mathematics

MDPI

Article

Key Vulnerable Nodes Discovery Based on Bayesian Attack Subgraphs and Improved Fuzzy C-Means Clustering

Yuhua Xu [1], Yang Liu [2], Zhixin Sun [3,4,*], Yucheng Xue [4], Weiliang Liao [4], Chenlei Liu [4] and Zhe Sun [4]

[1] Engineering Research Center of Broadband Wireless Communication Technology of the Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 210003, China; xuyh@njupt.edu.cn
[2] School of Computer Science and Technology, Nanjing University of Posts and Telecommunications, Nanjing 210003, China; 1222045809@njupt.edu.cn
[3] Engineering Research Center of Post Big Data Technology and Application of Jiangsu Province, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
[4] Research and Development Center of Post Industry Technology of the State Posts Bureau (Internet of Things Technology), Nanjing University of Posts and Telecommunications, Nanjing 210003, China; b21090431@njupt.edu.cn (Y.X.); b21090430@njupt.edu.cn (W.L.); 2019070270@njupt.edu.cn (C.L.); zhesunny@njupt.edu.cn (Z.S.)
* Correspondence: sunzx@njupt.edu.cn

Abstract: Aiming at the problem that the search efficiency of key vulnerable nodes in large-scale networks is not high and the consideration factors are not comprehensive enough, in order to improve the time and space efficiency of search and the accuracy of results, a key vulnerable node discovery method based on Bayesian attack subgraphs and improved fuzzy C-means clustering is proposed. Firstly, the attack graph is divided into Bayesian attack subgraphs, and the analysis results of the complete attack graph are quickly obtained by aggregating the information of the attack path analysis in the subgraph to improve the time and space efficiency. Then, the actual threat features of the vulnerability nodes are extracted from the analysis results, and the threat features of the vulnerability itself in the common vulnerability scoring standard are considered to form the clustering features together. Next, the optimal number of clusters is adaptively adjusted according to the variance idea, and fuzzy clustering is performed based on the extracted clustering features. Finally, the key vulnerable nodes are determined by setting the feature priority. Experiments show that the proposed method can optimize the time and space efficiency of analysis, and the fuzzy clustering considering multiple features can improve the accuracy of analysis results.

Keywords: Bayesian attack graphs; key vulnerability discovery; community division; fuzzy clustering

MSC: 93C42

Citation: Xu, Y.; Liu, Y.; Sun, Z.; Xue, Y.; Liao, W.; Liu, C.; Sun, Z. Key Vulnerable Nodes Discovery Based on Bayesian Attack Subgraphs and Improved Fuzzy C-Means Clustering. *Mathematics* **2024**, *12*, 1447. https://doi.org/10.3390/math12101447

Academic Editor: Hsien-Chung Wu

Received: 17 April 2024
Revised: 30 April 2024
Accepted: 4 May 2024
Published: 8 May 2024

1. Introduction

With the continuous progress of science and technology, the network has become an indispensable part of modern society. The development of the network has broken the restrictions of time and space and promoted the dissemination and sharing of information. Attackers penetrate and hijack data through device vulnerabilities, causing economic losses to individuals, institutions, large companies, and even countries [1]. Therefore, it is necessary to analyze the vulnerabilities in network systems and take corresponding defensive measures to prevent hacker attacks.

The large number and complex types of vulnerabilities in network systems always threaten the security and stability of the system. Many scholars have applied various methods to study system vulnerability analysis. Some methods evaluate the vulnerability threat in the network system by considering the threat characteristics of the vulnerability itself and the threat of its associated assets [2–5]. However, these methods only evaluate the stable vulnerability threat influencing factors, and do not take the actual changing network

environment into account, so the factors considered are not comprehensive enough. In order to take various factors into account, some scholars use machine learning to evaluate the threat degree of vulnerabilities [6–10]. These studies combine various characteristic information of vulnerabilities and train various models to improve the effect of vulnerability detection. However, these methods only detect the vulnerability of a single device, and the correlation between the detection results and other vulnerabilities in the network and the whole attack process is not strong. The attack graph is a graphical security assessment technique that contains various network configurations and vulnerability information. It reveals all potential vulnerability combinations and their relationships and lists all potential attack paths from the perspective of the attacker to reflect the security state of the network, such as the number of attack paths, the length of the shortest attack path, and the key vulnerability. Multi-step attacks can be effectively prevented based on the attack graph [11,12]. To enhance the relevance between vulnerability assessment and network systems, many studies have conducted network vulnerability analysis based on attack graphs. They realized the association analysis of key vulnerabilities in network systems by studying attack graph construction techniques [13,14], node analysis techniques [15–17] and attack path analysis [18,19]. However, the attack graph does not have the ability of quantitative analysis. The Bayesian theory is a statistical method to deal with uncertainty through observation data. The key of Bayesian theory is to predict possible risks in advance by mathematical methods, and it does not focus on the random attack itself [20]. Many risk analysis methods based on attack graphs combine Bayesian theory to realize risk quantification and prediction analysis [21–23]. In the network attack graph with a large number of nodes, the existing studies have problems such as low efficiency and single consideration when searching for key vulnerable nodes. These problems coincide with the advantages of fuzzy C-means (FCM) clustering, which can integrate various characteristics of vulnerabilities and classify them spontaneously. Thus, a set of key vulnerable nodes with similar threat degrees can be effectively obtained.

To sum up, the temporal and spatial efficiency of attack graph analysis in large-scale networks needs to be improved, and the factors taken into account in searching key vulnerable nodes in network systems are not comprehensive enough. This paper uses the attack graph combined with the network division to divide the attack graph into multiple subgraphs for correlation analysis. Based on the analysis results, the actual threat features of vulnerability nodes are extracted. The actual threat features and inherent threat features of vulnerabilities are taken as FCM clustering indicators. The main contributions of this paper are as follows:

1. An analysis method based on Bayesian attack subgraphs is proposed. It divides the attack graph based on the idea of community division, quantifies the threat of nodes, constructs and analyzes Bayesian attack subgraphs to form the subgraph analysis information group, and aggregates information groups to quickly obtain the final analysis results of all paths so as to improve the spatiotemporal efficiency of the results;
2. A method based on improved FCM to discover key vulnerable nodes is proposed. It uses variance to design the total difference value between classes (TDVC) and determines the optimal number of FCM by maximizing the TDVC. Then, the actual threat features and inherent threat features of vulnerabilities are extracted based on the Common Vulnerability Scoring System (CVSS) and the analysis results of the attack graph. Next, FCM is used to cluster the vulnerability nodes based on the extracted features so as to improve the accuracy of the results. Finally, the feature priority is set, and the key vulnerability node cluster with the highest threat level is found according to the results.
3. The experimental scenario is designed and the data from the National Vulnerability Database (NVD) are collected for the experiment. The temporal and spatial efficiency improvement of attack graph analysis and the accuracy improvement of key vulnerability nodes search results are verified by comparing with other methods.

2. Related Works

At present, there is much research on key vulnerability search. Hao et al. [24] and Tang et al. [25] train neural network models to identify key vulnerabilities on network devices using static analysis, but this method is only applicable to a single device and cannot be dynamically combined with other devices in the network. Li et al. [26] use the Kemeny constant as a global connectivity measure to identify network key connections and network decomposition is used to cut off connections to minimize global connectivity measures, thereby obtaining key vulnerabilities. However, this method is not intuitive enough and only considers a single influencing factor. Huang et al. [27] build an attack tree and conduct Bayesian inference to find key vulnerable nodes by tracing the attack path. However, the application scope of the attack tree is limited and vulnerabilities could not be associated.

To conduct association analysis of network nodes and make the analysis results more accurate and intuitive, many studies use the attack graph as the basic analysis method to search for key vulnerabilities. Yang et al. [28] quantify the asset value of the host through the attribute value and topology perspective on the attack graph and search for key vulnerable nodes in combination with attack probability. However, this method does not consider the influence of the location of the vulnerability in the attack path. Li et al. [29] use attack distance and atomic weight to optimize the complexity of the attack graph and improve the ant colony algorithm to solve the minimum key attack set through the pheromone adaptive update principle and local search strategy so as to obtain the key vulnerable nodes, but this method does not consider the threat characteristics of vulnerability nodes themselves. Qian et al. [30] optimize the attack graph with maximum hop count and reachability probability and quantify the reachability probability of vulnerable nodes. According to the vulnerability measurement value of nodes and paths, the key vulnerable nodes in the network are found, but this method only considers the attack path with the highest vulnerability measurement value. Xie et al. [31] use the Bayesian attack graph model to continuously carry out probabilistic correction learning according to the attack data, quantify the dynamic risk, and evaluate the risk value of key nodes according to the quantitative results. However, this method analyzes the complete attack graph and dynamically adjusts the attack graph model, which reduces the time efficiency of analysis. Li et al. [32] combine particle the swarm optimization algorithm and a grey wolf optimization algorithm to find the maximum weight spanning tree from the attack graph, evaluate the key nodes in the spanning tree based on interpretable structure modeling, and improve the simulated annealing algorithm. This method optimizes the time efficiency through heuristic ideas but reduces the accuracy of the result due to the pruning of part of the edges.

Previous studies have some problems when using the Bayesian attack graph to search for key vulnerable nodes. Many studies do not consider the threat characteristics of the vulnerability itself and the threat characteristics in the actual network at the same time, which leads to the accuracy of the search results of key vulnerable nodes needing to be improved. Some studies will lead to low search efficiency when they are applied to large-scale attack graphs, and large-scale attack graphs also need a lot of storage space. Therefore, this paper proposes a key vulnerable node discovery method based on Bayesian attack subgraphs and improved FCM. The large-scale network is divided by community division and the Bayesian attack subgraphs are constructed. The analysis result of the complete attack graph is formed by aggregating the attack path analysis information inside the attack subgraphs, and the threat characteristics of nodes are extracted from the result. Then, the TDVC is designed to adaptively determine the optimal number of clusters, and the FCM is used to cluster the nodes. Finally, the clustering results are analyzed based on the designed feature priority to obtain the key vulnerable nodes so as to improve the accuracy of results and the spatiotemporal efficiency of search analysis.

3. Key Nodes Discovery Model Based on Attack Subgraph Aggregation Search and Fuzzy C-Means Clustering

The key vulnerable nodes discovery model based on attack subgraph aggregation search and fuzzy C-means clustering is mainly divided into three steps: data preprocessing, attack path aggregation search based on Bayesian attack subgraph, and key vulnerable nodes discovery based on fuzzy C-means clustering. The overall process of the key node discovery model based on attack subgraph aggregation search and fuzzy C-means clustering is shown in Figure 1.

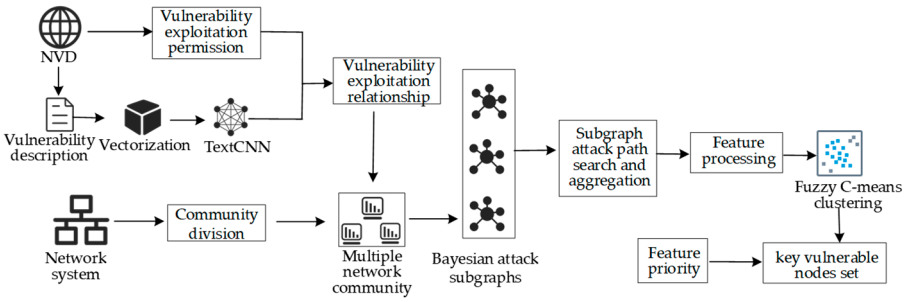


Figure 1. Key nodes discovery model based on attack subgraph aggregation search and fuzzy C-means clustering.

In the data preprocessing stage, the topology graph of the network system is constructed, and vulnerability scanning tools such as Nessus are used to obtain the vulnerability list existing on the device. At the same time, the vulnerability exploit relationship is analyzed for the subsequent construction of the attack graph. Since the current attack graph construction technology generally uses a manual analysis method to obtain the exploitation relationship between vulnerabilities, this method cannot be effectively implemented when there are a large number of vulnerabilities, and manual analysis has a strong subjective will. To make the construction of vulnerability exploitation relationships more accurate, the model in this paper uses the method based on Word2Vec and TextCNN in reference [33] to obtain the exploitation relationship between vulnerabilities. Firstly, the basic information such as permission requirements and description of vulnerabilities is obtained through the interface provided by the NVD, and then the description information is organized into a corpus to train the Word2Vec model. The output of the Word2Vec model is used as the input of TextCNN to train the text classification model. Finally, the permissions obtained after the vulnerability has been attacked are divided into three categories: other, user, and root through the text classification model, and the exploitation relationship between the vulnerabilities is obtained according to the comparison results of the permissions requirements of the vulnerability and the permissions obtained after the attack.

In the attack path aggregation search phase based on the Bayesian attack subgraph, the large-scale network is first divided into multiple communities with close internal connections by the community division algorithm. Then, the Bayesian attack subgraph is constructed in each community based on CVSS, network connection relationship, and vulnerability utilization relationship. Next, the attack path is searched in each subgraph, and the attack probability and other related information of each path are recorded for the aggregation of subsequent paths. Finally, the attack path and its information are aggregated based on the connection relationship between subgraphs to obtain the complete attack path information. Compared with the search of attack paths on the whole attack graph directly, the search of attack paths based on the subgraph can not only save the storage consumption of the attack graph but also improve the search efficiency of attack paths.

In the discovery stage of key vulnerable nodes based on fuzzy C-means clustering, the sample characteristics during clustering are determined first. Two inherent threat features and two actual threat features of the vulnerability are selected as sample features in this method. Inherent threat features include exploitability score and impact score, which

can be directly obtained according to CVSS. The exploitability index of a vulnerability in CVSS reflects the difficulty of exploiting the vulnerability, and the impact index reflects the severity of the consequence of exploiting the vulnerability. Both of them are related to the severity of the vulnerability, but both of them are fixed characteristics of the vulnerability itself, so their applicability is weak. Therefore, this paper extracts two actual threat features of vulnerability attack probability and vulnerability occurrence frequency from the relevant information on attack paths. These two features can reflect the threat degree of vulnerabilities combined with specific attack paths in different networks and improve the accuracy of discovering key vulnerable nodes. Then, the optimal number of clusters is determined by maximizing the difference between the clustering results, and the fuzzy C-means clustering is realized by setting the membership degree, the maximum number of iterations, and other parameters. Finally, the feature priority is set to classify the clustering results so as to obtain the final set of key vulnerable nodes.

4. Attack Path Aggregation Search Based on Bayesian Attack Subgraph

At present, the attack graph analysis scheme involving the attack path is difficult to implement in large-scale network systems due to its space-time complexity. Therefore, an attack path aggregation search method based on a Bayesian attack subgraph is proposed in this paper to improve the spatiotemporal efficiency of this attack graph analysis. The information obtained based on this method will be used as the key features of subsequent clustering algorithms to find key vulnerable nodes. The process of the attack path aggregation search based on attack subgraphs is shown in Figure 2.

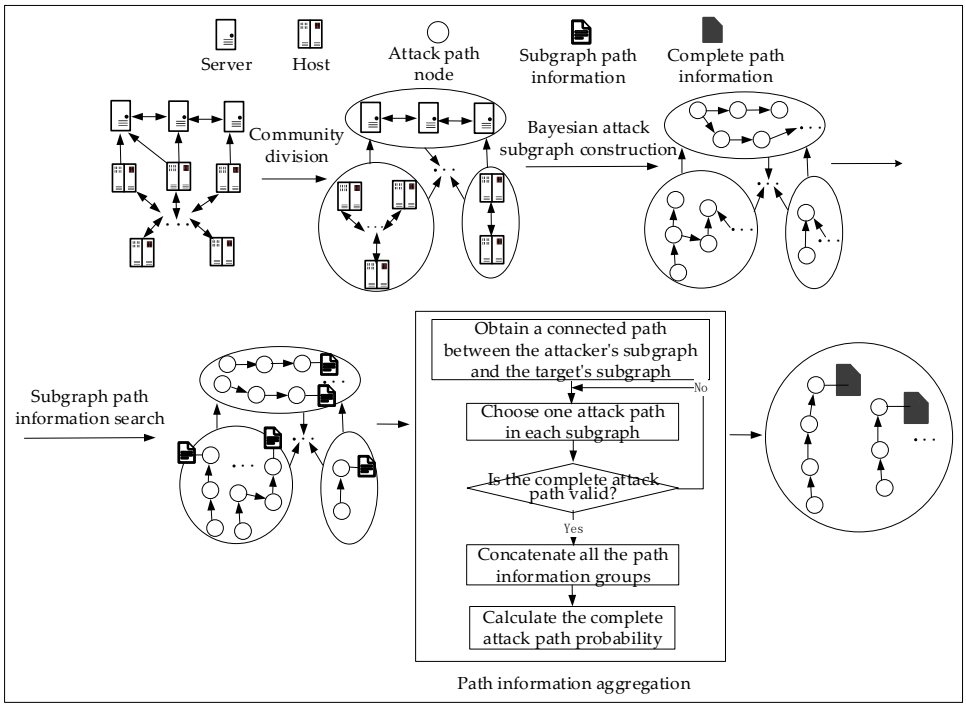


Figure 2. The process of the attack path aggregation search based on attack subgraphs.

In this method, the large-scale network is divided into multiple subnetworks using the network community partitioning algorithm, and the Bayesian attack subgraphs are constructed in the subnetworks first. All Bayesian attack subgraphs form the whole Bayesian attack graph. Then, the attack path is searched in each Bayesian attack subgraph, and the related information of the attack path is recorded. Finally, according to the attack paths in different attack subgraphs and the related information recorded, the attack paths and related information of the whole Bayesian attack graph are obtained.

4.1. Bayesian Attack Subgraph Construction

Definition 1 (The Bayesian attack graph BAG)**.** *The BAG is a directed acyclic graph defined as a quintuple* $< N, E, R, P_a, P_s, P_c >$.

1. N *is the set of nodes.* $N = \left\{ N_{begin} \cup N_{middle} \cup N_{target} \right\}$. N_{begin} *is the set of nodes where the attacker is located in the Bayesian attack graph.* N_{target} *is the node set of the attack target.* N_{middle} *is the set of the remaining nodes. The value of* N_i *can be 0 or 1.* $N_i = 1$ *means that the node i has been compromised.* $N_i = 0$ *means that the node i is not compromised;*
2. E *is the set of directed edges between nodes.* $E = \{E_i | i = 1, 2 \ldots\}$. $E_k =< i, j >$ *means that an attacker at node i can attack node j after having sufficient privileges;*
3. R *is the set of parent–child node relationships in the attack graph.* $R = \left\{ r_{i,par(i)} \middle| i = 1, 2, \ldots \right\}$. $par(i)$ *is the set of parents of node i.* $r_{i,par(i)} =$ *or means that node i can be attacked when any of its parents has been compromised.* $r_{i,par(i)} =$ *and means that node i can only be attacked after all its parents have been compromised;*
4. P_a *is the set of node breach probabilities.* $P_a = \{P_a(i) | i = 1, 2 \ldots\}$. $P_a(i)$ *means the probability that node i is successfully attacked;*
5. P_s *is the set of node selection probabilities.* $P_s = \{P_s(i) | i = 1, 2 \ldots\}$. $P_s(i)$ *means the probability that node i is selected by the attacker as an attack target;*
6. P_c *is the set of conditional probabilities of nodes.* $P_c = \{P_c(i|Parent(i)) | i = 1, 2 \ldots\}$. $P_c(i|Parent(i))$ *means the conditional probability that the node i will be attacked after its parent is compromised.*

Aiming at the problem of efficiency caused by searching the attack path on the whole attack graph in traditional methods, this paper adopts the method of attack path analysis based on Bayesian attack subgraphs. The first step is to partition the network to form Bayesian attack subgraphs. The construction of Bayesian attack subgraphs includes three parts: network partition, attack subgraph construction, and node quantification.

Partitioning a large-scale network into multiple subnetworks is the basis for generating Bayesian attack subgraphs. In this paper, the Lovain algorithm based on modularity evaluation is used to divide the network. Modularity is used to evaluate the closeness of the community structure. The modularity gain reflects the comparison of the modularity of the whole graph when a node is merged from one community to another. The goal of Lovain algorithm partitioning is to maximize the modularity increment. The calculation formula of the modularity increment ΔQ is given in Equation (1).

$$\Delta Q = \frac{1}{2\varpi}(\kappa_{i,in} - \frac{\sum_{tot} \kappa_i}{\varpi}), \tag{1}$$

where $\kappa_{i,in}$ is the sum of edge weights between node i and all nodes in the merged target community. $\sum tot$ is the sum of edge weights related to nodes in the target community. κ_i is the sum of edge weights of node i. ϖ is the sum of all edge weights. In the directed unweighted graph, the weight of each edge can be regarded as one.

The attack subgraph is constructed based on subnetworks formed by community division. For each community network, the vulnerabilities of each host are obtained first. Then, it determines whether there is an exploitation relationship between any two vulnerabilities i and j on any two connected hosts; if there is, a directed edge $E_k =< i, j >$ is added between i and j to represent the attack relationship. After all the vulnerabilities are processed, the vulnerabilities in the subnetwork form N in the corresponding attack subgraph, and the attack relationship between the vulnerabilities form E.

To use Bayesian theory for analysis in attack subgraphs, vulnerability nodes in each attack subgraph need to be quantified to form a Bayesian attack subgraph. Each node i needs to quantify $P_a(i)$ and $P_s(i)$ based on CVSS. $P_a(i)$ is related to the difficulty of the node i being exploited, and the lower the difficulty, the easier it is to be compromised. $P_s(i)$ is related to the attack cost of the node i, and the lower the attack cost, the easier it is to be

selected as an attack object. In CVSS, the attack vector score S_{AV} and the attack complexity score S_{AC} can measure the difficulty of exploiting the vulnerability, while the privileges required score S_{PR} and user interaction score S_{UI} can measure the exploitation cost of the vulnerability. Therefore, $P_a(i)$ and $P_s(i)$ are calculated by Equations (2) and (3), respectively.

$$P_a(i) = S_{AV} \times S_{AC}, \tag{2}$$

$$P_s(i) = S_{PR} \times S_{UI}, \tag{3}$$

According to the different relationship $r_{i,par(i)}$ between the node i and its parent node $par(i)$, the conditional probability $P_c(i|Par(i))$ will be quantized by different methods based on P_a and P_s. When $r_{i,par(i)} = and$, the conditional probability of node i is the probability that each parent node is compromised multiplied by the probability that node i is also compromised. When $r_{i,par(i)} = or$, the conditional probability of node i is the probability that any parent node is compromised multiplied by the probability that node i is also compromised. The calculation formulas of $P_c(i|Par(i))$ in the above two cases are shown as Equations (4) and (5), respectively. In particular, for the node j that has no parent node in the entire attack graph, its conditional probability is calculated as Equation (6).

$$P_c(i|Parent(i)) = \begin{cases} 0,\ N_k \in Parent(i),\ N_k = 0 \\ \prod P_a(Parent(i)) \times P_s(i) \times P_a(i)' \end{cases}, \tag{4}$$

$$P_c(i|Parent(i)) = \begin{cases} 0,\ N_k \in Parent(i),\ N_k = 0 \\ \{1 - \prod[1 - P_a(Parent(i))]\} \times P_s(i) \times P_a(i)' \end{cases}, \tag{5}$$

$$P_c(j|Parent(j)) = P_s(j) \times P_a(j), \tag{6}$$

4.2. Attack Subgraph Paths Search

Definition 2 (The basic attack path L)**.** *L is an attack path inside the Bayesian attack subgraph, which consists of nodes with an attack relationship. $L = (N_0, N_1, \ldots N_n)$. N_i is the node inside the Bayesian attack subgraph.*

Definition 3 (The attack path information group $\boldsymbol{I}$)**.** *$\boldsymbol{I}$ is the matrix used to record the node information that appears in the corresponding attack path. $\boldsymbol{I} = (\boldsymbol{i}_1, \ldots \boldsymbol{i_n})$. $\boldsymbol{i_k}$ is a two-dimensional column vector where the first row is the node number and the second row is the depth of the node.*

Definition 4 (The path reachability probability P)**.** *P describes the possibility of an attacker attacking through a certain path.*

The second step of attack path analysis based on the Bayesian attack subgraph is to search the basic attack path and its corresponding attack path information group inside each attack subgraph, which will be used for subsequent path aggregation and then used as indicators for clustering. Since the analysis speed inside the attack subgraph is better than the analysis speed in the whole attack graph, and the storage space requirement of the attack subgraph is lower than that of the whole attack graph, the purpose of improving the time and space efficiency of the analysis can be achieved. This section will introduce how to search the attack path and record the attack path information inside the established Bayesian attack subgraph.

For each attack subgraph, firstly, the set of nodes whose out-degree value and in-degree value are both 0 are obtained, respectively. Then, the search starts from the node with in-degree 0 as the initial node, and the node connected to the current node is the next node on the attack path. The conditional probability of the nodes on the path is multiplied to calculate the reachability probability P of the basic attack path. After that, the search continues from the next node until the node with out-degree 0 is reached. Finally, a basic attack path is formed, and the information group $\boldsymbol{I}$ corresponding to the basic attack path

is recorded during the search process for subsequent attack path aggregation. The formula for calculating P is shown as Equation (7).

$$P = \prod P_c(i|Parent(i)), \tag{7}$$

The attack subgraph path search algorithm is shown in Algorithm 1.

Algorithm 1: BasicPathSearch

Input: The Bayesian attack subgraph set *BAGs*
Output: The basic path set *Ls* in the Bayesian attack subgraph, the information group set *Is*, and the basic path reachability probability set *Probs*

```
FOR each attack subgraph BAG in BAGs
  The set N_out of nodes whose out-degree is 0 and the set N_in of nodes whose in- degree is 0 in BAG are counted
  FOR each node n in N_in
    Add n to L, store the information of n to I, Prob = P_c(n | Parent(n))
    FOR n's each neighbor node n'
      Add n' to L, store the information of n' to I, Prob = Pc(n' | Parent(n'))
      IF n' ∈ N_out
        Add L, I, Prob to the result sets Ls, Is, and Probs, respectively
      ELSE
        n = n'
```

4.3. Attack Paths and Its Information Aggregation

The reachability probability of the attack path in the Bayesian attack graph describes the possibility of the attacker attacking along the vulnerable nodes on the path. The larger the reachability probability is, the higher the threat degree of the nodes on the path is. Therefore, the reachability probability of the attack path is used as an evaluation index for the subsequent clustering algorithm. This method considers the influence of the node's position in the attack path on its threat degree and makes the search results of key vulnerable nodes more accurate. However, the attack probability is for the attack path, and cannot be used as the vulnerability node feature for clustering. It needs to combine the node depth information in the information group to convert it into the characteristics of the vulnerability node.

The attack path and its information group in the Bayesian attack graph are aggregated based on the basic attack path L in each attack subgraph, the corresponding path information group $\boldsymbol{I}$ and the basic path reachability probability P. Firstly, a directed path from the subgraph of the attacker to the subgraph of the attack target is selected according to the connectivity of attack subgraphs. Let the length of the path be $\mathcal{L}'$. Then, a basic path L_k is selected in each attack subgraph, and the path reachability probability vector $\widehat{\boldsymbol{P}}$ and the attack relation vector $\widehat{\boldsymbol{AR}}$ between paths are constructed based on these paths according to Equations (8) and (9).

$$\widehat{\boldsymbol{P}} = (P_{L_{k_1}}, P_{L_{k_2}}, \ldots, P_{L_{k_{\mathcal{L}'-1}}}, P_{L_{k_{\mathcal{L}'}}}), \tag{8}$$

$$\widehat{\boldsymbol{AR}} = \left(1, ar_{L_{k_1}L_{k_2}}, \ldots, ar_{L_{k_{\mathcal{L}'-1}}L_{k_{\mathcal{L}'}}}\right), ar_{L_{k_{\mathcal{L}'-1}}L_{k_{\mathcal{L}'}}} = 0 \; or \; 1, \tag{9}$$

where P_{Lk1}, P_{Lk2} and so on are the reachability probabilities of the selected basic paths. $\mathcal{L}'$ is the number of subgraphs in the path between subgraphs. $ar_{L_{k_1}L_{k_2}} = 1$ indicates that the tail node of the basic path L_{k_1} can utilize the head node of the basic path L_{k_2}. $ar_{L_{k_1}L_{k_2}} = 0$ indicates that the tail node of the basic path L_{k_1} can not utilize the head node of the basic path L_{k_2}. ψ is the control variable of the attack path, which is used to control the effectiveness of the obtained attack path. When $\psi = 0$, it means that some selected basic

paths are not connected, that is, the corresponding attack path is invalid. Only valid paths are taken into account. The calculation formula of ψ is given in Equation (10).

$$\psi = \prod \widehat{\boldsymbol{AR}} = \prod_{i=0}^{\mathcal{L}'-1} ar_{L_{k_i} L_{k_{i+1}}}, \tag{10}$$

Finally, the reachability probability P' of the aggregated attack path is obtained through Equation (11), and the corresponding information group $\boldsymbol{I}'$ is obtained by merging the information groups of each basic path according to Equation (12).

$$P' = \widehat{\boldsymbol{P}} \times \widehat{\boldsymbol{AR}}^T, \tag{11}$$

$$\boldsymbol{I}' = \left[\boldsymbol{I}_{L_{k_1}}, \boldsymbol{I}_{L_{k_2}}, \ldots, \boldsymbol{I}_{L_{k_{\mathcal{L}'}}}\right], \tag{12}$$

The set of all effective attack paths, the set of corresponding attack path reachability probabilities, and the set of corresponding information groups of the whole attack graph formed by the aggregation will be used as the indicators of the subsequent clustering algorithm. The attack path aggregation algorithm based on the basic path and its information group is shown in Algorithm 2.

Algorithm 2: AttackPathAggregation

Input: The attack connection information C between subgraphs, the utilization relation U of nodes, the basic path reachability probability P, the basic path information group $\boldsymbol{I}$
Output: The aggregated reachability probability P' of the attack path and its corresponding information group $\boldsymbol{I}'$

1. Select an inter-subgraph path $SubBAGPath$ from the subgraph where the attacker is located to the subgraph where the attack target is located according to C.
2. $\widehat{\boldsymbol{AR}} = [1]$, $\widehat{\boldsymbol{P}} = []$, $\psi = 1$, $\boldsymbol{I}' = []$
3. FOR each attack subgraph BAG' in $SubBAGPath$
4. Select a basic path L_k from BAG', whose tail node is $tail$ and head node is $head$, and add P_{L_k} to $\widehat{\boldsymbol{P}}$
5. IF BAG' is not the attack subgraph where the attacker is located
6. IF $U(lastTail, head) = 1$
7. $ar_{L_{last} L_k} = 1$, update $\widehat{\boldsymbol{AR}}$ and ψ
8. ELSE
9. BREAK
10. ELSE
11. $L_{last} = L_k$
12. $lastTail = tail$, add I_{L_k} to $\boldsymbol{I}'$
13. $P' = \widehat{\boldsymbol{P}} * \widehat{\boldsymbol{AR}}^T$

5. Discovery of Key Vulnerable Nodes Based on Improved FCM

When the attack graph is used for security analysis, the threat degree of the vulnerability node is related to various factors, such as the exploitability score and impact score of the vulnerability node in CVSS, the position of the vulnerability node in the attack graph, and the occurrence times of the vulnerability node. Only one factor will lead to inaccurate search results for key vulnerability nodes. Therefore, this paper takes a variety of factors that affect the vulnerability threat degree as the characteristics of vulnerability nodes and uses FCM to cluster them. Then, the feature priority is set to find out the set of key vulnerability nodes with the highest priority. The discovery process of key vulnerable nodes based on improved FCM is shown in Figure 3.

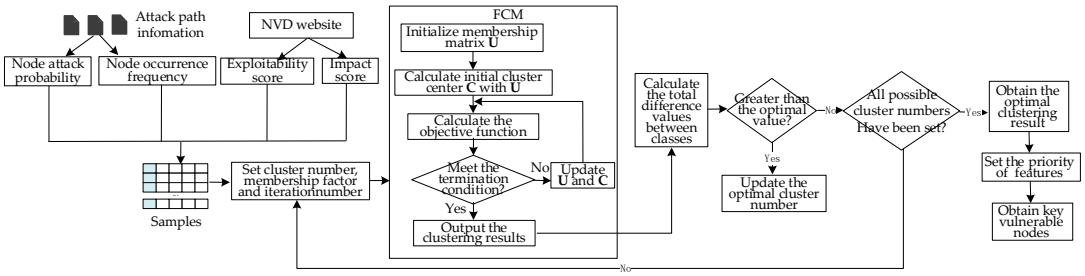


Figure 3. The discovery process of key vulnerable nodes based on improved FCM.

Firstly, it is necessary to determine the features of the participating clustering samples, which are considered from two aspects in this paper. The first is the inherent threat features of the vulnerability node, including the exploitability score and impact score. The inherent threat is the threat evaluation standard set by the National Vulnerability Database of the United States based on CVSS, combined with the damage degree of the vulnerability in various scenarios. It represents the comprehensive threat degree of the vulnerability. These two values do not change with the actual network environment of the vulnerability. The second is the actual threat features of the vulnerable nodes. Due to the vulnerability in different network environments, the threat degree is different. Its inherent threat cannot accurately reflect the threat of vulnerabilities in the actual situation. In this paper, an attack graph is constructed based on the actual network environment to realize the search for attack path information, which can reflect the actual threat situation of the vulnerability node in the current network environment. Therefore, this section will extract the actual threat features of the vulnerability node based on the attack path information, including the occurrence frequency and attack probability of the vulnerability node.

The occurrence frequency *Ocr* of the vulnerability node is the total number of occurrences of each node in all attack paths, which reflects the possibility that the attacker uses the node as the entry point to carry out the attack. It can be directly counted in the search process. The calculation formula of *Ocr* is shown in Equation (13).

$$Ocr_n = \sum_{p=1}^{tot} \sum_{i=0}^{N_p-1} \boldsymbol{I}_{0,i}^p,\ \boldsymbol{I}_{0,i}^p = \begin{cases} 1, \boldsymbol{I}_{0,i}^p = n \\ 0, \boldsymbol{I}_{0,i}^p \neq n \end{cases}, \tag{13}$$

where Ocr_n is the occurrence frequency of node n, tot is the total number of attack paths, N_p is the total number of nodes in the p-th attack path, $\boldsymbol{I}_{0,i}^p$ is the value of row 0 and column i in the p-th attack path information group. The attack probability *Pb* of the vulnerable node reflects the probability that the node is compromised, which is obtained from the attack probability of its attack path and the corresponding information group. The calculation formula of *Pb* is shown in Equation (14).

$$Pb_n = \frac{\sum_p \frac{\boldsymbol{I}_{1,m}^p - d}{\boldsymbol{I}_{1,m}^p} \times P_p'}{number},\ (d = \boldsymbol{I}_{1,i}^p, \boldsymbol{I}_{0,i}^p = n)\ and\ (\forall p, \exists j, \boldsymbol{I}_{0,j}^p = n), \tag{14}$$

where Pb_n is the attack probability of node n, $\boldsymbol{I}_{1,m}^p$ is the maximum depth of the p-th attack path, d is the depth of n in the p-th attack path, P_p' is the attack probability of the p-th attack path, *number* is the number of attack paths including n.

Secondly, the traditional FCM algorithm has no means to determine the optimal number of clusters, but determining the number of clusters only by subjective methods will reduce the accuracy of clustering results, making the difference between classes not obvious, and it is difficult to determine the key vulnerability nodes. To make the difference in the degree of clustering results obvious, combined with the demand characteristics of this method, this paper determines the optimal number of clusters based on the discrimination degree between clusters. For the current clustering results, this method first sets a cluster

number, calculates the intra-class average vector $\boldsymbol{m_j}$ of each class, and obtains the total average vector $\boldsymbol{m}$ through the intra-class average vector. The calculation formula of $\boldsymbol{m}$ is shown in Equation (15).

$$\boldsymbol{m_j} = \frac{\sum_i \boldsymbol{F_i}}{d_j}, i = 1, 2, \ldots, d_j, \ \ \boldsymbol{m} = \frac{\sum_{j=1}^{C} \boldsymbol{m_j}}{C}, \tag{15}$$

where $\boldsymbol{F_i}$ is the feature vector of the i-th sample in the j-th class, C is the total number of clusters, and d_j is the total number of samples of the j-th class. Then, the optimization objective is to maximize the sum of the squared Euclidean distance between the average vector within each class and the total average vector. This value is called the total difference value between classes (TDVC). Thus, the optimal number of clusters can be obtained. The optimization objective is shown in Equation (16).

$$maxf(c) = \frac{\sum_{j=1}^{c} \left(\boldsymbol{m_j} - \boldsymbol{m}\right)^2}{c}, c \in [2, N], \tag{16}$$

where c is the number of clusters currently set, N is the total number of vulnerability nodes, and $f(c)$ is the TDVC corresponding to c.

Finally, FCM clustering is carried out after determining the sample features and the optimal number of clusters. FCM uses membership degree to assign uncertainty to classification. It allows a data point to belong to more than one class and assigns each sample to the class with the largest membership degree. This ability makes it better reflect the fuzziness and complexity of data in the real world. Therefore, the key of FCM is to determine the cluster center $\boldsymbol{V}$ and the membership degree matrix $\boldsymbol{U}$. The objective function of FCM is shown in Equation (17).

$$J_m = \sum_{i=1}^{D} \sum_{j=1}^{C} u_{ij}^{\alpha} \left\| x_i - v_j \right\|^2, \sum_{j=1}^{C} u_{ij} = 1, i = 1, 2, \ldots D, \tag{17}$$

where α is the membership factor, i is the sample number, j is the class number, u_{ij} is the membership degree of sample i for class j, x_i is the i-th sample, C is the total cluster number, v_j is the j-th cluster center, and D is the total number of samples. FCM constantly updates $\boldsymbol{U}$ and $\boldsymbol{V}$ to minimize the objective function, thus completing the clustering. Constraints are added to the objective function by Lagrange multiplier method. Based on this formula, the partial derivatives of variables u and v are obtained, respectively. Then, the updated formula for $\boldsymbol{U}$ and $\boldsymbol{V}$ are obtained by setting the derivative to 0. These two formulas are shown in Equation (18).

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\left\| x_i - v_j \right\|}{\left\| x_i - v_k \right\|} \right)^{\frac{2}{\alpha - 1}}}, v_j = \frac{\sum_{i=1}^{D} u_{ij}^{\alpha} \cdot x_i}{\sum_{i=1}^{D} u_{ij}^{\alpha}}, \tag{18}$$

The vulnerability nodes clustering algorithm based on improved FCM is shown in Algorithm 3.

After the clustering is completed, the priority among features is set according to the feature characteristics. The clustering results are compared according to priority, and the nodes in the category ranked first are selected as the key vulnerable node set. Since the vulnerability impact score and exploitability score are authoritative indicators in CVSS, they should be considered first, and they have the same importance in CVSS calculation formula. For the node occurrence frequency and the node attack probability, the attack probability takes more factors into account, including the path location of the node and the attack path probability. It can divide the threat degree of the vulnerability node more fine-grained, so the attack probability is considered when the first three are close. To sum up, the order of

feature priority is the impact score, exploitability score, occurrence frequency, and attack probability of the vulnerability.

Algorithm 3: VulnerabilityClusteringByImprovedFCM

Input: The vulnerability node samples *samples*, the total number of samples D
Output: The number of optimal clusters *number*, and the corresponding clustering result *result*

Initialize *number*, *result*, and the corresponding TDVC $value = 0$.
FOR $d = 1, 2, \ldots, D$
  Initialize $\boldsymbol{U}$ and $\boldsymbol{V}$, set the membership factor α, set the number of clusters $C = d$, set the maximum number of iterations of the cluster to T
  WHILE($T > 0$)
    Update $\boldsymbol{U}$ and $\boldsymbol{V}$ by Equation (17)
    $T = T - 1$
  FOR $i = 1, 2, \ldots, D$
    Set the class of sample i to the class to which the maximum membership degree of sample i belongs in $\boldsymbol{U}$
  FOR $c = 1, 2, \ldots, C$
    Calculate $\boldsymbol{m_c}$ by Equation (15)
  Calculate $\boldsymbol{m}$ by Equation (15), and calculate the current TDVC f_C by Equation (16)
  IF $f_C > value$
    $value = f_C$, $number = C$, update *result* to the current cluster result

6. Results

6.1. Experimental Scenario

This paper constructs an experimental scenario to verify the effectiveness of the proposed method. The experimental environment includes web servers, ftp servers, smtp servers, sql servers and multiple user devices. The vulnerabilities of each device can be obtained by using the vulnerability scanning tool. Information about vulnerabilities comes from NVD. Vulnerability quantification is based on CVSS. The attacker launched the attack from an external network, targeting important data on the SQL server. In addition, to obtain the utilization relationship between vulnerabilities, this paper uses Word2Vec combined with TextCNN to conduct a semantic analysis of vulnerability description to obtain the exploitation relationship. All code is written in Java and Python. Finally, in order to fully verify the effectiveness of the proposed method, this paper also verifies the effectiveness of the proposed method under different attack graph scales by extending the network.

6.2. Experimental Process

Firstly, the network topology of the experimental environment is abstracted. Use the vulnerability scanning tool to obtain vulnerabilities on each device to form a vulnerability set and prepare the information required for constructing Bayesian attack subgraphs. In order to obtain the unknown utilization relationship of vulnerabilities during the construction of the attack graph, the experiment uses NVD's official interface to obtain description information of common vulnerabilities and forms a corpus after word segmentation to train Word2Vec+TextCNN model to predict the utilization consequences of vulnerabilities. The basic score, impact score, exploitability score, and utilization conditions of each vulnerability in the vulnerability set are obtained from NVD. The utilization relationship is constructed according to the utilization conditions and the predicted utilization consequences. The vulnerability utilization relationship is shown in Equation (19).

$$\begin{cases} other \rightarrow NONE \\ user \rightarrow NONE, LOW \\ root \rightarrow NONE, LOW, HIGH \end{cases}, \tag{19}$$

where $con_1 \rightarrow con_2$ indicates that the vulnerability with utilization condition con_2 can be attacked after breaching the vulnerability with utilization consequence con_1. The vulnera-

bility information and predicted exploitation consequences for some of the vulnerabilities in the vulnerability set are shown in Table 1.

Table 1. Information and predicted utilization consequences of partial vulnerabilities in the vulnerability set.

CVEID	Base Score	Exploitability Score	Impact Score	Utilization Condition	Utilization Consequence
CVE-2022-27502	7.8	5.9	1.8	LOW	user
CVE-2022-28704	7.2	5.9	1.2	HIGH	root
CVE-2022-29525	9.8	5.9	3.9	NONE	root
CVE-2022-29797	9.8	5.9	3.9	NONE	user
CVE-2022-20148	6.4	5.9	0.5	HIGH	user
CVE-2021-32546	8.8	5.9	2.8	LOW	other

The Louvain algorithm is used to divide the network into communities. The vulnerability on each device is abstracted as an attack graph node. In each community, an attack subgraph is constructed according to the attack graph nodes, network topology, and vulnerability utilization relationship. For each attack graph node, its node selection probability and breach probability are quantified to form multiple Bayesian attack subgraphs. In each community, the community connection relationship is constructed according to the network topology and vulnerability utilization relationship to quickly determine the connection between communities. The partial Bayesian attack subgraph structure is shown in Figure 4.

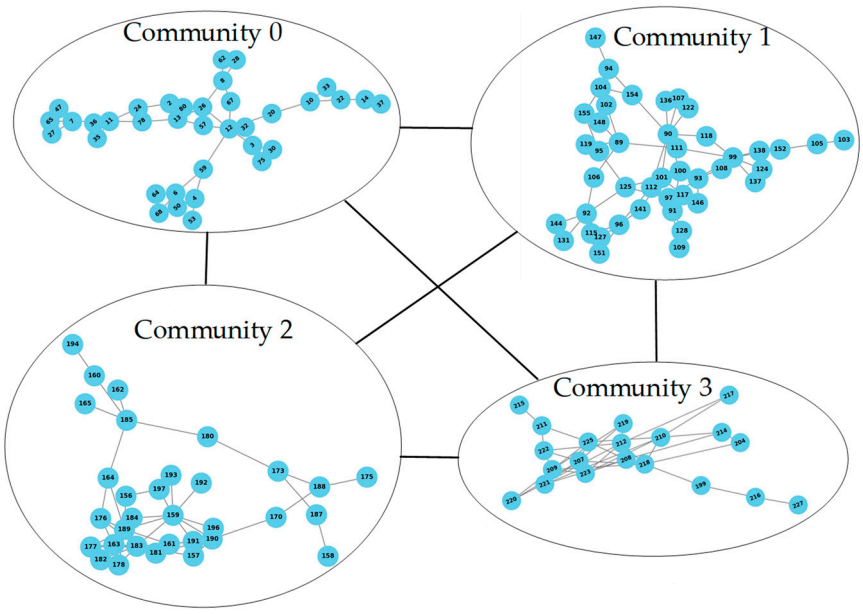


Figure 4. Partial Bayesian attack subgraphs.

Then, the method in Section 4 is used to search and aggregate the attack paths and their information groups in the constructed Bayesian attack subgraphs. The path search results in the attack subgraph and the aggregated complete path information are shown in Table 2.

Four features for FCM are extracted according to the aggregated complete attack paths and vulnerability information. The features of some samples are shown in Table 3.

Table 2. Part of the attack paths.

	Community	Attack Probability	Information Group
Attack subgraph paths	0	1.91×10^{-4}	$\begin{pmatrix} 1 & 23 & 49 & 18 & 29 & 52 & 36 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$
	1	2.36×10^{-2}	$\begin{pmatrix} 89 & 100 & 92 & 99 \\ 1 & 2 & 3 & 4 \end{pmatrix}$
	2	7.71×10^{-2}	$\begin{pmatrix} 155 & 188 & 162 \\ 1 & 2 & 3 \end{pmatrix}$
Aggregated paths	-	8.14×10^{-3}	$\begin{pmatrix} 226 & 215 & 198 & 217 & 37 & 189 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix}$
	-	3.34×10^{-4}	$\begin{pmatrix} 226 & 215 & 198 & 217 & 49 & 18 & 64 & 160 & 189 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{pmatrix}$
	-	3.94×10^{-5}	$\begin{pmatrix} 226 & 215 & 198 & 217 & 127 & 71 & 5 & 58 & 189 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{pmatrix}$

Table 3. Feature information of some samples.

Node Number	Exploitability Score	Impact Score	Occurrence Frequency	Attack Probability
1	5.9	1.0	2146	2.43×10^{-7}
18	2.7	2.8	27,762	3.39×10^{-7}
49	3.6	2.8	15,500	5.39×10^{-7}
160	3.6	2.8	15,186	4.46×10^{-7}
215	5.9	3.9	118,313	2.51×10^{-6}

Next, the improved FCM in Section 5 is used to determine the optimal number of clusters. According to the optimal cluster number, fuzzy C-means clustering is performed on the samples, and the result vulnerability set is obtained by analyzing the clustering results. The experimental results show that when the number of clusters is set to 14, the total value of the difference degree between classes reaches the maximum, that is, the optimal cluster number is 14. Some of the results of FCM clustering under the optimal number of clusters are shown in Table 4.

Table 4. Partial FCM clustering results under the optimal number of clusters.

Class Number	Node Number Contained in This Class
1	(141,189,215,217)
2	(1,2,3,8,11,26,51,71,74,121,136)
7	(4,14,28,52,55,63,116,117,120,124,140,182,183,192)
11	(6,9,12,21,30,32,72,78,80,83,87,185)
14	(18,29,39,46,47,155)

Finally, the average value of each feature in each class is calculated, and the clustering results are compared according to the set feature priority. Firstly, the exploitability score and the impact score are compared, and the comparison results are shown in Figure 5. It can be seen that the set of class numbers for which both features take the maximum mean value at the same time is (1,5,8,13).

After that, the node occurrence frequency and the node attack probability of these four types of nodes are compared. The comparison results are shown in Figure 6a,b, respectively. It can be seen from the comparison results that the mean value of each feature of Category 1 is the maximum value, so the key vulnerable node set obtained is (141,189,215,217).

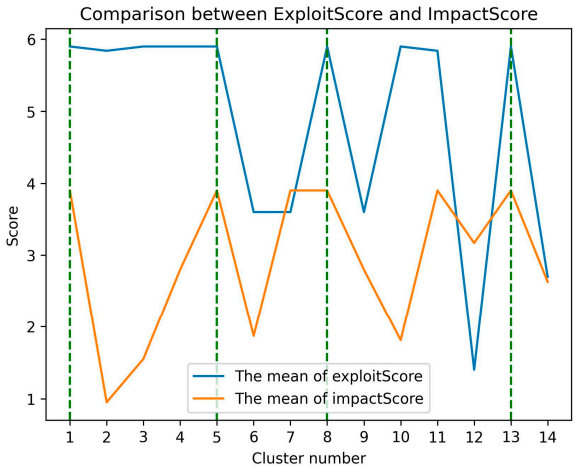
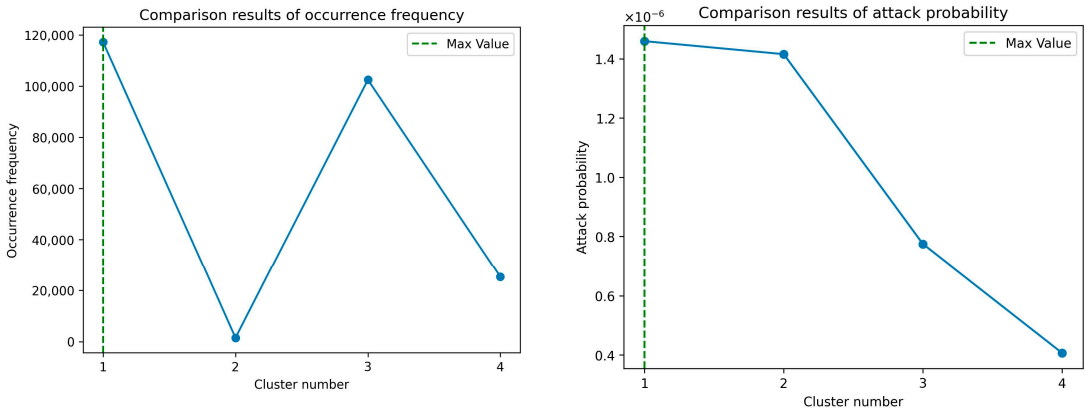


Figure 5. Comparison of the mean values of the exploitability score and the impact score.



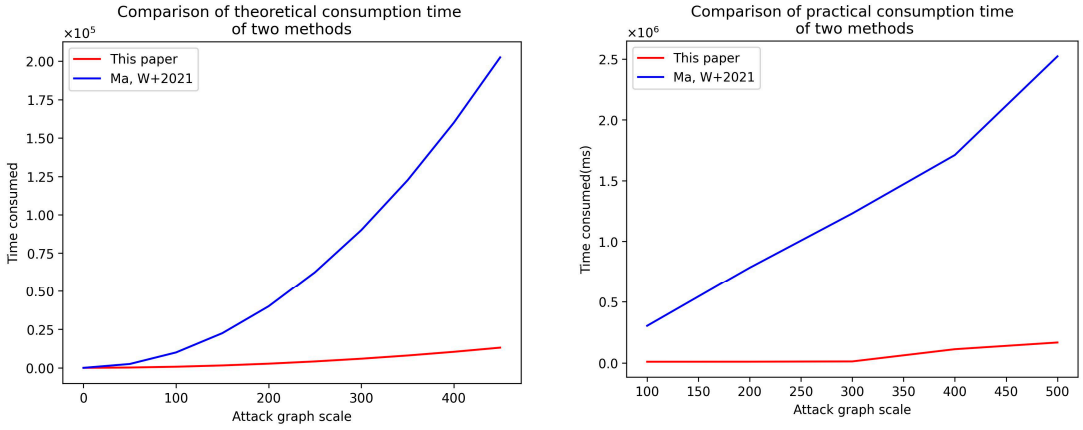
(**a**) (**b**)

Figure 6. Comparison of the mean values of the node occurrence frequency and the node attack probability: (**a**) comparison results of the node occurrence frequency; (**b**) comparison results of the node attack probability.

6.3. Experiment Results Analysis

Firstly, in order to verify the effectiveness of the optimization time efficiency of the attack path aggregation search method based on Bayesian attack subgraphs in this paper, this section makes an analysis with reference [34] from both theoretical and practical aspects. Reference [34] uses the method of searching and storing the complete attack graph. Since we cannot determine the attack graph storage strategy and search strategy of refs. [35,36], we can not check their time/space consumption.

Suppose that breadth-first search is selected as the basic search algorithm, the graph structure is stored by the adjacency matrix, and the number of nodes in the attack graph is $\mathbb{N}$. When the method of reference [34] is used to search, in the worst case, each node and its neighbors need to be traversed, and the time consumption is $\mathbb{N}^2$. When the method in this paper is used, it is assumed that the attack graph is evenly divided into m subgraphs, and each subgraph has $\mathbb{N}/m$ nodes. Then, the path search efficiency in the subgraph is $(\mathbb{N}/m)^2$. The number of paths in each subgraph is no more than $\mathbb{N}/m$ and the aggregation times of each complete path are no more than m. In the worst case, the number of traversal times to obtain all complete paths is $\mathbb{N}$, and the total time is $(\mathbb{N}/m)^2 + \mathbb{N}$. Since $m^2 > \mathbb{N}/(\mathbb{N}-1) > 1$, $\mathbb{N}/m^2 < \mathbb{N} - 1$, that is $(\mathbb{N}/m)^2 + \mathbb{N} < \mathbb{N}^2$. Therefore, the efficiency of the search method proposed in this paper is theoretically better than the efficiency of the search on the complete attack graph. Figure 7a,b show the effect of different $\mathbb{N}$ on the search efficiency of the two methods in theory and experiment, respectively.



(**a**) (**b**)

Figure 7. Comparison of search time consumption between the method in this paper and the method in reference [34]: (**a**) comparison of theoretical consumption time of two methods; (**b**) comparison of practical consumption time of two methods.

Because the attack graph is divided into four subgraphs in the experiment, $m = 4$ is taken. It can be seen from Figure 7 that with the increase in the graph scale, the time consumed by the proposed method increases slightly both in theory and experiment, which can optimize the search time. The optimization effect of the proposed method on time efficiency becomes more and more obvious with the increase in the graph scale.

Then, from the perspective of space efficiency, this paper adopts the method of storing the attack graph based on subgraphs rather than the complete attack graph. Suppose that the total number of nodes in the attack graph is $\mathbb{N}$, and it is evenly divided into m attack subgraphs, each subgraph contains $\mathbb{N}/m$ nodes. Theoretically, the space consumption of storing the complete attack graph using the adjacency matrix is $\mathbb{N}^2$, and the storage method based on subgraphs consumes $(\mathbb{N}/m)^2 \times m$. Since $m > 1$, obviously $(\mathbb{N}/m)^2 \times m < \mathbb{N}^2$. Figure 8 shows the comparison of the space consumed by the two methods under different attack graph scales in the experiment.

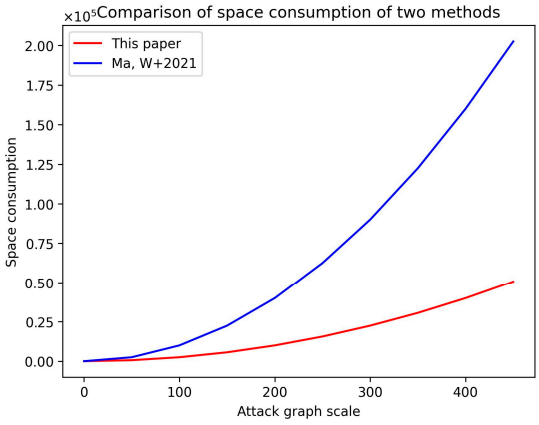


Figure 8. Comparison of the space consumed by the two methods under different attack graph scales [34].

It can be seen from Figure 8 that when the attack graph scale is small, the space consumption of the two methods is similar. However, with the increase in the graph scale, the storage method based on subgraphs has a more and more obvious effect on storage space optimization. When the attack graph has 500 nodes, the space optimization has reached 75%, so the latter can effectively improve the utilization efficiency of space.

After that, in order to verify the effectiveness of the method of determining the optimal cluster number in this paper and the superiority of improved FCM over other clustering methods, the clustering results of different clustering methods are analyzed. When only FCM is used for longitudinal analysis, the changes in the TDVC of different cluster numbers are shown in Figure 9.

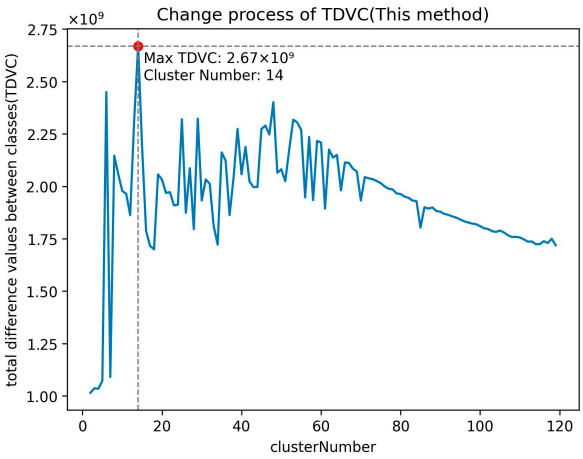
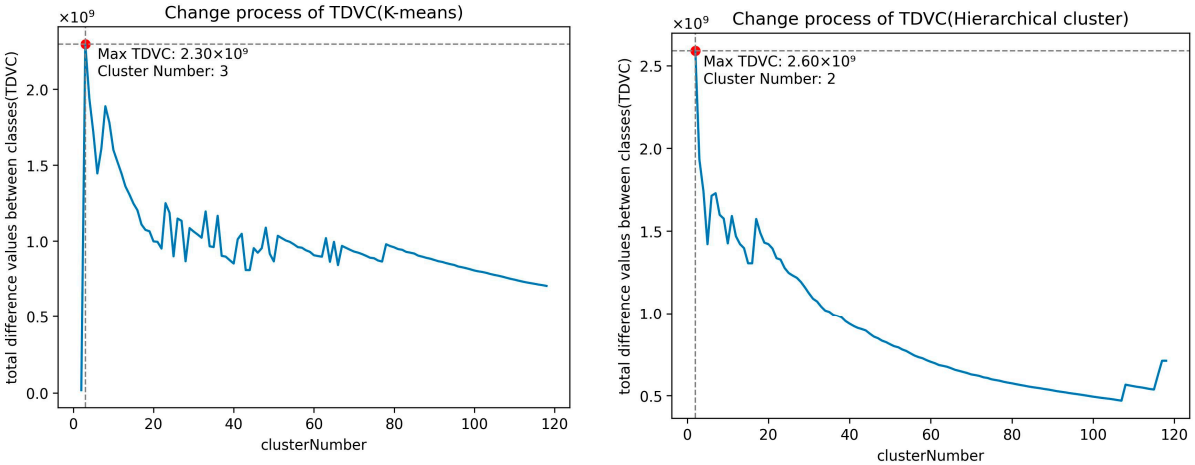


Figure 9. The change process of the TDVC of FCM.

It can be seen from Figure 9 that when the number of clusters is 14, the TDVC of clustering results is the largest. At this time, the difference between classes is the largest, and it is easy to distinguish the characteristics of various types, and the clustering effect is optimal.

Next, in the experiment, different clustering methods are selected for horizontal analysis, such as the common k-means clustering, hierarchical clustering and Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The changes in TDVC for K-means clustering and hierarchical clustering are shown in Figure 10a,b, respectively.



(**a**) (**b**)

Figure 10. The changes in TDVC for K-means clustering and hierarchical clustering: (**a**) the TDVC change process of K-means clustering; (**b**) the TDVC change process of hierarchical clustering.

DBSCAN, as a density clustering algorithm, does not need to specify the number of clusters in advance to complete clustering. The comparison results of the maximum TDVC of the three methods and the cluster number corresponding to the maximum TDVC are shown in Table 5.

Table 5. Comparison results of the maximum TDVC of the three methods.

Methods	FCM (This Method)	K-Means	Hierarchical Clustering	DBSCAN
Maximum TDVC	2.6692×10^9	2.2973×10^9	2.5907×10^9	5.3945×10^7
Corresponding cluster number	14	3	2	-

It can be seen from Table 5 that the maximum value of the TDVC obtained by the other clustering methods is lower than the maximum value of FCM. It shows that compared with other clustering results, the difference between the clusters in the clustering results of FCM is larger, which is more in line with the requirements of the proposed method. So using FCM as the clustering method is the optimal choice.

Finally, in order to verify the accuracy of the collection results of key vulnerability nodes in the proposed method, this experiment compares the results of this paper with the search results of [35–37], respectively. The search results of key vulnerable nodes, the reachability of target nodes, and the changes in the number of attack paths after repairing the four methods are shown in Table 6.

Table 6. Comparison of effectiveness of search results of four methods.

Methods	Key Vulnerability Node	Target Nodes Reachability	Number of Remaining Attack Paths
This paper	(92,141,152,189,215,217)	unreachable	0
[35]	(32,37,130,190,215)	reachable	79,983
[36]	(141,189,217)	reachable	635
[37]	(92,141,189,215,217)	reachable	13

Reference [35] only considers the CVSS score threat characteristics of vulnerabilities and takes the node with the highest score as the key vulnerable node. The attack probability of the attack path of nodes 32, 37, and 30 is only 3.1615×10^{-9}. In practice, it can be regarded as an impossible event that the attacker attacks along this path. Therefore, after repairing the nodes, the target node is still reachable, and the key vulnerable nodes are not accurately obtained. Reference [36] considers both the threat of vulnerability itself and the actual threat and removes part of the attack paths by pruning to improve the analysis efficiency in large-scale attack graphs. However, the removed attack paths affect the search accuracy of key vulnerable nodes, and there are 92,152,215 missing nodes. This results in that although the number of remaining attack paths is greatly reduced after repairing the key vulnerable nodes, the target nodes are still reachable. After analysis, the remaining 635 paths all contain one or more of the 92,152,215 nodes. Reference [37] adopts the current advanced neural network method. The results obtained in the set experimental environment are (92,141,189,215,217). However, like reference [36], partial pruning paths reduce the accuracy of search results. This results in a missing node 152 in the search results. After the key vulnerable nodes obtained by the proposed method are repaired, the target node is unreachable, which shows the accuracy and effectiveness of the search results. Because the method in this paper not only considers all attack paths but also considers the inherent and actual threat characteristics of the vulnerability. The characteristics of this paper compared with other reference methods are summarized in Table 7.

Table 7. Comparison of the characteristics of this paper with other references.

Methods	Time Optimization	Space Optimization	Consider Vulnerability Features	Consider Actual Features	Consider All Paths	Adaptive Adjustment of Cluster Number	Get Key Vulnerabilities
[19]	Yes	No	Yes	Yes	Yes	No	No
[29]	Yes	No	Yes	No	No	No	Yes
[31]	No	No	Yes	No	Yes	No	Yes
[35]	No	No	Yes	No	No	No	Yes
[36]	Yes	No	Yes	Yes	No	No	Yes
This paper	Yes	Yes	Yes	Yes	Yes	Yes	Yes

7. Conclusions

It is an important problem in network security analysis to obtain the key vulnerable nodes in large-scale networks quickly and accurately. In this paper, the large-scale network is divided into multiple subnetworks by the idea of community division, and the Bayesian attack subgraphs are constructed by quantifying the subgraph nodes. Then, the analysis results of the attack path information in the subgraph are aggregated to quickly obtain the analysis results of the complete attack graph. The experimental results show that under the attack graph scale of 500 nodes, the time consumption of the analysis method based on subgraphs is only 10% of that of the analysis method based on the complete attack graph, and the space consumption is only 25% of the latter, which has a great improvement in time and space efficiency. Next, the optimal number of clusters is adaptively determined by using the idea of variance, and the actual threat features of the vulnerability nodes in the network are extracted from the analysis results of the attack graph. The threat features of the vulnerability themselves proposed in CVSS are combined for fuzzy C-means clustering, and the key vulnerable nodes are obtained by setting feature priorities according to the clustering results. Fuzzy clustering can take into account a variety of features that affect the threat of vulnerabilities, and improve the accuracy of the search results of key vulnerable nodes. The experimental results confirm the effectiveness of the method in this paper. However, the method of quantifying the threat value of vulnerable nodes in this paper is relatively simple, and only CVSS is used as a single quantification standard. In addition, there is no pruning method in the aggregation process of attack subgraph paths, and the amount of attack path information in the aggregation process is small, so the accuracy of the results needs to be improved. In the future, the quantization scheme of node attack probability will be optimized by Bayesian theory, and the accuracy of key node search results will be improved by increasing the amount of information in path aggregation. For the obtained key vulnerable nodes, game theory can also be used to lay out the node defense scheme.

Author Contributions: Conceptualization, Y.X. (Yuhua Xu) and Y.L.; methodology, Y.X. (Yuhua Xu) and Y.L.; writing—review and editing, Y.X. (Yuhua Xu) and Y.L.; investigation, Y.X. (Yucheng Xue), W.L., C.L. and Z.S. (Zhe Sun); software, Y.X. (Yucheng Xue), W.L., C.L. and Z.S. (Zhe Sun); writing—original draft preparation, Y.X. (Yucheng Xue), W.L., C.L. and Z.S. (Zhe Sun); validation: Y.X. (Yuhua Xu) and Y.L.; supervision: Y.X. (Yuhua Xu) and Z.S. (Zhixin Sun). All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China under Grant 62272239; Jiangsu Agriculture Science and Technology Innovation Fund (JASTIF) CX(22)1007; the Natural Science Foundation of the Jiangsu Higher Education Institutions of China Grant 22KJB520027; Natural Science Research Start-up Foundation of Recruiting Talents of Nanjing University of Posts and Telecommunications Grant NY222029; Guizhou Provincial Key Technology R&D Program [2023]272; the Postgraduate Research & Innovation Plan of Jiangsu Province under Grant KYCX20_0761.

Data Availability Statement: The dataset supporting this study is available through the official interface provided at https://nvd.nist.gov, accessed on 20 February 2024.

Conflicts of Interest: The authors of this study declare that no conflict of interest could influence the study of this paper.

References

1. Aslan, Ö.; Aktuğ, S.S.; Ozkan-Okay, M.; Yilmaz, A.A.; Akin, E. A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. *Electronics* **2023**, *12*, 1333. [CrossRef]
2. Ferrara, P.; Mandal, A.K.; Cortesi, A.; Spoto, F. Static analysis for discovering IoT vulnerabilities. *Int. J. Softw. Tools Technol. Transf.* **2021**, *23*, 71–88. [CrossRef]
3. Vallabhaneni, R.; Vaddadi, S.; Maroju, A.; Dontu, S. Analysis on Security Vulnerabilities of the Modern Internet of Things (IOT) Systems. *Int. J. Recent Innov. Trends Comput. Commun.* **2024**, *11*, 9.
4. Jbair, M.; Ahmad, B.; Maple, C.; Harrison, R. Threat modelling for industrial cyber physical systems in the era of smart manufacturing. *Comput. Ind.* **2022**, *137*, 103611. [CrossRef]
5. Xiong, W.; Legrand, E.; Åberg, O.; Lagerström, R. Cyber security threat modeling based on the MITRE Enterprise ATT&CK Matrix. *Softw. Syst. Model.* **2022**, *21*, 157–177.
6. Cao, S.; Sun, X.; Bo, L.; Wei, Y.; Li, B. Bgnn4vd: Constructing bidirectional graph neural-network for vulnerability detection. *Inf. Softw. Technol.* **2021**, *136*, 106576. [CrossRef]
7. Liu, Z.; Qian, P.; Wang, X.; Zhuang, Y.; Qiu, L. Combining graph neural networks with expert knowledge for smart contract vulnerability detection. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 1296–1310. [CrossRef]
8. Zheng, Y.; Pujar, S.; Lewis, B.; Buratti, L.; Epstein, E.; Yang, B.; Su, Z. D2a: A dataset built for ai-based vulnerability detection methods using differential analysis. In Proceedings of the 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), Madrid, Spain, 25–28 May 2021; pp. 111–120.
9. Chakraborty, S.; Krishna, R.; Ding, Y.; Ray, B. Deep learning based vulnerability detection: Are we there yet? *IEEE Trans. Softw. Eng.* **2021**, *48*, 3280–3296. [CrossRef]
10. Steenhoek, B.; Rahman, M.M.; Jiles, R.; Le, W. An empirical study of deep learning models for vulnerability detection. In Proceedings of the 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), Melbourne, Australia, 14–20 May 2023; pp. 2237–2248.
11. Almazrouei, O.S.M.B.H.; Magalingam, P.; Hasan, M.K.; Hasan, M.K.; Shanmugam, M. A review on attack graph analysis for iot vulnerability assessment: Challenges, open issues, and future directions. *IEEE Access* **2023**, *11*, 44350–44376. [CrossRef]
12. Zenitani, K. Attack graph analysis: An explanatory guide. *Comput. Secur.* **2023**, *126*, 103081. [CrossRef]
13. Hankin, C.; Malacaria, P. Attack dynamics: An automatic attack graph generation framework based on system topology, CAPEC, CWE, and CVE databases. *Comput. Secur.* **2022**, *123*, 102938.
14. Mohammadzad, M.; Karimpour, J.; Mahan, F. MAGD: Minimal Attack Graph Generation Dynamically in Cyber Security. *Comput. Netw.* **2023**, *236*, 110004. [CrossRef]
15. Presekal, A.; Ştefanov, A.; Rajkumar, V.S.; Palensky, P. Attack graph model for cyber-physical power systems using hybrid deep learning. *IEEE Trans. Smart Grid* **2023**, *14*, 4007–4020. [CrossRef]
16. Shin, G.Y.; Hong, S.S.; Lee, J.S.; Han, I.-S.; Kim, H.-K.; Oh, H.-R. Network security node-edge scoring system using attack graph based on vulnerability correlation. *Appl. Sci.* **2022**, *12*, 6852. [CrossRef]
17. Al-Araji, Z.J.; Ahmad, S.S.S.; Abdullah, R.S. Propose vulnerability metrics to measure network secure using attack graph. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 51–58. [CrossRef]
18. Al-Araji, Z.; Syed Ahmad, S.S.; Abdullah, R.S. Attack prediction to enhance attack path discovery using improved attack graph. *Karbala Int. J. Mod. Sci.* **2022**, *8*, 313–329. [CrossRef]
19. Kholidy, H.A. Multi-layer attack graph analysis in the 5G edge network using a dynamic hexagonal fuzzy method. *Sensors* **2021**, *22*, 9. [CrossRef] [PubMed]
20. Saravanakumar, T.; Lee, T.H. Hybrid-driven-based resilient control for networked T-S fuzzy systems with time-delay and cyber-attacks. *Int. J. Robust Nonlinear Control* **2023**, *33*, 7869–7891. [CrossRef]
21. Jiang, S.; Yang, L.; Cheng, G.; Gao, X.; Feng, T.; Zhou, Y. A quantitative framework for network resilience evaluation using Dynamic Bayesian Network. *Comput. Commun.* **2022**, *194*, 387–398. [CrossRef]
22. Xie, J.; Zhang, S.; Wang, H.; Chen, M. Multiobjective network security dynamic assessment method based on Bayesian network attack graph. *Int. J. Intell. Comput. Cybern.* **2024**, *17*, 38–60. [CrossRef]
23. Luo, Z.; Xu, R.; Wang, J.; Zhu, W. A Dynamic Risk Assessment Method Based on Bayesian Attack Graph. *Int. J. Netw. Secur* **2022**, *24*, 787–796.
24. Hao, J.; Luo, S.; Pan, L. A novel vulnerability severity assessment method for source code based on a graph neural network. *Inf. Softw. Technol.* **2023**, *161*, 107247. [CrossRef]
25. Tang, G.; Yang, L.; Zhang, L.; Cao, W.; Meng, L.; He, H.; Kuang, H.; Yang, F.; Wang, H. An attention-based automatic vulnerability detection approach with GGNN. *Int. J. Mach. Learn. Cybern.* **2023**, *14*, 3113–3127. [CrossRef]
26. Li, H.J.; Wang, L.; Bu, Z.; Cao, J.; Shi, Y. Measuring the network vulnerability based on markov criticality. *ACM Trans. Knowl. Discov. Data (TKDD)* **2021**, *16*, 1–24. [CrossRef]
27. Huang, B.; Liu, Y. A network vulnerability assessment method using general attack tree. In Proceedings of the 2022 5th International Conference on Data Science and Information Technology (DSIT), Shanghai, China, 22–24 July 2022; pp. 1–4.

28. Yang, H.; Yuan, H.; Zhang, L. Risk assessment method of IoT host based on attack graph. *Mob. Netw. Appl.* **2023**, 1–10. [CrossRef]
29. Li, Y.; Li, X. Reseasrch on multi-target network security assessment with attack graph expert system model. *Sci. Program.* **2021**, *2021*, 9921731.
30. Qian, K.; Jin, M.; Zhang, D.; Huang, H.C. Research on Evaluation Method of Network Vulnerability in Power Monitoring System. In *Advances in Intelligent Information Hiding and Multimedia Signal Processing: Proceeding of the IIH-MSP 2021 & FITAT 2021*; Springer: Singapore; Kaohsiung, Taiwan, 2022; Volume 2, pp. 113–123.
31. Xie, J.; Keda, S.; Xubing, L. Risk assessment method of power plant industrial control information security based on Bayesian attack graph. *J. Electr. Syst.* **2021**, *17*, 529.
32. Li, Z.; Liu, H.; Wu, C. Computer network security evaluation method based on improved attack graph. *J. Cyber Secur. Technol.* **2022**, *6*, 201–215. [CrossRef]
33. Ying, Y. *Research and Implementation of Network Security Measurement Technology Based on Attack Path Threat Analysis*; Beijing University of Posts and Telecommunications: Beijing, China, 2021.
34. Ma, W. Research on network vulnerability assessment based on attack graph and security metrics. *J. Phys. Conf. Ser. IOP Publ.* **2021**, *1774*, 012070. [CrossRef]
35. Vasilyev, V.; Kirillova, A.; Vulfin, A.; Nikonov, A. Cybersecurity risk assessment based on cognitive attack vector modeling with CVSS Score. In Proceedings of the 2021 International Conference on Information Technology and Nanotechnology (ITNT), Samara, Russia, 20–24 September 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
36. Kalogeraki, E.M.; Papastergiou, S.; Panayiotopoulos, T. An attack simulation and evidence chains generation model for critical information infrastructures. *Electronics* **2022**, *11*, 404. [CrossRef]
37. Fan, W.; Xu, H.; Jin, W.; Liu, X.; Tang, X.; Wang, S.; Li, Q.; Tang, J.; Wang, J.; Aggarwal, C. Jointly attacking graph neural network and its explanations. In Proceedings of the 2023 IEEE 39th International Conference on Data Engineering (ICDE), Anaheim, CA, USA, 1 April 2023; pp. 654–667.