerability analysis model consists of five parts: network elements, attack graph construction, vulnerability hazard assessment, attack path generation and vulnerability assessment. The specific process of evaluation includes the following four steps:

1) On the basis of existing network elements, such as vulnerability information, network topology information and attack scenes, traverse and match the vulnerability nodes, and generated attack graphs. After optimizing the attack map, the loop of the graph will be eliminated by using the sub-graph optimization algorithm, and the global attack graph is constructed.
2) According to the score of CVSS, combining with the attack cost and attack profit, calculate the maximum loss.
3) According to the maximum loss, start the augmented path search with depth priority order. the attack path is set up according to the crowding distance, thus the priority order of the attack path is determined, and the best attack path is determined.
4) According to the best attack path, the key fragile links of the network system are determined, which provides the basis for the security and reinforcement of the administrators.

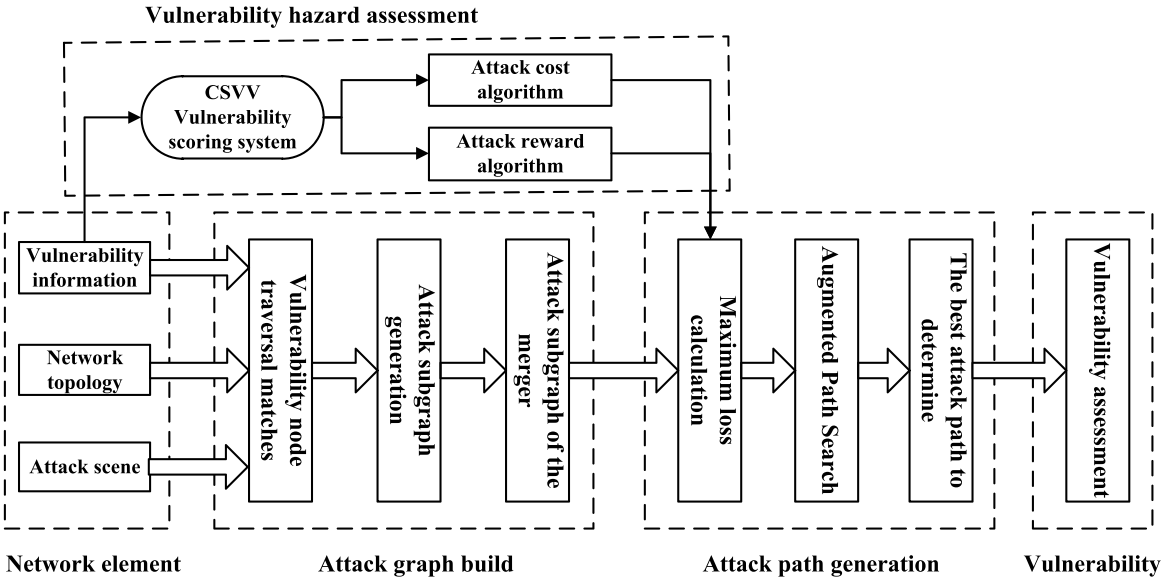


FIGURE 1. Vulnerability assessment framework based on attack graph.

B. MODEL REPRESENTATION OF VULNERABILITY ATTACK GRAPH

In this paper, the traditional attack map is improved, the CVSS index is included in the vulnerability assessment, and the maximum loss degree is used to quantify the network vulnerability.

In the construction of the model, we assume that the attacker is monotonous. That is, the attacker will not attack the completed attack, and will not discard the condition that the attacker has already obtained. Based on the traditional attack graph model, the vulnerability attack graph model can be represented as a nine tuple.

$$VAG = (N, E, C, M, D, IMP, EXP, LC, LF)$$

1) N is node set in network, includes start node N_s, attack target node N_t and attack path node N_p, which is $N = N_s \cup N_t \cup N_p$. If $n_p \in N_p$, exist $e_1, e_2 \in E$, meet $n_p \in pre\,(e_1 \wedge e_2) \in next\,(e_2)$, then N_p is the node on the attack path, set the logical value of node to true.
2) E is edge set of all node, which is $E \subset (N_s \times N_p) \rightarrow (N_p \times N_t)$, each element e in E represents an attack.
3) C is physical connectivity between nodes, which is $C \subseteq (N \times N \times P_t)$, P_t is the interface linking the nodes.
4) M is a map from edge set E to vulnerability set V, which is $M{:}E \rightarrow V$. In vulnerability attack graph, each attack $(e \in E)$ is corresponding to a vulnerability $V\,(e)$, according to the vulnerability information, the loss capacity LC and loss flow LF are confirmed.
5) D is dependence set of vulnerabilities, which is any element in D, meet $d_i{:}v_m \rightarrow v_n$, it is means if the attacker want to use vulnerability v_n, the vulnerability v_m is necessary.
6) EXP is the attack cost along with one path. The detailed calculation process and basis are detailed in the 3th section of the vulnerability quantification process.
7) IMP is the attack reward along with one path. The detailed calculation process and basis are detailed in the 3th section of the vulnerability quantification process.
8) LC is the constraint set of loss capacity on edge E. The loss capacity of edge $eLC\,(e)$ is the maximum loss of whole system when attack with vulnerability corresponding to edge e. The calculation basis is the content of CVSS. The specific process is detailed in the 3th section of the vulnerability quantification.
9) LF is the loss flow set on edge E, which $lf_e \in LF$ is the actual loss flow by attack with edge e, $e \in E$. The loss flow is affected by the attack process and changes as the attack changes. The loss flow and loss capacity also satisfy the capacity constraint and the conservation of flow theorem.

Theorem 1 (Capacity Constraint): The loss flow of an arbitrary edge e is less than equal to its loss capacity, which is

$$\forall e \in E, \begin{cases} lf\,(e) \geq 0 \\ lc\,(e) \geq lf\,\,(e) \end{cases} \tag{1}$$

Prove: In the actual attack, whether an attacker can successfully exploit the vulnerability is related to the individual ability of the attacker and the protection of the network. Therefore, the damage to the system by exploiting the vulnerability is not only related to the CVSS vulnerability score, but also affected by the success probability of the vulnerability $p\,(e) \in [0, 1]$. Thus, $lf\,(e) = p\,(e) * lc\,(e)$. Therefore, The capacity constraint theorem for the loss capacity and loss capacity on the side e is established.

TABLE 1. The measure elements and values of the basic metric group.

Item	Value	Quantification
Attack Vector(AV)	Local(L)	0.395
	Adjacent Network(A)	0.646
	Remote Network(N)	1.000
Attack Complexity(AC)	High(H)	0.350
	Medium(M)	0.610
	Low(L)	0.710
Authentication(AT)	Multiple(M)	0.450
	Single(S)	0.560
	None(N)	0.704
Confidentiality Impact(CI)	None(N)	0.000
Integrity Impact(II)	Partial(P)	0.275
Availability Impact(AI)	Complete(C)	0.660

Theorem 2 (Flow Conservation): The inflow and outflow traffic satisfies the following relationship for any node n_p in the node set N_p of the path.

$$\forall n_p \in N_p, \quad lf^+(n_p) = lf^-(n_p) \tag{2}$$

Prove: $lf^+(n_p)$ is the sum of the loss to attack the node n_p.$lf^-(n_p)$ is potential loss to next node after attack node n_p successfully. According to the monotonicity hypothesis before, the loss caused by the attack will be passed along the attack path and kept in and out of each node.

III. VULNERABILITY QUANTIFICATION BASED ON CVSS

In order to solve the problem of the low degree of attack path quantization and change the calculation mode of the original attack probability, the CVSS scale of the vulnerability is introduced in this paper. CVSS is currently the most widely used vulnerability scoring system [18]. It is part of the secure content automation protocol (SCAP), and is supported by the US national vulnerability Library (NVD) [19].

This article consists of the basic metric group, the time measure group, and the environment measure group to form the CVSS scale. The main function of the time measure group and the environment measure group is to modify the basic metric group. Because this chapter focuses on the vulnerability assessment of network system, in order to simplify the problem, we do not consider the impact of time measurement group and environment measure group on quantization. We consider only the elements of basic measurement group on the path loss flow calculation. The measurement elements and values of the basic metric group are shown as shown in Table 1.

The following concepts and quantization processes used in the process of generating the vulnerability attack map are given.

Let p be an attack path, e is an edge on the path, the attack cost EXP and attack reward IMP can be calculated by formula(3) and (4).

$$IMP(e) = 10.41 * (1 - (1 - CI) * (1 - NI) * (1 - AI)) \tag{3}$$

$$EXP(e) = 20 * AV * AC * AT \tag{4}$$

$LC(e)$ is the difference between attack reward $IMP(e)$ and attack cost $EXP(e)$

$$lc(e) = IMP(e) - EXP(e) \tag{5}$$

Maximum loss d_m is the max flow loss in the preceding attack. The calculation process is:

$$d_m(p) = \sum\nolimits_{p \in P} lf^- \tag{6}$$

Loss saturation γ is the ratio of maximum loss d_m to the sum of loss capacity in the preceding attack. The calculation process is:

$$\gamma = \frac{d_m(p)}{\sum_{e \in p} lc(e)} \tag{7}$$

Node connectivity is represented by $C(n_i, n_j, pt)$, n_i and n_j is connectedness node, pt is connection port. The whole network connected matrix M_c is shown in formula (8).

$$M_c = \begin{bmatrix} - & y & \cdots & pt_{1n} \\ y & - & \cdots & n \\ \vdots & \vdots & \ddots & \vdots \\ pt_{n1} & n & \cdots & - \end{bmatrix} \tag{8}$$

The element at row n and column m is the connectivity between node n and node m. If nodes is connect, the element value is pt, otherwise the value is n. If nodes is physical connected, but not logical connected, the value is y.

IV. VULNERABILITY ATTACK GRAPH GENERATION

The formation process of the vulnerability attack map includes three parts: the traversing of the fragile node, the generation of the attack graph, and the generation of the global attack graph by the sub-graph. The following three parts are described respectively.

A. TRAVERSAL AND MATCHING OF VULNERABILITY NODES

The basic idea of traversing the vulnerability node is to match the condition of the vulnerability node attack process. An attack tree is generated for each vulnerability node that consists of only the current vulnerability node and its sub nodes.

Before the vulnerability of nodes is traversed, the attack mode is instantiated based on the network connectivity C and the vulnerability information V. The instantiated pattern is expressed in EP, including the preconditions and consequences of the attack. $EP = Cdn\_P \cup Cdn\_N$, Cdn_P is attack precondition set, Cdn_N is attack consequence set. Then call the vulnerability node matching algorithm *VNodesMatch*, shown in Algorithm 1, to generate vulnerability node.

Algorithm 1 Vulnerability Node Matching Algorithm

Input: precondition set *PNodes*, attack instance set *EP* and newly added node *Node*
Output: vulnerability node *VNodes* and edge set *E*

```
if EP ≠ ∅ then
  return
end
for each ep ∈ EP
  if Node ∈ ep.Cdn_P ⋀ (ep.Cdn_P − {Node}) ⊂ PNodes then
    create start attack node ts
    VNodes ← VNodes ∪ {ts}
    for each tp ∈ ep.Cdn_p
      E ← E ∪ {< tp, ts >}
      TNodes ← ∅
    for each tn ∈ ep.Cdn_n
      if tn ∉ PNodes then
       PNodes ← PNodes ∪ {tn}
      end
      TNodes ← TNodes ∪ {tn}
      E ← E ∪ {< ts.tn >}
      EP ← EP − ep
      for each t ← TNodes
        VNodesMatch (PNodes, EP, t)
      end
    end
  end
return VNodes, E
```

VNodesMatch is a recursive algorithm, include three parameters, *PNodes*, *EP* and *Node*. The output is *VNodes* and edge set *E*. *VNodes* is vulnerability node set, *Node* is new node at each iterator. At initial state, *VNodes* and *E* are empty, all nodes are in the *PNodes*. *EP* is a set of all attack preconditions and consequences. The element in *EP* should be removed when the element was used until *EP* is empty.

First in *EP*, find out if there is an available attack instance. If this instance exists, find out if there is an attack instance with the new node *Node* as the premise of the attack, and the other premises of the attack instance are also conformed, indicating that the node is an usable vulnerability node. Then, put the vulnerability node into the *VNodes*, find out the previous node, build the edge *e* with previous node and add it to the edge set *E*. If the result node of the attack is not in the former node set *PNodes*, the node is added to the set *PNodes* of the former node and the temporary node set *TNodes*. Establish the edge of the node and the consequence node and put it in the edge set *E* and remove the attack instance from the *EP*. Finally, call the method *VNodesMatch* for each element in set *TNodes* until the set *EP* is empty.

B. ATTACK GRAPH GENERATION

The generation of the vulnerability attack graph is based on the traversal of the last section of the vulnerability node, which will generate the attack sub-tree, merge and eliminate the loop process. The process of generating the vulnerability attack map includes two processes of the attack sub-tree (the vulnerability node) and the attack graph optimization.

In the process of merging the vulnerability nodes, firstly go through the vulnerability node set *PNodes*, then search the edge which start with the current node, get the node which end with the edge and take the node as a root node to generate global attack graph *VAG*. This process is relatively simple, not too much to explain, and then focus on the optimization of the attack map process.

After the attack target is determined, the whole attack graph is searched by the target node as the starting point of the reverse depth. The optimized algorithm is recursive, includes four parameters n_s, n_t, *VAG* and output result *OAG*. *VAG* is global attack graph, n_s is target node, n_t is child node of target node, *OAG* is optimal attack graph to be generated.

The algorithm starts with the target node n_s, search the vulnerability node *VNodes* outside the ring from the attack full graph *VAG* and the side of the node, put the edges and nodes to the optimal attack graph *OAG*. Then, set the target node n_s as the parent of current node, the child node n_t as the target node. Call the function *OptGraph* recursively to traverse all the node.

When the algorithm searches for a certain path, if the node is already in the optimal attack graph, it indicates that the node has been extended, and add the edge of the node n_t to the sub-graph *OAG*. In order to do this, the algorithm uses the set *TP* to record the traversal of the former node on a certain path. If the node appears in the collection *TP*, it indicates that the loop appears in the path, and the search is continued according to the other path.

C. ALGORITHM COMPLEXITY ANALYSIS

The generation of vulnerability attack graph consists of two parts: the traversal matching of the vulnerability node and the generation of the attack graph. The number of elements in the set of attack instances *EP* is n_{ep}, and the number of elements of the reachable precursor property *PNodes* is n_p, and the number of elements of the edge set *E* is n_e. In the process of traversal of the vulnerability node, the n_p times of the matching algorithm need to be called recursively, and each call requires *EP* traversal of the n_{ep}. So, the complexity of the frailty node traversal matching algorithm is $O\left(n_{ep} \cdot n_p\right)$. In the process of attack graph generation and optimization, algorithm 2's step 3 recursively calls along the edge of the attack path, the number of edges is n_e, which is the complexity of algorithm, that is $O\left(n_e\right)$. The time complexity of the generation algorithm of the vulnerability attack graph is $O\left(n_e\right) + O\left(n_{ep} \cdot n_p\right) = O\left(n_{ep} \cdot n_p\right)$.

V. OPTIMAL ATTACK PATH DETERMINATION

A. MAXIMUM LOSS FLOW ALGORITHM

The attack graph built through the last section can record attack and attack process very well, and analyze the vulnerability of the system through its purpose and intention. Therefore, the maximum flow method can be used to calculate the

Algorithm 2 Attack Graph Optimization Algorithm

Input: global attack graph *VAG*, target node n_s, child node n_t
Output: optimal attack graph*OAG*

if $N_t = PNodes$ **then**
  if $PNodes \neq \emptyset$ **then**
    $E \leftarrow E \cup \{< n_s, n_t >\}$
  end
  if $n_s \in PNodes$ **then**
    return
  else
    $PNodes \leftarrow PNodes \cup \{n_s\}$
    $TP \leftarrow TP \cup \{n_s\}$
  end
end
for each $t \in Parent\ (n_s)$
  $OptGraph\ (VGA, t, n, OAG)$
  $TP \leftarrow TP - \{v\}$
  if $n_s \in VNodes \bigwedge \left(pre\ (n_s) \bigcap TP = \emptyset\right) \subset PNodes$ **then**
    $VNodes \leftarrow VNodes \cup \{n_s\}$
  end
  $E \leftarrow E \cup \{< n_s, n_t >\}$
  for each $s \in pre\ (n_s)$
    $OptGraph\ (VGA, t, n, OAG)$
  end
end
return *OAG*

loss saturation γ and the maximum loss d_m on the path to represent the resulting vulnerability loss.

When the attack path of the vulnerability attack graph is generated, the attack graph is converted to a weighted matrix T_w. The value of the element $T_w\ [i, j]$ in the matrix is the loss flow on the $e\ (i, j)$ between node i and node j, that is, $lc\ (e)$.

The algorithm starts from the root N_s of the optimized attack graph *OAG*, and searches for the path of the target node N_t according to the depth first. According to the formula (3) and (4), the return and attack returns of each search path are calculated and the reward matrix T_d is generated. The specific content of T_d is shown as a formula (9). Each column $col\ (i) = [IMP\ (i)\ , EXP\ (i)\ , Index\ (i)]$ in the matrix represents the return of the attack, the overhead, and the number of the path of the path i.

$$T_d = \begin{bmatrix} IMP\ (1) & \cdots & IMP\ (n) \\ EXP\ (1) & \cdots & EXP\ (n) \\ Index\ (1) & \cdots & Index\ (n) \end{bmatrix} \tag{9}$$

Each attack path is sorted by the function *MT_Sort*. The sort is based mainly on the overhead of the attack *EXP* and the return *IMP*. The sorting result of the function *MT_Sort* determines the augmented order of the loss flow in the graph. The attacker will continue to search forward according to the augmented order, and then calculate the loss saturation γ and the maximum loss d_m. The pseudo code description of the maximum loss flow algorithm *MaxLossFlow* is shown in algorithm 3.

Algorithm 3 Maximum Loss Flow Algorithm

Input: matrix with weight T_w, root node n_s
Output: loss saturation γ, maximum loss d_m, key path P

Initialize n_s, P
$P = Path_Seek\ (n_s)$
for each $p \in P$
  Compute $IMP\ (p)$, $EXP\ (p)$
  $T_d \leftarrow T_d \cup \left\{[IMP\ (p)\ , EXP\ (p)\ , Index\ (p)]^T\right\}$
end
$MT_Sort\ (T_d)$
$d_m \leftarrow 0$
while $T_d \neq \emptyset$ **then**
  $p \in P$ **by** $Index\ (i)$
  $\omega \leftarrow min\ \{sub\ (lc\ (i, j)\ , lf\ (i, j)) \mid (i, j) \in p\}$
  $T_w\ [i, j] \leftarrow \omega$
  $T_w \leftarrow T_w \cup \{T_w\ [i, j]\}$
  Update T_w
  $d_m = \sum lf_p^-$, $\gamma = \frac{d_m}{\sum_{e \in p} lc(e)}$
end
return γ, d_m, P

B. PATH SEEKING

The main function of *Path_Seek* in algorithm 3 is to find the possible attack path. The process of searching for algorithms is to find an augmented edge. The augmented edges of the algorithm are defined as the edges that conform to the capacity constraint theorem 1. Because the attackers are greedy in the actual attack process, they hope to increase the probability of success through as many ways and means as possible, and reach the goal with the least steps. To this end, the algorithm uses a forward depth first method to search. First, the algorithm starts from the initial node n_s to find an augmented edge. In the augmented process, the state of each of the direct child nodes n_c of the n_s is checked. If the direct child node n_c is not labeled as access, the n_c is used as the next hop, and the side between the n_s and the n_c is placed in the path p. On the other hand, if n_c has been accessed, it selects its direct child node to check until the path reaches the target node. The algorithm uses the roulette algorithm *RWS* to mark the access of nodes. It uses three colors of white, gray and black to indicate the three states that nodes are not accessed, need to continue searching, and have been visited.

C. MULTI-OBJECTIVE AUGMENTED ROAD SORTING

The main function of the function *MT_Sort* in algorithm 3 is the multi-objective ordering of the augmented path. Function *MT_Sort* is sorted by calculating crowding distance. The calculation method of crowding distance is shown in formula (10).

$$\begin{aligned} \Delta d\ (i) = &|IMP\ (i-1) - IMP\ (i+1)| \\ &+ |EXP\ (i-1) - EXP\ (i+1)| \end{aligned} \tag{10}$$

Algorithm 4 Path Seeking Algorithm

Input: root node n_s
Output: key path P
Initialize P
for each $n_c \in ChildrenNodes\ (n_s)$
  $RWS\ (n_c)$
switch $color\ (n_c)$
  case *white*:
    $n_s \leftarrow preaecessor\ (n_c)$
    $p \rightarrow add\ (n_c)$
    $dfs\ (n_c)$
  case *gray*:
    $n_s \leftarrow traceback\ (n_c)$
  case *black*:
    if $n_c \rightarrow next = n_g$ **then**
      $p \rightarrow add\ (p)$
      $p = null$
    end
end
return P

Algorithm 5 Multi-Objective Augmented Road Sorting Algorithm

Input: reward matrix T_d
Output: sorted matrix $T_d^{'}$
for each $col\ (i) \in T_d$
  sort *IMP* by **asc**
  if $IMP\ (i) = IMP\ (i+1)$ **then**
    sort *EXP* by **asc**
  end
end
compute $\Delta d\ (i)$
while $\Delta d \neq Sort\ (\Delta d)$ **then**
  $T_d^{'} = crowdSort\ (T_d)$
end
return $T_d^{'}$

The pseudo code of algorithm *MT_Sort* is shown in algorithm 5.

D. ALGORITHM COMPLEXITY ANALYSIS

The optimal path determination algorithm based on the maximum flow includes three parts: the maximum loss flow algorithm, the path search algorithm and the multi-objective augmented road sorting algorithm. The number of elements of the vulnerability node *VNodes* is n_v, and the number of bars of the attack path is n_p. According to algorithm 3, it is known that its time consumption is mainly the traversing process of path P, and its complexity is $O\left(n_p\right)$. According to algorithm 4, it is known that the time consumption is mainly a recursive process with node as the root, and the number of algorithms is n_v^2, and its complexity is $O\left(n_v^2\right)$. The time complexity of the augmented road sorting algorithm 5 is $O\left(n_p log\left(n_p\right)\right)$. It can be seen that the time complexity of the optimal path generation algorithm is $O\left(n_p\right) + O\left(n_v^2\right) + O\left(n_p log\left(n_p\right)\right) = O\left(n_v^2\right)$.

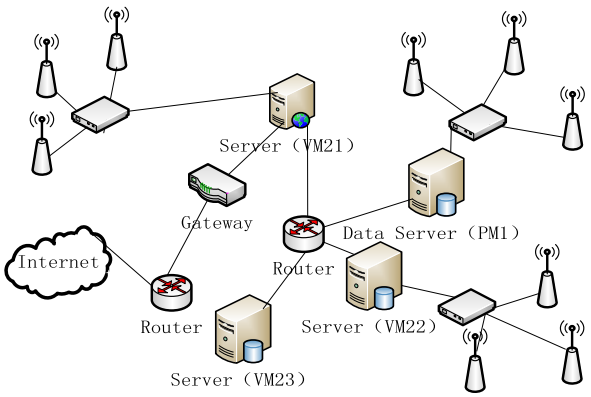


FIGURE 2. Network topology.

TABLE 2. Host configuration information.

Host	OS	Function	Server	Vulnerability
VM21	Redhat 5.4	Web Server	HTTP, SSH	Remote, Execute arbitrary code (2011-2688)
PM1	Redhat 5.4	Database Server	SSH	Remote, Bypass access mechanism (2012-2122) Local, Elevated permissions (2010-2693)
VM22	Redhat 5.4	File Server	FTP, SSH	Remote, Execute arbitrary code (2011-4130) Remote, Bypass access mechanism (2012-6067)
VM23	Redhat 5.4	Host	SSH	Remote, Obtain arbitrary permissions (2008-3234)

VI. EXPERIMENTAL SIMULATION

A. EXPERIMENTAL ENVIRONMENT

The network topology of the experimental environment selection is shown in Fig.2. It is composed of Virtualization Server and physical server, including 2 virtualization servers and 1 physical servers. The configuration and function information of the server are shown in Table 2.

In real life, the purpose of all kinds of attackers to carry out network attacks is to obtain all kinds of resources and data on the server [20]–[22]. In the experiment, it is assumed that

an attacker takes the Root permissions in the target database PM1 as the ultimate goal to obtain business data.

B. AN ATTACK GRAPH OF THE EXPERIMENTAL SCENE

In order to achieve this goal, an attacker can attack by a variety of ways and means. The first way, attackers can scan through Web Vulnerability Scanner [23], XScan [24] and other tools to find the SQL injection vulnerability on the web server VM21 (2011-2688). Through this vulnerability, the attacker gets the user rights of the VM21 and establishes a connection with the database server PM21 on the VM21 with a legitimate identity. Then through the (2012-2122) and (2010-2693) vulnerabilities on the server PM21, the access mechanism is bypassed, and the local authority is carried out. Finally get the root permissions of the database server PM21. Second way, attacker can also improve VM22 permissions through a vulnerability (2012-6067) and (2011-4130) on the file server VM22, then infiltrate to the database server PM21, and finally get the root permissions. Third way, attackers reach the ultimate goal by directly attacking the database server PM21.

As a result of the above description, it can be seen that an attacker can achieve a goal in a variety of ways. According to the definition of the attack graph model and the attack graph generation algorithm in the 4th section, the attack graph of the experimental scene can be obtained, and the specific information is shown in Fig.3.

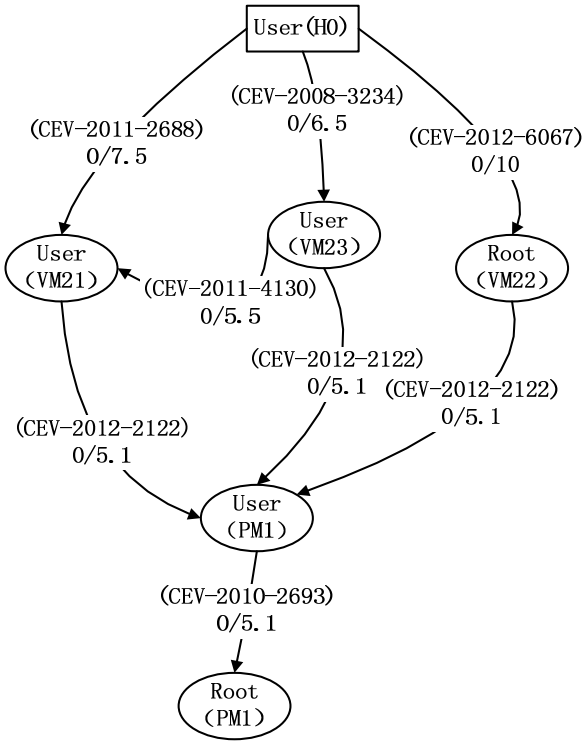


FIGURE 3. An attack graph of the experimental scene.

C. PERFORMANCE ANALYSIS OF ATTACK PATH SEARCH

The last section has already given the attack map of the experimental scene, and then we are looking for the best attack path. Firstly, the performance of the path search algorithm is compared and analyzed. From the 5th section of path search algorithm, we can see that the depth loss algorithm based on depth first proposed in this paper is consistent with the time complexity of the wired search algorithm. In order to verify the rationality of the maximum loss flow algorithm, the experimental simulation method is used to compare this method and the breadth first augmented road search [25]–[27]. In the process of simulation, the scale of the nodes is set to 100, 200, 300, 400, 500, 600 respectively, and the search time is compared. The comparison results are shown in Fig.4. From the contrast results of the graph, it can be seen that both of the two are increasing with the size of the node, and the time of the algorithm is increasing. But we can see that when the size of nodes is larger than 300, the depth loss algorithm based on depth first proposed in this paper, the time consumption growth rate is significantly less than that based on breadth first search algorithm.

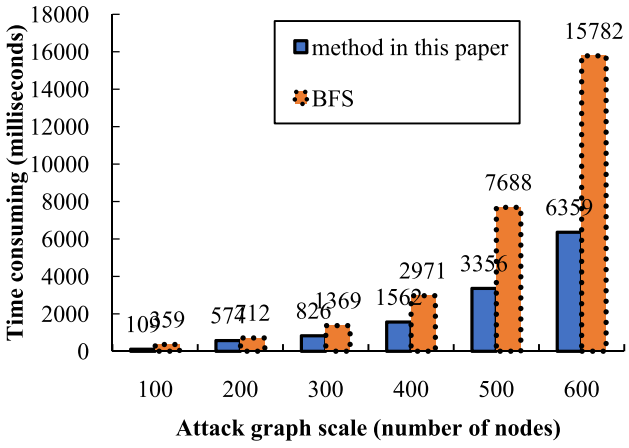


FIGURE 4. Path search performance of different nodes.

D. ANALYSIS OF QUANTITATIVE RESULTS

As described in Section 2 and 3, this article incorporates CVSS indicators into the vulnerability assessment process. The attack cost and reward are introduced into the attack graph, and a vulnerability assessment model based on attack graph is established, so that we can grasp the overall vulnerability of the network and identify the key vulnerable links. In order to evaluate the vulnerability of the sample scenario, we calculate the attack cost and reward on each side of the attack map based on table I given in the 3th section, and then calculate the attack cost and reward on different attack paths. Attack cost and reward on each side, as shown in Table 3, attack cost and reward on each attack path as shown in Table 4.

Tables 3 and 4 show attack cost and reward on each side and path of the attack diagram, respectively. Because of the different vulnerabilities on each side, the cost and reward of the attack are not the same. The attack path, which consists of

TABLE 3. Cost and reward on each side.

Index	Edge	Cost	Reward
$e_{1,2}$	$S1 \rightarrow S2$	9.9968	0.6189
$e_{1,3}$	$S1 \rightarrow S3$	7.9520	0.6189
$e_{1,4}$	$S1 \rightarrow S4$	9.9968	0.9607
$e_{2,5}$	$S2 \rightarrow S5$	4.9280	0.6189
$e_{3,2}$	$S3 \rightarrow S2$	7.9520	0.9607
$e_{3,5}$	$S3 \rightarrow S5$	9387	0.9607
$e_{4,5}$	$S4 \rightarrow S5$	4.9280	0.6189

TABLE 4. Cost and reward on each path.

No.	Edge	Cost	Reward	Index
$R1$	$e_{1,2}, e_{2,5}$	14.9248	1.2378	1
$R2$	$e_{1,3}, e_{3,5}$	11.8907	1.5796	3
$R3$	$e_{1,4}, e_{4,5}$	14.9248	1.5796	4
$R4$	$e_{1,3}, e_{3,2}, e_{2,5}$	20.8320	2.1985	2

different sides, has different reward. Through the data in the table, we know that attackers will always choose the attack cost with the least cost, but will get the most aggressive attack path. Therefore, the vulnerability assessment of the network should also take into account the characteristics of the attacker.

In order to better compare the attack paths, according to the data in Table 4, the cost and reward contrast curves of the attack path can be drawn, as shown in Fig.5.

From the graph, we cannot easily compare any two of the paths in the path. For example, path $R1$ and $R4$, although $R1$'s attack returns are lower than $R4$, the corresponding attack overhead $R1$ is also lower than $R4$. To this end, the multi-objective sorting algorithm in the 5th section is used to analyze and sort the path. In the attack graph model, the order of attack is the same as the augmented order of the loss flow. In the process of multi-objective sorting, the use formula (10) is used to calculate the crowding distance and order it. The result of the sorting is the augmented order in Table 4. According to the results of multi-objective ordering, the augmented process of the loss flow in the experimental network is shown in the order of Figure 6.

From Figure 6, it can be seen that the attacker started from the path $R1$ and augmented in sequence $R1 \rightarrow R4 \rightarrow R2 \rightarrow R3$. Firstly, the attacker increases the loss flow value on the edge of the $R1$. In the process of enlargement, if any edge loss traffic reaches the upper limit of capacity, it will continue to augment other paths until the end of path $R1$ is extended, until all paths cannot find the edge that continues to increase the loss traffic. At this time, the network reaches the maximum flow and satisfies the law of conservation of flow.

In order to better evaluate the crowding distance sorting scheme based on maximum loss of flow, with the increase of random [28] and greedy strategy [29] based on augmented

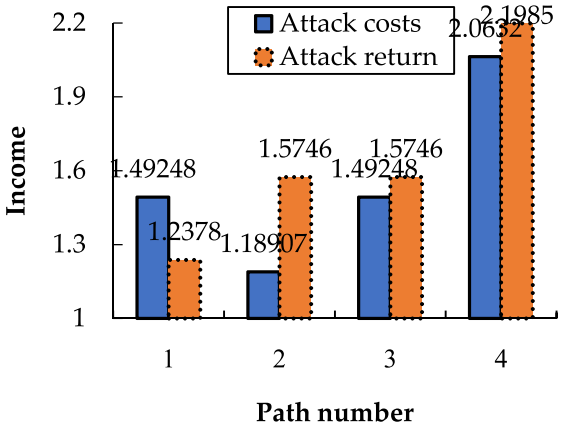


FIGURE 5. Cost and reward on each path.

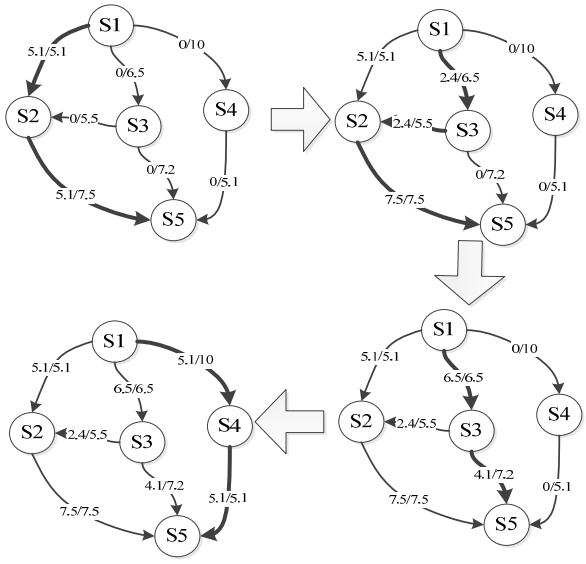


FIGURE 6. Augmented order process of maximum loss flow.

method comparison. Fig.7 is an unordered and random augmentation method, which randomly selects the edge in the augmented process. The order in which the path is augmented is $R4 \rightarrow R1 \rightarrow R2 \rightarrow R3$.

Fig.8 is an augmented process of adopting a greedy strategy. In the process of augmentation, it is the choice to augment the largest edge of the current loss flow. It can be seen from the figure that the order of augmentation is $R2 \rightarrow R1 \rightarrow R3 \rightarrow R4$.

The two methods can all solve their maximum loss flow. The maximum loss flow obtained by the unordered stochastic augmentation is 16, and the maximum loss flow in the way of greedy augmentation is 16.7. The detailed comparison between the two methods and the method based on the crowding distance sorting is shown in Table 5.

The zero flow in the table represents an attack path that the attacker completely ignores. Compared with the other two

TABLE 5. Comparison of three different augmented ways.

Method	Order	Max Loss Flow	Zero Flow	Full Flow	Loss Saturation
Crowding Distance Sorting	$R1 \rightarrow R4 \rightarrow R2 \rightarrow R3$	16.7	No	$e_{1,2}, e_{1,3}, e_{2,5}, e_{4,5}$	76.17%
Unordered Random	$R4 \rightarrow R1 \rightarrow R2 \rightarrow R3$	16	No	$e_{1,3}, e_{3,2}, e_{2,5}, e_{4,5}$	71.06%
Unordered Greed	$R2 \rightarrow R1 \rightarrow R3 \rightarrow R4$	16.7	$\boldsymbol{e_{3,2}}$	$e_{1,2}, e_{1,3}, e_{4,5}$	58.94%

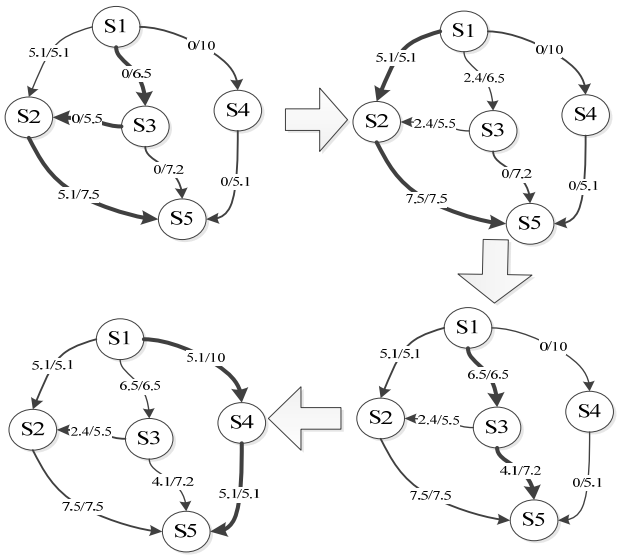


FIGURE 7. Random augmentation process.

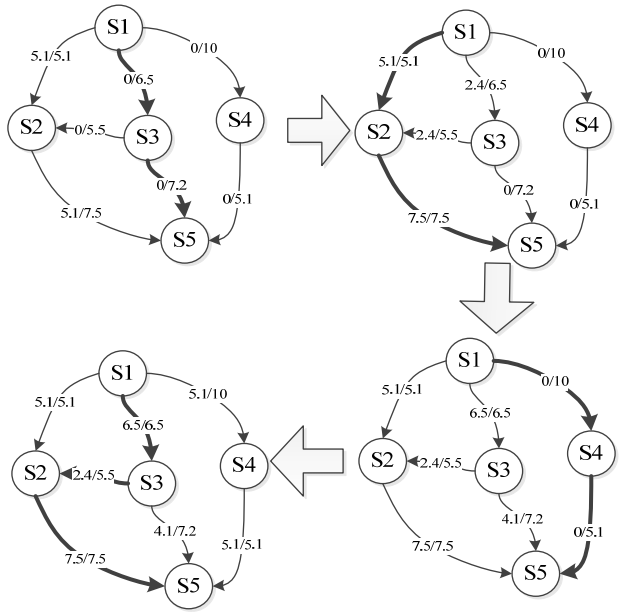


FIGURE 8. The augmented process of greed.

schemes, the maximum loss flow scheme based on crowding distance sorting is the best result. Mainly based on the following reasons:

1) The maximum loss flow scheme of crowding distance ranking, and the maximum loss flow value of the best path is obtained.
2) There is no zero-flow phenomenon in the final result of the solution, which is in accordance with the actual attack process.
3) The final result of this method includes four sides of full flow, and the loss saturation is also the largest of the three schemes.

VII. CONCLUSION

This study focuses on vulnerability assessment of industrial internet of things and proposes a new vulnerability assessment method. Many experts and scholars quantify and assess vulnerability based on Bayesian network separation, attack graph and network centricity. However, there are some problems that the attack path has a low degree of quantization and a complicated way of seeking paths. This study proposes a vulnerability graph model based on attack graph and a vulnerability algorithm based on maximum loss stream. The CVSS scale is used to quantify and calculate the potential risk of attack path. The mechanism of maximum loss flow is used to characterize the attack path in the network, and the network vulnerability is evaluated according to the maximum loss and loss saturation. The augmented path is used to estimate the global path searched for and determine the optimal attack path. The proposed vulnerability assessment method solves the problems such as the portrayal of vulnerability relation and repeated computation of attack path, and realizes the fast and accurate solution to the potentially critical vulnerable path of the network.

ACKNOWLEDGEMENTS

The authors express their sincere appreciation to the editors and the anonymous reviewers for their helpful comments.

REFERENCES

[1] J. Li, L. Huang, Y. Zhou, S. He, and Z. Ming, "Computation partitioning for mobile cloud computing in a big data environment," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2009–2018, Aug. 2017.
[2] H. Wang, J. Gu, X. Q. Di, D. Liu, J. P. Zhao, and X. Sui, "Research on classification and recognition of attacking factors based on radial basis function neural network," *Cluster Comput.*, vol. 20, pp. 1–13, Nov. 2017, doi: 10.1007/s10586-017-1371-9.
[3] Y. Yun, S. X. Xi, and J. Yan, "An attack graph-based probabilistic computing approach of network security," *J. Comput. Sci.*, vol. 33, no. 10, pp. 1987–1996, 2010.

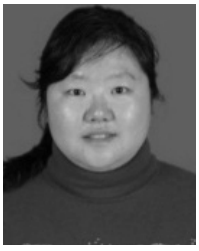
[4] A. Yang, Y. Han, and Y. Pan, "Optimum surface roughness prediction for titanium alloy by adopting response surface methodology," *Results Phys.*, vol. 7, no. 4, pp. 1046–1050, 2017.
[5] L. Wang, S. Noel, and S. Jajodia, "Minimum-cost network hardening using attack graphs," *Comput. Commun.*, vol. 29, no. 18, pp. 3812–3824, 2006.
[6] Y. Sun, H. Qiang, and X. Mei, "Modified repetitive learning control with unidirectional control input for uncertain nonlinear systems," *Neural Comput. Appl.*, vol. 28, pp. 1–10, Apr. 2017, doi: 10.1007/s00521-017-2983-y.
[7] S. Jajodia and S. Noel, "Topological vulnerability analysis: A powerful new approach for network attack prevention, detection, and response," *Algorithms, Archit. Inf. Syst. Secur.*, vol. 3, no. 11, pp. 285–305, 2008.
[8] K. Cui, W.-H. Yang, and H. Gou, "Experimental research and finite element analysis on the dynamic characteristics of concrete steel bridges with multi-cracks," *J. Vibroeng.*, vol. 19, no. 6, pp. 4198–4209, 2017.
[9] R. E. Sawilla and X. Ou, "Identifying critical attack assets in dependency attack graphs," in *Proc. 13th Eur. Symp. Res. Comput. Secur.*, Málaga, Spain, Oct. 2008, pp. 18–34.
[10] Y. Jin, W. Shu, and Y. Zhi, "An attack graph analysis method based on network flow," *Comput. Res. Develop.*, vol. 48, no. 8, pp. 1497–1505, 2011.
[11] F. Han, S. Zhao, L. Zhang, and J. Wu, "Survey of strategies for switching off base stations in heterogeneous networks for greener 5G systems," *IEEE Access*, vol. 4, pp. 4959–4973, 2016.
[12] W. Jia, D. G. Feng, and Y. F. Lian, "Network-vulnerability evaluation method based on network centrality," *J. Graduate Univ. Chin. Acad. Sci.*, vol. 29, no. 4, pp. 529–535, 2012.
[13] A. Sadighian, S. T. Zargar, J. M. Fernandez, and A. Lemay, "Semantic-based context-aware alert fusion for distributed intrusion detection systems," in *Proc. 7th Int. Conf. Risks Secur. Internet Syst. (CRiSIS)*, Cork, Ireland, Oct. 2012, pp. 1–6.
[14] G. D'Aniello, V. Loia, and F. Orciuoli, "A multi-agent fuzzy consensus model in a Situation Awareness framework," *Appl. Soft Comput.*, vol. 30, pp. 430–440, May 2015.
[15] A. A. Ramaki, M. Khosravi-Farmad, and A. G. Bafghi, "Real time alert correlation and prediction using Bayesian networks," in *Proc. 12th Int. ISC Conf. Inf. Secur. Cryptol.*, Rasht, Iran, Sep. 2015, pp. 98–103.
[16] A. A. Ramaki, M. Amini, and R. E. Atani, "RTECA: Real time episode correlation algorithm for multi-step attack scenarios detection," *Comput. Secur.*, vol. 49, pp. 206–219, Mar. 2014.
[17] P. Szwed and P. Skrzyński, "A new lightweight method for security risk assessment based on fuzzy cognitive maps," *J. Appl. Math. Comput. Sci.*, vol. 24, no. 1, pp. 213–225, 2014.
[18] Q. Hui and W. Kun, "Real-time network attack intention recognition algorithm," *Int. J. Secur. Appl.*, vol. 10, no. 4, pp. 51–62, 2016.
[19] P. L. Shrestha, M. Hempel, F. Rezaei, and H. Sharif, "A support vector machine-based framework for detection of covert timing channel," *IEEE Trans. Depend. Sec. Comput.*, vol. 13, no. 2, pp. 274–283, Mar./Apr. 2016.
[20] Y. Liang, H. V. Poor, and L. Ying, "Secure communications over wireless broadcast networks: Stability and utility maximization," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 682–692, Sep. 2011.
[21] X. Zhu, "Computer network vulnerability assessment and safety evaluation application based on Bayesian theory," *Int. J. Secur. Appl.*, vol. 10, no. 12, pp. 359–368, 2016.
[22] E. Jenelius and L. G. Mattsson, "Road network vulnerability analysis of area-covering disruptions: A grid-based approach with case study," *Transp. Res. A, Policy Pract.*, vol. 46, no. 5, pp. 746–760, 2012.
[23] P. Cynthia and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proc. 9th Conf. Comput. Commun. Secur. Assoc. Comput. Mach. (ACM)*, Washington, DC, USA, Nov. 2002, pp. 71–79.
[24] J. Li, S. He, S. Cai, and Z. Ming, "An intelligent wireless sensor networks system with multiple servers communication," *Int. J. Distrib. Sensor Netw.*, vol. 11, no. 8, p. 960173, 2015, doi: 10.1155/2015/960173.
[25] J.-Q. Li, F. R. Yu, G. Deng, C. Luo, Z. Ming, and Q. Yan, "Industrial Internet: A survey on the enabling technologies, applications, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1504–1526, 3rd Quart., 2017.
[26] K. Cui and X. Qin, "Virtual reality research of the dynamic characteristics of soft soil under metro vibration loads based on BP neural networks," *Neural Comput. Appl.*, vol. 28, pp. 1–10, Jan. 2017, doi: 10.1007/s00521-017-2853-7.
[27] W. Wei, X. Fan, H. Song, and X. Fan, and J. Yang, "Imperfect information dynamic stackelberg game based resource allocation using hidden Markov for cloud computing," *IEEE Trans. Serv. Comput.*, vol. 11, no. 1, pp. 78–89, Jan./Feb. 2016.
[28] C. Ge, Z. Sun, N. Wang, K. Xu, and J. Wu, "Energy management in cross-domain content delivery networks: A theoretical perspective," *IEEE Trans. Netw. Serv. Manage.*, vol. 11, no. 3, pp. 264–277, Sep. 2014.
[29] W. Wei *et al.*, "Gradient-driven parking navigation using a continuous information potential field based on wireless sensor network," *Inf. Sci.*, vol. 408, pp. 100–114, Oct. 2017.

HUAN WANG received the master's degree from the Changchun University of Science and Technology, China. He is currently an Engineer with the School of Computer Science and Technology, Changchun University of Science and Technology.

ZHANFANG CHEN received the master's and Ph.D. degrees from the Changchun University of Science and Technology, China. He is currently an Associate Professor with the School of Computer Science and Technology, Changchun University of Science and Technology.

JIANPING ZHAO received the master's and Ph.D. degrees from the Changchun University of Science and Technology, China. He is currently a Professor with the School of Computer Science and Technology, Changchun University of Science and Technology.

XIAOQIANG DI received the master's and Ph.D. degrees from the Changchun University of Science and Technology, China. He is currently an Associate Professor with the School of Computer Science and Technology, Changchun University of Science and Technology.



DAN LIU received the master's degree from the Changchun University of Science and Technology. She is currently a Lecturer with the School of Computer Science and Technology, Changchun University of Science and Technology.