Randomness analysis of end-to-end delay in random forwarding network

Xiaowen Wang [1] , **Jie Huang** [Corresp., 1, 2] , **Zhenyu Duan** [1] , **Yao Xu** [1] , **Yifei Yao** [1]

[1] School of Cyber Science and Engineering, Southeast University, Nanjing, Jiangsu, China

[2] Purple Mountain Laboratories, Nanjing, Jiangsu, China

Corresponding Author: Jie Huang
Email address: jhuang@seu.edu.cn

Random forwarding networks play a significant role in solving security and load balancing problems. As a random quantity easily obtained by both sender and receiver, the end-to-end delay of Random forwarding networks can be utilized as an effective random source for cryptography-related applications. In this paper, we propose a mathematical model of Random forwarding networks and give the calculation method of end-to-end delay distribution. In exploring the upper limit of the randomness of end-to-end delay, we find that the end-to-end delay collision of different forwarding routes is the main reason for the decrease of end-to-end delay randomness. Some of these collisions can be optimized by better network deployment, while others are caused by some interesting network topology, which is unavoidable. For further analysis, we propose an algorithm to calculate the inevitable collision in Random forwarding networks skillfully by using Symbol Matrix, and we give the optimal node forwarding strategy with the maximum randomness of the end-to-end delay for a given number of middle forwarding nodes and forwarding times. Finally, we introduce a specific application of generating symmetric keys by using the randomness of the end-to-end delay.

Randomness analysis of end-to-end delay in Random forwarding networks

Xiaowen Wang[1], Jie Huang[1,2], Zhenyu Duan[1], Yao Xu[1], and Yifei Yao[1]

[1]School of Cyber Science and Engineering, Southeast University, Nanjing, Jiangsu, China

[2]Purple Mountain Laboratories, Nanjing, Jiangsu, China

Corresponding author:

Jie Huang[1]

Email address: jhuang@seu.edu.cn

ABSTRACT

Random forwarding networks play a significant role in solving security and load balancing problems. As a random quantity easily obtained by both sender and receiver, the end-to-end delay of Random forwarding networks can be utilized as an effective random source for cryptography-related applications. In this paper, we propose a mathematical model of Random forwarding networks and give the calculation method of end-to-end delay distribution. In exploring the upper limit of the randomness of end-to-end delay, we find that the end-to-end delay collision of different forwarding routes is the main reason for the decrease of end-to-end delay randomness. Some of these collisions can be optimized by better network deployment, while others are caused by some interesting network topology, which is unavoidable. For further analysis, we propose an algorithm to calculate the inevitable collision in Random forwarding networks skillfully by using Symbol Matrix, and we give the optimal node forwarding strategy with the maximum randomness of the end-to-end delay for a given number of middle forwarding nodes and forwarding times. Finally, we introduce a specific application of generating symmetric keys by using the randomness of the end-to-end delay.

INTRODUCTION

A drunk started from a bar to go home. When he arrived at a crossroads, he couldn't recognize the way back because of drunkenness. There were two choices in front of him. One was to stay in place for a while, the other was to choose a road in front of him at random. The streets of this city extend in all directions, and the drunk could go anywhere he could go. After walking a few blocks, the drunk woke up and went back to his home directly. Because the drunk goes to the bar every day, his wife at home is curious about what regularity of the time he comes back?

The problem of drunk returning home can be modeled by random forwarding networks and the time drunk spends on the road is the end-to-end delay of a random forwarding route. Suppose we have a random forwarding network G consisting of m middle forwarding nodes $Z_1, Z_2, \ldots, Z_m$. G plays the role of forwarding the delay measurement data packet sent by Alice to Bob. The forwarding rules are as follows:

- Firstly, Alice randomly selects a middle forwarding node to send the initial delay measurement data packet.
- Secondly, this middle forwarding node randomly selects other middle forwarding nodes as the next hop of forwarding or forwards the packet to itself, and stipulates that the total forwarding times of the delay measurement data packet is N, which is recorded in the data packet. Every time the data packet is forwarded, the remaining forwarding times is reduced by 1 by the currently receiving middle forwarding node.
- Finally, when the remaining forwarding time becomes 0, the current middle forwarding node directly forwards this delay measurement data packet to Bob and finishes this forwarding.

Obviously, the end-to-end delay is related to the number of middle forwarding nodes, random

46 forwarding strategy ,and forwarding times. The main content of this paper is to reveal the relationship
47 between them.

48 First, let us introduce the definition of Random forwarding networks. **Random forwarding networks**
49 **(RFNs)** are a kind of network consisting of several network nodes called middle forwarding nodes with
50 random forwarding as the forwarding strategy. Different from the Open Shortest Path First (OSPF)
51 forwarding strategy, RFNs do not focus on efficient data transmission, but on the security application and
52 load balancing in the process of data forwarding.

53 **Security application** is embodied in the attacker's inability to track the data in the Random forwarding
54 networks because the forwarding node is randomly selected rather than determined by some forwarding
55 rules (Duan et al., 2013). The famous Tor network takes advantage of the anonymity of random forwarding,
56 and Tor agents replace users to visit service sites to keep users secure. Using onion routing technology,
57 access requests are randomly forwarded among several Tor network agents, hiding users' real addresses
58 (Syverson et al., 2001). In Optical Transport Networks (OTN), random forwarding is potentially more
59 secure than explicit forwarding, and the probability that a wiretapper recovers a whole secure data as the
60 first try is in the range of 10^{-7} (Engelmann et al., 2014).

61 while **Load balancing** can evenly distribute tasks to multiple working nodes, which is an essential
62 technology in high-performance web services (Liu et al., 2013). In wireless sensor networks (WSNs),
63 random forwarding can provide a more stable and longer lifetime of networks (Li and Kim, 2015). In
64 addition, RFNs are strongly extensible because the random forwarding strategy makes every node have
65 equal status, and it can flexibly add new nodes without changing the basic forwarding logic. Because
66 of the flexibility, RFNs also has strong robustness. When an abnormal node in an RFN is detected, the
67 whole RFN can still work effectively by deleting the abnormal node from the forwarding list.

68 The whole network delay from Alice to Bob is the **end-to-end delay**. In RFNs, the end-to-end delay
69 has strong randomness, and it can be easily measured by both sender and receiver, which is of great
70 significance in cryptography (Abdelkefi and Jiang, 2011). The delay between middle forwarding nodes
71 has stability and reciprocity, in which stability means that there is no obvious fluctuation in the delay
72 between middle forwarding nodes within a short period time (within a few minutes), while reciprocity
73 means that the communication round-trip delay is approximately equal (Choi et al., 2004).

74 In physical layer security, it is a valuable technology to generate security keys by using the reciprocity
75 and randomness of wireless channels, which can enable both parties to quickly establish a secure commu-
76 nication channel (Sánchez et al., 2020). The lightweight security solutions relying on key generation from
77 wireless channels are eminently suitable for the Internet of Things (IoTs) (Zhang et al., 2020). Similarly,
78 the end-to-end delay with reciprocity and randomness in RFNs can also be used to achieve the same
79 purpose. However, the difference is that using wireless channel characteristics to generate keys has great
80 restrictions on communication distance while using network characteristics has no such restrictions, which
81 can achieve cross-regional key negotiation.

82 Therefore, in order to further explore the potential of RFNs in multi-node cross-domain secret sharing
83 and key distribution, this paper mainly discusses the randomness of end-to-end delay in RFNs. The main
84 contributions of this paper are summarized as follows:

85 1. We proposed the mathematical model of RFNs and derived the mathematical formula of the
86 end-to-end delay distribution.
87 2. We presented a quantitative calculation method of the end-to-end delay randomness based on
88 information entropy and give a theoretical explanation.
89 3. We explored the forwarding strategy that maximizes the randomness of end-to-end delay when the
90 number of middle forwarding nodes and forwarding times is constant. We revealed the main reason
91 for the decrease of the randomness of end-to-end delay is delay collision and provided the optimal
92 forwarding strategy and the theoretical upper limit of end-to-end delay randomness under different
93 numbers of middle forwarding nodes and random forwarding times.
94 4. We introduced the application of cross-domain key distribution using the randomness and reciprocity
95 of end-to-end delay.

RFNS MODEL

The end-to-end delay probability distribution

In this section, we will first give the algebraic relationship between end-to-end delay distribution and the forwarding strategy of middle forwarding nodes.

Because the time delay between each node in the forwarding network is stable in the short term, once the deployment of forwarding network G is completed, the time delay between nodes is determined in the short term. Here are some symbol habits used in this paper, the delay and the forwarding probability between Alice and middle forwarding nodes Z_i are denoted as d_{ai} and p_{ai} respectively, the delay and the forwarding probability between middle forwarding nodes Z_i and middle forwarding nodes Z_j are denoted as d_{ij} and p_{ij} respectively, and the delay between Bob and middle forwarding nodes Z_i are denoted as d_{ib}.

In the process of forwarding, we use the delay monomial px^d keep the cumulative information of probability and the cumulative information of delay because of such property: $p_1x^{d_1} \cdot p_2x^{d_2} = p_1p_2x^{d_1+d_2}$. take Figure 1 as an example, the delay and the probability of $Alice \rightarrow Z_1 \rightarrow Z_2 \rightarrow Z_3 \rightarrow Bob$ for the forwarding route r can be calculated as

$$p_rx^{d_r} = \prod p_ix^{d_i} = (p_{a1}p_{12}p_{23}p_{3b})x^{(d_{a1}+d_{12}+d_{23}+d_{3b})}$$
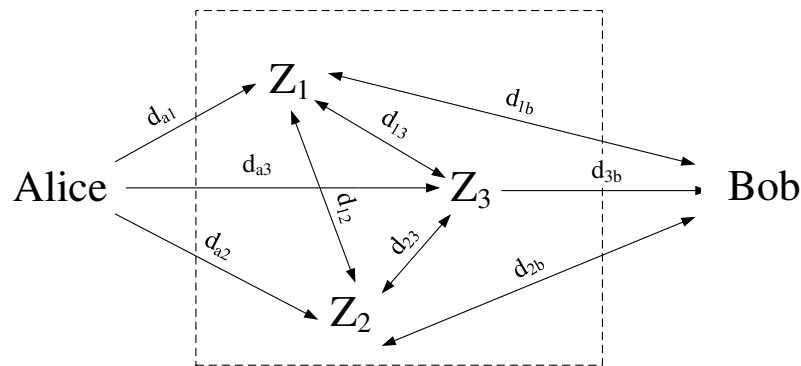
Figure 1. Random forwarding networks for $m = 3$

Figure 2 describes all possible forwarding routes from Alice to Bob under general conditions of m nodes and N times and defines the set of these routes as S. Since each route corresponds to a delay monomial p_rx^r, then the distribution of end-to-end delay is the sum of the delay multinomial p_rx^r corresponding to all routes, and we express this sum as

$$p(x) = \sum_{r \in S} p_rx^{d_r}$$

After simplification, we get $p(x) = \sum_{i=1}^{n} p_ix^{d_i}$, which means that the probability of taking d_i as the end-to-end delay is p_i.

$p(x)$ is the polynomial form of the end-to-end delay probability distribution. Considering the multi-layer network structure of forwarding routes, the **vector form** of the end-to-end delay distribution polynomial $p(x)$ can be calculated as follows

$$p(x) = s^T P^N t \tag{1}$$

where $s = \begin{pmatrix} p_{a1}x^{d_{a1}} & p_{a2}x^{d_{a2}} & \dots & p_{am}x^{d_{am}} \end{pmatrix}^T$ is the initial forwarding vector forwarded by Alice to the middle forwarding nodes, $t = \begin{pmatrix} x^{d_{1b}} & x^{d_{2b}} & \dots & x^{d_{mb}} \end{pmatrix}^T$ is the end forwarding vector forwarded by middle forwarding nodes to Bob, and P is the forwarding matrix of middle forwarding nodes forwarding
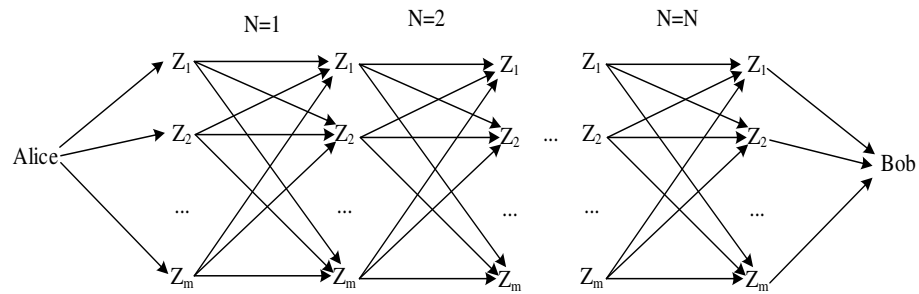
Figure 2. All possible forwarding routes of *m* nodes for *N* times

to each other, that is

$$P = \begin{pmatrix} p_{11}x^{d_{11}} & \cdots & p_{1m}x^{d_{1m}} \\ \vdots & \ddots & \vdots \\ p_{m1}x^{d_{m1}} & \cdots & p_{mm}x^{d_{mm}} \end{pmatrix}$$

When the forwarding network is deployed, the delay d_{ij} between any two nodes is determined. According to Eq. (1), $p(x)$ is uniquely determined by the random forwarding strategy. Let $P_A = \begin{pmatrix} p_{a1} & p_{a2} & \cdots & p_{am} \end{pmatrix}^T$ denotes the initial random forwarding strategy forwarded by Alice to the middle forwarding nodes. Let P_Z denotes the random forwarding strategy of middle forwarding nodes forwarding to each other, that is

$$P_Z = \begin{pmatrix} p_{11} & \cdots & p_{1m} \\ \vdots & \ddots & \vdots \\ p_{m1} & \cdots & p_{mm} \end{pmatrix}$$

Measurement of end-to-end delay randomness

As shown in Figure 3, the network model of end-to-end time delay generated by the random forwarding network is regarded as a black box. Given forwarding strategy (P_A, P_Z), this black box will randomly generate end-to-end time delay data, which will obey the probability distribution defined by $p(x)$. This is similar to a discrete source sending uncertain symbols in communication. The randomness of a source sending symbols can be measured by information entropy, which reflects the uncertainty of a source by calculating the average self-information amount of symbols (Shannon, 1948).

Figure 3. End-to-end delay generation model

Therefore, by calculating the information entropy of end-to-end delay, we can quantitatively analyze its randomness. If the randomness of end-to-end delay is exploited to generate the secret key, the effective length of the secret key is proportional to the randomness of end-to-end delay. For example, if the end-to-end delay is given by the front and back of a coin thrown, d_1 will be generated on the front side and d_2 will be generated on the backside, that is to say, the end-to-end delay will only generate two possible values with the same probability, so there are at most two corresponding secret keys. Although the secret key length can be expanded by some algorithms like Hash (Bellare et al., 1996), the effective key code length is actually only 1bit, which is the information entropy of the end-to-end delay.

The measurement formula of end-to-end delay randomness is as follows

$$H_d = -\sum_i p_i \log p_i \tag{2}$$

131   where p_i are the coefficients of $p(x)$ calculated by Eq. (1).

OPTIMIZATION OF THE RANDOMNESS OF END-TO-END DELAY

133   This section mainly discusses how to improve the randomness of end-to-end delay, which is of great
134   significance in cryptography.

evitable collision and inevitable collision of end-to-end delay

136   End-to-end delay collision (hereinafter referred to as collision) means that two different forwarding routes
137   have the same end-to-end delay. Collision is one of the main reasons leading to the decrease of the
138   randomness of end-to-end delay because of the reduction of end-to-end delay sample space.
139   Collisions that can be solved by adjusting RFNs deployment are referred to as evitable collisions.
140   Otherwise, they are referred to as inevitable collision. These two collisions are described in detail below.

141   **evitable collision**   In order to show this collision intuitively, an example as shown in Figure 4 is
142   provided, which is an equal delay forwarding network with two middle forwarding nodes, in which the
143   delay between any two nodes is approximately the same (replaced by 1).
144   Taking single forwarding as an example, it is easy to find from Figure 4 that the end-to-end delay of
145   route $Alice \to Z_1 \to Z_1 \to Bob$ is the same as $Alice \to Z_2 \to Z_2 \to Bob$, and the end-to-end delay of route
146    $Alice \to Z_1 \to Z_2 \to Bob$ is the same as $Alice \to Z_2 \to Z_1 \to Bob$, that is to say, the end-to-end delays of
147   these two pairs of routes collide.
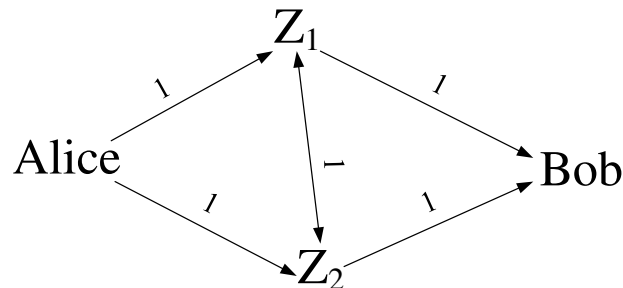
Figure 4. Equal delay forwarding network for $m = 2$

The end-to-end delay distribution polynomial corresponding to Figure 4 is

$$p(x) = \begin{pmatrix} p_{a1}x \\ p_{a2}x \end{pmatrix}^T \begin{pmatrix} p_{11} & p_{12}x \\ p_{21}x & p_{22} \end{pmatrix} \begin{pmatrix} x \\ x \end{pmatrix} = p_{a1}p_{11}x^2 + p_{a2}p_{22}x^2 + p_{a1}p_{12}x^3 + p_{a2}p_{21}x^3$$

148   The collision of end-to-end delay is reflected by the existence of like terms in the end-to-end delay
149   distribution polynomial, and the existence of like terms reduces the randomness of end-to-end delay. For
150   example, $Alice \to Z_1 \to Z_1 \to Bob$ corresponds to $p_{a1}p_{11}x^2$, $Alice \to Z_2 \to Z_2 \to Bob$ corresponds to
151    $p_{a2}p_{22}x^2$, which are like terms.
152   Equal delay forwarding networks are prone to delay collisions. To avoid such collisions, the de-
153   ployment of forwarding networks can be adjusted, such as the forwarding network shown in Figure
154   5.
155   Similarly, taking a single forwarding as an example, the corresponding end-to-end delay distribution
156   polynomial is

$$p(x) = \begin{pmatrix} p_{a1}x \\ p_{a2}x^2 \end{pmatrix}^T \begin{pmatrix} p_{11} & p_{12}x^3 \\ p_{21}x^3 & p_{22} \end{pmatrix} \begin{pmatrix} x \\ x^3 \end{pmatrix} = p_{a1}p_{11}x^2 + p_{a2}p_{22}x^5 + p_{a1}p_{12}x^7 + p_{a2}p_{21}x^6$$
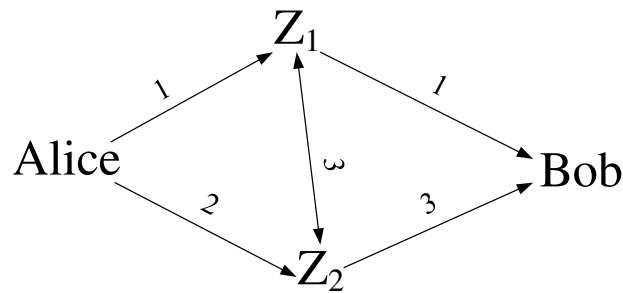
Figure 5. Adjust the deployed forwarding network for $m = 2$

There is no like term in the adjusted end-to-end delay distribution polynomial, that is to say, the end-to-end delay corresponding to each possible forwarding route is different, which improves the randomness of the measurement delay. This kind of collision is called evitable collision.

Inevitable collision   Taking $m = 2$ and $N = 2$ as an example, the end-to-end delay distribution polynomial is as follows

$$p(x) = \boldsymbol{s}^T \boldsymbol{P}^2 \boldsymbol{t} = \boldsymbol{s}^T \begin{pmatrix} p_{11} & p_{12}x^d \\ p_{21}x^d & p_{22} \end{pmatrix}^2 \boldsymbol{t} = \boldsymbol{s}^T \begin{pmatrix} p_{11}^2 + p_{12}p_{21}x^{2d} & p_{11}p_{12}x^d + p_{12}p_{22}x^d \\ p_{21}p_{11}x^d + p_{22}p_{21}x^d & p_{22}^2 + p_{21}p_{12}x^{2d} \end{pmatrix} \boldsymbol{t}$$

It can be found that the internal elements of matrix $\boldsymbol{P}^2$ have like terms, such as $p_{11}p_{12}x^d + p_{12}p_{22}x^d$ in the second column of the first row and $p_{21}p_{11}x^d + p_{22}p_{21}x^d$ in the first column of the second row, which will lead to the existence of like terms in the expansion. The collision caused by such like terms can not be avoided by adjusting the deployment. So, we call this kind of collision inevitable collision.

Taking the forwarding network in Figure 5 as an example, make a forwarding route map under two forwarding, which is shown as Figure 6. The blue route (-··) is $Alice \rightarrow Z_1 \rightarrow Z_1 \rightarrow Z_2 \rightarrow Bob$ and the yellow route (-·) is $Alice \rightarrow Z_1 \rightarrow Z_2 \rightarrow Z_2 \rightarrow Bob$, which correspond to $p_{11}p_{12}x^d$ and $p_{12}p_{22}x^d$ from the second column of the first row in matrix $\boldsymbol{P}^2$ respectively. Since the two routes share all the edges that can be changed by deployment, they are bound to collide.
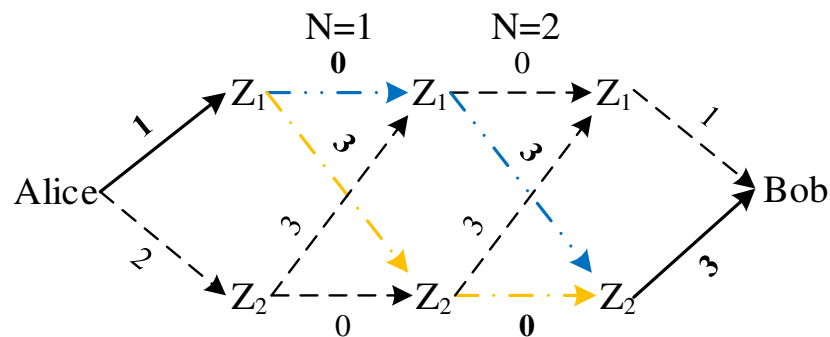
Figure 6. Inevitable collision of end-to-end delay (take forwarding network in Figure 5 as an example)

Fast Calculation of Inevitable Collision Using Symbol Matrix

The collision of end-to-end delay is the main reason for the decrease of the randomness of end-to-end delay. The evitable collision can be solved by adjusting the deployment, while the inevitable collision is an unavoidable problem in the optimization of the randomness of end-to-end delay. Therefore, this subsection introduces a method for quickly calculating the inevitable collision in RFNs.

We have known that the inevitable collision depends on whether there are like terms in the internal elements of matrix $\boldsymbol{P}^N$, which is an inherent property of matrix power operation and is independent of the

178 value of the specific elements of the matrix itself. Symbol matrix is a matrix composed of simple symbols,
179 which is very suitable for revealing the structure of like terms in the internal elements of matrix $\boldsymbol{P}^N$.
180   The diagonals of the symbol matrix are all replaced by 1, which represents that the nodes forward to
181 themselves will not change the end-to-end delay. The non-diagonals represent the delay between different
182 nodes and are replaced by symbols. In fact, the symbol matrix is only a simplification of the forwarding
183 matrix $\boldsymbol{P}$. In this paper, $\boldsymbol{S}_m$ is used to denote the symbol matrix of the forwarding matrix $\boldsymbol{P}$ with m nodes.
184 Note that $\boldsymbol{S}_m$ is symmetric.

For example, the symbol matrix $\boldsymbol{S}_2$ for $m = 2$ is

$$\boldsymbol{S}_2 = \begin{pmatrix} 1 & a \\ a & 1 \end{pmatrix} \leftarrow \begin{pmatrix} p_{11} & p_{12}x^d \\ p_{21}x^d & p_{22} \end{pmatrix}$$

If $N = 2$, the symbol matrix $\boldsymbol{S}_2^2$ is

$$\boldsymbol{S}_2^2 = \begin{pmatrix} 1+a^2 & 2a \\ 2a & 1+a^2 \end{pmatrix} \leftarrow \begin{pmatrix} p_{11}^2 + p_{12}p_{21}x^{2d} & p_{11}p_{12}x^d + p_{12}p_{22}x^d \\ p_{21}p_{11}x^d + p_{22}p_{21}x^d & p_{22}^2 + p_{21}p_{12}x^{2d} \end{pmatrix}$$

185 where $2a$ is the result of merging like terms, the coefficient represents that the number of inevitable
186 collision routes is 2.

With the help of symbol matrix, it is easier to calculate the inevitable collision in complex cases.
Taking $m = 3$ as an example, the symbol matrix $\boldsymbol{S}_3$ is

$$\boldsymbol{S}_3 = \begin{pmatrix} 1 & a & b \\ a & 1 & c \\ b & c & 1 \end{pmatrix}$$

When $N = 2$, the symbol matrix $\boldsymbol{S}_3^2$ is

$$\boldsymbol{S}_3^2 = \begin{pmatrix} 1+a^2+b^2 & 2a+bc & 2b+ac \\ 2a+bc & 1+a^2+c^2 & 2c+ab \\ 2b+ac & 2c+ab & 1+b^2+c^2 \end{pmatrix}$$

187   We find that the form of the elements on the main diagonal of $\boldsymbol{S}_3^2$ is consistent, and the form of the
188 elements on the upper triangle and the lower triangle (except the main diagonal) of $\boldsymbol{S}_3^2$ is consistent. The
189 difference only exists in the rotation of symbols, which is called **rotation consistency**. That is to say, as
190 long as the first two elements of the first line of $\boldsymbol{S}_3^2$ are calculated, the remaining elements can be recovered
191 by rotation consistency.

So $\boldsymbol{S}_3^2$ can be compressed as

$$\boldsymbol{S}_3^2 = (1+a^2+b^2, 2a+bc)_{a,b,c}$$

192 Where the elements in $()$ is the first two elements in $\boldsymbol{S}_3^2$ and the subscript a, b, c denote the symbols of
193 rotation.
194 Two operators is used to recover the original $\boldsymbol{S}_3^2$ from the compressed $\boldsymbol{S}_3^2$.

- The first operator is the cyclic permutation transformation R:

$$\begin{pmatrix} f_1(a,b,c) \\ f_2(a,b,c) \\ f_3(a,b,c) \end{pmatrix} \xrightarrow{R} \begin{pmatrix} f_3(\sigma(a,b,c)) \\ f_1(\sigma(a,b,c)) \\ f_2(\sigma(a,b,c)) \end{pmatrix}$$

where permutation operator $\sigma = \begin{pmatrix} a & b & c \\ c & a & b \end{pmatrix}$ and it makes

$$\boldsymbol{S}_3^2 = (\boldsymbol{\mu} \quad R(\boldsymbol{\mu}) \quad R^2(\boldsymbol{\mu})), \quad \boldsymbol{\mu} = \begin{pmatrix} 1+a^2+b^2 \\ 2a+bc \\ 2b+ac \end{pmatrix}$$

Algorithm 1 FPSSM(m,N)

Input: m: Dimensions of Symbol Matrix $\boldsymbol{S}_m$; N: Power of Symbol Matrix Multiplication

1: $\boldsymbol{S}_m$ = Symbol_Matrix_Generate(m)
2: R = Cyclic_Permutation_Generate($\boldsymbol{S}_m$)
3: e_{ij} = Replacement_Generate($\boldsymbol{S}_m$)
4: $f,g = \boldsymbol{S}_m[0,0],\boldsymbol{S}_m[0,1]$
5: $\boldsymbol{\mu_0},\boldsymbol{\mu_1} = \boldsymbol{S}_m[:,0],\boldsymbol{S}_m[:,1]$
6: **for** i in $[1,2,\ldots,N-1]$ **do**
7: $\quad\boldsymbol{\gamma} = [f,g,e_{23}(g),e_{24}(g),\ldots,e_{2m}(g)]^T$
8: $\quad f \leftarrow \boldsymbol{\gamma}^T \boldsymbol{\mu_0}$
9: $\quad g \leftarrow \boldsymbol{\gamma}^T \boldsymbol{\mu_1})$
10: $\boldsymbol{\gamma} = [f,g,e_{23}(g),e_{24}(g),\ldots,e_{2m}(g)]^T$
11: $\boldsymbol{S}_m^N = [\boldsymbol{\gamma},R(\boldsymbol{\gamma}),\ldots,R^{m-1}(\boldsymbol{\gamma})]$
12: **return** $\boldsymbol{S}_m^N$

- The second operator is replacement transformation e_{ij}:

$$f_i(a,b,c) \xrightarrow{e_{ij}} f_j(a,b,c) = f_i(e_{ij}(a,b,c)), \quad i,j \geq 2$$

where e_{ij} can be generated by $\boldsymbol{S}_m[:,j] = e_{ij}(\boldsymbol{S}_m[:,i])$. $\boldsymbol{S}_m[:,i]$ denotes the ith column of $\boldsymbol{S}_m$. In recovering the compressed $\boldsymbol{S}_3^2$, we need $e_{23} = \begin{pmatrix} a & b \\ b & a \end{pmatrix} = a \leftrightarrow b$ to recover $\boldsymbol{\mu}$ as

$$\boldsymbol{\mu} = \begin{pmatrix} f_1(a,b,c) \\ f_2(a,b,c) \\ f_2(e_{23}(a,b,c)) \end{pmatrix}$$

By using operators R and e_{ij}, the complete matrix $\boldsymbol{S}_3^2$ can be recovered from the first two elements of the $\boldsymbol{S}_3^2$. This property is universal, and there is such rotation consistency for any number of nodes and any number of forwarding times (See **APPENDIX** for proof).

Now, we will show how to use these two operators to calculate $\boldsymbol{S}_3^3$ easily:

$$\boldsymbol{S}_3 = (1,a)_{a,b,c}, \boldsymbol{\mu_0} = \begin{pmatrix} 1 \\ a \\ b \end{pmatrix}, \boldsymbol{\mu_1} = R(\boldsymbol{\mu_0}) = \begin{pmatrix} a \\ 1 \\ c \end{pmatrix}, \boldsymbol{\gamma} = \begin{pmatrix} 1 \\ a \\ e_{23}(a) \end{pmatrix} = \boldsymbol{\mu_0}$$

$$\boldsymbol{S}_3^2 = (\boldsymbol{\gamma}^T \boldsymbol{\mu_0},\boldsymbol{\gamma}^T \boldsymbol{\mu_1})_{a,b,c} = (1+a^2+b^2,2a+bc)_{a,b,c}, \boldsymbol{\gamma} = \begin{pmatrix} 1+a^2+b^2 \\ 2a+bc \\ e_{23}(2a+bc) \end{pmatrix} = \begin{pmatrix} 1+a^2+b^2 \\ 2a+bc \\ 2b+ac \end{pmatrix}$$

$$\boldsymbol{S}_3^3 = (\boldsymbol{\gamma}^T \boldsymbol{\mu_0},\boldsymbol{\gamma}^T \boldsymbol{\mu_1})_{a,b,c} = (1+3a^2+3b^2+2abc,a^3+ab^2+ac^2+3a+3bc)_{a,b,c}$$

where $\boldsymbol{\gamma}$ is the first column of $\boldsymbol{S}_3^N$.

Generally, **the fast power of symmetric symbol matrix (FPSSM)** is given by Algorithm 1 to calculate matrix $\boldsymbol{S}_m^N$ easily. Because every loop in FPSSM only needs to calculate two times vector multiplication, the algorithm reduces the time complexity of polynomial matrix multiplication from $O(Nm^3)$ to $O(Nm)$ and the space complexity from $O(m^2)$ to $O(1)$. The complexity here refers to the complexity of polynomial multiplication, not the complexity of conventional numerical multiplication.

Now we have powerful tools to study the inevitable collision of RFNs in complex conditions. As long as we calculate $\boldsymbol{S}_m^N$, all possible inevitable collisions can be obtained. Take $m=3, N=3$ as an example, every term in $\boldsymbol{S}_3^3$ whose coefficient is not 1 represents an inevitable collision. Figure 7 shows the inevitable collision of $3a^2$ and $2abc$ in $\boldsymbol{S}_3^3$. Among them, the first figure labeled $3a^2$ shows a kind of inevitable collision caused by self forwarding, while the second figure labeled $2abc$ shows another kind of inevitable collision caused by symmetry in the forwarding route map. Of course, these two types are not mutually exclusive. There are also inevitable collisions caused by both self-forwarding and symmetry in forwarding route maps with more middle forwarding nodes.
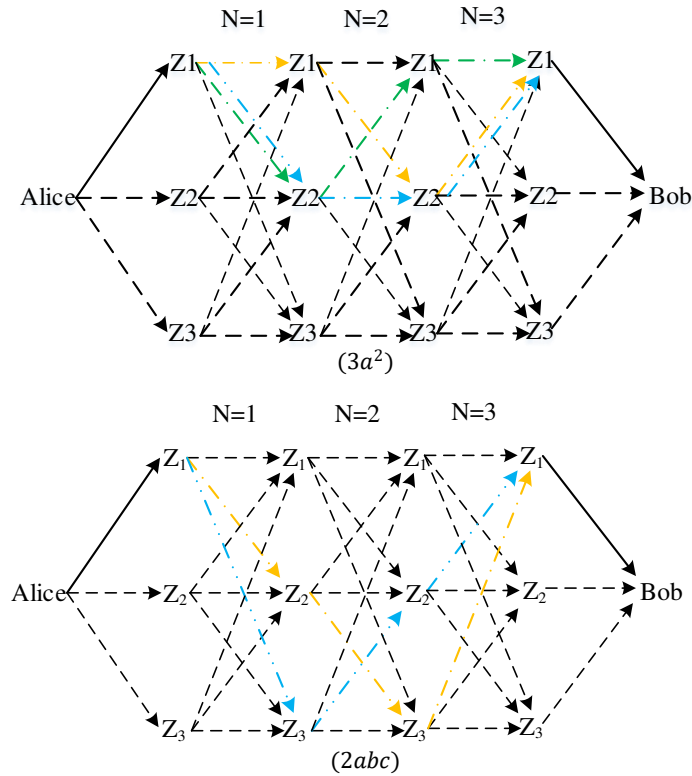
Figure 7. Two typical inevitable collision of end-to-end delay for $m = 3, N = 3$

The upper limit of end-to-end delay randomness and the optimal forwarding strategy

In this subsection, we will explore how to formulate random forwarding strategies to achieve the upper limit of end-to-end delay randomness. We have known that the collision of end-to-end delay will lead to the decrease of randomness, so the first step is to adjust the deployment to remove all evitable collisions. In this way, our goal becomes the optimal forwarding strategy under the inevitable collision deployment.

Our optimization problem is that, for a given non-evitable collision random forwarding network G (including Alice and Bob), what is the optimal forwarding strategy to maximize the entropy of the end-to-end delay information? The mathematical form is described as follows

$$Given : G = (V, E), V = \{Alice, Z_1, Z_2, \ldots, Z_m, Bob\}$$

$$Goal : \max_{\boldsymbol{P_A}, \boldsymbol{P_Z}} H_d = -\sum_i p_i \log p_i$$

where p_i are the coefficients of $p(x)$ calculated by Eq. (1).

The maximum entropy problem is a convex optimization, and its optimal solution exists and is unique(Boyd and Vandenberghe, 2004), which is the key to solving this optimization problem.

First, let's define a cyclic shift permutation operator C on the matrix $\boldsymbol{A} \in \mathbb{R}^{m \times m}$ as

$$C : \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{pmatrix} \rightarrow \begin{pmatrix} a_{22} & a_{23} & \ldots & a_{21} \\ a_{32} & a_{33} & \ldots & a_{31} \\ \ldots & \ldots & \ldots & \ldots \\ a_{12} & a_{13} & \ldots & a_{11} \end{pmatrix}$$

In fact, C is a compound operation of cyclic left shift and cyclic upward shift on the matrix, so any element in the matrix is permuted as follows under the transformation of C

$$a_{ij} \xrightarrow{C} a_{[i+1]_m [j+1]_n}$$

where $[i+1]_m = (i \bmod m) + 1$ ensures the cyclic property of the shift.

Operator C has the following three important properties:

- **Property 1**

$$C^m(\boldsymbol{A}) = \boldsymbol{A}, \quad \boldsymbol{A} \in \mathbb{R}^{m \times m}$$

- **Property 2**

$$C(\boldsymbol{A})C(\boldsymbol{B}) = C(\boldsymbol{AB}), \quad \boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times m}$$

- **Property 3**

$$C(\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{y}, \quad \boldsymbol{A} \in \mathbb{R}^{m \times m}, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^m$$

Then, rewrite the end-to-end delay distribution polynomial $p(x)$ with **Hadamard Product** as

$$p(x) = \boldsymbol{s}^T \boldsymbol{P}^N \boldsymbol{t} = (\boldsymbol{P_A} \circ x^{\boldsymbol{D_A}})^T (\boldsymbol{P_Z} \circ x^{\boldsymbol{D_Z}})^N x^{\boldsymbol{D_B}} \tag{3}$$

where $x^{\boldsymbol{D_A}} = \begin{pmatrix} x^{d_{a1}} & x^{d_{a2}} & \dots & x^{d_{am}} \end{pmatrix}^T$, $x^{\boldsymbol{D_B}} = \begin{pmatrix} x^{d_{1b}} & x^{d_{2b}} & \dots & x^{d_{mb}} \end{pmatrix}^T$ and $x^{\boldsymbol{D_Z}} = \left(x^{d_{ij}} \right)_{m \times m}$. The operator $\circ$ is the Hadamard product operator defined by

$$(\boldsymbol{A} \circ \boldsymbol{B})_{ij} = (\boldsymbol{A})_{ij}(\boldsymbol{B})_{ij}$$

Because H_d is calculated by p_i, which are the coefficients of $p(x)$, and p_i is distributed by $\boldsymbol{P_A}^T \boldsymbol{P_Z}^N \mathbf{1}$ according to the end-to-end delay like term, that is to say, H_d is decided by $\boldsymbol{P_A}^T \boldsymbol{P_Z}^N \mathbf{1}$ (The notation $\mathbf{1}$ represents a vector of ones of appropriate length).

Since the optimization objective is $\boldsymbol{P_A}$ and $\boldsymbol{P_Z}$, by cyclic shifting $\boldsymbol{P_A}$ and $\boldsymbol{P_Z}$ in $p(x)$ using C, we get

$$C(p(x)) = (C(\boldsymbol{P_A}) \circ x^{\boldsymbol{D_A}})^T (C(\boldsymbol{P_Z}) \circ x^{\boldsymbol{D_Z}}))^N x^{\boldsymbol{D_B}}$$

According to **Property 2** and **Property 3**, we have

$$C(\boldsymbol{P_A})^T C(\boldsymbol{P_Z})^N \mathbf{1} = C(\boldsymbol{P_A}^T) C(\boldsymbol{P_Z}^N) \mathbf{1} = C(\boldsymbol{P_A}^T \boldsymbol{P_Z}^N \mathbf{1}) = \boldsymbol{P_A}^T \boldsymbol{P_Z}^N \mathbf{1}$$

Therefore,

$$H_d(\boldsymbol{P_A}, \boldsymbol{P_Z}) = H_d(C(\boldsymbol{P_A}), C(\boldsymbol{P_Z}))$$

It is known from the uniqueness of the optimal solution of convex optimization that

$$\begin{cases} \boldsymbol{P_A} = C(\boldsymbol{P_A}) \\ \boldsymbol{P_Z} = C(\boldsymbol{P_Z}) \end{cases}$$

Similarly,

$$\begin{cases} \boldsymbol{P_A} = C(\boldsymbol{P_A}) = C^2(\boldsymbol{P_A}) = \dots = C^{m-1}(\boldsymbol{P_A}) \\ \boldsymbol{P_Z} = C(\boldsymbol{P_Z}) = C^2(\boldsymbol{P_Z}) = \dots = C^{m-1}(\boldsymbol{P_Z}) \end{cases}$$

That is

$$\begin{cases} p_{a1} = p_{a2} = \dots = p_{am} = \frac{1}{m} \\ p_{11} = p_{22} = \dots = p_{mm} \\ p_{12} = p_{23} = \dots = p_{m1} \\ \dots \\ p_{1m} = p_{21} = \dots = p_{mm-1} \end{cases}$$

In addition, according to the **rotation consistency** of the $\boldsymbol{S}_m^N$, we know that the forwarding object $Z_2, Z_3, \dots, Z_m$ can rotate for node Z_1, that is

$$p_{12} = p_{13} = \dots = p_{1m}$$

Let $p_{11} = p$, $p_{12} = q$, $\boldsymbol{P_A}$ and $\boldsymbol{P_Z}$ are updated as

$$\begin{cases} \boldsymbol{P_A} = \frac{1}{m}\mathbf{1} \\ \boldsymbol{P_Z} = (p-q)\boldsymbol{I} + q\mathbf{1}\mathbf{1}^T \end{cases}$$

where $\boldsymbol{I}$ is the identity matrix with ones down the diagonal. In fact, p represents the self-forwarding probability of middle forwarding nodes, and q represents the forwarding probability between middle forwarding nodes.

Substituting back into Eq. (3), we have

$$p(x) = \boldsymbol{s}^T \boldsymbol{P}^N \boldsymbol{t} = \frac{1}{m} x^{\boldsymbol{D_A}^T} (((p-q)\boldsymbol{I} + q\boldsymbol{1}\boldsymbol{1}^T) \circ x^{\boldsymbol{D_Z}})^N x^{\boldsymbol{D_B}} \tag{4}$$

Then, our optimization goal is simplified as

$$\max_{p,q} H_d = -\sum_i p_i \log p_i$$

$$s.t. \quad p + (m-1)q = 1, 0 \le p, q \le 1$$

where p_i are the coefficients of $p(x)$ calculated by Eq. (4).

This optimization can be solved by the Karush-Kuhn-Tucker (KKT) conditional of Lagrange multiplier method as

$$\begin{cases} (m-1)\frac{\partial H_d}{\partial p} = \frac{\partial H_d}{\partial q} \\ p + (m-1)q = 1 \end{cases} \tag{5}$$

Considering

$$\boldsymbol{P} = ((p-q)\boldsymbol{I} + q\boldsymbol{1}\boldsymbol{1}^T) \circ x^{\boldsymbol{D_Z}} = \begin{pmatrix} p & qx^{d_{12}} & \cdots & qx^{d_{1m}} \\ qx^{d_{21}} & p & \cdots & qx^{d_{2m}} \\ \cdots & \cdots & \cdots & \cdots \\ qx^{d_{m1}} & qx^{d_{m2}} & \cdots & p \end{pmatrix}$$

Because $x^{d_{ij}} = x^{d_{ji}}$, $\boldsymbol{P}$ is a symmetric symbolic matrix. Algorithm 1 can be used to calculate $\boldsymbol{P}^N$ quickly and get the expression of H_d.

Take $m = 3, N = 2$ as an example, because $S_3^2 = (1 + a^2 + b^2, 2a + bc)_{a,b,c}$, we get

$$\boldsymbol{P}^2 = (p^2 + q^2 x^{2d_{12}} + q^2 x^{2d_{13}}, 2pqx^{d_{12}} + q^2 x^{d_{13}+d_{23}})_{x^{d_{12}}, x^{d_{13}}, x^{d_{23}}}$$

Then, H_d for $m = 3, N = 2$ is calculated by Eq. (2) as

$$H_d(m = 3, N = 2) = \log 3 - p^2 \log(p^2) - 4pq \log(2pq) - 4q^2 \log q^2$$

Figure 8 shows the change of $H_d(m = 3, N = 2)$ (bits) with the change of p. It can be clearly seen from the figure that the best p corresponding to the maximum entropy is the position marked by the red dot.
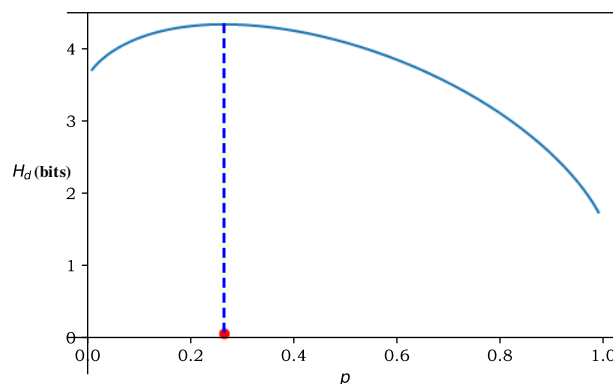
Figure 8. The change of H_d (bits) with the change of p for $m = 3, N = 2$

By substituting back into Eq. (5) and simplifying, we have

$$\begin{cases} \left(\frac{p}{q}+2\right)\log\frac{p}{q} = \left(\frac{p}{q}-2\right)\log 2 \\ p+2q = 1 \end{cases}$$

239   Through Newton's Method, the optimal forwarding strategy is

$$\begin{cases} p \approx 0.265 \\ q \approx 0.3675 \end{cases}$$

240   Then we know the best p in Figure 8 is 0.265, and the maximum entropy H_{dmax} is 4.333 bits.

241   Similarly, we can calculate the optimal forwarding strategy under other m and N. Some results are
242 given in the Table 1 and Table 2. Table 1 provides the p value of the optimal forwarding strategy, which
243 is the probability of self-forwarding. While the probability q representing the forwarding probability
244 between middle forwarding nodes can be calculated by $p = \frac{1-p}{m-1}$. Table 2 provides the maximum entropy
245 H_{dmax}, which is the upper limit of end-to-end delay randomness. From these two tables, we can find
246 that with the increase of forwarding times N, the p value of the best forwarding strategy tends to be
247 stable gradually and the growth rate of the maximum entropy H_{dmax} is gradually decreasing, that is to
248 say, it is impossible to increase the end-to-end delay randomness by the unlimited number of forwarding
249 times. When the number of forwarding times cannot increase the end-to-end delay randomness, the only
250 effective way is to add middle forwarding nodes.

     Noted that when the number of middle forwarding nodes $m = 2$, since p is always equal to 0.5, we
can get the expression of H_{dmax} about the number of forwarding times N as

$$H_{dmax}(N) = N+1 - \frac{1}{2^N}\sum_{i=0}^{N} C_N^i \log_2 C_N^i \approx \frac{1}{2}\log_2 N + 2$$

251   which shows that the impact of forwarding times on end-to-end delay is logarithmic.

p		N								
		1	2	3	4	5	6	7	8	9
	2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
	3	0.333	0.265	0.237	0.231	0.231	0.232	0.233	0.234	0.236
	4	0.25	0.175	0.146	0.135	0.132	0.132	0.132	0.132	0.132
	5	0.2	0.13	0.104	0.092	0.088	0.086	0.085	0.085	0.084
m	6	0.167	0.103	0.08	0.069	0.064	0.062	0.06	0.06	0.059
	7	0.143	0.086	0.065	0.055	0.05	0.047	0.046	0.045	0.044
	8	0.125	0.073	0.055	0.045	0.041	0.038	0.036	0.035	0.034
	9	0.111	0.064	0.047	0.039	0.034	0.031	0.03	0.029	0.028

Table 1. Optimal forwarding strategy (p value)

H_{dmax}		N								
		1	2	3	4	5	6	7	8	9
	2	2	2.5	2.811	3.03	3.2	3.333	3.447	3.544	3.63
	3	3.17	4.334	5.273	6.018	6.613	7.101	7.51	7.86	8.163
	4	4	5.664	7.129	8.4	9.483	10.415	11.226	11.94	12.573
	5	4.644	6.691	8.565	10.267	11.788	13.145	14.361	15.458	16.455
m	6	5.17	7.523	9.725	11.778	13.666	15.395	16.979	18.436	19.782
	7	5.615	8.221	10.697	13.04	15.235	17.283	19.19	20.969	22.634
	8	6	8.824	11.53	14.12	16.577	18.898	21.097	23.152	25.102
	9	6.34	9.352	12.26	15.061	17.745	20.305	22.74	25.057	27.263

Table 2. The upper limit of end-to-end delay randomness (bits)

Randomness Analysis of End-to-End Delay in Equal Delay Forwarding Network

We have known that collision leads to the decrease of the randomness of end-to-end delay in RFNs and the Equal Delay Forwarding Network (EDFN) is the most collision-prone network theoretically, which is worth some analysis.

EDFN is defined as a forwarding network, in which the delay between nodes is approximately the same. In EDFN, for any node Z_i, there is no difference between forwarding to Z_{j1} or to Z_{j2}. From the symbolic point of view, Z_{j1} and Z_{j2} can rotate. As shown in Figure 9, let p denotes the self-forwarding probability of middle forwarding nodes and q denotes the forwarding probability between middle forwarding nodes.
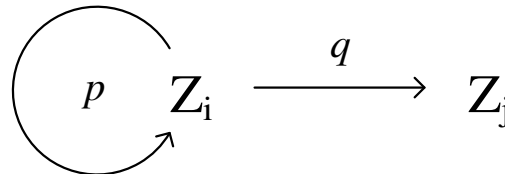
Figure 9. Optimal forwarding strategy for EDFN

For convenience, the delay between middle forwarding nodes is normalized to 1, then the forwarding matrix $\boldsymbol{P}$ of EDFN is

$$\boldsymbol{P} = \begin{pmatrix} p & qx & \cdots & qx \\ qx & p & \cdots & qx \\ \cdots & \cdots & \cdots & \cdots \\ qx & qx & \cdots & p \end{pmatrix} = (qx)\mathbf{1}\mathbf{1}^T + (p - qx)\boldsymbol{I}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{1}$ is the m dimensional vector of ones.

Therefore, for the EDFN with m nodes and N forwarding times, the end-to-end delay distribution polynomial $P(x)$ is

$$p(x) = \frac{1}{m}\mathbf{1}^T \boldsymbol{P}^N \mathbf{1} = (\frac{1}{m}\mathbf{1}^T \boldsymbol{P}\mathbf{1})^N = (p + (m-1)qx)^N$$

That is to say, the end-to-end delay of EDFN obeys binomial distribution, and the maximum entropy of binomial distribution is obtained at $p = 0.5$, so the optimal forwarding strategy for EDFN is

$$\begin{cases} p = 0.5 \\ q = \frac{1}{2(m-1)} \end{cases}$$

Then the end-to-end delay distribution polynomial $p(x)$ under the optimal forwarding strategy is

$$p(x) = \frac{1}{2^N}(1+x)^N = \frac{1}{2^N}\sum_{i=0}^{N} C_N^i x^i$$

So the end-to-end delay distribution of EDFN under the optimal forwarding strategy is $p(d = i) = \frac{1}{2^N}C_N^i$, and the maximum entropy of the end-to-end delay in EDFN is

$$H_{d\,max} = N - \frac{1}{2^N}\sum_{i=0}^{N} C_N^i \log_2 C_N^i$$

It can be found that the maximum entropy of EDFN is only related to the number of forwarding times N, and is irrelevant to the number of middle forwarding nodes m. What's worse, the maximum entropy of EDFN is 1 bit lower than the maximum entropy of RFNs with 2 nodes under the inevitable collision deployment. So, it just proves the conclusion that collision is the main reason for the decrease of randomness.

APPLICATION: USING THE RANDOMNESS OF END-TO-END DELAY TO GENERATE SYMMETRIC KEYS

Key generation needs random sources. The original key distribution channel tends to be unsafe, so the original key exchange is a difficult problem. One idea is using the key distribution center (KDC) to generate random numbers and then delay the key distribution through the secure key exchange protocol (D'Arco, 2001). In 1976, Diffie-Hellman proposed a key exchange scheme using discrete logarithm, but there is also a man-in-the-middle attack problem, and the security is rooted in the NP problem of discrete logarithm in the finite field on classical computers (Diffie and Hellman, 1976). The development of quantum computing has impacted the cryptography algorithm based on discrete logarithm problems. P.Shor has proved that there exist polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer (Shor, 1999).

Another way of thinking is to abandon the idea that the key is distributed by the center, and choose the scheme that both sides of the communication measure the channel to obtain reciprocity characteristics. This process does not need secret information exchange, so it avoids the risk that secret information eavesdrops. For example, the key is generated by using the frequency selective fading characteristic of the wireless channel, including measuring the received signal strength (RSS) (Awan et al., 2019), the channel impulse response (CIR) in time-frequency domain (Walther et al., 2019), and the phase (Zeinali and Hossein, 2016), delay and envelope of the received channel (Ye et al., 2010). The only problem is that the spatial distance between sender and receiver is limited in wireless channel key exchange, and the information exchange is mainly carried out by wire for the equipment with a far geographical distance. There is also a lot of randomness in RFNs, and the end-to-end delay, which is mainly studied in this paper, is an ideal feature that satisfies both long-term randomness and short-term reciprocity and can be used for key generation. So, this section mainly introduces how to use the randomness of end-to-end delay to generate symmetric keys.
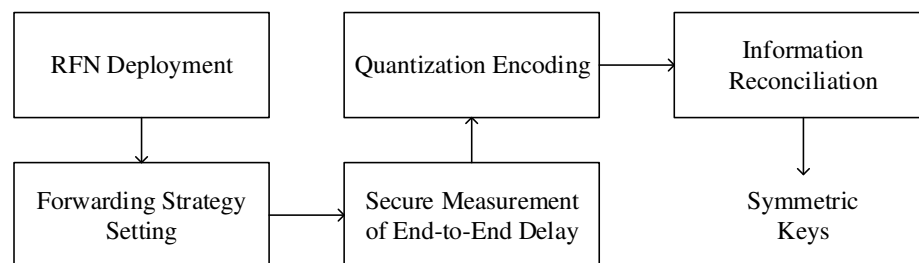
Figure 10. Key generation process based on RFNs

As shown in Figure 10, the whole process of symmetric key generation based includes RFNs deployment, forwarding strategy setting, secure measurement of end-to-end delay, quantization encoding, and information reconciliation. Each part is described in detail below.

RFNs Deployment RFNs can be applied in many scenarios, such as the large scenario of host group distributed between cities, or the small scenario of communication node cluster within the scope of LAN, especially in the scenario of encrypted communication needs between IoT device clusters. It is very convenient to generate the symmetric key with end-to-end delay. The deployment of RFNs mainly concerns two indicators, one is the number of middle forwarding nodes, the other is whether there is an evitable collision. The former affects the deployment cost, while the latter affects the efficiency of key generation.

The number of middle forwarding nodes is determined by the demand of the real scene key generation rate. From the perspective of the economy, we hope to achieve the highest key generation rate with the least number of nodes. For example, if the key generation rate of $r = 128$ bit/s is required, then suppose the average time $\bar{t}$ required for a single measurement is 100ms, a single measurement can generate at least 12.8bit key. From the data in Table 2, when the number of middle forwarding nodes $m = 5$ and the number of forwarding times $N = 6$, the key length is 13.145bits, which can meet the requirement. That is

308 to say, the key length is determined by $\frac{H_d}{\bar{t}} > r$, and the number of middle forwarding nodes is determined
309 by looking up Table 2.
310   The evitable collision can be checked by calculating $p(x)$. The number of inevitable collisions can be
311 obtained by calculating the symbol matrix $\boldsymbol{S}_m^N$ and counting the coefficients, and other like terms are all
312 evitable collisions. These evitable collisions can be avoided as far as possible by adjusting the deployment.

313 **Forwarding Strategy Setting**   When the RFNs network is deployed, the optimal forwarding strategy p
314 can be found through Table 1, and then the internode forwarding probability q can be calculated by $\frac{1-p}{m-1}$.
315 For the above example, the optimal forwarding strategy is $p = 0.086$ and $q = 0.2285$ for $(m = 5, N = 6)$.
316 Because of the rotation among nodes, the forwarding strategies set by each node are the same, which is
317 also very helpful in security, because attackers cannot identify forwarding nodes by counting forwarding
318 rules. Although the forwarding strategy seems to be static, the dynamically adjusted forwarding strategy
319 often divulges the information of the network itself, so that attackers can take advantage of it. When the
320 number of forwarding nodes or forwarding times changes, the forwarding strategy of deployed nodes can
321 be easily switched by looking up Table 1.

322 **Secure Measurement of End-to-End Delay**   The consistency of generated keys depends on the accurate
323 measurement of end-to-end delay (Fabini and Abmayer, 2013). In order to ensure that both sides of the
324 communication can measure approximately the same end-to-end delay and meet the security requirements,
325 we design a secure end-to-end delay measurement scheme as shown in Figure 11. The scheme steps are
326 as follows:
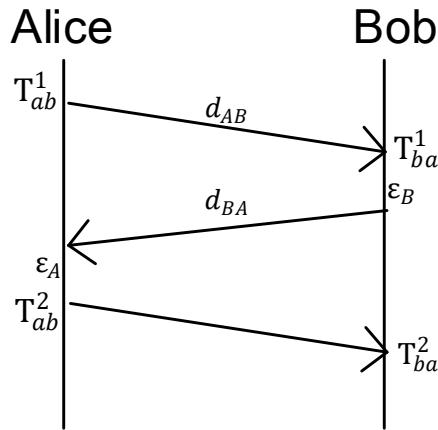
Figure 11. Secure end-to-end delay measurement scheme

327   1. Alice sends a request message to Bob and records the sending time T_{ab}^1,
328   2. Bob receives the request and records the receiving time T_{ba}^1, and send the reply package to Alice
329   with a delay of ε_B,
330   3. Alice receives the reply and sent it to Bob with a delay of ε_A. Then record the receiving time T_{ab}^2,
331   calculate the data transmission delay ΔT_{ab},
332   4. Bob receives the reply and records the receiving time T_{ba}^2, calculate the data transmission delay
333   ΔT_{ba}.

   Let d_{AB} denotes the end-to-end delay from Alice to Bob and d_{BA} denotes the end-to-end delay from
Bob to Alice. Then according to this scheme, Alice and Bob can calculate ΔT_{ab} and ΔT_{ba} as measurement
end-to-end delay as

$$\Delta T_{ab} = T_{ab}^2 - T_{ab}^1 = d_{AB} + d_{BA} + \varepsilon_A + \varepsilon_B$$
$$\Delta T_{ba} = T_{ba}^2 - T_{ba}^1 = d_{AB} + d_{BA} + \varepsilon_A + \varepsilon_B$$

334   Since $\Delta T_{ab} = \Delta T_{ba}$, the end-to-end delays measured by Alice and Bob are equal.

335   In terms of security, because each node only records the last hop node, Alice is anonymous in the
336   forwarding packet, and only Bob's information is in the forwarding packet, so it is impossible to measure
337   the end-to-end delay directly from the sending and receiving nodes. It is also difficult to obtain the end-to-
338   end delay by obtaining the forwarding route. Because the forwarding strategy is random, the probability
339   of each node in the next hop is the same, so it cannot be traced. To obtain a complete forwarding route,
340   the attacker needs to attack almost all forwarding nodes, which means that the cost of the attack is far
341   greater than the benefit. So in general, the security of the scheme is guaranteed.

342   **Quantization Encoding**   When we get the end-to-end delay data, we need to use quantization coding
343   technology to convert it into a key. We use nonlinear quantization, and the distribution of quantization
344   interval is consistent with that of end-to-end delay. Gray code is used in coding because Gray code
345   belongs to reliability coding, which is an error minimization coding method (Mecklenburg et al., 1973).
346   Another scheme is to encode the distribution of end-to-end delay by Huffman coding (Huffman, 1952),
347   and then make the nearest neighbor decision on the measured end-to-end delay and the theoretically
348   calculated possible value.

349   **Information Reconciliation**   An information reconciliation protocol is used to discard or correct the
350   difference of key bits generated by the sender and the receiver, which is a common method for key
351   agreement in physical layer security. Existing information reconciliation methods are mainly divided
352   into reconciliation protocols and error correction coding. The reconciliation protocols mainly include
353   BBBSS, Cascade ,and Winnow protocol. Error correction coding includes Hamming code, BCH code,
354   Turbo code, LDPC code, etc (Huth et al., 2016). Of course, if the process of information reconciliation
355   causes information leakage, then privacy amplification is needed to discard some leaked bits (Maurer and
356   Wolf, 2003).

357   In Purple Mountain Laboratory of Nanjing, we design a symmetric key generation system according
358   to the application introduced in this section (Huang et al., 2021). The practical results show that this
359   scheme is effective. According to our statistics, the key agreement rate of sending and receiving can be
360   over 91%, which can meet our communication needs.

CONCLUSIONS

362   This paper studies the randomness of end-to-end delay in Random forwarding networks (RFNs) through
363   the problem of drunks returning home. In this paper, we solved six problems in the study of end-to-end
364   randomness in RFNs. By establishing a mathematical model, we solved the first problem of what kind
365   of distribution does end-to-end delay obey by deriving the formula Eq. (1) for calculating the random
366   distribution of end-to-end delay; Then the second question of how to measure the randomness of end-
367   to-end delay was answered by analyzing the end-to-end delay generation model, and the conclusion is
368   that the randomness of end-to-end delay can be quantitatively measured by information entropy; In the
369   process of answering the third question of what is the reason for decline of the randomness of end-to-
370   end delay, we found that the end-to-end delay collision is the main reason, among which the evitable
371   collision can be solved by adjusting RFNs deployment, while the inevitable collision can not be avoided;
372   Then, we proposed a fast algorithm FPSSM (Algorithm 1) for calculating inevitable collisions by using
373   symbolic matrix and solved the optimization problem of maximizing the randomness of end-to-end
374   delay to answer the fourth and fifth questions of what is the upper limit of end-to-end delay and how to
375   reach the upper limit. We gave the flow of solving the optimization problem in detail, and then gave the
376   optimization results in Table 1: the upper limit of the randomness of end-to-end delay and Table 2: the
377   optimal forwarding strategy; Finally, we introduced the application of symmetric key generation based on
378   end-to-end delay randomness to answer the final question of how to use the RFNs to share keys.

ACKNOWLEDGMENTS

380   I would like to thank my supervisor Prof. Huang for this interesting research direction, the research
381   environment of Purple Mountain Laboratory, and the lab team for their great help.

REFERENCES

Abdelkefi, A. and Jiang, Y. (2011). A structural analysis of network delay. In *2011 Ninth Annual Communication Networks and Services Research Conference*, pages 41–48.

Awan, M. F., Kansanen, K., Perez-Simbor, S., Garcia-Pardo, C., Castelló-Palacios, S., and Cardona (2019). Rss-based secret key generation in wireless in-body networks. In *2019 13th International Symposium on Medical Information and Communication Technology (ISMICT)*, pages 1–6.

Bellare, M., Canetti, R., and Krawczyk, H. (1996). Keying hash function for message authentication. *Lecture Notes in Computer Science*, 1109:1–15.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Convex Optimization.

Choi, B. Y., Moon, S. B., Zhang, Z. L., Papagiannaki, K., and Diot, C. (2004). Analysis of point-to-point packet delay in an operational network. In *Joint Conference of the IEEE Computer & Communications Societies*.

D'Arco, P. (2001). On the distribution of a key distribution center. In *Italian Conference on Theoretical Computer Science*.

Diffie, W. and Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654.

Duan, Q., Al-Shaer, E., and Jafarian, H. (2013). Efficient random route mutation considering flow and network constraints. In *2013 IEEE Conference on Communications and Network Security (CNS)*, pages 260–268.

Engelmann, A., Zhao, S., and Jukan, A. (2014). Improving security in optical networks with random forwarding and parallel transmission. In *GLOBECOM 2015 - 2015 IEEE Global Communications Conference*.

Fabini, J. and Abmayer, M. (2013). Delay measurement methodology revisited: Time-slotted randomness cancellation. *IEEE Transactions on Instrumentation and Measurement*, 62(10):2839–2848.

Huang, J., Wang, X., Wang, W., Duan, Z., and Zhu, Y. (2021). A novel key distribution scheme based on transmission delays. *Security and Communication Networks*, 2021.

Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101.

Huth, C., Guillaume, R., Strohm, T., Duplys, P., Samuel, I. A., and Güneysu, T. (2016). Information reconciliation schemes in physical-layer security: A survey. *Computer Networks*, 109:84–104. Special issue on Recent Advances in Physical-Layer Security.

Li, S. and Kim, J. G. (2015). Maximizing the lifetime of wireless sensor networks with random forwarding. *AEU - International Journal of Electronics and Communications*, 69(1):455–457.

Liu, M., Jin, Y., and Yang, T. (2013). Research-on load balance in distributed network measurement system. In *2013 5th IEEE International Conference on Broadband Network Multimedia Technology*, pages 25–29.

Maurer, U. and Wolf, S. (2003). Secret-key agreement over unauthenticated public channels .ii. privacy amplification. *IEEE Transactions on Information Theory*, 49(4):839–851.

Mecklenburg, P., Pehlert, W., and Sullivan, D. (1973). Correction of errors in multilevel gray coded data. *IEEE Transactions on Information Theory*, 19(3):336–340.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*.

Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *Siam Review*, 41(2):303–332.

Syverson, P., Tsudik, G., Reed, M., and Landwehr, C. (2001). Towards an analysis of onion routing security. *Springer Berlin Heidelberg*.

Sánchez, J., Urquiza-Aguiar, L., Paredes, M., and Osorio, D. (2020). Survey on physical layer security for 5g wireless networks. *Annals of Telecommunications*.

Walther, P., Franz, E., and Strufe, T. (2019). Blind synchronization of channel impulse responses for channel reciprocity-based key generation. In *2019 IEEE 44th Conference on Local Computer Networks (LCN)*, pages 76–83.

Ye, C., Mathur, S., Reznik, A., Shah, Y., Trappe, W., and Mandayam, N. B. (2010). Information-theoretically secret key generation for fading wireless channels. *IEEE Transactions on Information Forensics and Security*, 5(2):240–254.

Zeinali, Vajihe, K. B. and Hossein (2016). Shared secret key generation protocol in wireless networks based on the phase of mimo fading channels. *Wireless Pers Commun*, 89:1315–1334.

437 Zhang, J., Li, G., Marshall, A., Hu, A., and Hanzo, L. (2020). A new frontier for iot security emerging
438   from three decades of key generation relying on wireless channels. *IEEE Access*, 8:138406–138446.