Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.076043

ARTICLE

Active Defense Method for Network Hopping Based on Dynamic Random Graph

Zhu Fang[1,2,*], **Zhengquan Xu**[1,2], **Weizhen He**[3] **and Bohao Xu**[3]

[1]School of Electronic Information, Wuhan University, Wuhan, China
[2]State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, China
[3]Key Laboratory of Cyberspace Situation Awareness of Henan Province, Information Engineering University, Zhengzhou, China

*Corresponding Author: Zhu Fang. Email: fangzhu@whu.edu.cn

Received: 13 November 2025; Accepted: 26 January 2026; Published: 09 April 2026

ABSTRACT: In view of the problem that the IP address jump law is easy to predict in the current mobile target defense, this paper proposes a network address jump active defense method based on a dynamic random graph, designed to improve the unpredictability of IP address translation. Firstly, in order to make IP address transformation unpredictable in space and time, a random graph model is designed to generate a pseudo-random sequence of IP address randomization; these pseudo-random can meet the unpredictability of IP address translation in both space and time. Then, based on these pseudo-random sequences and IP address pool, a random map generation algorithm is proposed, which generates highly random IP address sequences through chaotic mapping (Logistic mapping) combined with encryption perturbation technology, meeting the requirements of resisting analysis attacks, while these transformed IP addresses are adapted to network target defense. And finally, this article uses buildMininet to build a cloud network trusted environment, by testing the spatial randomization and temporal randomization of the Random mapping model (CRM), the results show that the CRM model has a good effect on improving the local randomness. The test results of the ablation experiment further show that the CRM model can improve the local randomness while maintaining the global randomness.

KEYWORDS: Network active defense; IP hopping; dynamic random graph; logistic chaotic sequence; RC4 encryption

1 Introduction

With the development and iteration of digital technology, the functions of the Internet have been further expanded, gradually becoming a significant force in the production, daily life, and governance of human society. However, cyber attacks have occurred frequently in recent years. In February 2024, the "ESXiArgs" organization launched a blackmail attack targeting customers running the VMware ESXi virtual machine monitor, affecting over 3800 servers. In April 2024, Cactus ransomware attacked Aero Dynamic Machine, an American aerospace manufacturer, to steal 1.1 TB of data, including confidential drawings of SpaceX, Boeing, and other enterprises, directly threatening national security.

In the face of these network attacks, the current mainstream passive defense methods, such as firewalls, antivirus software, and intrusion detection systems, have significant shortcomings. For example, the essence of a firewall is a static rule-based defense system [1]; in the face of rapid iteration of attack means, it will cause the defense to lag [2]. Antivirus software relies on static file analysis and cannot detect fileless attacks in memory [3,4]. To address the shortcomings of these traditional defense methods, moving target defense (MTD) [5–7] emerged, increasing the difficulty and cost of attacks by dynamically changing system configurations, such as dynamic network topology, service virtualization, and protocol confusion [8,9].

IP address hopping, as one of its core technologies, can randomize or periodically change IP addresses, making it difficult for attackers to establish a persistent attack path [10].

Currently, the defense side enhances the unpredictability of IP address conversion to increase the cost of attacks, which can be achieved by dynamically adjusting the selection of IP addresses and the timing of hopping [11–13]. However, the IP hopping defense technology still faces the problem of identifying the IP address selected by the attacker to bypass the proxy server or SDN network hopping mechanism [14], as well as the challenge of predicting the current or next period's IP address based on historical data analysis.

Based on the above problems, to increase the unpredictability of IP address transformation and prevent attackers from analyzing historical data to predict the IP address for the next period, this paper proposes a network jump defense method based on a dynamic random graph. The main contributions of this paper are as follows:

(1) Constructs a dynamic random graph model integrating chaotic systems and cryptographic techniques. This achieves dual unpredictability in both temporal and spatial dimensions through spatial randomization (IP address selection) and temporal randomization (hopping timing), addressing the predictability issues inherent in traditional fixed graphs.

(2) Introduces a Chaos-Randomization Model (CRM) combining chaotic (Logistic) and cryptographic (RC4) random mapping. This generates global random sequences through mapping while enhancing local randomness via cryptographic algorithms,resolving the issue of local non-uniformity in chaotic sequences. Concurrently proposes IP address conversion and anti-collision mechanisms to ensure algorithmic universality.

(3) Experiments show the new method produces nearly uniform IP address distributions. It proves resistant to analysis attacks, confirming its unpredictability. Tests also confirm high time-based randomness.

2 Related Work

According to the different hopping strategies, IP hopping can be roughly divided into three types: random hopping, rule-based hopping, and intelligent adaptive hopping. This article discusses related work from these three main types.

The randomized hopping algorithm is the most basic and widely used type of algorithm in IP hopping technology, and its core idea is to generate unpredictable hopping address sequences using random number generators or pseudo-random functions. The main advantages of such algorithms are simple implementation, low overhead and the ability to provide a high hopping frequency. For exampleJafarian et al. used OpenFlow to develop an MTD architecture that transparently changes host IP addresses with high unpredictability and rate, while maintaining configuration integrity and minimizing overhead [15]. Heydari based on the shared key and timestamp, a hash chain is used to synchronously generate random IPv6 address suffixes to achieve end-to-end hopping [16]. Kong et al. used the time window adaptive adjustment (TWAA) algorithm to reduce the dependence on strict time synchronization, taking into account safety and efficiency [17]. Majid et al. tried to estimate the cost of path randomization in a message transmission system using IP hop randomization in the network, and obtained some experience with random hop cost strategies [18]. Random hopping is easy to implement and has good robustness in the face of different network attacks. However, random hopping lacks clear policy guidance, high-frequency hopping will increase network overhead, and random sequences with insufficient security are vulnerable to cracking and prediction attacks.

Rule-based hopping algorithminstead of being completely random, the hoppings are made according to a preset policy or rule, usually to achieve a specific security goal or to optimize resource utilization.

For example, Liu et al. used the Multi-Homing feature of IPv6 nodes to hop between multiple address prefixes according to policies, provide "fast switching" and "over-retention" policies to ensure communication continuity [19]. Mei et al. used "pre-announcement" and "post-clear" mechanisms to reduce the communication interruption caused by the hopping of the subnet prefix and interface identifier at the same time according to the policy [20]. Han et al. proposed a random selection of addresses in the virtual address pool by weight in the SDN-controlled Internet of Things, rather than uniform random, to distinguish device status or address history [21]. Zhang et al. proposed a multi-constraint and multi-strategy hopping model with adaptive switching of hopping paths and intervals to deal with different attack scenarios [22]. The rule-based hopping algorithm can reduce the network overhead to some extent while ensuring security. However, the effectiveness is highly dependent on the designer's pre-understanding of attack patterns, and the lack of adaptability in the face of new attacks or zero-day attacks. And the attacker can break the hopping law through long-term observation and pattern analysis.

An intelligent algorithmthe hopping strategy can be dynamically and adaptively adjusted according to network status, security threats or business requirements. For example, Gu et al. proposed a hopping strategy that provides differentiated hopping patterns and parameters in an SDN environment based on the identified types of communication services and their reliability requirements [23]. The Gao team used reinforcement learning to develop a complex deception defense mechanism called DSHopping (deception scene hopping), which intelligently converts the hopping probability by calculating the network overhead [24]. Shi et al. proposed an adaptive IP hopping method for moving target defense (MTD) using a one-dimensional convolutional neural network, which adaptively triggers the corresponding IP hopping strategy by detecting the type of attack [25]. Xu et al. proposed a lightweight convolutional neural network (CNN) detector composed of three convolution modules and a judgment module to sense scanning attacks, and the detection result is used to trigger IP hop points [26]. He et al. propose an intelligent hopping mechanism based on differential game methods, which can adaptively adjust the IP frequency hopping frequency to maximize defense benefits [27]. The intelligent algorithm can sense the type of attack and give a more efficient IP hopping strategy to maximize the hopping revenue. However, the training and inference process of deep neural networks requires considerable computing resources, which may be unaffordable on resource-constrained IoT devices, limiting their deployment range.

In summary, the current IP hopping in practical applications there is a large network and computing resource overhead, while the attacker can identify the proxy server or hopping mechanism to predict the hopping IP and other issues. In view of the above problems, this paper proposes a Dynamic random map IP hopping model based on chaotic sequence generation, and uses a logistic chaotic sequence to enhance the pseudo-random number generator (PRNG) the quality [28]. By designing the map generation algorithm and the enhanced perturbation algorithm, the high randomness and uniform distribution of the IP hopping sequence are realized, so that under the premise of small network resource overhead, make the hopping have a higher randomness, while improving the anti-analysis and anti-interference ability, effectively avoiding the attacker through continuous analysis to crack the hopping system, to avoid non-monotonic security, expands its deployment scope and application scenarios.

3 Random Map Model Based on Chaos and Encryption

This section builds a random graph model based on chaos and encryption, uses the random IP generation module as the basis for generating Dynamic random graphs, and optimizes the results through the chaotic sequence perturbation module, high entropy randomization of single-node IP addresses is realized from the spatial dimension, and a cross-period dynamic association mechanism is established from the time dimension to form a two-dimensional and space-time collaborative enhanced IP hopping defense system.

In view of the fact that the collaborative defense mechanism of the model in two dimensions of space-time is realized by specific technical paths respectively, the two sub-modules of the random IP generation module and the chaotic sequence disturbance module will be described in detail in the following, the design principles, key algorithms and synergy mechanisms of each sub-model are revealed to fully present the technical architecture and defense effectiveness improvement path of the random map model.

3.1 Random IP Generation Model Based on Chaotic Sequence

Chaos refers to the movement of uncertainty that occurs in a deterministic system, which is characterized by uncertainty, unrepeatability and unpredictability. Logistic mapping is a simple nonlinear differential equation, its characteristics reflect the chaos phenomenon, is a typical application in the chaotic system [29]. The chaotic sequence generated based on the Logistic map has the characteristics of long period, non-repetition and sensitivity to the initial value, and its most typical application is the one-dimensional chaotic map, the expression is as follows:

$$X_1 = \mu X_0(1 - X_0) \tag{1}$$

$\mu \in [3.5699, 4]$, μ and X_0 as the input parameter of Formula (1), after θ iterations, a set of chaotic sequences distributed in the range of (0,1) is generated. $\{X_i\}|_{i=1,2,\ldots,N}$, the sequence has long-term unpredictability. Similarly, after performing the above process back and forth for one round, a chaotic sequence in a two-dimensional array is obtained, denoted as $\{X_i|i = 1, 2, \ldots, I; n = 1, 2, \ldots, N\}_{I\times N}$, which is equivalent to $\{X_i\}_{I\times N}$.

In getting the randomization results $\{X_i\}_{I\times N}$ after that, it needs to be further processed, divided into subspaces and normalized to lay the foundation for subsequent IP address conversion, as shown in Fig. 1.

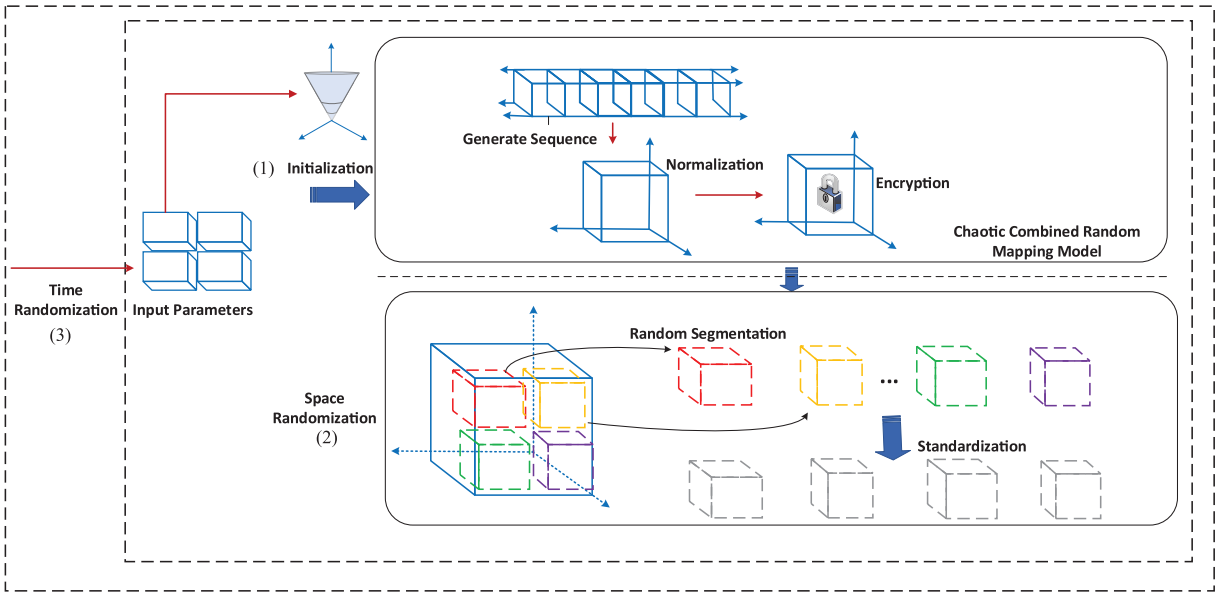

Figure 1: Schematic diagram of chaos combined with random mapping model.

The mapping process of chaotic sequence to IP address can be summarized into the following two parts:

(1) Normalization

Assumption $x_{ij} = b_n b_{n-1} \ldots b_1 b_0$, its length may exceed the IP address parameter pool V the range that can be accommodated. Therefore, these sequences need to be normalized to the IP address pool parameter pool V the range $(0, M)$ that can be accommodated. Specifically, will x_{ij} to decimal, the conversion process is as follows:

$$D = b_n \times 2^n + b_n \times 2^n + \ldots + b_1 \times 2^1 + b_0 \times 2^0 \tag{2}$$

where D is the converted decimal value. Next, normalize each value to $(0, M)$, a common method is to first D_{k} normalized to the (0,1) interval and then adjusted according to the target interval $(0, M)$. For each D_{k}, Normalize it to (0,1), the formula is as follows:

$$\mathrm{N_k} = \frac{\mathrm{D_k}}{2^{\mathrm{m}} - 1} \tag{3}$$

where $\mathrm{N_k}$ is the value of the converted $(0, 1)$ interval. Then you need $\mathrm{N_k}$ adjusted to the interval $(0, M)$, the adjustment process is shown in the formula:

$$M_{\mathrm{k}} = M \cdot \mathrm{N_k} \tag{4}$$

where M_{k} is the converted interval $(0, M)$. Two-dimensional sequence $X(i) = \{x_n(i)\}_{I \times N}$ the normalized subspace can be expressed as a formula:

$$X(i) = M_{\mathrm{k}}(i)_{I \times N} \tag{5}$$

Among them, $M_{\mathrm{k}}(i)$ for No. I the KTH normalized subspace of the period.

(2) IP address translation

IP address translation is a random mapping method that combines Knuth Shuffle (Shuffle) and subspace, aiming to realize the conversion of normalized subspace to IP address. In IP address translation, starting from the last IP address in the IP address parameter pool, it is exchanged with the previous randomly selected IP address. Then, the corresponding IP address is selected from the shuffled IP address parameter pool according to the subspace as the converted IP address. Since this process is based on the subspace generated according to the model in this paper, the probability of each IP address in the pool being selected is the same, thus ensuring the equal probability of the resulting converted IP address. Knuth Shuffle (Shuffle) generated random permutation of π satisfies:

$$\pi = \sigma_{n-1}, \cdots, \sigma_1(A) \tag{6}$$

where $\sigma_{\{k\}}$ is the random permutation of step k. The Knuth Shuffle ensures a uniform distribution of random IPs. That is, the probability that each IP is mapped to IS $\frac{1}{M}$ where M is the range of the normalized subspace of the IP address pool.

For the kth subspace $M_{\mathrm{k}}(i)$ and IP address parameter pool V for IP address parameter pool V the IP address in the is reshuffled once, according to the subspace. $M_{\mathrm{k}}(i)$ get converted IP address.

First generate a random integer ja from 0 to j, and then randomly rearrange the IP address pool V:

$$(V[j], V[ja]) \leftarrow (V[ja], V[j]) \tag{7}$$

After that complete the base mapping process:

$$h_n = V\left[M_n\right] \tag{8}$$

In the above process, although the possibility of IP address collision between subspace and Knuth Shuffle is very small, there is still the possibility of IP address collision in theory. Therefore, collision detection and processing are required in the process of IP address conversion:

$$h_n = \begin{cases} \text{Knuth Shuffle}(V) \longrightarrow h_{\mathrm{k}}(\mathrm{i}) & \text{if } h_n = h_{n-1} \\ V\left[M_n\right] & \text{otherwise} \end{cases} \tag{9}$$

where $h_{\{n-1\}}$ is the IP address of the previous moment.

From the above two processes, it has been possible to realize the function of generating random IP addresses from chaotic sequences, and make the generated random IP addresses uniform and legal, and will not be used by attackers. Known Intercept Results easily analyze other IP addresses with high unpredictability.

3.2 Dynamic Randomization Model on Perturbation Enhancement of Chaotic Sequence

The random map composed of the sequence of the above random IP and other network parameter changes has good randomness in general, but because the chaos algorithm and linear shift register have randomness in the small set range, and there may be a mapping degradation problem within an infinite set, and thus it may be difficult to meet the requirement that a random map whose length is theoretically Approximately Infinite should also be random, in order to ensure that the random IP generated by the chaotic sequence has better randomness and unpredictability in both space and time, the perturbation technique is introduced to process the chaotic sequence twice. Let the perturbation function be E_{y}, in which y is a dynamic parameter (such as the current period sequence number or timestamp), and the sequence after perturbation is:

$$X_i^{'} = E_y(X_i) \tag{10}$$

Considering that RC4 can generate random disturbance to the local part of the sequence, RC4 is chosen as the disturbance function to generate the sequence with global and local uniformity. RC4 is an efficient stream encryption algorithm, the core of which is to generate a highly random byte stream through the dynamic parameter scheduling algorithm (KSA) and the pseudo-random generation algorithm (PRGA) [30] Which can effectively destroy the short-term correlation of chaotic sequences and prevent attackers from predicting subsequent IPs through local patterns. The specific disturbance process is as follows:

(1) We initialize the RC4 state vector(S) using a seed y, which combines the current hopping period index and node identifier. This step utilizes the Key Scheduling Algorithm (KSA) to generate the initial permutation array S. The approach ensures the uniqueness of perturbation rules for each period and node. It provides a dynamic and independent foundation for subsequent keystream generation.

(2) We generate the pseudo-random byte stream using the state vector(S) produced in Step (1). We invoke the Pseudo-Random Generation Algorithm (PRGA) of RC4 to produce a keystream $rc4stream$ of the same length as the chaotic sequence(X_i). This keystream inherits the uniqueness of the dynamic seed. It directly supports the subsequent bitwise perturbation, ensuring that the perturbation pattern varies across periods and nodes.

$$rc4stream = r_1, r_2, \ldots, r_m \tag{11}$$

(3) Sequence perturbation: the chaotic sequence and the pseudo-random flow are bit-by-bit different:

$$X_i^{'} = X(i) \oplus rc4stream \tag{12}$$

where $\oplus$ denotes a bitwise XOR operation, ensuring that the perturbed sequence $X_i^{'}$ local randomness is significantly enhanced.

Chaotic sequence after perturbation $X_i^{'}$ local randomness is enhanced, which destroys the short-term correlation of chaotic sequences and prevents attackers from predicting subsequent IP addresses through local patterns. The time dimension is dynamic, and the parameter k changes with the period, ensuring that the IP generation rules for different periods are completely independent. The information entropy of RC4 Pseudo-random flow increases, which significantly improves the unpredictability of the sequence and improves the difficulty of the attacker to analyze the graph.

4 Chaos Combined with Random Map Model Random Map Generation Algorithm

Given according to the random map model (CRM) above $X(i) = \{x_n(i)\}_{I\times N}$ and the parameter pool V this section presents a random map generation algorithm based on chaos combined with random mapping model, which generates maps by transforming IP addresses in both space and time dimensions $H = \{h_{ij}\}_{I\times N}$. The algorithm strictly follows the randomness enhancement logic of the model design, and organically integrates the chaotic sequence generation, RC4 disturbance processing and IP address mapping process to ensure that the algorithm output meets the high unpredictability requirements of the model.

4.1 Algorithm Overview

The flow of random map generation algorithm based on chaos combined with random mapping model is shown in Fig. 2.

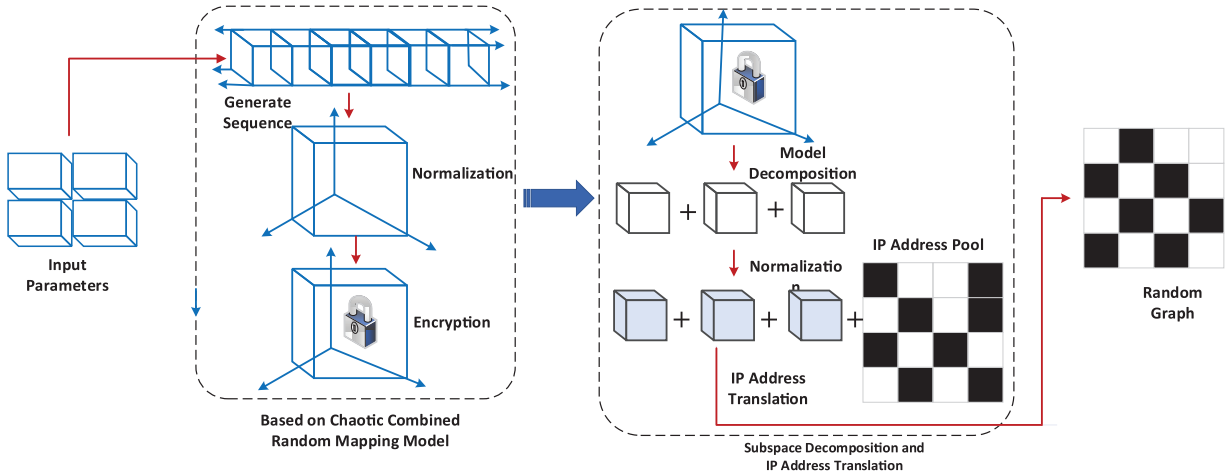

Figure 2: Flow chart of random map generation algorithm based on chaos combined with random mapping model.

As illustrated in Fig. 2, the chaos-random mapping model actively drives the random graph generation algorithm through three core steps: chaotic entropy generation with perturbation enhancement, subspace decomposition coupled with IP address mapping and collision resolution, and construction of the two-dimensional random graph.

First, the algorithm utilizes the Logistic map to generate the initial chaotic sequence and then applies RC4 perturbation to enhance local randomness. This dual mechanism effectively overcomes the potential local non-uniformity and degradation risks inherent in standalone chaotic systems, producing a hybrid entropy source that exhibits both global and local randomness.

Next, the algorithm normalizes the perturbed sequence, divides it into subspaces, and combines Knuth Shuffle to perform a uniform random permutation of the IP address pool. This ensures unbiased mapping from each subspace index to any IP in the pool. Simultaneously, an active collision detection and resolution mechanism enforces IP uniqueness: different nodes within the same period receive distinct addresses, and the same node across consecutive periods obtains different addresses.

Finally, the algorithm extends these steps to the spatiotemporal domain and constructs a random graph, where represents the number of hopping periods, denotes the number of nodes, and specifies the IP address assigned to the node in the ith period. The resulting framework achieves dual spatiotemporal independence: IP addresses remain uncorrelated among nodes in the same period and uncorrelated for the same node across periods, thereby forming a highly random and analysis-resistant IP hopping defense graph.

4.2 Random Mapping Algorithm Based on Logistic Chaos and RC4 Perturbation

Inheriting the Section 3.1 of the chaotic sequence generation mechanism, the use of Logistic mapping to generate the initial chaotic sequence, the specific generation process as shown in the Algorithm 1.

Algorithm 1: Logistic chaos generation algorithm

Input: Chaos parameter μ, initial value x_0, the number of iterations iter
Output: Chaotic sequence value x
1: $x = x_0$
2: for t= 1 to iter do
3: $x = \mu \times x \times (1 - x)$
4: end
5: return x
Return: ReturnChaos Sequence$x$

In Algorithm 1, by inputting the initial value x_0 and the number of iterations $iter$, the initialization of the Logistic chaotic sequence can be controlled. The Lyapunov exponent analysis indicates that these parameters ensure the most significant chaotic characteristics, making the randomness of the Logistic chaotic sequence the strongest.

After obtaining the initial randomization results, further enhancement is required through disturbance processing using the RC4 disturbance algorithm, which includes the Key Scheduling Algorithm (KSA) and the Pseudo-Random Generation Algorithm (PRGA). First, a dynamic parameter y_n (such as a timestamp or period number) is used as a seed to generate a 256-byte state vector through the KSA algorithm. This vector contains real-time information about the seed, ensuring that the perturbation rules for different periods are completely independent. The PRGA algorithm then generates a pseudo-random stream of the same length as the chaotic sequence. After performing a bit-by-bit XOR operation, the perturbation sequence is obtained, and the local randomness is significantly enhanced. This process is illustrated in Algorithm 2.

Algorithm 2: RC4 Enhanced Perturbation Algorithm

Input: Chaotic sequence value x, dynamic parameter y_n
Output: perturbed sequence value x'
1: S = KSA(y_n)
2: z =PRGA(S, 1)
3: $x' = x$ XOR z//bitwise XOR perturbation
4: return x'
Return: ReturnChaos Sequence x'

The random sequence with local and global randomness is obtained, which prevents the chaotic degradation of the Logistic chaotic sequence and further enhances the randomness of the chaotic sequence.

4.3 Knuth Shuffle and IP Address Anti-Collision Algorithm

In getting the randomization results $\{x_n(i)\}_{I\times N}$ after that, it needs to be further processed to ensure that the probability of each IP address being selected is strictly equal, eliminating the possible deviation of the traditional pseudo-random generation algorithm. It is divided into subspaces and normalized to lay the foundation for subsequent IP address conversion.

In The Algorithm 3, first enter the k-th subspace $M_{\mathrm{k}}(i)$ and IP address parameter pool V, and then through the first 6 steps of the IP address parameter pool V. The IP address in the is reshuffled once. Then, in step 7, according to the subspace $M_{\mathrm{k}}(i)$ get converted IP address.

Algorithm 3: Knuth Shuffle (Shuffle) Algorithm

Input: $M_{\mathrm{k}}(i)$, V
Output: Converted IP address $h_{\mathrm{k}}(i)$
1: for j = M − 1 to 1 do
2: ja = andom integer(0, j);
3: temp = V[j];//IP address exchange
4: $V[j] = V[ja]$;
5: $V[M_{\mathrm{k}}(i)]$ = temp;
6: end for
7: $h_{\mathrm{k}}(i) = V[M_{\mathrm{k}}(i)]$;
Return: Return $h_{\mathrm{k}}(i)$

In the process of IP address translation, there is theoretically the possibility of IP address collision. Therefore, this paper proposes an IP address anti-collision algorithm, if $h_k(i) = h_{k-1}(i)$ the IP address is changed. This mechanism ensures that the IP addresses of the same node must be different in successive periods, further enhancing the unpredictability of the time dimension. The detailed description is shown in Algorithm 4.

Algorithm 4: IP collision avoidance algorithm

Input: $h_{k-1}(i), M_k(i), V$,
Output: Converted IP address $h_k(i)$
1: if $h_{k-1}(i) == V[M_k(i)]$;
2: $V[M_k(i)] \leftarrow$ Knuth Shuffle $(M_k(i), V)$;
3: else
4: $h_k(i) = V[M_k(i)]$;
Return: Return $h_k(i)$

In the Algorithm 4, the input includes the IP address of the k − 1 $h_{k-1}(i)$ and the k-th subspace $M_k(i)$. When starting to execute the algorithm, first, proceed to step 1. If the condition is true, it is directly determined Knuth Shuffle provide $V[M_k(i)]$ the return value. Otherwise, the algorithm will skip the intermediate step, execute step 4 directly, and finally output the return value $h_k(i)$.

4.4 Random Map Generation Algorithm

In the generation of random maps, the models and methods described in the previous two sections are combined to form a random map generation algorithm based on chaos combined with random mapping model, as shown in Algorithm 5. This algorithm describes the process of random map generation.

Algorithm 5: Random map generation algorithm based on chaos combined with random mapping model CandEM

Input: Result of model generation $X(i) = \{M_k(i)\}_{I \times N}$
IP address parameter pool $V = \{v_1, v_2, \cdots, v_M\}$;
Output: Graph $H = \{h_k(i)\}_{I \times N}$
1: Initialization:
2: Setting Chaos Parameters $\mu \in [3.5699, 4]$, initial value x_0
3: Parameters of time randomization required to generate RC4 perturbations a, b, $x_n(i)$ and $y_n(i)$, submitted to the computing network device;
4: for $i = 0$ to $I - 1$ do
5: for k = 0 to $N - 1$ do
6: $X(i,k) = Logistic(\mu, x_0, i * N + k)$;//Generate chaotic sequence
7: $X'(i,k) = RC4(X(i,k), y_n)$;//RC4 perturbation
8: $M_k(i) =$ Normalize (X′ (i, k) , V) ;//Normalized
9: $V[M_k(i)] \leftarrow$ Knuth Shuffle $(M_k(i), V)$;
10: $h_k(i) \leftarrow$ IP Collision Avoidance $(M_k(i), h_{k-1}(i))$;
11: $H \leftarrow h_k(i)$;
12: end for
13: end for
Return: Return Graph $H = \{h_k(i)\}_{I \times N}$

In the Algorithm 5, the normalized subspace of the RCM model is obtained by the first 8 steps. $X(i) = \{M_k(i)\}_{I \times N}$. Among them, in steps 5 to 9, step 5 pairs $\{1, 2, \cdots, N\}$ to traverse, and then pass Knuth Shuffle return the converted IP address for each traversal, then perform IP anti-collision on the converted IP address in step 7 to obtain the converted IP address after collision check $h_k(i)$, followed by Step 8 to map H complete the assignment. In steps 4 to 10, repeat to complete N steps 5 to 9. Finally return to the map H.

4.5 Performance Analysis of Algorithm

This section will analyze the two core dimensions of time complexity and security of the algorithm, aiming to systematically verify the feasibility and universality of the algorithm.

4.5.1 Time Complexity

It is assumed that there are i hopping period, n nodes and m alternative IP addresses. In each hopping period, the IP addresses of these nodes are randomly replaced with IP addresses, and the newly selected IP addresses are changed from this Mselect an IP address. Where, through Step 3, the normalized subspace at each period and each node can be calculated. $\{M_{\mathrm{k}}(i)\}_{I\times N}$, the time complexity of this calculation process is $O(in)$. Reuse the converted IP address obtained by normalizing the subspace in step 6. $h_{\mathrm{k}}(i)$, the time complexity of this calculation process is $O(m)$. Next, the IP anti-collision check is performed in step 7 and the checked IP address is obtained. The time complexity of this calculation process is $O(n)$. Combined with the above analysis, the total time complexity of this algorithm is the sum of the calculation stages, that is, the time complexity of the Algorithm 3.3 is $O(in + m + n)$.

4.5.2 Safety Analysis

Proposed under Section 4, this section performs a security analysis of the random graph generated by the algorithm to evaluate its ability to resist analytical attacks.

Assuming there are k IP addresses used for hopping, and the hopping period is T(seconds). The time required for an attacker to attack an IP address is T_a. In the period T, when the attacker finishes in the T_a, the IP address obtained by the second attack is the same as k of a successful attack, which occurs when an IP address exactly matches. The following calculates the completion of the attacker T_a/T, the secondary attack k IP address, and the attack retention rate.

When the attack time T_a is less than the hopping period T, the attacker has enough time in the hopping period. part of the IP address of the internal attack. During the hopping period T, the probability of a secondary IP address can be calculated by considering the following factors. Consider the attacker's strategy; the attacker uses a uniform random strategy. IP selects the attack target in the address pool. During the hopping period, the probability of success for a single attack is the p. The probability of success of the secondary IP address is $p^{\frac{T_a}{T}k}$, considering the attacker's strategy, false, if the attacker uses a uniform random strategy. Secondly, considering the dynamic nature of the IP address pool, the algorithm based on the IP address pool generates a random map to achieve the unpredictability of the random map, and builds a model to capture the characteristics of the IP address pool over time. Model definition: First, time is divided into a series of discrete moments $(t_1, t_2, \ldots, t_\varphi)$.

Each moment represents the point at which the IP address pool may change. Second, define a state variable (t) to indicate at time t, the number of IP addresses available for attack at any time. Finally, the change of IP pool is described as a random process, such as a Poisson process, assuming that in each hopping period T, new IP addresses are added to the IP address pool and existing IP addresses are removed from it. From the mathematical description, in the hopping period, the number of new IPs can be simulated by

Poisson distribution, i.e., $P_{add}(k) = \frac{e^{-\lambda T(\lambda T)^k}}{k!}$, in which representation k similarly, the number of IP addresses leaving can also be described by Poisson distribution, i.e., $P_{remove}(k) = \frac{e^{-\varsigma T(\varsigma T)^k}}{k!}$. Therefore, in each hopping period T, the overall change in the IP address pool can be expressed as the difference between the new IP address and the leaving IP address. Set $\vartheta(t + T)$ is the number of IP addresses at the moment, then there is the number of IP addresses $\vartheta(t + T) = \vartheta(t) + N_{add} - N_{remove}$, in which N_{add} and N_{remove} represents

the number of IP addresses actually added and left respectively, which is determined by the above-mentioned Poisson distribution.

The above model takes into account the randomness of new IP addresses and IP address departures, and then introduces the k IP address parameter pool. V is to emphasize addressing the range of variation, i.e., The number of addresses available for attack at any given time may be different. Hopping period T divided T the length of each time period is $\Delta t = \frac{T}{\tau}$.

Attackers need to attack T_a/T time to the IP address. Inside T_a, the attackers have k the second chance to attack different IP addresses. Therefore, the attacker needs $\frac{T_a}{T} \cdot k$ one T_a complete the attack.

The attacker is T_a within a successful attack IP the probability of the address is p. Therefore, the attackers inner T Attack $\frac{T_a}{T}$ times, success k one IP address the attack retention rate can be expressed as Formula (13).

$$P_{at} = \binom{\tau}{\frac{T_a}{T}k} p^{\frac{T_a}{T}} (1-p)^{\tau - \frac{T_a}{T}k} \tag{13}$$

Among them, the left part of the formula $\binom{\tau}{\frac{T_a}{T}k}$ is a combination, which means from T selected in a time period $\frac{T_a}{T} \cdot k$, the number of combinations of time periods, $p^{\frac{T_a}{T} \cdot k}$ indicates that in the selected $\frac{T_a}{T} \cdot k$ successful attacks within a time period $\frac{T_a}{T} \cdot k$, the probability of the second IP address, the left part of the formula $(1-p)^{\tau - \frac{T_a}{T}k}$ for the remaining $\tau - \frac{T_a}{T} \cdot k$. A small period of time did not successfully attack any IP the probability of the address.

From the above analysis, it can be seen that in the hopping period T, the attacker's k IP address initiation $(T_a/T) \cdot k$, the success rate of the attack is T_a, T, k, and T influence of factors. As can be seen from Eq. (12), the smaller the hopping period, the lower the success rate of the attack. The time spent on attacking an IP address is less; When for hopping the number of IP addresses, the larger the attack, the lower the success rate.

Taking into account the hopping period T, single attack time T_a number of IP addresses k and T. The total number of successful attacks during the attack period, as well as other factors, these are then analyzed using information entropy. Anti-interception analysis capability of random maps.

Assume that the address hopping device is provided. In given period T, the attackers attacked $(T_a/T) \cdot k$ secondary IP address. The probability of each IP address being selected is $p_1, p_2, \ldots, p_k$, assuming the IP addresses are independent and evenly distributed. The attacker independently selects the IP address for each attack, and the probability of each IP address being selected is equal; that is $p_1 = p_2 = \ldots = p_k$, the attacker cannot rely on the previous interception results to infer the value of the next IP address.

To quantitatively analyze the anti-interception ability of random maps, we introduce information entropy. This section uses information entropy to evaluate the resistance of IP addresses against interception and analysis. The information entropy $H(X)$ of a random variable is calculated using Eq. (14).

$$H(X) = \sum_{i=0}^{n} P(x_i) \cdot \log_b P(x_i) \tag{14}$$

where $P(x_i)$ is the probability of the random variable X taking the value x_i. For a uniformly distributed random variable, each value has an equal probability, i.e., $P(x_i) = \frac{1}{\mathrm{n}}$, where n is the number of possible IP addresses an attacker may encounter.

Assume that an addr-ss hopping device provides one IP address from a pool of n possible addresses. The probability of an attacker targeting any specific IP address is $\frac{1}{n}$. Substituting this into the information entropy formula, the entropy of the IP address distribution is:

$$H(X) = \log_2 n \tag{15}$$

Eq. (15) indicates that, from the perspective of information entropy, the difficulty for an attacker to infer the next IP address depends on three factors:

(1) The hopping period T (shorter periods increase entropy),

(2) The time spent attacking a single IP address T_a (less time increases entropy),

(3) The total number of IP addresses nnn (a larger pool increases entropy).

When T is smaller, T_a is shorter, or nnn is larger, the information entropy of the IP address space increases. This means it becomes harder for an attacker to rely on intercepted data to predict future IP addresses. Additionally, assuming each IP address is independently and uniformly distributed, even if an attacker obtains a subset of IP addresses, they cannot accurately infer the remaining unknown addresses.

Suppose an attacker intercepts a certain number $\frac{T_a}{T}$ IP address, this paper random map generation method randomly generated. k IP has the ability to resist interception, making attackers cannot rely on known interception results to infer the value of other IP addresses.

Proof: Assume that the set of IP addresses intercepted by the attacker is A, map generation method randomly generated IP the address set is X. Definition $H(A)$ is the information entropy of the IP address intercepted by the attacker, $H(X)$ is the information entropy of randomly generated IP addresses. Attack the attacker cannot rely on the known interception results to infer the value of other IP addresses, can be expressed, in given collection of IP addresses that have been intercepted by the attacker A in the case, the map generation method randomly generates a collection of IP addresses. X the information entropy under the condition of should not be equal to the information entropy under the unknown condition $H(X)$. If $H(X|A)$ and $H(X)$ the difference between) is large, indicating that the attacker cannot rely on the known interception results to infer the value of other IP addresses. Therefore, the goal of this paper is to prove $H(X|A) \neq H(X)$.

Introducing the conditional entropy pair $H(X|A)$ and $H(X)$ are analyzed, and the conditional entropy is defined as (16).

$$H(X) = -\sum_{x \in X} P(x) \log P(x) \tag{16}$$

$$H(X|A) = -\sum_{x \in X} P(x|A) \log P(x|A) \tag{17}$$

Next to prove $H(X|A) \neq H(X)$, in order to prove this point, use the basic properties in information theory to prove this point.

Assumption $H(X|A) = H(X)$, that is, assuming a collection of IP addresses intercepted by a known attacker A the information entropy of the randomly generated IP address is equal to the information entropy of the unknown condition. But, by the definition of conditional entropy, $H(X|A) = H(X, A) - H(A)$, in which, $H(X, A)$ is X and A the joint entropy, $H(A)$ is A of entropy.

According to the definition of entropy: $H(X) = H(X, A) - H(A|X)$

where is a collection of IP addresses that are randomly generated in a known X case, the attacker intercepts a collection of IP addresses A the conditional entropy.

Assumption $H(A|X) = H(X)$ is established, then: $H(X, A) - H(A) = H(X, A) - H(A|X)$, Formula (18) is expressed:

$$H(A) = H(A|X) \tag{18}$$

But this contradicts the nature in information theory, i.e., $H(A|X) \leq H(A)$. So assume $H(X|A) = H(X)$ is not established.

From the above analysis, it can be concluded that, suppose an attacker intercepts a certain number of IP addresses, it is also impossible to rely on known interception results to infer other IP addresses. This shows randomized algorithm in this paper randomly generated atlas with randomness and independence, they are not affected by the attacker's known interception results, andhave the ability to resist analysis attacks.

5 Experimental Results and Analysis

5.1 Experiment Setup

In the simulation experiment, this paper uses Minine. Built a cloud network trusted environment that includes multiple OpenFlowswap virtual machines and instances, and deploy the developed Atlas randomization algorithm in Ryuon. The server's configuration information used is Intel (R) Xeon (R) e-2124, with a frequency is 3.31 GHz, 16 GB of memory, and Windows 10 as the operation system. The configuration information oincludes the CPU model as Intel i3-12400F and the memory capacity as 8 GB × 2.

The data in the experiment mainly includes IP address database and data set, where the IP address database is qqzeng-ip database, the data set is NSL-KDD data set, DDoSattack data set.

In the evaluation of spatial randomization and time randomization, the choice of Leap-forward type linear feedback register (LFSR), pseudo-random number generator (PRNG), Chaotic Sequence (CS), CS algorithm is compared with the randomization algorithm proposed in this paper.

5.2 Evaluation Index

This section uses three safety indicators to evaluate the effectiveness of the proposed method. These three indicators include: attack cost (AC), Attack Return (AR) and system availability (A). These indicators are described below:

(1) Cost of attack (AC): Used to measure the difficulty of the attacker attacking the system and to quantify the cost of the attacker attacking the entire system.

$$AC = \sum AC_{IP_i}, IP_i \in \text{ap}$$

where AC_{IP_i} cost for attacks using IP addresses [176], apan attack path from the attacker to the target.

(2) Attack returns (AR): Refers to the willingness of the attacker to choose the attack path by quantifying the cost and benefits of the attack.

$$AR = \sum AR_p, p \in \text{AP}$$

where AR_p is the sum of the attack returns for each IP address in a single path, AP for all possible attack paths of the attacker to the target, p the probability of a successful attack on an IP address.

(3) System availability (A): is used to quantify the availability of the system in the event of an attack.

$$A = 1 - \sum_{i=1}^{n} R_i \times I_i$$

where R_i for No. i. The risk of an IP address being attacked, I_i for a successful attack on the No. i. The impact of IP addresses on the system.

5.3 Spatial Randomization Experiment

In the spatial randomization evaluation, the choice Kolmogorov-Smirow (K-S) the test Index evaluates the similarity of the random map generated by the map generation algorithm.

The K-S test is a non-parametric probability statistical test that tests whether the test sample and the reference sample come from the same probability distribution. The main parameters include K-S statistics, significance level, α, h-value, and p-value, where the K-S statistic is used to describe the difference in the probability distribution of the two samples, the h value is a logical value (0 or 1), and under the Zero hypothesis that the test and reference samples come from a uniform distribution, A value of 1 indicates rejection of the Zero hypothesis, a value of 0 indicates that the zero hypothesis cannot be rejected, and a p-value is a probability value that indicates the probability of observing the value of the two sample difference statistics if the zero hypothesis is true. In the similarity measure of random maps, K-S statistics are used to test the difference between the probability distribution of randomly generated maps and the uniform distribution of the null hypothesis. For convenience, in the K-S test, the cumulative distribution function (CDF) in the statistical process is set $cdf = struct('type', 'normal', 'params', \{0,1\})('\text{normal}'$ is uniformly distributed), the level of significance α set to 0.05. And the critical value of the output is $\frac{1.36}{\sqrt{n}}$ (n to verify the number of IP addresses). Among them $\frac{1.36}{\sqrt{n}}$ is based on the significance level in the critical value reference table. ($\alpha = 0.05$). And the number of IP addresses tested ($n \gg 40$) obtained. Fig. 3 shows the results of the K-S test.

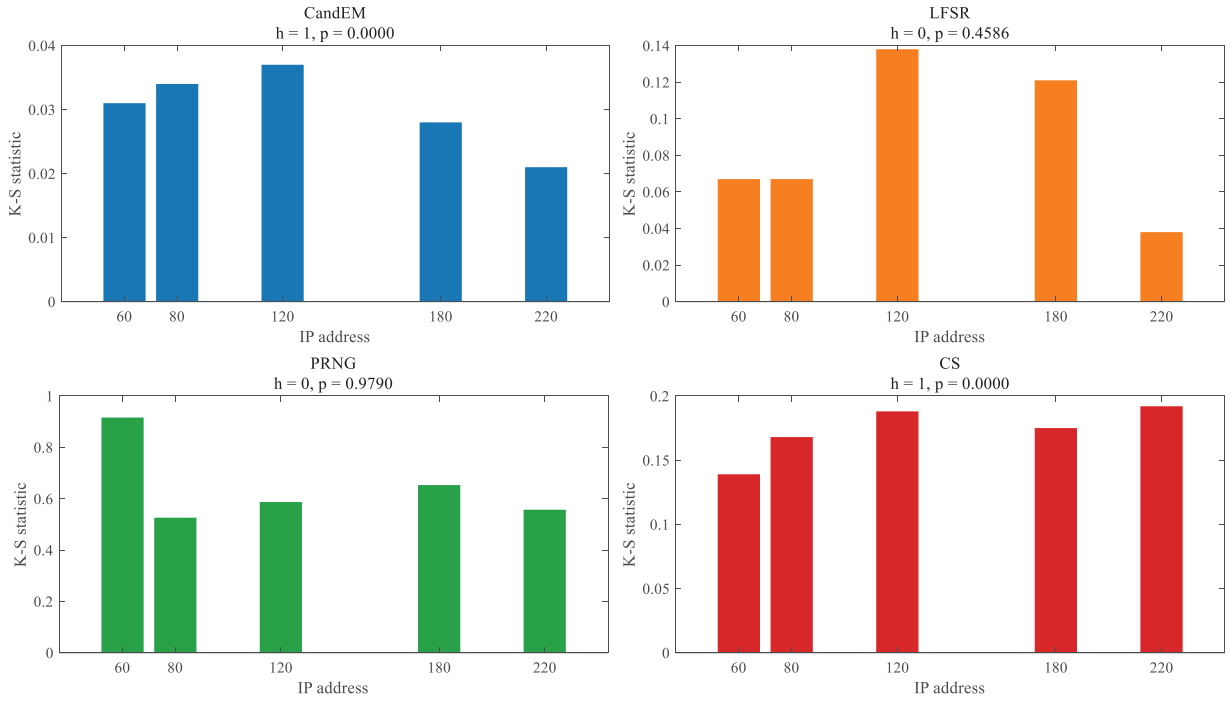

Figure 3: K-S comparison results.

As can be seen, when the graph 3 the h-value calculated by the four algorithms based on K-S statistics is 0, i.e., after supporting the Zero hypothesis, as the number of IP addresses in the test sample increases gradually from 60, 80, 120, 180 to 220, the value of the K-S statistic fluctuates slightly, CandEM the K-S statistics of the algorithm are smaller overall. So from a statistical point of view, the Zero hypothesis that these algorithms follow a uniform distribution based on the results of the h-values generated by the K-S statistics is valid. In other words, this indicates that the distribution of IP addresses generated by these algorithms is statistically the same as the uniform distribution. However, there are differences in the *p*-values calculated by these algorithms based on K-S statistics. Among them, CandEM the *p*-value obtained by the algorithm is the smallest and less than the significance level ($\alpha = 0.05$). The *p*-values obtained by the CS, LSFS, and PRNG algorithms are 0.6406, 0.5288, 0.9519, greater than the significance level. This is because the CS algorithm generates locally uneven subspaces through chaotic direct mapping without considering encryption, thereby generating uneven IP addresses. At the same time, due to the inherent periodicity of linear shift registers and pseudo-random sequences, the LSFS and PRNG algorithms will have repetitions based on the subspaces they generate, thus generating repeated IP addresses. Thus, compared to these algorithms, CandEM the algorithm is able to generate more uniform IP addresses.

5.4 Time Randomized Experiment

The time randomization test is a non-parametric probability statistical test method used to analyze whether the probability distribution of sample occurrence in a time series is uniform. In the time randomization test, multiple sets of experiments were conducted on the similarity of IP addresses periodically generated by the Atlas generation algorithm through the K-S test, and the results are shown in Fig. 4.

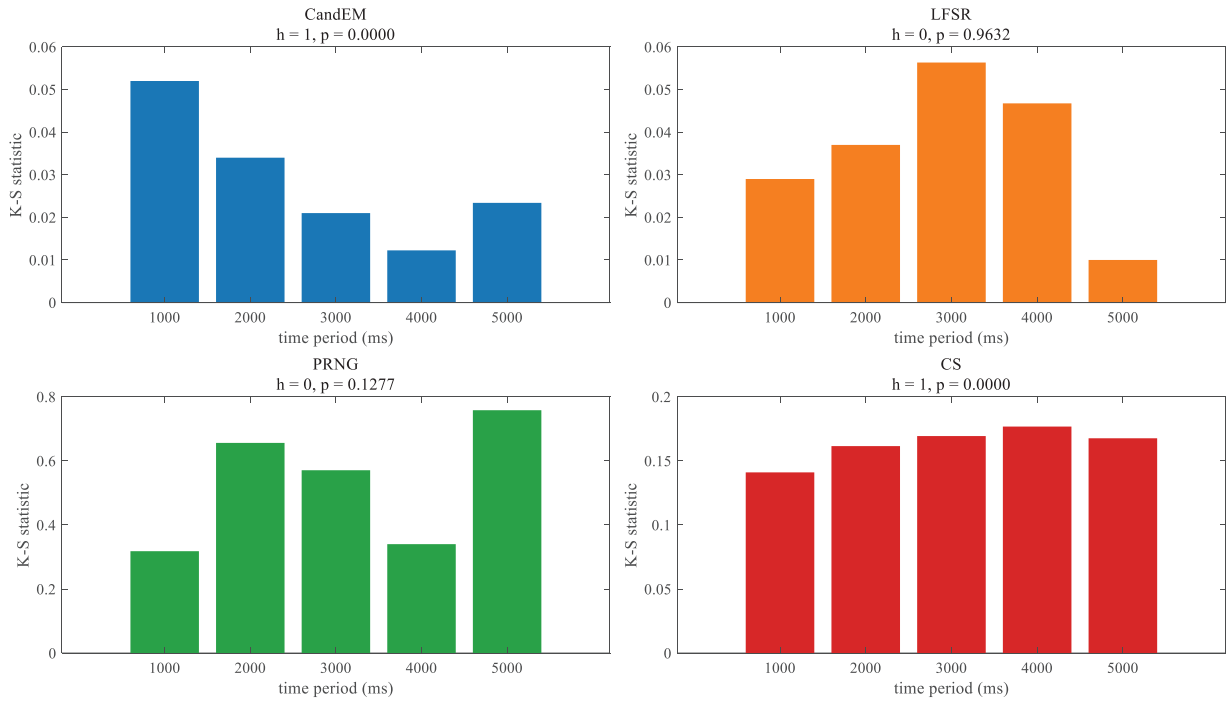

Figure 4: K-S statistics results.

Fig. 4 shows how several algorithms obtain the statistics of the K-S test. It can be seen that the h values in the upper left, lower left, and upper right graphs are all 0, indicating that the probability statistical distribution

of IP addresses generated by these algorithms is consistent with the null hypothesis of uniform distribution. In the following figure, h is equal to 1, indicating that the statistical distribution of the CS algorithm for IP addresses is not the same as the statistical distribution of the reference sample. In addition, it can be seen that in the K-S test, with the gradual increase of the hopping period from 1000, 2000, 3000, 4000 to 5000 MS, the K-S statistics in the upper left and lower left graphs show a downward trend. Since the magnitude of the K-S statistic in the upper left graph is lower than the magnitude of the K-S statistic in the lower right graph, CandEM the algorithm is better than the PRNG algorithm in testing the effect of K-S statistics. This means that the statistical distribution of the test sample is closer to the statistical distribution of the reference sample. In the lower right panel, the K-S statistic value remains essentially the same as the hop period increases, but is below the critical value (0.05). In the upper right panel, the value of the K-S statistic is higher than the critical value, but the *p*-value is higher than the significance level ($\alpha = 0.05$). This is because the periodic characteristics of the sequence of linear feedback shift registers and chaotic systems have an impact on the statistical distribution of IP addresses of the LFSR algorithm and the CS algorithm.

5.5 Anti-Analytical Attack Experiment

This section analyzes the security of the CRM model by four indicators:*R*, *AC*, *R*, and *AR*. Assume that the attacker has gained root privileges, leaving the system with a system vulnerable to a system-level exploit. Setup the benchmark defense rate to be 0.75, the attack rate on CRM obeys the exponential distribution, and its parameter value is set to 0.3 (attack every 4 h). According to the currently defined security indicators, these security indicators are calculated through different scales of 50 to 150 IP addresses. Fig. 5 shows the changes in availability metrics corresponding to different numbers of IP addresses (50, 150, and 200 IP addresses) over time. As show in the Fig. 5, the availability index value of 50 IP addresses dropped the fastest. The availability value of 150 IP addresses decreases more slowly than that of 50 IP addresses. Availability values for 200 IP addresses dropped the slowest. This indicates that increasing the number of IP addresses has improved the system's security. However, increasing the number of IP addresses will also lead to a decline in system availability.

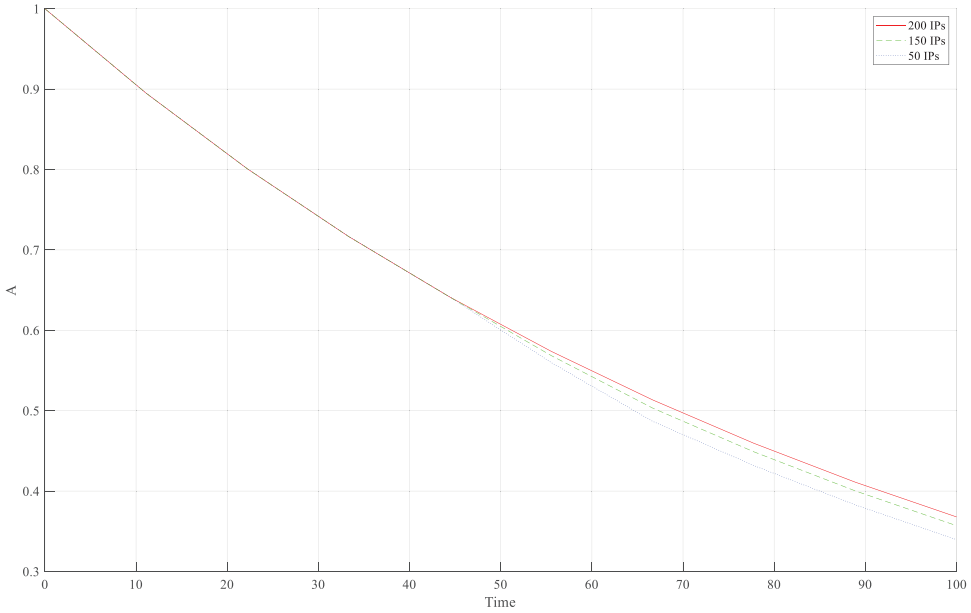

Figure 5: Availability indicators for random maps.

Fig. 6 shows the attack costs of the three metrics R, AC, and AR at different IP address sizes. As can be seen from the figure, as the number of IP addresses increases, the R index, AC index, and AR index all increase at the same time. This shows that as the number of IP addresses increases, the cost of the R, AC, and AR metrics increases accordingly.

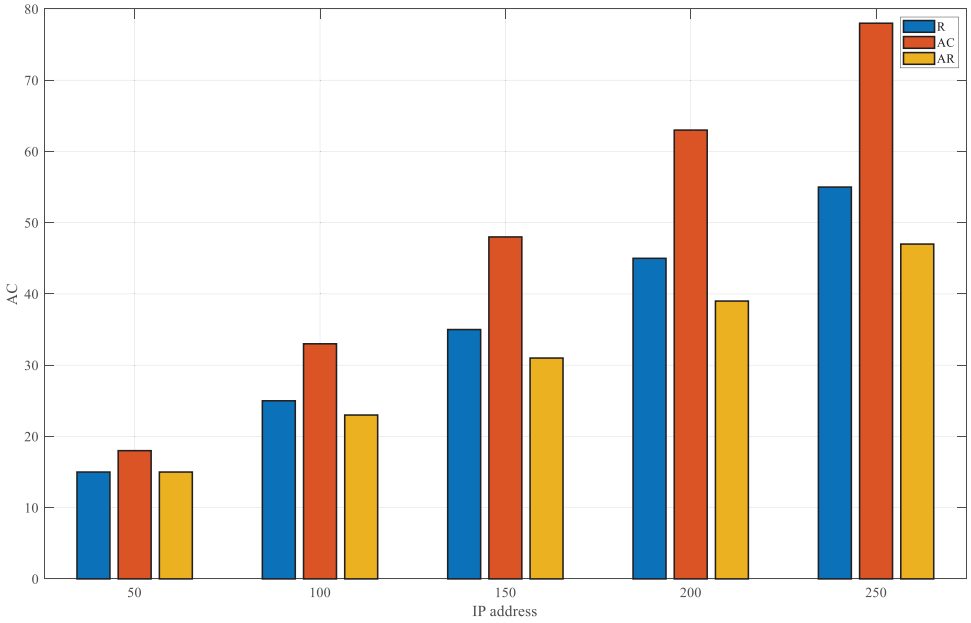

Figure 6: Attack costs of IP addresses.

Fig. 7 compares the defense success rate ($p_D(h)$) of random and fixed IP address strategies at different attack times. We plot curves for different numbers of IP addresses in the pool (n = 100, n = 150) alongside the benchmark defense rate of 0.7. You can see that the defense success rate of the fixed IP address policy is lower than that of the random IP address policy. This is because, in the initial stage of the attack, the defense success rate of the fixed IP address strategy is equal to that of the random IP address strategy, but as time goes by, the attacker may recognize the IP address transformation rule of the fixed IP address strategy, this has led to a gradual decline in the success rate of its defense. In contrast, the defense success rate of the random IP address strategy decreases with the extension of the attack time. However, it still maintains a high defense success rate, which is higher than the benchmark defense rate 0.7. The reason is that its IP address transformation law is random and has the ability to resist analysis attacks.

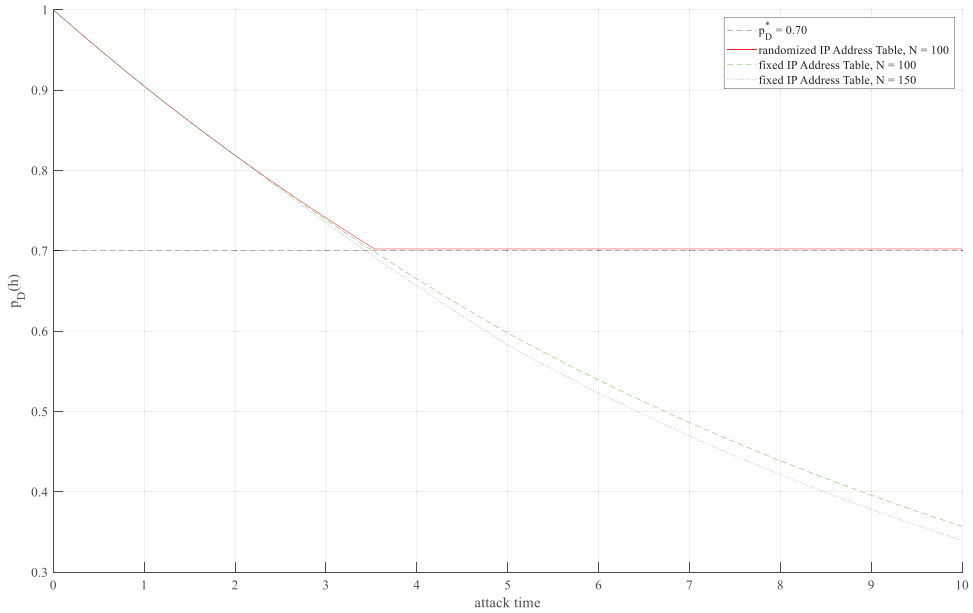

Figure 7: Defense success rate of random and fixed IP strategies over Attack Time.

6 Conclusion

For the problem that attackers can easily obtain the transformed IP address by analyzing and predicting the rules of IP address transformation in static IP hopping defense, this paper proposes a network hopping active defense method based on dynamic random graph. Firstly, a random graph model is designed to generate the normalized subspace, which lays the foundation for the spatial and temporal randomization of IP addresses. Secondly, based on subspace, a map randomization algorithm is used to achieve spatial and temporal randomization of IP addresses. In addition, the algorithm introduces a method to further disturb the IP address, and design an IP local anti-collision algorithm to prevent IP address collision. The experimental results show that this method is superior to several other test methods. By testing the spatial randomization and time randomization of the CRM model, the results indicate that the CRM model has a positive impact on enhancing the local randomness. The test results of the ablation experiment further demonstrate that the CRM model can enhance the local randomness while preserving global randomness.

Acknowledgement: We would like extend our appreciation to the colleagues and students from the School of Electronic Information at Wuhan University, the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, and the Key Laboratory of Cyberspace Situation Awareness of Henan Province for their valuable discussions and technical support during the course of this research. Special thanks are due to the experimental platform provided by Mininet, which facilitated the simulation and validation of our proposed model.

Funding Statement: This work was supported by the National Natural Science Foundation of China. Funding number: 41971407. The funding bodies had no role in the design of the study, data collection, analysis, interpretation of results, or the writing of the manuscript.

Author Contributions: Zhu Fang: Conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing—original draft. Zhengquan Xu: Supervision, conceptualization, methodology, resources, writing—review & editing, project administration, funding acquisition. Weizhen He: Software, validation, investigation, writing—review & editing. Bohao Xu: Methodology, software, validation, writing—review & editing. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The datasets and materials generated during the current study are available from the corresponding author on reasonable request. The source code for the dynamic random graph-based IP hopping algorithm and the Mininet simulation environment can be accessed at http://192.168.62.56/data2/code. Any additional data or materials required to replicate the findings of this study are also available upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liang J, Kim Y. Evolution of firewalls: toward securer network using next generation firewall. In: Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC); 2022 Jan 26–29; Las Vegas, NV, USA. p. 752–9. doi:10.1109/ccwc54503.2022.9720435.
2. Wang Q, Chen J, Jiang Z, Guo R, Liu X, Zhang C, et al. Break the wall from bottom: automated discovery of protocol-level evasion vulnerabilities in web application firewalls. In: Proceedings of the 2024 IEEE Symposium on Security and Privacy (SP); 2024 May 19–23; San Francisco, CA, USA. p. 185–202. doi:10.1109/sp54263.2024.00129.
3. Sudhakar, Kumar S. An emerging threat Fileless malware: a survey and research challenges. Cybersecurity. 2020;3(1):1. doi:10.1186/s42400-019-0043-x.
4. Sen S, Aydogan E, Aysan AI. Coevolution of mobile malware and anti-malware. IEEE Trans Inform Forensic Secur. 2018;13(10):2563–74. doi:10.1109/tifs.2018.2824250.
5. Carvalho M, Ford R. Moving-target defenses for computer networks. IEEE Secur Privacy. 2014;12(2):73–6. doi:10.1109/msp.2014.30.
6. Xu J, Guo P, Zhao M, Erbacher RF, Zhu M, Liu P. Comparing different moving target defense techqiques. In: Proceedings of the 1st ACM Workshop on Moving Target Defense; 2014 Nov 7; Scottsdale, AZ, USA. New York, NY, USA: ACM; 2014. p. 97–107.
7. Cai G, Wang B, Wang T, Luo Y, Cui X. Research and development of moving target defense technology. J Comput Res Dev. 2016;53(5):968–87.
8. Moghaddam T, Yang G, Thapa C, Camtepe S, Kim DD. MTD in plain sight: hiding network behavior in moving target defenses. In: Proceedings of the 2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks—Supplemental Volume (DSN-S); 2024 Jun 24–27; Brisbane, Australia. p. 50–2. doi:10.1109/dsn-s60304.2024.00022.
9. Steinberger J, Kuhnert B, Dietz C, Ball L, Sperotto A, Baier H, et al. DDoS defense using MTD and SDN. In: Proceedings of the NOMS 2018—2018 IEEE/IFIP Network Operations and Management Symposium; 2018 Apr 23–27; Taipei, China. p. 1–9. doi:10.1109/noms.2018.8406221.
10. Chang SY, Park Y, Ashok Babu BB. Fast IP hopping randomization to secure hop-by-hop access in SDN. IEEE Trans Netw Serv Manage. 2019;16(1):308–20. doi:10.1109/tnsm.2018.2889842.
11. Krylov V, Kravtsov K. IP fast hopping protocol design. In: Proceedings of the 10th Central and Eastern European Software Engineering Conference in Russia; 2014 Oct 23–25; Moscow, Russia. p. 1–5. doi:10.1145/2687233.2687238.
12. Wang P, Zhou M, Ding Z. A two-layer IP hopping-based moving target defense approach to enhancing the security of mobile *ad-hoc* networks. Sensors. 2021;21(7):2355. doi:10.3390/s21072355.
13. Zhao Z, Liu F, Gong D, Chen L, Xiang F, Li Y. An SDN-based IP hopping communication scheme against scanning attack. In: Proceedings of the 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN); 2017 May 6–8; Guangzhou, China. p. 559–64. doi:10.1109/iccsn.2017.8230174.
14. Zhang L, Wei Q, Tang XC, Fang J. SDN network active defense technology based on path and end address hopping. Comput Res Dev. 2017;54(12):2761–71. (In Chinese). doi:10.1109/fskd.2016.7603498.
15. Jafarian HJ, Al-Shaer E, Duan Q. Openflow random host mutation. Charlotte, NC, USA: University of North Carolina at Charlotte; 2012.
16. Heydari V. Moving target defense for securing SCADA communications. IEEE Access. 2018;6:33329–43. doi:10.1109/access.2018.2844542.

17. Kong Y, Zhang L, Wang Z. IPv6 address hopping active defense model based on sliding time window. Comput Appl. 2018;38(07):1936–73. (In Chinese). doi:10.11772/j.issn.1001-9081.2018010073.
18. Ghaderi M, Jero S, Nita-Rotaru C, Safavi-Naini R. On randomization in MTD systems. In: Proceedings of the 9th ACM Workshop on Moving Target Defense; 2022 Nov 7; Los Angeles, CA, USA. p. 37–43. doi:10.1145/3560828.3564016.
19. Liu H, Wang Z, Guo Y. An IPv6 active defense model based on multi-point jump. J Electron Inf Technol. 2012;34(07):1715–20. doi:10.3724/sp.j.1146.2011.01350.
20. Mei Z, Wang Z, Wang Y, Zhang L. An IPv6 MTD model based on subnet hopping. Comput Appl Softw. 2016;33(12):301–24. (In Chinese). doi:10.3969/j.issn.1000-386x.2016.12.070.
21. Han L, Song J, Sun S. Research on mobile target defense mechanism based on device address in SD-IoT. Inf Netw Secur. 2022;22(11):36–46. (In Chinese).
22. Zhang B, Li H, Zhang S, Sun J, Wei N, Xu W, et al. Multi-constraint and multi-policy path hopping active defense method based on SDN. Future Internet. 2024;16(4):143. doi:10.3390/fi16040143.
23. Gu Y, Hu Y, Ding Y, Xie J. Random address hopping method based on traffic awareness. Comput Eng. 2018;44(10):28–41. (In Chinese). doi:10.19678/j.issn.1000-3428.0051042.
24. Gao Y, Yang L, Zhu R, Yang F, Cao Y, Zhang L. Intelligent hopping mechanism for deception defense scenarios based on reinforcement learning. In: Proceedings of the 2024 IEEE 49th Conference on Local Computer Networks (LCN); 2024 Oct 8–10; Normandy, France. p. 1–8. doi:10.1109/lcn60385.2024.10639792.
25. Shi F, Zhou Z, Yang W, Li S, Liu Q, Bao X. AHIP: an adaptive IP hopping method for moving target defense to thwart network attacks. In: Proceedings of the 2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD); 2023 May 24–26; Rio de Janeiro, Brazil. p. 1300–5. doi:10.1109/cscwd57460.2023.10152746.
26. Xu X, Hu H, Liu Y, Zhang H, Chang D. An adaptive IP hopping approach for moving target defense using a light-weight CNN detector. Secur Commun Netw. 2021;2021:8848473. doi:10.1155/2021/8848473.
27. He Y, Zhang M, Yang X, Sun QT, Luo J, Yu Y. The intelligent offense and defense mechanism of Internet of vehicles based on the differential game-IP hopping. IEEE Access. 2020;8:115217–27. doi:10.1109/access.2020.3004255.
28. Kietzmann P, Schmidt TC, Wählisch M. A guideline on pseudorandom number generation (PRNG) in the IoT. ACM Comput Surv. 2022;54(6):1–38. doi:10.1145/3453159.
29. Kanso A, Smaoui N. Logistic chaotic maps for binary numbers generations. Chaos Solitons Fractals. 2009;40(5):2557–68. doi:10.1016/j.chaos.2007.10.049.
30. Jindal P, Singh B. RC4 encryption-a literature survey. Procedia Comput Sci. 2015;46:697–705. doi:10.1016/j.procs.2015.02.129.