electronics

MDPI

Article

A Network Attack Surface Evaluation Method Based on Optimal Attack Strategy

Peng Xie [1,2], Lin Zhang [2], Zhichao Lian [1,*] and Jianxin Yang [2]

[1] School of Cyber Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
[2] Information Center of China North Industries Group Corporation, Beijing 100089, China; yangjianxin@norincocloud.com.cn (J.Y.)
* Correspondence: newlzcts@njust.edu.cn

Abstract: In the era of the rapid development of information technology, it is particularly important to ensure the security of information systems. The network attack surface, as an important index for measuring information system security, has become the focus of practitioners. At present, the accuracy and practicability of network attack surface evaluations are insufficient. In order to solve this problem, this paper proposes a network attack surface evaluation method based on an optimal attack strategy. This method first identifies the main attack targets of network resources and then uses advanced optimization techniques to determine the best attack strategy. Finally, the network resources closely related to system network security are selected, and the network attack surface is calculated according to the filtering results. A series of simulation experiments show that the method proposed in this paper is more closely related to penetration testing results, more sensitive to changes in network attack surfaces, and more consistent with the real situation compared to other methods. The results demonstrate the method's balance of practicality and effectiveness.

Keywords: target defense; security risk evaluation; network attack surface; arithmetic optimization algorithm

Academic Editor: Zbigniew Kotulski

Received: 13 November 2024
Revised: 6 January 2025
Accepted: 9 January 2025
Published: 11 January 2025

Citation: Xie, P.; Zhang, L.; Lian, Z.; Yang, J. A Network Attack Surface Evaluation Method Based on Optimal Attack Strategy. *Electronics* **2025**, *14*, 274. https://doi.org/10.3390/electronics14020274

1. Introduction

Network security evaluation technology is capable of identifying security risks and vulnerabilities within a network information system that can be exploited by intruders. It enables the detection and resolution of these issues before any attack, significantly enhancing the overall security of a network information system. The information system can be understood as an integrated human–machine system composed of computer hardware, network and communication equipment, computer software, information resources, information users, and regulations for processing information flow, such as personal computers, network devices, applications, and even the infrastructure of the entire organization.

Due to the inherent flaws in network applications and other programs, intruders can easily exploit vulnerabilities to launch attacks. If the intruder successfully attacks, it is easy to obtain sensitive information and even gain management privileges on the target system [1]. These actions pose a threat to the confidentiality, integrity, and availability of a system. With the rapid advancement of electronic government and e-commerce, there has been a increased emphasis on network security assessments and related information and system security engineering within government agencies, military entities, businesses, and research institutions due to its extensive research and development opportunities.

The severity of the vulnerability of a network information system is a prerequisite for the occurrence of network attacks. The interdependence of vulnerabilities, the interdependence of hosts, the dynamism of services, and the complexity of network information system connections indicate that network security assessment, especially the study of assessment models, is a very complex task [2].

The primary methods of evaluating network security include attack trees, attack graphs, and network attack surfaces [3]. The construction of attack trees and attack graphs often relies on the configuration information of the attacked nodes, their causal relationships, and the attacker's capabilities. The selection of initial nodes is based on vulnerability scanning results, without an analysis of the system's internal characteristics. Network attack surface measurement has gained significant attention in recent years and has been applied to the security analysis of software, hardware, networks, and physical systems. The network attack surface refers to the available software, hardware resources, and other network resources that a system exposes to attackers, which can be used to measure the security of the system. Currently, most research on network attack surfaces has focused on the set of system-related elements that can be exploited by attackers. System-related elements can be system components, such as interfaces, methods, data, and protocols. Security risk refers to the risk that an information system is attacked by an attacker and loses control. The network attack surface does not represent the quality of the software code, and a large network attack surface does not necessarily mean that there are many security defects in the code, but it indicates that the system has a greater security risk. Current work in the field of network attack surfaces has focused on creating empirical and theoretical measures for moving target defense, software system evaluation, etc. The work on evaluating network attack surfaces is just starting [4], and the calculation is based on expert knowledge to assign privileges and access permissions, which involves subjectivity.

Currently, the main methods for evaluating network attack surfaces are based on attack paths and Bayesian network resource graphs [4]. The Bayesian network resource graph formalizes and graphically characterizes the causal relationships between all network resources. Based on Bayesian network resource graphs, potential attack paths in the network can be predicted and evaluated. Therefore, methods based on Bayesian network resource graphs have gradually become mainstream.

However, the current method assumes that the impact of different network resources on system security is consistent and indistinguishable in the calculation process of the network attack surface, which is inconsistent with the actual situation. For example, in a dynamically networked control system, edge nodes such as environment-aware nodes and data collection nodes have less impact on the network attack surface than key nodes such as control nodes or data processing nodes. Additionally, when the network scale is large, it usually has a large number of leaf nodes. Although each leaf node has a small impact on the network attack surface, when calculating the network attack surface based on the unified weight of all nodes, the calculation results deviate far from the actual situation. Different network resources exert varying degrees of influence on system security based on their respective positions. Network resources located on the periphery have a lesser impact on overall information system security compared to those in core control positions. Therefore, in order to improve the accuracy of network attack surface evaluation results, this paper proposes a network attack surface security evaluation method based on an optimal attack strategy. The core attack target of network resources is defined, the optimal attack strategy is found based on the arithmetic optimization algorithm, and the network attack surface is calculated based on the optimal strategy to measure the network security performance of resources. The method proposed in this paper can be utilized to quantitatively evaluate the survivability of network resources in complex network environments.

2. Related Work

2.1. Network Attack Surface

The concept of the network attack surface was initially proposed by Howard et al. [5] for Windows, using 17 vectors with weights ranging from 0.1 to 1 as the evaluation dimension of the network attack surface. The size of the network attack surface is the sum of the contribution values of all dimensions and requires specialized knowledge in a specific field to develop and implement. Later, this concept was promoted through formal models, refined, and applied to large-scale software. Based on the attack process, the evaluation indicators were refined to three elements: entry and exit points, channels, and data [6]. Its calculation can be achieved through the call graph generated automatically by time vector machines. In terms of calculation, it relies on expert knowledge to assign privileges and access permissions. Manadhata et al. [7] analyzed many attack cases against operating systems and concluded that attackers can use system functions, channels, and data items in the system environment to attack the system. The network attack surface of a system is defined as a subset of method, channel, and data triplets. Kurmus et al. [8] defined the combination of publicly available functions and their direct and indirect calls in the Linux kernel as a call graph and, based on the call graph, defined the network attack surface as a subset of the triple of call graph, attack callable function set, and blocking attack function set. Peng et al. [9] defined the network attack surface as the sum of all externally accessible resources. Foreman et al. [10], Sun et al. [11], and Cyberko et al. [12] defined a network attack surface as the method and means by which an attacker performs an attack, connects to a system, and destroys the system. Bopche et al. [13] defined it as a subset of network resources such as the network configuration and vulnerabilities that attackers can exploit. According to Zhuang et al. [14], the network attack surface refers to the sum of various available software and hardware resources (such as the various software installed on the terminal system, open protocol ports, and existing system vulnerabilities) that the system exposes to attackers, as well as other network resources that can be used to carry out network attacks. Based on the literature above, the network attack surface can be used as a measure of information system network security. The larger the network attack surface, the higher the probability of malicious attacks such as vulnerability attacks [15], traffic-based attacks [16], and side channel attacks [17] breaking through the information system.

The network attack surface has attracted widespread attention for many years and has been modified to evaluate the security of different network information systems. Younis et al. [18] used Apache as an example to compare vulnerability severity with the dimensions of the method on the network attack surface, analyze the complementary relationship between the two, and use entry points and accessibility to analyze the vulnerability exploitability of the software. Gu [2] proposed a risk evaluation model for air traffic control information systems based on the network architecture. Tran et al. [19] used static analysis to identify the associated entry and exit points, identify the network attack surface of Android applications, and evaluate the security of the applications. Dovgalyuk et al. [20] used virtual machines to simulate complex systems and evaluated the security characteristics of complex systems based on the network attack surface of the programs processing input data. Sunkavilli et al. [21] proposed a method to measure and defend the security of open source FPGA CAD tools based on network attacks. Fursova et al. [22] used pollution analysis to detect programs and modules that may be affected by input data in virtual machines in order to evaluate whether the software is safe. Zhang et al. [23] proposed a differential dynamic monitoring method for IoT devices based on network attack surface feature recognition to monitor IoT device security. Ghanshyam et al. [24] used Boolean similarity to measure the network attack surface of large-scale networks and evaluate their security. Wang et al. [25] studied the security evaluation of Node.js based on network attack surfaces. It can be seen that

researchers have applied network attack surfaces for different evaluations of the security of information systems, and each method has been personalized and modified for a certain system, which lack scalability and have limited practicality. Recently, Douglas et al. [26] studied the problem of network attack surface mapping, introduced tools for network attack surface mapping, and proposed several promising research directions for the future.

2.2. Intelligent Optimization Algorithms

Intelligent optimization algorithms are characterized by easy implementation, inspiration, and parallel processing in solving optimization problems by simulating certain natural phenomena or mimicking the self-organizing behaviors of biological populations.

The genetic algorithm [27] is a type of random search optimization algorithm based on the theory of biological evolution and the genetic mechanism. In the process of strategy selection, the initial strategy is encoded with genes, and a fitness function is established based on various constraints. Through genetic operators such as selection, crossover, and mutation, the optimal strategy evolves generation by generation. The genetic algorithm is robust and has implicit parallelism and a strong global search ability [28].

The differential evolution algorithm [29] is a type of random search optimization algorithm based on population differences. Similar to genetic algorithms, it establishes fitness functions for various constraints on the initial solution encoding and iteratively obtains the optimal strategy through mutation, crossover, selection, and other operations. The differential evolution algorithm has a simple structure and strong robustness. Its unique memory function gives it strong global search ability, making it suitable for solving combinatorial optimization problems. However, it has drawbacks such as premature convergence and contraction stagnation.

The particle swarm optimization algorithm [30] is a swarm intelligence search optimization algorithm that simulates the foraging behavior of flocks of birds. In strategy selection, each particle represents a strategy, and a fitness function is established to evaluate the quality of the particles. Based on the information sharing between the particles, the motion of the particles is guided to evolve from disorder to order, ultimately obtaining the optimal strategy. The particle swarm optimization algorithm has advantages such as fewer parameter settings and a fast search speed, but its disadvantages, such as being easily trapped in local optima and slow convergence speed, affect the application effect of the algorithm. The coyote optimization algorithm [31] is a heuristic optimization algorithm that simulates the behavior of a coyote population through four main operations: initialization of the population, growth of the coyote, life and death of the coyote, and expulsion and acceptance of the coyote. Using their cooperation and competition strategies, the solution space is explored and optimized, and the optimal solution is finally selected. This algorithm has excellent performance in solving combinatorial optimization and parameter optimization, but its convergence speed is slow [32].

The arithmetic optimization algorithm (AOA) [33] is a latest metaheuristic algorithm that utilizes the distribution behavior of four basic arithmetic operations in mathematics (addition, subtraction, multiplication, and division) for optimization. The algorithm uses a mathematical function accelerator to select optimization strategies, dividing the entire optimization process into exploration and development, that is, using multiplication and division operations for global exploration and using addition and subtraction operations for local development. The sand cat swarm optimization algorithm (SCSO) is a metaheuristic algorithm that mimics the predatory behavior of sand cats, giving the algorithm strong optimized performance.

Recently, researchers have proposed a type of optimization algorithm inspired by human life and social behavior. For example, researchers drew on the concept of group teaching and proposed the group teaching optimization algorithm [34] by simulating the

idea of teachers using different teaching methods for different students during the teaching process. The real estate market-based optimization algorithm [35] mimics the interaction between the supply and demand sides in the market, constantly adjusting the equilibrium point to find the optimal solution. Alpine ski optimization (ASO) [36] is inspired by skiers participating in championships, mimicking the physical stamina and sprinting process of skiers to simulate the algorithm's exploration and exploitation. The coronavirus mask protection algorithm (CMPA) [37] simulates the onset process of human infectious diseases, which is divided into three stages: virus infection, virus spread, and virus immunity.

The latest optimization algorithm based on the metaheuristic algorithm is the sand cat swarm optimization algorithm (SCSO) [38]. Its inspiration comes from the hunting process of sand cats, which simulates the stage of looking for prey during the exploration stage and the stage of hunting prey during the exploitation stage. The improved sand cat swarm iptimization algorithm [39] incorporates low-frequency noise into the search strategy to improve search efficiency, which accelerates the convergence speed of the algorithm and enhances its global optimization capability.

In addition to the commonly used algorithms, researchers have also developed various intelligent optimization algorithms such as cuckoo search algorithm [40], bat algorithm [41], whale optimization algorithm [42], and moth-to-flame optimization algorithm [43]. Each algorithm has different applicable scenarios. Experiments have shown that compared to different optimization algorithms [33], arithmetic optimization algorithms perform excellently in most cases. Existing research has typically used all network resources to evaluate the network attack surface. Different network resources are processed according to unified standards and the calculation results are higher than the actual simulation results. Alternatively, targeted security evaluation methods are used for the in-depth analysis of specific software or a specific device, which has limited scalability, cannot be transferred, and has low robustness. This paper delineates core attack targets in network information systems, searches for optimal attack strategies, screens core network resources to calculate network attack surfaces, and improves the accuracy and robustness of network attack surface evaluation methods.

3. Methodology

3.1. Information System Network Attack Surface

In the era of digitization and intelligence, information systems are equipped with more and more intelligent devices, and the risk of network attacks is increasing. During the use of network information systems, such as information collection, transmission, processing, and storage, attackers can exploit vulnerabilities in any node to carry out effective attacks, resulting in information system paralysis.

Network resources that can be used to attack information systems are categorized into three groups: ports, protocols, and untrusted data. M represents the set of all ports, C represents the set of protocols, and I represents the set of untrusted data. Therefore, resources related to the network attack surface can be represented by sets M, C, and I. For a given information system E and its related network resource $R_e s$, the network attack surface of the information system E can be defined as a triplet (M^{Res}, C^{Res}, I^{Res}).

To calculate the network attack surface, the Bayesian network resource graph is first established according to the three types of network resource subsets and their relationships. The Bayesian network resource graph is a directed acyclic graph, where each node in the graph represents resources such as the port M, protocol C, and untrusted data I contained in the network information system. The directed edge represents the dependency between the nodes, that is, the information flow between resource objects. A typical Bayesian network resource graph is shown in Figure 1. If there is a dependency relationship between two

nodes, an attack between the two nodes is possible. If there is no dependency relationship and the two nodes are independent of each other, an attack cannot be carried out. The attacker can establish the maximum attack path according to the layer-by-layer dependency relationship between the child node and the parent node. However, for child nodes, they can only be attacked once the parent node has been successfully compromised. Therefore, the severity of the vulnerability of the parent node is typically greater than that of the child node, which also determines the initial probability that a node is attacked.

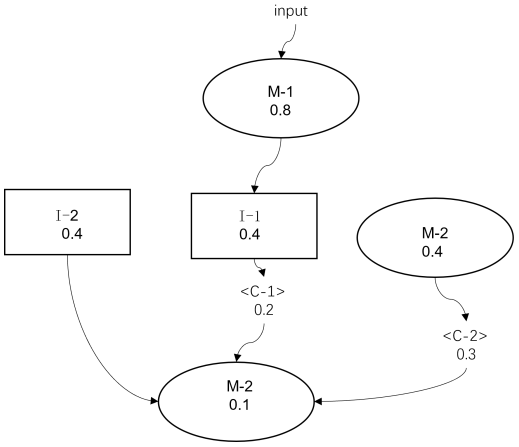

Figure 1. Bayesian network resource graph.

Having established the Bayesian framework, we proceed to calculate the attack surfaces based on these dependencies. To calculate the probability that each node is attacked in the information system, the threat model used in this article is the zero-day vulnerability threat model [44]. This model assumes that each node has the possibility of being attacked; that is, the node has vulnerability, and attackers exploit the vulnerabilities in network protocols, network hardware and software, or malicious data packets to carry out attacks. According to the Bayesian network resource graph of information systems, two nodes connected by edges can trigger network attacks. The numerical values below each node in a Bayesian network represent the initial vulnerability severity of that node, that is, the conditional probability distribution of each node in the network. The root nodes of Bayesian networks are conditionally independent, while the conditional probabilities of any other node depend on the conditional probabilities of the parent node. In Bayesian network resource graphs, the probability between the nodes is represented by P, which is the initial probability of being attacked. For any node in a Bayesian network resource graph X_i, $i \in (M^{Res}, C^{Res}, I^{Res})$, given the value of $pa(X_i)$ for its parent node, the degree of dependency of each child node on the parent node can be determined and represented as the probability of the likelihood in the Bayesian formula, that is, the severity of the vulnerability of the child node relative to the parent node. After determining the Bayesian network structure and the initial vulnerability severity of the nodes, a Bayesian network model is used for inference to ultimately calculate the vulnerability severity $p'(X_i)$ of each node under the previous conditions.

In order to obtain a more reasonable and comprehensive definition of a network attack surface, this paper integrates previous definitions of network attack surfaces and takes into account node vulnerability, attack efficiency, and resource quantity when defining the network attack surface. Suppose the vulnerability severity of port m is $p'(m)$ and its network attack cost ratio is $der_m(m)$, $m \in M^{Res}$; The vulnerability severity of protocol c is $p'(c)$, and its network attack cost ratio is $der_c(c)$, $c \in C^{Res}$; The vulnerability severity of untrusted data d is $p'(d)$, and its network attack cost ratio is $der_d(d)$, $d \in I^{Res}$. The network attack surface metric can be represented by the following triplet.

$$(p'(m)\sum_{m\in M^{Res}} der_m(m), p'(c)\sum_{c\in C^{Res}} der_c(c), p'(d)\sum_{d\in I^{Res}} der_d(d)),$$
$$der_i(i) = \frac{dl_i(i)}{ac_i(i)}\partial(i), i \in (M^{Res}, C^{Res}, I^{Res}) \quad (1)$$

where dl_i is an indicator of the damage degree of the network attack on the resource, ac_i is an indicator of the attack cost paid by the attacker to attack the resource, and $\partial(i)$ is the quantity of resource type i, where the higher the metric value obtained, the larger the network attack surface.

3.2. Evaluate Network Attack Surface

Currently, the quantitative evaluation of network security performance based on the network attack surface is typically conducted by considering all exposed ports, protocols, and untrusted data [2]. However, not every resource on the network attack surface in a complex network system can be leveraged by attackers to achieve their objectives. Therefore, relying solely on the network attack surface for network security assessment may result in significant discrepancies between the assessment results and the actual state.

For attackers, the launch of a network attack usually aims to achieve the attack goal at the minimum cost. Different attack targets have different attack methods, and the attacker usually does not blindly test any network resource exposed by the attack network. Therefore, the main threat to the network is the network resources involved in different attack methods. This paper starts from the perspective of the attackers. Firstly, a Bayesian network resource graph is constructed on the basis of the information system to sift out the core modules of the information system. Secondly, the improved AOA is used to solve the optimal attack approach for the core modules of the network information system. Finally, the significant resources of the top K involved in different attack approaches are selected to calculate the network attack surface. This offers a reference for the optimized design of the security defense system of a network information system. The schematic diagram of the network attack surface evaluation framework is shown in Figure 2.

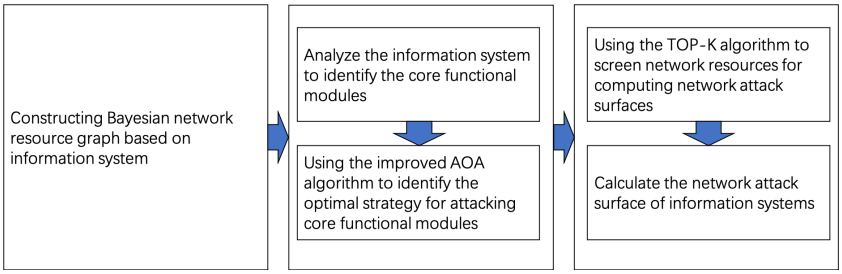

Figure 2. The complete computational framework of network attack surfaces.

3.2.1. The Network Security Evaluation Framework for Information Systems Filter Network Resources

This section describes in detail the selection of network resources from the information system network resource set to evaluate the network attack surfaces. To evaluate the network security performance of information systems, different attack methods are selected to simulate attacks on network information systems composed of N subsystems. Select the optimal Φ attack strategy for each subsystem to achieve the goal of disabling the system at the lowest cost. The optimization of the optimal attack strategy is an optimization problem, and this paper improves the AOA algorithm to solve this problem. Compared to other optimization algorithms, AOAs converge faster and have better results in finding the optimal solution [33]. We add a convergence check to the AOA. When the difference between two iterations is less than a given threshold, it indicates that the algorithm has converged, and

the iteration is stopped. This change improved the efficiency of the algorithm. The entire framework of the algorithm is shown in Figure 3.

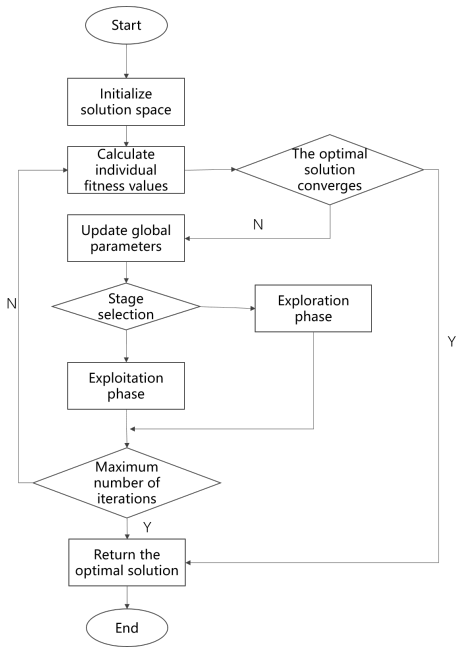

Figure 3. Process of improved arithmetic optimization algorithm.

In this paper, the solution space is encoded as a binary vector. Assuming that the network resources of information system E are R_es and the number of network resources is $|R_es|$, the length of a solution is set to $|R_es|$, and each bit corresponds to a network resource r, $r \in R_{es}$. If the current strategy includes the network resource r, the value at the position corresponding to r in the solution vector is 1; otherwise, it is 0. A solution corresponds to a vector of length $|R_es|$, and different solutions correspond to different vectors.

The fitness function measures the quality of solutions and guides the optimization process. The operation of updating each solution is controlled by changing the fitness value. The choice of fitness function directly affects the convergence speed of the algorithm. Due to the fact that arithmetic optimization algorithms do not use external information in evolutionary search, they only rely on fitness functions, and each solution is evaluated and searched for based on fitness. Therefore, the design of the fitness function must be as simple as possible to minimize the search time.

For attack strategies, the shorter the time required to achieve the attack effect, the better. The greater the impact of attack results on system indicators, the better. At the same time, the higher the success rate of the attack, the better. Therefore, Formula (2) can describe the fitness function.

$$f(s) = \frac{1}{Time(s)} + Attack(s) * Influ(s) \tag{2}$$

where s is a vector representing a solution, that is, an attack strategy. $Time(s)$ represents the time to complete the attack according to the attack strategy s, $Attack(s)$ represents the probability of successful attack according to the attack strategy s, and $Influ(s)$ represents the degree of impact on the system after completing the attack according to the attack strategy s.

The attack time of different network resources is calculated based on historical simulation data. The attack time of the network resources involved in the attack policy is summed up in the attack time of the attack policy, which is represented in Formula (3).

$$Time(s) = \sum_{i=1}^{i=|Res|} s^i * r_t^i \tag{3}$$

where s^i represents the i-th bit of vector s, and r_t^i represents the attack time corresponding to the i-th bit of the network resources in the system. r_t^i is calculated based on historical data $r_t^i = \frac{\sum historical(r_t^i)}{N}$, where $historical(r_t^i)$ represents the historical data of r_t^i.

The probability of a successful attack is calculated on the basis of historical simulation data analysis. Assuming that the probabilities of each network resource being breached are independent of each other, the probability of successful attack on the attacked resource in attack strategy s can be expressed as Formula (4):

$$Attack(s) = \prod_{i=1}^{i=|Res|} s^i * p^i \tag{4}$$

where p^i represents the probability that the i-th bit network resource in the system is successfully attacked. p^i is calculated based on historical data: $p^i = \frac{\sum historical(p^i)}{N}$, where $historical(p^i)$ represents the historical data of p^i.

The cumulative mechanism is used to calculate the impact of attack policies on system performance indicators. That is, the comprehensive impact is equal to the accumulation of different impacts, which is expressed in Formula (5).

$$Influ(s) = \sum_{i=1}^{i=|Res|} s^i * D^i \tag{5}$$

where D^i represents the degree to which the i-th bit network resources in the system affect the performance index of the network system.

To sum up, the final form of the fitness function is shown in Formula (6):

$$f(s) = \frac{1}{\sum_{i=1}^{i=|Res|} s^i * r_t^i} + \prod_{i=1}^{i=|Res|} s^i * p^i * (\sum_{i=1}^{i=|Res|} s^i * D^i) \tag{6}$$

The fitness function is also an important basis for judging whether the current optimal solution converges. At the beginning of the optimization algorithm, based on the fitness function, the algorithm rapidly iterates in the direction of the superoptimal solution, and the optimization effect is significant. When approaching the optimal solution, the difference between the results of two adjacent iterations becomes smaller. When the change is less than the set threshold, the iteration can be stopped. In this paper, the convergence judgment is based on the optimal solution generated by each iteration. When the difference between the moving mean of t times and the moving mean of $t-1$ times is less than the threshold value θ, the algorithm is considered converged, and the iteration is ended. To avoid becoming stuck in local optima, the sliding average is used here to compare the results between adjacent iterations. The convergence conditions are formulated as Formula (7).

$$\left| \overline{\mathrm{f}(s^{best})}_t - \overline{\mathrm{f}(s^{best})}_{t-1} \right| < \theta \tag{7}$$

where $\overline{\mathrm{f}(s^{best})}_t$ represents the moving mean of the fitness function of the optimal solution at the time of iteration t; θ is a very small positive number 0.000001. The calculation window for the moving average is set to 5 based on the characteristics of the experimental data, and smaller windows may be affected by noise and outliers.

There are two crucial parameters in the AOA iterative optimization process: math optimizer accelerated (MOA) and math optimizer probability (MOP). MOA controls the optimization process and enters different search stages according to the current discriminant conditions. The calculation formula for the MOA is shown in Formula (8):

$$\mathrm{MOA}(T_{Iteration}) = Min + T_{Iteration} * (\frac{Max - Min}{MAX\_Iteration}) \tag{8}$$

where $T_{Iteration}$ indicates the number of current iterations, and MOA($T_{Iteration}$) indicates the value of the MOA during this iteration. *Min* and *Max* represent the minimum and maximum MOA values, respectively. $MAX\_Iteration$ is the maximum number of iterations.

Under the AOA framework, when entering the exploitation or exploration phase, two stages are randomly selected. During the exploration phase, addition and subtraction are used, as shown in Formula (9), while during the exploitation phase, multiplication or division is used, as shown in Formula (10).

$$s_{i,j}(T_{\text{Iteration}}+1) = \begin{cases} s_j^{\text{best}} - MOP \times \left((UB_j - LB_j) \times \mu + LB_j\right), r < 0.5 \\ s_j^{\text{best}} + MOP \times \left((UB_j - LB_j) \times \mu + LB_j\right), 0.5 \le r \end{cases} \tag{9}$$

$$s_{i,j}(T_{\text{Iteration}}+1) = \begin{cases} s_j^{\text{best}} \div (MOP + \varepsilon) \times \left((UB_j - LB_j) \times \mu + LB_j\right), r < 0.5 \\ s_j^{\text{best}} \times MOP \times \left((UB_j - LB_j) \times \mu + LB_j\right), 0.5 \le r \end{cases} \tag{10}$$

where $s_{i,j}(T_{\text{Iteration}}+1)$ represents the updated value of $s_{(i,j)}$ during the $T_{\text{Iteration}}$ iteration process, and $s_{(i,j)}$ represents the j-th bit of the i-th vector. s_j^{best} represents the j-th position of the current optimal solution, while UB_j and LB_j represent the upper and lower bounds on the value of the j-th position of the vector, respectively. μ is a control parameter used to adjust the search process, ε is a very small integer to ensure that the divisor is not 0, and r is a randomly generated number between 0 and 1 each time.

MOP, as a key parameter shared by different update operations, is calculated using Formula (11).

$$\text{MOP}(T_{Iteration}) = 1 - \frac{T_{Iteration}^{\frac{1}{\alpha}}}{MAX\_Iteration^{\frac{1}{\alpha}}} \tag{11}$$

MOP($T_{Iteration}$) represents the value of the MOP in the $T_{Iteration}$ iteration; α is an accuracy parameter that controls the magnitude of changes in each update operation.

The complete process optimized based on the AOA framework is expressed using algorithms (Algorithm 1) as follows:

Algorithm 1 Optimization algorithm based on AOA framework.

```
Initialize the parameters for algorithm operation α, μ
Randomly generate solution vectors S = {s1, s2, s3.....sN}
while (T_Iteration < MAX_Iteration) do
    Calculate the fitness function value of each solution according to Formula (2) and find the optimal solution up to the current point
    Determine whether convergence is achieved according to Formula (7)
    Update MOA according to Formula (8) and update MOP according to Formula (11)
    for i = 0 to n do
        for j = 0 to |Res| do
            Generate random values r1, r2, r3 between [0,1]
            if r1 > MOA then
                Updates_{i,j}(T_Iteration + 1) according to Formula (10)
            else
                Updates_{i,j}(T_Iteration + 1) according to Formula (9)
            end if
        end for
    end for
    T_Iteration = T_Iteration + 1
end while
return the best (S)
```

Network Attack Surface Calculation

This section provides a detailed introduction to calculating the network attack surface of information systems based on the core resources of the network. Each network resource in an information system has a different impact on the security of the entire system. There

are a large number of edge network resources, but they are far away from the control center of the system and have little impact on system security. Using edge network resources to calculate the network attack surface easily interferes with the final evaluation results. This article uses a network attack surface composed of the top K core network resources to measure system security, and the optimal value of K is selected through experiments.

After selecting the attack strategies of different attack methods, the system network security evaluation matrix A is constructed. The rows of the matrix represent the resources of the network within the system, while the columns represent specific attack methods targeting the network system. A typical example is shown in Figure 4.

	M_1	M_2	M_3	.....	M_n
NetRes_1	1	0	1		0
NetRes_2	0	1	1		0
NetRes_3	1	1	0		0
.....					
NetRes_m	0	0	0		1

Figure 4. An example of network security evaluation matrix A, where rows represent network resources and lists attack methods, and the first column (101... 0) indicates that attack method M_1 utilizes two types of network resources NetRes_1 and NetRes_3.

According to the previous analysis, the network resources used to calculate the network attack surface of information systems should have an impact on network security and are important resources. Resources that have little or no impact on network security can be ignored [20]. In order to more accurately evaluate the network security of information systems, based on the network security evaluation matrix A of information systems, core network security resources are selected.

Firstly, based on the network security assessment matrix A of the information system, solve matrix B, which can characterize the importance of nodes to the network security of the information system. Suppose that an attack method involves nodes with the same contribution rate to achieving the attack objective; for example, if three network resources are used in a single attack, the contribution rate of each network resource to the attack is 1/3 = 33.33%. Based on this assumption, the expression of matrix B can be expressed as Formula (12).

$$B_{i,j} = \frac{A_{ij}}{\sum_{k=1}^{k=m} A_{kj}} \tag{12}$$

After obtaining matrix B, the size of matrix B is mn, and the importance of the i-th network resource ψ_i can be expressed as Formula (13).

$$\psi_i = \sum_{j=0}^{j=n} B_{ij} \tag{13}$$

After completing the above operations, matrix B can be used to calculate the importance of each network resource for system network security. In order to select core network resources and improve the accuracy of network evaluation in evaluating information system network security, a threshold K is set and the top K network resources are selected as the input set of network attack surfaces, denoted as E_{input}, for evaluating information system network security. The top K selection is used to sort each network resource based on its importance, and the K nodes with the highest importance are selected. After obtaining the input set E_{input} to evaluate the network attack surface, the network attack surface can be calculated according to Formula (14).

$$\left(p'(m)\sum_{m\in M^{E_{\text{input}}}}\text{der}_m(m), p'(c)\sum_{c\in C^{E_{\text{input}}}}\text{der}_c(c), p'(d)\sum_{d\in I^{E_{\text{input}}}}\text{der}_d(d)\right) \quad (14)$$

The complete information system network security evaluation model includes the above two parts, which are mainly divided into two steps. In the first step, the core network resource module is selected as the i-th attack target T_i, $T_i \in R_{es}$, and the optimal attack strategy S_i for attack $targetT_i$ is found. The optimal attack strategy of multiple attack targets is composed of an attack policy set S. In the second step, from the network resources R_s, $R_s \subseteq R_{es}$ involved in the policy set S, the core network resources are selected as input for the measurement of the network attack surface, and the network attack surface is calculated as the value of the evaluation of network security of the information system.

For ease of expression and understanding, suppose that $G(m) = p'(m)\sum_{m\in M^{Res}} der_m(m)$, $G(c) = p'(c)\sum_{c\in C^{Res}} der_c(c)$, $G(d) = p'(d)\sum_{d\in I^{Res}} der_d(d)$. Assuming that there are $|T_i|$ core attack targets for information system E, the value of the evaluation of the network security of the information system E can be expressed as Formula (15).

$$\begin{aligned} V(E) &= \left(\frac{p^p(m)\sum_{m\in M^{Rs}} der_m(m)}{|T_i|}, \frac{p'(c)\sum_{c\in C^{Rs}} der_c(c)}{|T_i|}, \frac{p^p(d)\sum_{d\in I^{Rs}} der_d(d)}{|T_i|}\right) \\ V(E) &= \left(\frac{G^{RS}(m)}{|T_i|}, \frac{G^{Rs}(c)}{|T_i|}, \frac{G^{RS}(d)}{|T_i|}\right) \end{aligned} \quad (15)$$

4. Experiment and Discussion

In this study, a series of experiments with different specifications were carried out to highlight the performance and efficiency of the proposed method. This section is divided into two parts. The first part introduces the experimental setup, including the dataset used in the experiment, comparison methods, and evaluation indicators. The other part analyzes the experimental results.

4.1. Experiment Setup

In order to verify the performance of the proposed method in evaluating the network attack surface of information systems, this paper selected three real intelligent control systems $E1$, $E2$ and $E3$ for the experiments. Firstly, we used the network situational awareness platform as an experimental environment to simulate three different network information systems, conducted penetration tests and attacks, and collected simulation data. Secondly, we created application system network resource topology structures based on business processes. Then, we assigned different attributes to network resources in different systems based on simulation data and indicators defined in the Common Vulnerability Scoring System (CVSS) version 3.1 of the National Institute of Standards and Technology (NIST). Finally, we established the Bayesian network resource map for the network information system, calculated the network attack surface of different application systems using different algorithms, and compared and analyze the experimental results of different algorithms.

4.1.1. Dataset Description

The three systems used in the experiment were real systems. $E1$ is the scheduling control system, $E2$ is the production control system, and $E3$ is the environmental control system. They had different network sizes and contained different types and quantities of network resources. Each of these three systems included multiple data acquisition modules, data processing modules, data communication modules, data management models, databases, and firewalls. We built simulation environments for the three systems on the network situation simulation awareness platform and used automated scripts to

simulate attacks. The simulated attacks mainly included denial of service (DoS), remote-to-login (R2L), user-to-room (U2R) and port (PAT) attacks, and the collected data included attack traffic samples and normal traffic samples. For example, the simulation data based on E1 included 734 normal traffic samples and 3730 attack traffic samples. Detailed statistical information about the simulation data is shown in Table 1.

Table 1. Statistical information of the simulation data.

Sample Type	E1	E2	E3
Normal	734	711	715
DoS	927	858	742
R2L	995	754	1063
U2R	152	200	213
PAT	1656	2421	2402
Total Attack Count	3730	4233	4420

Based on the simulation data and Bayesian network resource diagrams of the structural components of the system, they are represented as Network 1, Network 2, and Network 3. Network1 represents a simulated network built on the information system $E1$, which includes all resources of information system $E1$. Network 2 represents a simulation network built on network information system $E2$, including all network resources of information system $E2$; Network 3 represents a simulated network constructed based on network information system E3 and includes all network resources of information system E3. Each row in the Network dataset represents a network resource, containing detailed information such as the network resource ID, the network resource name, the type, related network resource IDs, relationships, communication protocols, and network resource attributes. The statistical characteristics of the different datasets from the network are shown in Table 2.

Table 2. Statistical characteristics of the Network dataset.

	M	C	I
Network 1	80	20	600
Network 2	200	80	800
Network 3	500	200	1000

4.1.2. Baseline Algorithms

In order to verify the performance of the proposed evaluation method, this paper selected the most typical method based on the network evaluation surface and the latest method as comparative analysis methods. The selected comparative methods are introduced in what follows.

Security Threat Evaluation Method Based on Attack Surface (STEM) [19]: This method integrates the ports, protocols, and data of various resource components to model the network attack surface of resource nodes. Based on Bayesian networks, a resource graph is established, and the network attack surface measurement is defined as the triple combination of the severity of the vulnerability of the resource node and the sum of the cost-effectiveness ratios of various resource attacks.

Security Evaluation Method Based on Attack Surface (SEM): This method is based on the network attack surface proposed by Howard et al. [21] and incorporates the attack cost-effectiveness ratio.

Network Attack Surface Evaluation Method Based on Optimal Attack Strategy (NASEM): The method proposed in this paper first solves the optimal attack strategy for the core attack target and then calculates the network attack surface based on the optimal attack strategy to evaluate the security of network information systems.

Network Attack Surface Evaluation Method Based on Genetic Algorithm (NASEMGA): This method is based on the framework proposed in this article, and the selection of the optimal attack strategy is carried out using the latest improved genetic algorithm [27].

Network Situation Simulation Awareness Platform (NSSAP): The networking situation of the information system is simulated through the network situation simulation perception platform, and multiple rounds of simulation are conducted to derive the network attack surface of the network information system as a benchmark. The results of using NSSAP for the evaluation of the network attack surface have high accuracy, but the process is complex and inefficient, requiring a large amount of simulation.

4.1.3. Evaluation Measure

In order to objectively evaluate the performance of different algorithms, this article chose the evaluation results of the Network Situation Simulation Awareness Platform (NSSAP) as the benchmark. The smaller the error between the network attack surface evaluation results and the NSSAP calculation results, the better. To measure the results of the comparison, we chose error, difference, and similarity as the main indicators for the experimental evaluation.

The error is defined as the sum of the differences in the M, C, and I attributes between the network attack surface calculated according to an algorithm and the NSSAP evaluation value. The smaller the error, the better.

Difference refers to the difference between the evaluation values of two network attack surfaces for a system. When there is a change in the status of the evaluation system, the evaluation value should be able to reflect the change, where the larger the difference value, the better. This indicator was used to evaluate whether the network attack surface evaluation algorithm is sensitive to changes in the system. For example, if the current evaluation value of a system is A, and the evaluation value after reinforcing the attack surface of the system is B, the more obvious the difference between A and B, the better. The larger the difference, the more sensitive the evaluation algorithm to changes in the state of the system.

Similarity refers to the degree of similarity between the network resources used by an algorithm to calculate the network attack surface and the results of the NSSAP selection. The higher the repetition and similarity of the network resources, the better the results. This indicator is used to evaluate the accuracy of selecting network resources to compute network attack surfaces. If the similarity between the network resource, which is used for calculating the evaluation value of network attack surface using an algorithm, and the NASSP-selected network resources is high, their evaluation values are also similar.

4.2. Results and Discussion

In this part, the experimental results of the proposed algorithm and comparative methods on different datasets are analyzed in detail. This part is divided into three sections; the first section introduces the comparison between similar methods, the second part compares the performance of different optimization modules under the framework proposed in this paper, and the third part analyzes the key parameters. During the experiment, referring to the algorithm settings in [27], the maximum number of iterations in the network attack surface evaluation method based on the optimal attack strategy was set to 1000, the parameters of the network attack surface evaluation method based on the genetic algorithm were set to a population size of 100, the termination evolutionary algebra of the genetic algorithm was 1000, and the probability of crossover was 0.2. The probability of variation was 0.002. In order to obtain more accurate results, each round of experiments was repeated 100 times, and the average value was taken as the final experimental result. In order to

facilitate the comparison of the results of different algorithms, the experimental results of different algorithms were normalized in the experimental analysis.

4.2.1. Comparison Between Methods

The first part of the experiment analyzed the results of the different algorithms on the different datasets. The experimental results are shown in Figures 5–7, where the values of the different coordinate axes correspond to the values of the different attributes of the network attack surface. We used error to compare the performance of the different algorithms. The X axis in the figure represents the value of the network attack surface attribute M, the Y axis represents the value of attribute C, and the Z axis represents the value of attribute I. The closer the calculation results of the algorithm to the NSSAP evaluation results, the better. Observing the experimental results, it can be found that on different networks, the results of NASEM and NSSAP were the closest, and the results of STEM and SEM were significantly different from those of NSSAP.

After conducting network evaluations on Network 1, Network 2, and Network 3, each of the three information systems underwent network attack surface reinforcement to reduce system vulnerabilities and lower the vulnerability of the network resources. After reinforcement, we recalculated the network attack surface of the different information systems and compared the results of the two evaluations. We used the second metric, difference, to compare the performance of the different algorithms. As a measure of system network security, the more sensitive the network attack surface to changes in the system's network security status, the better. In theory, due to targeted security reinforcement, security has improved, and the measured value of the network attack surface should undergo significant changes. To compare the changes in the evaluation values of the different methods, we normalized the differences according to Formula (16).

$$Difference_i^{method} = \frac{\left| Previousvalues_i^{method} - Subsequentvalues_i^{method} \right|}{\left| Previousvalues_i^{NSSAP} - Subsequentvalues_i^{NSSAP} \right|} \tag{16}$$

where $Difference_i^{method}$ represents the change in the value of attribute i before and after the reinforcement of the network attack surface when using $method$ for network attack surface evaluation. $Previousvalues_i^{method}$ represents the value before strengthening the network attack surface, and $Subsequentvalues_i^{method}$ represents the value after strengthening the network attack surface, $i \in \{M, C, I\}$.

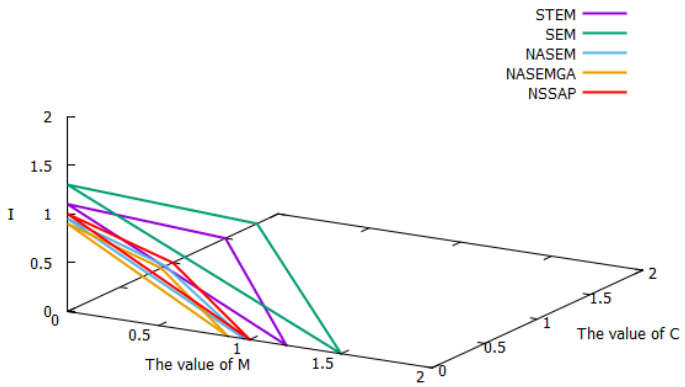

Figure 5. Experimental results of different algorithms on Network 1 dataset.

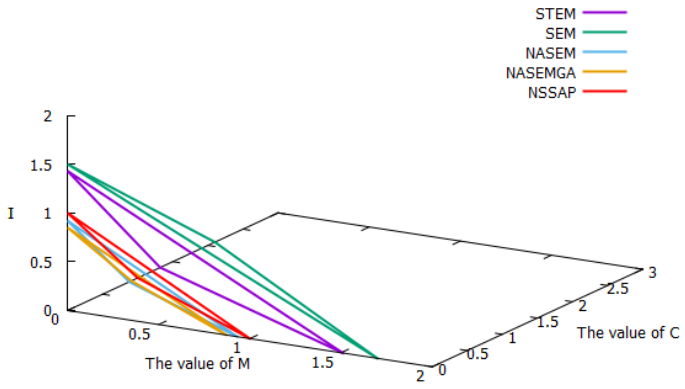

Figure 6. Experimental results of different algorithms on Network 2 dataset.

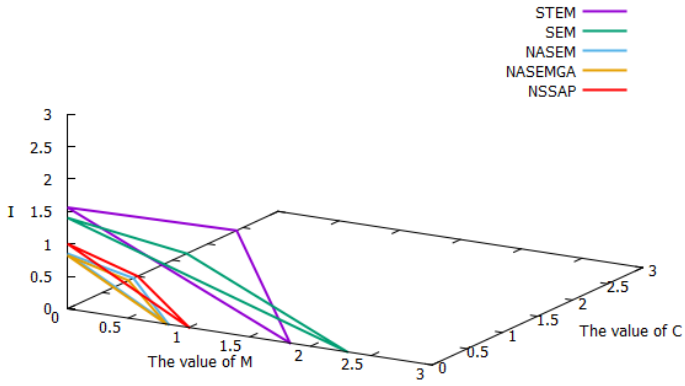

Figure 7. Experimental results of different algorithms on Network 3 dataset.

The experimental results are shown in Figures 8–10. Observing the experimental results, it can be found that on the three datasets, NSSAP and NASEM produced the most similar and significant changes in the measurement values before and after network attack surface enhancement. Among the four different methods, the NASEM results had the smallest error compared to the NSSAP results. The average error of NASEM was less than 20%.

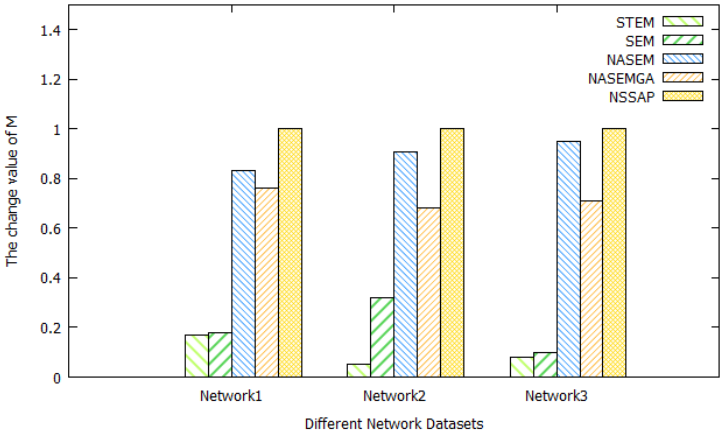

Figure 8. Changes in M values of different information systems before and after network attack surface reinforcement.

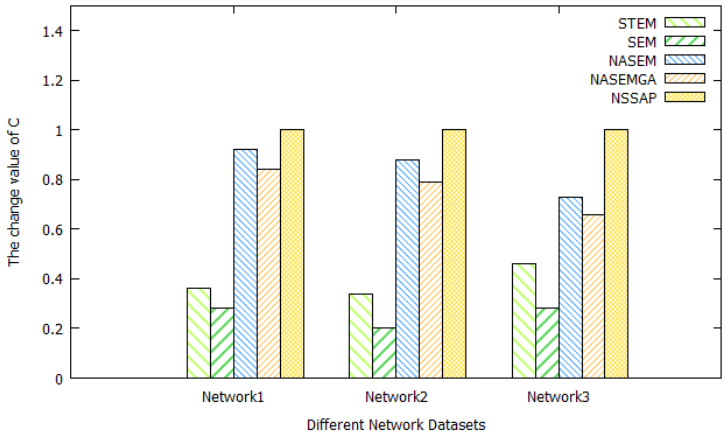

Figure 9. Changes in C values of different information systems before and after network attack surface reinforcement.

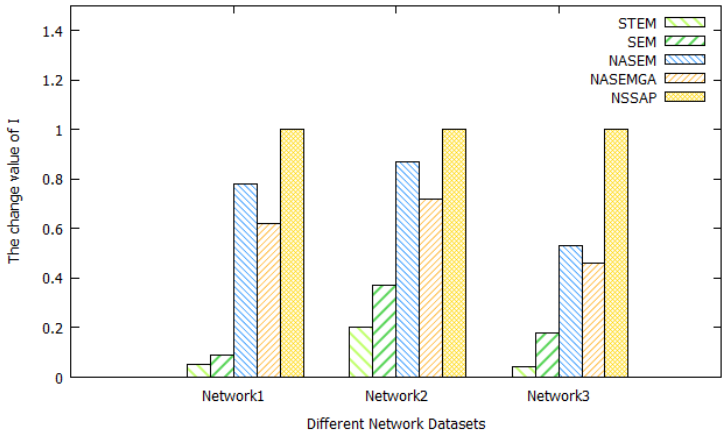

Figure 10. Changes in I values of different information systems before and after network attack surface reinforcement.

Based on Network 1 and using NSSAP as a benchmark, we analyzed the number of resources used to calculate the network attack surface. The experimental results are shown in Figure 11. To evaluate the different algorithms, we used a third metric, the similarity metric. Through experiments, it was found that NASEM was the most similar to NSSAP in calculating the network attack surface network resources, while STEM and SEM selected the network resources with the highest repetition rate compared to NSSAP.

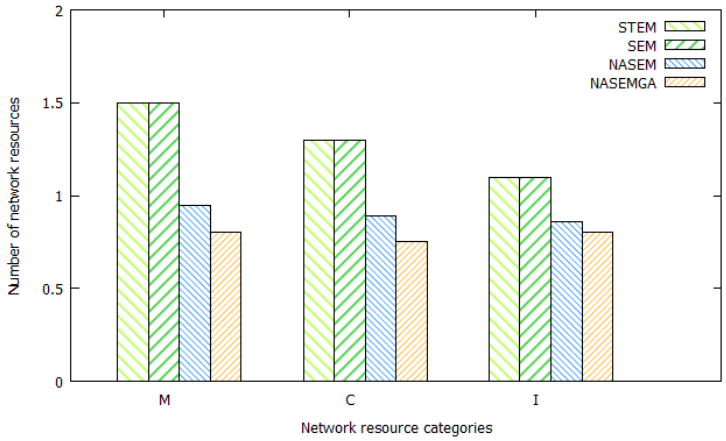

Figure 11. Comparison of network resource quantity for network attack surface computing using different algorithms.

The reason for the persistence of this phenomenon is that STEM and SEM use all the network resources in the process of calculating the network attack surface, and this operation causes the calculated result to be too large. The network resources selected by NASEM and NASEMGA are more accurate and more sensitive to the state changes of key network resources when evaluating the security of the system network.

4.2.2. Performance of Different Optimization Modules

This section compares the impact of the different optimization algorithms on the final results in the evaluation framework proposed in this paper through experiments.

During the experiment, the arithmetic optimization algorithm in the evaluated method was replaced with other optimization algorithms, and the results were compared. The results of the network situation simulation perception platform were used as a benchmark for comparison. The error between the evaluation results was calculated using the following Formulas (17) and (18):

$$V(E) = \left(\frac{p^p(m) \sum_{m \in M^{Rs}} der_m(m)}{|T_i|}, \frac{p'(c) \sum_{c \in C^{Rs}} der_c(c)}{|T_i|}, \frac{p^p(d) \sum_{d \in I^{Rs}} der_d(d)}{|T_i|} \right) \tag{17}$$

$$V(E) = \left(\frac{G^{RS}(m)}{|T_i|}, \frac{G^{Rs}(c)}{|T_i|}, \frac{G^{RS}(d)}{|T_i|} \right) \tag{18}$$

where M_t, C_t, and I_t, respectively, represent the different attribute values in the network attack surface calculated by the different algorithms. M_B, C_B, and I_B correspond to the different attribute values obtained from evaluating network attack surfaces using the benchmark algorithms. The error of the network attack surface evaluation values obtained using the different optimization algorithms in the different experimental environments is shown in Table 3. The arithmetic optimization algorithm used in this paper had the best overall performance.

Table 3. Evaluation errors of different optimization algorithms based on the evaluation framework in this article under different experimental environments.

	Network 1	**Network 2**	**Network 3**
Genetic algorithm	0.39	0.49	0.28
Differential optimization algorithm	0.34	0.47	0.53
Particle swarm optimization algorithm	0.51	0.52	0.49
Coyote optimization algorithm	0.68	0.35	0.37
Arithmetic optimization algorithm	0.26	0.41	0.21

4.2.3. Parameter Analysis

This section takes NSSAP as a benchmark and compares the impact of different experimental parameters on the evaluation results through experiments. There is an important parameter K in the method proposed in this paper, which we used to calculate the top K strategies for network attack surfaces. Different K values can result in significant fluctuations in the final evaluation results.When a certain K value is selected, the closer the evaluation result to the baseline NSSAP value, the better.

Fixing the other experimental conditions and continuously changing the K value, the ratio of the evaluation results to the benchmark results was analyzed. The experimental results are shown in Figure 12. The vertical axis in the figure represents the ratio of the evaluation results to NSSAP under different parameters. Observing the experimental results, it can be found that the results of NSSAP were the most similar when K was seven.

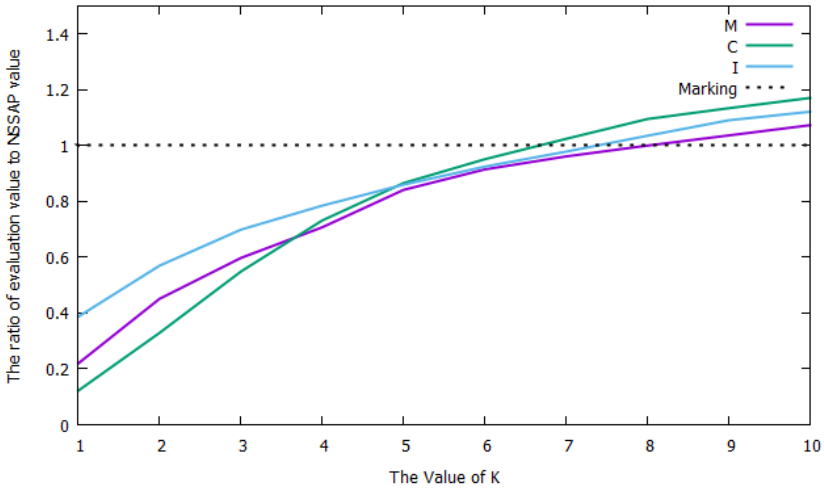

Figure 12. The impact of the K value on the evaluated algorithm.

5. Conclusions

With the continuous development of information technology, the cyberspace environment is becoming increasingly complex, and how to evaluate the security of network information systems has become a hot research topic. The network attack surface, as an important indicator used for measuring the security of network information systems, has attracted great attention from researchers. There are two types of methods for evaluating the network attack surface of information systems: one is to use all network resources for computational evaluation, ignoring the different levels of importance of the different network resources for network security; the other is to personalize information systems, limiting the scope of application of the evaluation methods. In order to find a balance between accuracy and robustness, this paper proposes a method based on an optimal attack strategy, which can identify the most important core resources of a network, distinguish the importance of various resources, and improve the accuracy of the final evaluation results. It is also suitable for different network information systems, with a wide range of applicability and strong robustness. A series of simulation experiments on different datasets confirmed that the proposed method is more sensitive to changes in the network attack surface of information system networks, and the results are closer to the real situation. This method is more flexible and efficient than using NSSAP for the evaluation of network attack surfaces. This method provides a new and more effective technical basis for testing and identifying the security of information system networks, which can guide the design and development of the system.

In the future, we would like to continue to conduct research on the evaluation of the security of information system networks based on our existing work. The current method assumes that the network is static and invariant, relying on accurate prior probabilities in Bayesian networks. A potential extension in the future is to evaluate dynamic network systems.

Author Contributions: Conceptualization, Z.L. and P.X.; methodology, P.X.; software, L.Z.; validation, L.Z. and P.X.; formal analysis, P.X. and J.Y.; investigation, P.X. and J.Y.; resources, Z.L.; data curation, P.X.; writing—original draft preparation, L.Z.; writing—review and editing, P.X. and J.Y.; supervision, Z.L.; project administration, Z.L.; funding acquisition, Z.L. All authors have read and agreed to the published version of this manuscript.

Funding: This research was funded by the Key R&D Program of Jiangsu (No. BE2022081).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to validate the outcomes of this investigation can be obtained by contacting the corresponding author.

Conflicts of Interest: Peng Xie, Lin Zhang and Jianxin Yang were employed by the Company Information Center of China North Industries Group Corporation. The remaining authors declare that the research was conducted without any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Zuo, J.; Guo, Z.; An, T.; Xu, Z.; Lu, Y. A Security Resilience Metric Framework Based on the Evolution of Attack and Defense Scenarios. *IEEE Internet Things J.* **2023**, *10*, 17007–17021. [CrossRef]
2. Zhaojun, G.; Rui, Y.; He, S. System Attack Surface Modeling Method in Network. *Netinfo Secur.* **2022**, *22*, 29–38.
3. Xuan, Z.; Hongzhi, L.; Tong, L.; Jing, X.; Qianru, Z.; Ye, Q. Software security measurement based on information entropy and attack surface. *J. Comput. Appl.* **2013**, *33*, 19–22, 44.
4. Lian Xinke, Y.Q. Security Evaluation System based on Attack Surface. *Commun. Technol.* **2020**, *53*, 2567–2572.
5. Howard, M.; Pincus, J.; Wing, J.M. *Measuring Relative Attack Surfaces*; Springer: Boston, MA, USA, 2005; pp. 109–137.
6. Theisen, C.; Herzig, K.; Murphy, B.; Williams, L. Risk-based attack surface approximation: How much data is enough? In Proceedings of the 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP), Buenos Aires, Argentina, 20–28 May 2017.
7. Manadhata, P.K.; Tan, K.M.C.; Maxion, R.A. *An Approach to Measuring A System's Attack Surface*; US Army Research Office (ARO): Durham, NC, USA, 2007.
8. Kurmus, A.; Tartler, R.; Dorneanu, D.; Heinloth, B.; Rothberg, V.; Ruprecht, A.; Schroder-Preikschat, W.; Lohmann, D.; Kapitza, R. *Attack Surface Metrics and Automated Compile-Time OS Kernel Tailoring*; NDSS: San Diego, CA, USA, 2013.
9. Peng, W.; Li, F.; Huang, C.T.; Zou, X. A moving-target defense strategy for Cloud-based services with heterogeneous and dynamic attack surfaces. In Proceedings of the 2014 IEEE International Conference on Communications (ICC), Sydney, NSW, Australia, 10–14 June 2014.
10. Chris Foreman, J.; Gurugubelli, D. Identifying the Cyber Attack Surface of the Advanced Metering Infrastructure. *Electr. J.* **2015**, *28*, 94–103. [CrossRef]
11. Sun, K.; Jajodia, S. Protecting Enterprise Networks through Attack Surface Expansion. In Proceedings of the 2014 Workshop on Cyber Security Analytics, Intelligence and Automation, Scottsdale, AZ, USA, 3 November 2014.
12. Cybenko, G.; Jajodia, S.; Wellman, M.P.; Liu, P. Adversarial and Uncertain Reasoning for Adaptive Cyber Defense: Building the Scientific Foundation. In Proceedings of the International Conference on Information Systems Security, Hyderabad, India, 16–20 December 2014.
13. Bopche, G.S.; Mehtre, B.M. Graph similarity metrics for assessing temporal changes in attack surface of dynamic networks. *Comput. Secur.* **2017**, *64*, 16–43. [CrossRef]
14. Zhuang, R.; Zhang, S.; Deloach, S.A.; Ou, X.; Singhal, A. Simulation-based Approaches to Studying Effectiveness of Moving-Target Network Defense. In Proceedings of the National Symposium on Moving Target Research, Annapolis, MD, USA, 11 June 2013.
15. Li, P.; Ye, D. Vulnerability Analysis of Distributed State Estimator Under False Data Injection Attacks. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 5235–5244. [CrossRef]
16. Li, J.; Zhou, H.; Wu, S.; Luo, X.; Wang, T.; Zhan, X.; Ma, X. FOAP: Fine-Grained Open-World Android App Fingerprinting. In Proceedings of the 31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, 10–12 August 2022; Butler, K.R.B., Thomas, K., Eds.; USENIX Association: Berkeley, CA, USA, 2022; pp. 1579–1596.
17. Ni, T.; Lan, G.; Wang, J.; Zhao, Q.; Xu, W. Eavesdropping Mobile App Activity via Radio-Frequency Energy Harvesting. In Proceedings of the 32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, 9–11 August 2023; Calandrino, J.A., Troncoso, C., Eds.; USENIX Association: Berkeley, CA, USA, 2023; pp. 3511–3528.
18. Younis, A.A.; Malaiya, Y.K. Relationship between attack surface and vulnerability density: A case study on apache http server. In Proceedings of the ICOMP'12, The 2012 International Conference on Internet Computing, Las Vegas, NV, USA, 12–15 July 2012.
19. Tran, A.D.; Yskout, K.; Joosen, W. AndrAS: Automated Attack Surface Extraction for Android Applications. In Proceedings of the 2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security (QRS), Chiang Mai, Thailand, 22–26 October 2023.
20. Dovgalyuk, P.; Klimushenkova, M.; Fursova, N.; Vasiliev, I.; Stepanov, V. Natch: Detecting Attack Surface for Multi-Service Systems with Hybrid Introspection. In Proceedings of the 2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion (QRS-C), Chiang Mai, Thailand, 22–26 October 2023; pp. 176–185.

21. Sunkavilli, S.; Zhang, Z.; Yu, Q. Analysis of Attack Surfaces and Practical Attack Examples in Open Source FPGA CAD Tools. In Proceedings of the 2021 22nd International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, 7–9 April 2021.
22. Fursova, N.; Dovgalyuk, P.; Vasiliev, I.; Klimushenkova, M.; Egorov, D. Detecting Attack Surface With Full-System Taint Analysis. In Proceedings of the 2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C), Hainan, China, 6–10 December 2021; pp. 1161–1162.
23. Zhang, Q.; Wu, B.; Zheng, M.; Ni, L.; Xue, B. IoT Device Attack Surface Feature Identification using Differentiated Dynamic Monitoring. In Proceedings of the 2022 IEEE 5th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), Shenyang, China, 18–20 November 2022; pp. 435–440.
24. Bopche, G.S.; Tiwari, D.; Rai, G.N. Boolean Similarity Measure for Assessing Temporal Variation in the Network Attack Surface. In Proceedings of the 2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 3–8 January 2023; pp. 31–36.
25. Wang, W.; Lin, X.; Wang, J.; Gao, W.; Gu, D.; Lv, W.; Wang, J. HODOR: Shrinking Attack Surface on Node.js via System Call Limitation. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, Copenhagen, Denmark, 26–30 November 2023; pp. 2800–2814.
26. Everson, D.; Cheng, L. A Survey on Network Attack Surface Mapping. *Digit. Threat.* **2024**, *5*, 1–25. [CrossRef]
27. Huang, F.; Guo, W.; Zhao, H. AGV Path Planning Based on Improved Genetic Algorithm. In Proceedings of the 2022 International Conference on Artificial Intelligence and Computer Information Technology (AICIT), Yichang, China, 16–18 September 2023; pp. 327–332.
28. Jiao, F.; Tao, L.; Nanjiang, D.; Rui, W. Multi-objective UAV Path Planning Based on an Improved NGSA-II Algorithm. *Fire Control Command. Control* **2022**, *47*, 43–48. 55.
29. Chai, X.; Zheng, Z.; Xiao, J.; Yan, L.; Qu, B.; Wen, P.; Wang, H.; Zhou, Y.; Sun, H. Multi-strategy fusion differential evolution algorithm for UAV path planning in complex environment. *Aerosp. Sci. Technol.* **2022**, *121*, 107287. [CrossRef]
30. Krell, E.; King, S.A.; Garcia, C.L.R. Autonomous Surface Vehicle energy-efficient and reward-based path planning using Particle Swarm Optimization and Visibility Graphs. *Appl. Ocean. Res.* **2022**, *122*, 103125. [CrossRef]
31. Pierezan, J.; Coelho, L.D.S. Coyote Optimization Algorithm: A New Metaheuristic for Global Optimization Problems. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018.
32. Chen, D.; Meng, X. UAV offline path planning based on self-adaptive coyote optimization algorithm. *Xi Tong Gong Cheng Yu Dian Zi Ji Shu/Syst. Eng. Electron.* **2022**, *44*, 603–611.
33. Abualigah, L.; Diabat, A.; Mirjalili, S.; Abd Elaziz, M.; Gandomi, A.H. The Arithmetic Optimization Algorithm. In *Computer Methods in Applied Mechanics and Engineering*; Elsevier: Amsterdam, The Netherlands, 2021; Volume 376.
34. Zhang, Y.; Jin, Z. Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems. *Expert Syst. Appl.* **2020**, *148*, 113246. [CrossRef]
35. Nobahari, H.; Eqra, N.; Bighashdel, A. Real Estate Market-Based Optimization Algorithm (REMARK): a market-inspired metaheuristic optimization algorithm based on the law of supply and demand. *J. Ambient. Intell. Humaniz. Comput.* **2023**, *14*, 12387–12405. [CrossRef]
36. Yuan, Y.; Ren, J.; Wang, S.; Wang, Z.; Mu, X.; Zhao, W. Alpine skiing optimization: A new bio-inspired optimization algorithm. *Adv. Eng. Softw.* **2022**, *170*, 103158. [CrossRef]
37. Yuan, Y.; Shen, Q.; Wang, S.; Ren, J.; Yang, D.; Yang, Q.; Fan, J.; Mu, X. Coronavirus Mask Protection Algorithm: A New Bio-inspired Optimization Algorithm and Its Applications. *J. Bionic Eng.* **2023**, *20*, 1747–1765. [CrossRef]
38. Seyyedabbasi, A.; Kiani, F. Sand Cat swarm optimization: A nature-inspired algorithm to solve global optimization problems. *Eng. Comput.* **2023**, *39*, 2627–2651. [CrossRef]
39. Jia, H.; Zhang, J.; Rao, H.; Abualigah, L. Improved sandcat swarm optimization algorithm for solving global optimum problems. *Artif. Intell. Rev.* **2025**, *58*, 5. [CrossRef]
40. Pan, J.S.; Nguyen, T.T.; Nguyen, T.D.; Nguyen, T.X.H.; Dao, T.K. An Optimal Power System Operation Planning Based on Enhanced Cuckoo Search Algorithm. In Proceedings of the International Conference on Genetic and Evolutionary Computing, Kaohsiung, Taiwan, 6–8 October 2024; Volume 1145 LNEE, pp. 533–544.
41. Liang, H.; Pang, A.; Lin, C.; Zhong, J. A Novel Hybrid Binary Bat Algorithm for Global Optimization. *Int. J. Swarm Intell. Res.* **2024**, *15*, 1–29. [CrossRef]
42. Li, H.; Yuan, H.; Zeng, D.; Wu, T.; Wang, Y.; Su, Y.; Zhang, B. Kalman Filter Based Improved Beluga Whale Optimization Algorithm. In Proceedings of the 2023 13th International Conference on Information Science and Technology (ICIST), Cairo, Egypt, 8–14 December 2023; pp. 222–231. [CrossRef]

43. Xiao-Jing, W.; Lei, X.; Ran, Z.; Xue-Li, W. Global and Local Moth-flame Optimization Algorithm for UAV Formation Path Planning Under Multi-constraints. *Int. J. Control Autom. Syst.* **2023**, *21*, 1032–1047.
44. Khalil, S.M.; Bahsi, H.; Korõtko, T. Threat modeling of industrial control systems: A systematic literature review. *Comput. Secur.* **2024**, *136*, 103543. [CrossRef]