Hindawi
Security and Communication Networks
Volume 2021, Article ID 3125820, 13 pages
https://doi.org/10.1155/2021/3125820

Research Article

A Novel Key Distribution Scheme Based on Transmission Delays

Jie Huang,[1,2] **Xiaowen Wang**,[1] **Wei Wang**,[1] **and Zhenyu Duan**[1,2]

[1]*School of Cyber Science and Engineering, Southeast University, Nanjing 210096, China*
[2]*Purple Mountain Laboratories, Nanjing 211111, China*

Correspondence should be addressed to Xiaowen Wang; xiaowenwang@seu.edu.cn

Received 29 September 2021; Accepted 10 November 2021; Published 30 November 2021

Academic Editor: Youwen Zhu

With the development of IoT (Internet of Things), the demand for security is increasing day by day. However, the traditional key distribution scheme is high in cost and complicated in calculation, so a lightweight key distribution scheme is urgently needed. In this paper, a novel key distribution scheme based on transmission delay is proposed. Based on the experimental observation, we find that the statistical characteristics of their transmission delays are about the same if any two terminals transmit the equal-length packets on the Internet and are different for different transmission paths. Accordingly, we propose a method to customize transmission delays. On the Internet, we have deployed 7 forwarding hosts. By randomly determining the forwarding path of packets, we can get customized transmission delay sets. Then, these sets are processed, respectively, by correcting outlier, normalizing, quantizing, encoding, and reconciling so as to be able to realize key distribution between two sides. Next, we design a key distribution protocol and a key distribution system, which consists of a Management Center, a Packet Forwarding Network, and Users. Finally, we reason the security of the key distribution protocol with formal analysis tools.

1. Introduction

The key is an important part of modern cryptography. For a long time, its distribution method has been one of the research focuses in secure communication, especially the distribution of symmetric keys. As we know, compared with the asymmetric encryption methods, the symmetric encryption methods have the advantage that the encryption and decryption are very fast, making them very suitable for transmitting large amounts of secure data. But either the sender or the receiver must first create a key, send it to the other party, and then encrypt the message to be sent. The traditional key distribution schemes involved face-to-face meetings, sending the key through an existing secure channel, of use of a trusted third party. The first is neither often practical nor always safe, so we mostly use the two latter.

However, with the development of Internet of Things technology and IPv6 technology, a large number of terminal devices, even smart devices, are connected to the Internet. They not only participate in network communications but also generate and transmit large amounts of data which need to be kept secret in many scenarios. So the key distribution technology plays an important role in protecting device-to-device communications, but the traditional key distribution methods cannot adapt to this change. Our goal in this paper is to find a low-cost and secure key distribution method. To this end, we not only comprehensively analyze the advantages and disadvantages of traditional key distribution schemes but also compare two new types of key distribution technologies, namely, quantum key distribution and key distribution based on wireless channel characteristics.

1.1. Key Distribution Based on Symmetric Cryptography. Needham and Schroeder first proposed the idea of key distribution center (KDC) [1]. In the scheme, the initial keys shared by the communicating participants and KDC are used to realize the session key distribution between the two principles. The security of the session key is guaranteed by encrypting it with the initial key. Although the number of the initial keys is far less than that of the session keys, there is still the problem of how to distribute these initial keys (also known as the "key establishment problem" [2]). And KDC destroys the confidentiality of the session key because it generates the key. Reference [3] proposed a method that exchanges a password between a user and a system. Instead of storing the user's password x, the system stored only the value $F(x)$, where F denotes a one-way function. To prevent an intruder from eavesdropping on the password, users encrypt the password by successive applications of F. Although the password is not transmitted in clear text during user authentication, it is still a problem how to share it between the user and the system when the password is generated. EKE protocol [4] uses a combination of asymmetric (public key) and symmetric (secret key) cryptography. A secret key, such as a password, is used to encrypt a randomly generated public key to exchange secret information between two parties, such as a session key, similar to the Kerberos protocol [5]. Their common feature is that two parties must share a secret message before the agreement goes.

SecurID authentication infrastructure [6, 7] was developed by SDTI/RSA, which now is known as RSA Security company. The main idea is the combination of a pseudorandom token code which is cyclically generated by a handheld authorization device, with the pin of the owner so that the subscriber can access a system. The user does not need to transmit the pseudorandom code to the system. The core of the scheme is the proprietary SecurID hash function. However, this function has never been made public. Reference [8] introduced a key distribution scheme based on artificial neural networks to prevent attackers from obtaining a key copy. Both partners use tree parity machines (TPMs) with the same structure. Each TPM starts with randomly chosen weights which are kept in secret. Each participant just knows the internal representation of his TPM. After the full synchronization, each party can use the weight vectors as the common secret key. In this process, both participants do not transmit any secret information, but they are vulnerable to man-in-the-middle attacks. Any attackers can use the TPM with the same structure as the participants to generate the same key, but any participants cannot identify it. At the same time, if a user needs to communicate confidentially with n users, then this user needs to save n TPMs with different structures.

1.2. Key Distribution Based on Asymmetric Cryptography. For a long time, public-key cryptography has been regarded as the ultimate solution to realize the symmetric key distribution. Diffie–Hellman algorithm [9] is the first public-key algorithm, and its security derives from the NP problem of solving the discrete logarithm over a finite field. DH algorithm is only used for key distribution, not for encrypting or decrypting information, nor for digital signature. And the algorithm cannot prevent man-in-the-middle attacks. For example, when Alice and Bob use the algorithm to exchange a session key, the attacker Eve intercepts the messages between the two parties and uses the private keys $Xe1$ and $Xe2$ generated randomly by himself to replace the private keys Xa and Xb generated by Alice and Bob, respectively. After completing the key exchange, both Alice and Bob think that they have shared a session key K. But in fact, Alice and Eve shared the key $K1$, and B and Eve shared the key $K2$. RSA algorithm [10] or elliptic curve algorithm [11, 12] with a digital certificate [13] or public-key distribution authority [14] can perfectly solve the problem of distributing keys. The user's public key and subject are signed by a Certificate Authority (CA) in a digital certificate. The sender uses the receiver's public key to encrypt the secret key and then sends it to the receiver through the public channel. Only the receiver can decrypt and obtain the secret key. The scheme in [15] is similar to them. Although these schemes can prevent man-in-the-middle attacks, the service cost is high. So its application scope is limited, such as device-to-device secure communication in IoT.

1.3. Quantum Key Distribution. With the relentless growth of computational power, public-key algorithms, which depend on NP problems to ensure security, are facing severe challenges. Once quantum computers [16] become available, public-key cryptosystems will no longer be computational security. Therefore, in 1984, physicist Bennett and cryptologist Brassard combined quantum physics with cryptography and first proposed a quantum key distribution (QKD) protocol, namely, BB84 protocol [17], which used a single photon with four polarization states for the encoding. In contrast to the above cryptographic methods that rely on NP problems, the security of QKD protocol is based on physical laws, that is, the measurement principle of quantum mechanics. This protocol utilizes the immeasurable property of quantum to realize secure key distribution. B92 protocol [18], which uses two polarization states instead of four, is the simplified version of the BB84 protocol. Ekert extended Bennett and Brassard's original ideas. He proposed a new protocol that uses the entanglement of photons for encoding and decoding instead of polarization state [19]. Quantum secret sharing [20–22] is a way that combines quantum cryptography with classical secret sharing. By using multiphoton entanglement, it is unnecessary to perform the classical secret-splitting procedure. The system with these QKD protocols is termed discrete variable QKD (DVQKD) systems. However, the polarization of single photons cannot be encoded and decoded efficiently because of the technological limitations of quantum devices. Therefore, continuous variable QKD (CVQKD) was proposed. Compared with DVQKD, CVQKD can enable higher key distribution rates in metropolitan areas. In a CVQKD system, the information is encoded in continuous variables by a Gaussian modulation utilizing the position or momentum quadrature of coherent quantum states [23–25]. Although Zhang's test [25] shows that quantum key can be effectively transmitted up to 50 kilometers, the CVQKD system is still unpractical in some scenarios because of its high cost, such as device-to-device communication in IoT.

1.4. Key Distribution Based on Wireless Channel Characteristics. Key distribution based on channel characteristics, such as signal strength (RSS), channel phase, channel impulse response (CIR), or channel state information (CSI), is to exploit the inherent randomness and reciprocity of communication channels to share secret key or information between two participants. In this kind of scheme, the computational resources or network parameter knowledge will not be limited for any eavesdropper [26–31]. The key elements of [26] were the information reconciliation and privacy amplification procedures. After then, Maurer and Wolf extended the secret key sharing analysis of [26] to account for the presence of an active eavesdropper [30]. Korapaty relied on the independence and randomness of wireless communication channels between the sender/receiver and the sender/eavesdropper to use the phase of the fading coefficient as the secret key [32]. In wireless channels, multipath propagation can provide a physical resource for generating a secret key [33]. Each pairwise wireless link can generate a secret key by estimating and quantizing the shared channel phases coefficients. In the scheme, the large number of random degrees of freedom in wideband wireless channels was exploited for generating longer secret keys. Ye et al. first proposed a scheme based on level crossings of the fading process, which is well suited for the Rayleigh and Rician fading models associated with a richly scattering environment [34]. Then, by observation from quantizing jointly Gaussian processes, they exploited empirical measurements to set quantization boundaries and a heuristic log-likelihood ratio estimate to improve secret key generation rate.

Although the above schemes make up for the shortcomings of traditional key distribution schemes and realize low-cost and decentralized key distribution, they cannot achieve end-to-end key distribution because one end cannot measure the channel characteristics of the other end.

1.5. Our Approach and Contributions. In this paper, we propose a novel secret key distribution scheme based on time delay and develop a secret key distribution protocol and system. Finally, we formally analyze the security of the protocol.

(1) Based on the experimental observation, we found that the statistical characteristics of their transmission delay are stable when packets with equal length are exchanged between any two hosts on the Internet, and different transmission paths have different delays. Accordingly, we propose a novel key distribution scheme with transmission delay.

(2) In order to realize the key distribution between any devices on the Internet, we have developed a key distribution protocol and system. In this system, we can generate random delays between two sides by selecting randomly the packet transmission route. Then, we encode the delays into a key by their reciprocity.

(3) We reason the security of the protocol with formal analysis tools. We first assume that attackers can only eavesdrop on all information on both sides. And we establish an attack model accordingly. Then, we formally reason that the proposed protocol can resist man-in-the-middle attacks.

The rest of this paper is organized as follows. In Section 2, we analyze the physical characteristics of the Internet and study its data transmission mechanism. By analyzing the delay collected continuously, we conclude that the statistical delay of the equal-length packets exchanged by any two ends on the Internet is stable. In Section 3, in order to generate random delays, we design different data transmission paths for each key agreement by deploying packet forwarding nodes on the Internet. We proposed a novel secret key distribution scheme in Section 4. In the scheme, we first correct outliers in the delay set and then normalize, quantize, encode, and reconcile them in turn. In order to realize the scheme, we develop a key distribution protocol and system in Section 5. In Section 6, we establish the attackers' eavesdropping model and assume their capabilities; that is, the attacker can eavesdrop on all the messages of both sides but cannot modify them. And the security of the protocol is formally analyzed by BAN logic. We conclude in Section 7.

2. Data Transmission Mechanism on the Internet

On the Internet, the main physical characteristics include throughput, channel utilization, rate, and time delay, which are the main indicators for evaluating Internet performance. The first three physical characteristics are inherent on the Internet. Once the network is built, the maximum throughput, channel utilization, and rate of each link will hardly change, which are only affected by the physical performance of the network equipment. At the same time, the terminal device cannot measure the throughput, channel utilization, or rate of all links on the transmission route. But the time delay characteristic is different, it is not only related to the physical performance of the network equipment but also affected by many factors, such as transmission path, and it is easy to measure. Therefore, in this paper, we choose the time delay characteristic to study a novel key distribution method on the Internet.

On the Internet, the data packets from the source to the terminal need to go through n links. Each link may have different branches. At this time, the routing protocol can help data packets choose different transmission paths. Therefore, from a microscopic point of view, the transmission route for each packet may be different. But we know that the router does not work blindly; it is to find an optimal transmission path for each data packet and then effectively transmit it to the terminal. For example, the OSPF protocol, which is a kind of unicast routing protocol and used mainly by HUAWEI routers and CISCO routers, is an internal gateway protocol (IGP). The protocol is used in a single autonomous system (AS) to exchange routing information between gateways, as shown in Figure 1. In AS 1, according to the OSPF protocol, two adjacent routers form an adjacency relationship by sending link-state information to each other, then calculate their respective routes by the shortest path algorithm, and store them in the OSPF routing table. Then OSPF routes are compared with the routes generated by other routing protocols, and the optimal route is added to the global routing table. Although the working principles of

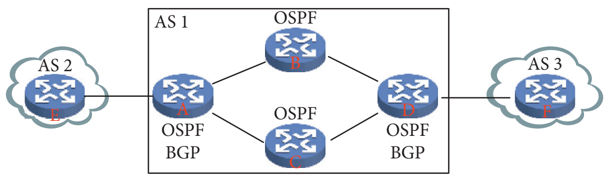

Figure 1: OSPF working principle.

RIP, BGP, and ISIS protocols are different from those of OSPF protocol, the routing information ultimately stored in global routing table must be optimal among all routing information. Therefore, on the Internet, the data packets are always forwarded through the globally optimal path. We estimate that network equipment on the Internet will not be replaced for a long period of time, so the global routing information is macroscopically unchanged. And if the location of the source and the terminal is fixed, the statistical delays of the two hosts are about the same when they exchange equal-length packets to each other. In order to verify our ideas, we deploy two hosts in Beijing and Shanghai of China and exchange packets with an equal length between them. The statistical delays of both sides are shown in Figure 2.

From Figure 2, we can find that in a very short time, the time delay of sending equal-length packets to each other is different. But through long-term observation, the fluctuation range of the statistical delays is so small that they can be almost regarded as the same. So there are several mutation points in the delays. We analyze that these outliers are mainly caused by network congestion. When the transmission delay of a certain link in the network is too large, the packets will choose another transmission path. But once it is restored, the packets will be transmitted from this link. During the entire experiment, these outliers did not affect the reciprocity of the transmission delays between the two sides. And we can correct them by other means. From this experiment, we can draw the following conclusions:

(1) From a macropoint of view, once the network is deployed, the packet transmission route between any two ends is hardly changed.

(2) On the Internet, if both sides exchange equal-length packets to each other, their statistical delay is about the same.

Based on the above conclusions, in Section 3, we will customize the transmission route for data packets on the Internet and obtain a random transmission delay by artificially changing the forwarding path of packets, which is the basis of key agreement.

3. Customizing Transmission Delay

In this paper, we assume that the source host first sends a packet to the destination host, and the destination host returns it immediately after receiving the packet. Once the source receives the sent packet from the destination, it immediately sends the packet to the destination again, and the destination receives the packet again, which is called one round of transmission. In each round of transmission, the transmission delay of the sender is defined as the time difference between the first bit sent by the source and the first bit received by the source, and the transmission delay of the receiver is defined as the time difference between the first bit sent by the destination and the first bit received by the destination. In order to realize customizable transmission delay, we first use OMNet++ simulation software to develop a simulation project. Its network structure is shown in Figure 3, where node A denotes Alice, node B denotes Bob, and the other 29 nodes represent different hosts in different geographic locations, which are responsible for forwarding packets. We assume that, among these 29 forwarding nodes, any two nodes can communicate with each other. Alice is the initiator of communication. Before Alice sends a packet, she first determines the forwarding times and chooses randomly the starting node of forwarding. In this simulation experiment, we exploit the concept of reciprocity in signal processing; that is, in the case of a single stimulus, reciprocity means that the response does not change due to the exchange of the excitation port and the response port. If A and B have the same transmission delay when they exchange packets of equal length with each other through the same forwarding node, we call that the transmission delay of A and B is reciprocal.

The working principle of this simulation project: before Alice sends a data packet, she first determines the number of packet forwarding hops and randomly selects the starting node to be forwarded. Then Alice begins to send a certain packet. She records the start time T_{ab}^1 when she sends the first bit. After receiving the packet, the forwarding node will randomly select the next hop node, and the number of hops will be automatically reduced by 1. When the hop count is zero, the packet will be sent directly to Bob. Bob receives the packet and immediately returns it to Alice through the same forwarding path. He will record the time T_{ba}^1 when he sends the first bit of the packet. Next, Alice receives the packet and immediately sends it to Bob again through the same forwarding path. When Alice receives the first bit of the packet,

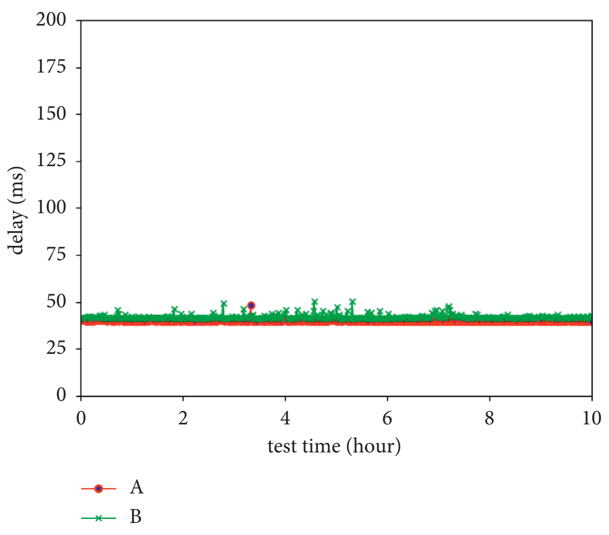

Figure 2: Transmission delays with the same packet.

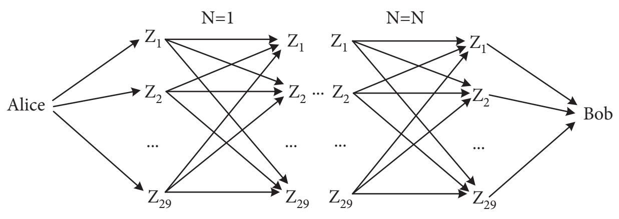

Figure 3: Simulating network structure.

she will record the time T^2_{ab}. Once receiving the first bit of the packet again, Bob records the time T^2_{ba}. After a round of communication, Alice and Bob can calculate the transmission delay $\Delta T_{ab} = T^2_{ab} - T^1_{ab}$ and $\Delta T_{ba} = T^2_{ba} - T^1_{ba}$, respectively. In the entire communication process, Alice and Bob do not know the complete transmission path, which ensures that attackers cannot trace the transmission path of the packet.

In this simulation, we generated 100 groups of transmission delays, as shown in Figure 4. Each group contains 2 values, namely, Alice's transmission delay and Bob's transmission delay $(\Delta T^i_{ab}, \Delta T^i_{ba})$. In this figure, the ordinate of each point is the mean value of the two rounds of transmission delay so as to eliminate the influence of transmission path fluctuations on the transmission delay as much as possible. In order to be able to simulate the real network environment, we randomly add a constant to each link as the delay caused by the physical characteristics of the link.

The simulation results show that the transmission delay of Alice is highly correlated with that of Bob, and the correlation coefficient is 99.96%. Moreover, the transmission delay of the two sides has strong randomness. Therefore, on the Internet, we can try to establish a custom forwarding network in which packets will be randomly forwarded so as to generate random transmission delays between the two parties. Accordingly, we deploy seven packet forwarding hosts in different places, including Beijing, Hangzhou, Zhangjiakou, Shenzhen, Chengdu, Qingdao, and Singapore. The network structure is shown in Figure 5. Among the seven hosts, any two hosts are linked to each other. Alice and Bob are two terminals on the Internet. The experimental steps are as follows:

Step 1. After Alice determines Bob's identity, she establishes a communication route with Bob and starts the key agreement.

Step 2. Alice selects a standard packet and packet forwarding times, N. Then, she determines the first forwarding node of the packet by the selection algorithm of forwarding nodes.

Step 3. After the first forwarding node receives the packet, N is automatically reduced by 1. And it randomly selects the next forwarding node from the remaining six nodes and sends the packet.

Step 4. Repeat Step 3 until $N = 0$. Then the last forwarding node directly sends the packet to Bob.

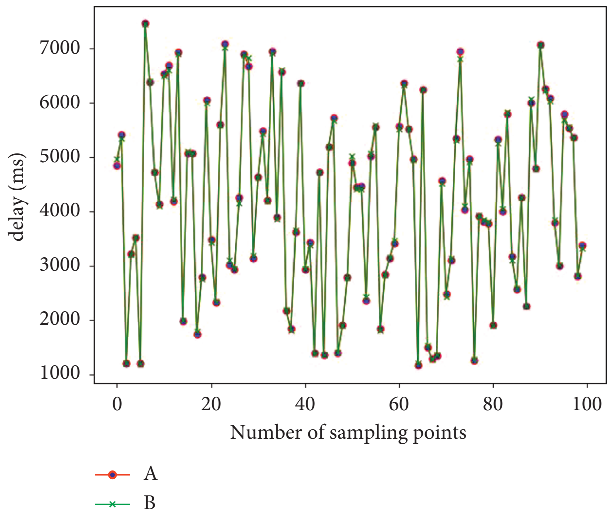

Figure 4: Reciprocity simulation of transmission delays.

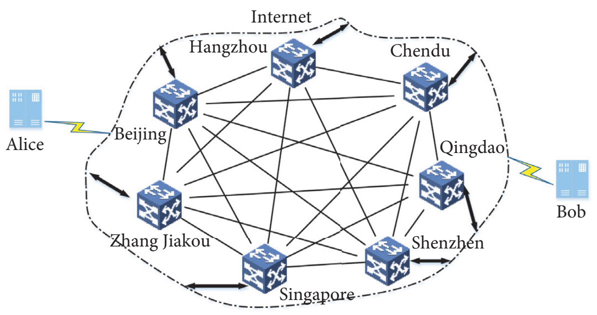

Figure 5: Packet forwarding network with 7 nodes.

Step 5. After Bob receives the packet, he immediately returns it to Alice. When Alice receives the packet, the transmission delay ΔT^i_{ab} is calculated.

Step 6. Similarly, Bob calculates the transmission delay ΔT^i_{ba};

Step 7. Repeat from Step 2 to Step 6 until all the transmission delays required by both sides are extracted.

We continuously collected 2000 groups of transmission delay for Alice and Bob and calculated their Pearson correlation coefficient, and the result was 88.4%. For the convenience of observation, we randomly selected 100 groups from 2000 groups and drew a line chart, as shown in Figure 6. We can intuitively see that, on the Internet, as long as the forwarding sequence and the number of packet forwarding hosts are the same, the transmission delays of both sides are about the same. Since in each round of transmission, Alice will choose a different packet forwarding path and the number of forwarding nodes, these transmission delays are random. Although the correlation coefficient of these transmission delays is reduced by 11.5% compared with the simulation experiment, it will not affect the conclusions in Section 2.

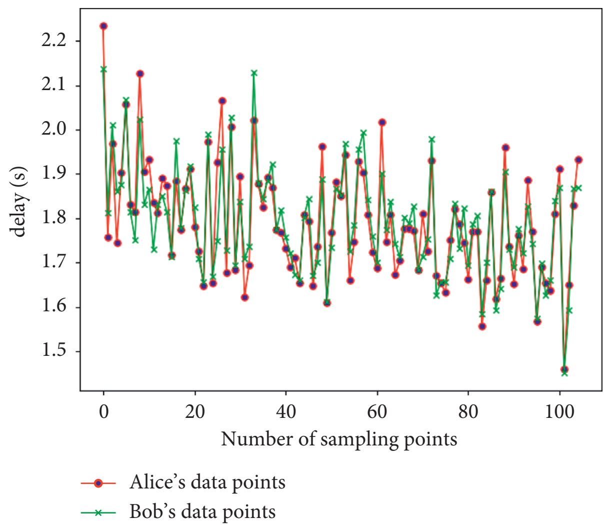

Figure 6: Transmission delays of Alice and Bob.

We analyze that the main reasons for the decrease of the correlation coefficient are as follows:

(1) Routers may change the packet forwarding path due to a sudden increase in traffic or even congestion on a certain link.

(2) There is a forwarding host in Singapore outside of China. Due to cross-border forwarding the packets through the host, even if their routes are the same, there will be some uncertainty in the transmission delay.

From the experiment, we can draw the following conclusion: although there are a few outliers, because of the high correlation of the transmission delay between both sides, we can realize the key agreement between both sides by the reciprocity of the transmission delays. But the questions now are as follows: (1) how to correct the outliers between both sides; (2) how to convert the transmission delays into keys.

4. Secret Key Distribution Scheme Based on Transmission Delays

4.1. Correcting Outliers. There are many ways to discriminate outliers in a dataset. In this paper, we will exploit the Chauvenet criterion to recognize outliers. The main reason is that we can correct outliers by the Chauvenet criterion without the complex calculations, such as sorting and iterating the transmission delays many times, and the amount of the transmission delays is not limited, so it is highly efficient and widely applicable.

The Chauvenet criterion is a strict gross error discrimination criterion based on the equal confidence probability. When we can determine a probability range of all the samples in a dataset by centering on a mean value, all the data outside of the range are regarded as outliers and to be corrected. If the absolute value of the difference between a measured value and the mean value is greater than the product of the standard deviation and the Chauvenet coefficient, the measured value is determined as containing gross errors, as shown in formula (1); otherwise, it is normal.

$$\left|x_i - \overline{x}\right| > sw, \tag{1}$$

$$s = \left[\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2\right]^{1/2}, \tag{2}$$

where x_i denotes the measured value, $\overline{x}$ denotes the mean value of the dataset, s denotes the standard deviation of the dataset, w denotes the Chauvenet coefficient, and n denotes the number of elements in the dataset.

The Chauvenet coefficient can be obtained by the look-up table, but it is not conducive to calculation on a computer. Here, we will exploit the empirical formula to calculate the Chauvenet coefficient, as shown in the following formula:

$$w = 1 + 0.4\ln(n). \tag{3}$$

Under formulas (1)–(3), the outliers of the transmission delays in Section 3 were corrected. After then, we calculated the Pearson correlation coefficient of the transmission delays of Alice and Bob, which result was 96.6%. For the convenience of observation, we randomly selected 100 groups from the corrected data set and drew the corrected transmission delay diagrams of Alice and Bob, as shown in Figure 7.

In Figure 7, although the consistency of transmission delays is poor at some points, the consistency of most points is very good. Therefore, we are able to use transmission delays to generate keys for both sides. But we must correct the different bits of transmission delays to generate the same key for both sides.

4.2. Key Generation Scheme Based on Transmission Delay Characteristics. After the above work, the dataset for the generating key is ready. In this section, we will process the dataset to generate the same key for both sides so as to realize key distribution on the Internet without sharing secret information between both sides or distributing keys by a trusted third party for both sides. Next, we need to normalize, quantize, encode, and reconcile the corrected dataset in turn, as shown in Figure 8.

4.2.1. Normalization. Due to the randomness of the transmission delays between Alice and Bob, the range of measured delays varies greatly, which makes it difficult to quantify these data. Therefore, we need to normalize these data, as shown in the following formula:

$$x_{\text{nor}}^{i} = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}, \tag{4}$$

where x_i denotes the measured value, $x_{\min}$ denotes the minimum value in the dataset, and $x_{\max}$ denotes the maximum value in the dataset.

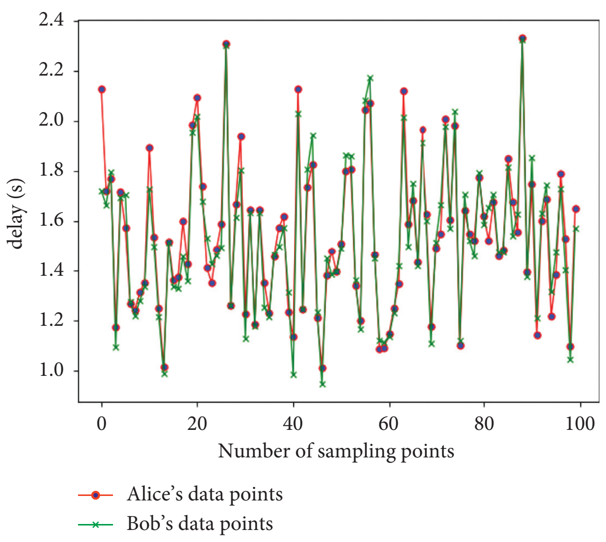

FIGURE 7: Corrected transmission delays of Alice and Bob.

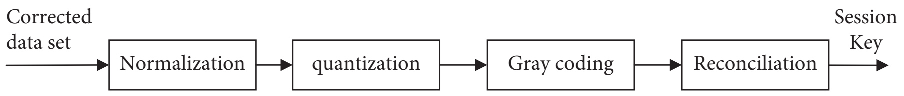

FIGURE 8: Key generation process based on transmission delays.

4.2.2. Quantization. The normalized data is uniformly quantized to obtain discrete measurement value q_i, as shown in the following formula:

$$q_i = \left[x_{\text{nor}}^i \times 2^R\right], \tag{5}$$

where [ ] denotes taking-integer operator and R denotes the order of quantization.

After these transmission delays are normalized, if $x_{\text{nor}}^i = 1$, then $q = 2^R$. But if $x_{\text{nor}}^i = 0$, then $q = 0$. So the transmission delays after quantization are not beyond the range of $[0, 2^R - 1]$.

4.2.3. Gray Coding. After these transmission delays are quantized, a set of Gray codes with a scale of 2^R is generated. The q value corresponds to the code words in the Gray code array in turn, and the range is $0\sim2^R - 1$, thus completing the mapping from the quantized data to the Gray code. For example, we generate a set of Gray codes with a scale of 2^4, that is, ['0000', '0001', '0011', '0010', '0110', '0111', '0101', '0100', '1100', '1101', '1111', '1110', '1010', '1011', '1001', '1000']. Then these quantized data are mapped to the corresponding Gray code in the order of $0\sim2^4 - 1$.

4.2.4. Information Reconciliation. Information coordination is a form of the key error correction performed by Alice and Bob to ensure that the two keys are the same, similar to channel error correction codes. When the consistency rate p of their keys is less than 85%, it is impossible to correct channel error.

After Alice and Bob encode the transmission delays, respectively, they need to evaluate the consistency rate p of these data. In this section, we focus on two important parameters that affect the encoding consistency rate, namely, the order R of the quantization degree and the number of the packet forwarding N.

In order to generate random transmission delays, the forwarding times N and the forwarding paths are different each time when the packet is forwarded in the PFN. First, we assume that the quantization order R is set to 2, 4, 8, and 16, respectively, and the range of the forwarding times N is $(0, N_{\max}]$. As shown in Figure 9, we plot four curves respectively, and the abscissa represents the maximum forwarding times $N_{\max}$. Each curve shows that the consistency rate p changes with the range of the forwarding times. From this figure, we can find that p will increase as $N_{\max}$ increases. And the randomness of transmission delay is also better. When R is greater than 4, p will be less than 85%, and then the transmission delay cannot be corrected. So it is impossible to reconcile information. However, the larger R is, the more the number of bits corresponding to the quantized data is and the longer the key that can be generated is. For example, Alice and Bob generate 1000 groups of transmission delays, and each group with the same code is regarded as valid data. The valid bits $b = 1000 \times 0.916 \times 2 = 1832$ when $R = 2$ and $b = 1000 \times 0.836 \times 4 = 3344$ when $R = 4$.

Secondly, we assume that the forwarding times are constant and variable, respectively. And the range of forwarding times N is [5, 10], [5, 15], [5, 20], and [5, 25], respectively. As shown in Figure 10, we plot five curves, and the abscissa represents the quantization order R. From this figure, we can find that the consistency of the transmission delays is the worst when N is constant. As the range of forwarding times increases, the consistency of transmission delay becomes better and better. But with the increase of quantization order, the consistency of transmission delay will become worse.

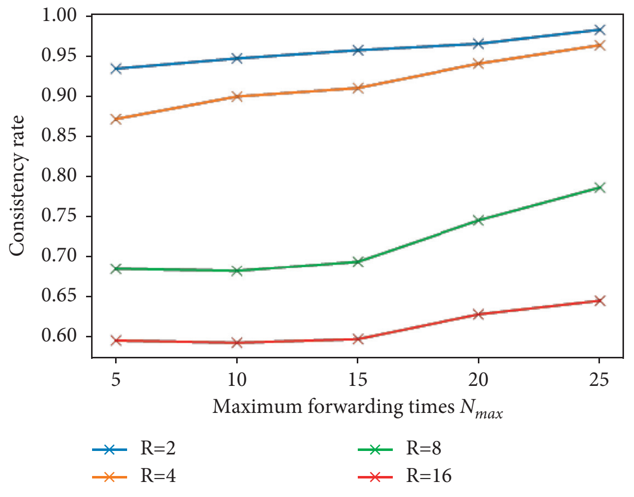

FIGURE 9: Maximum forwarding times $N_{\max}$ versus p with different R.

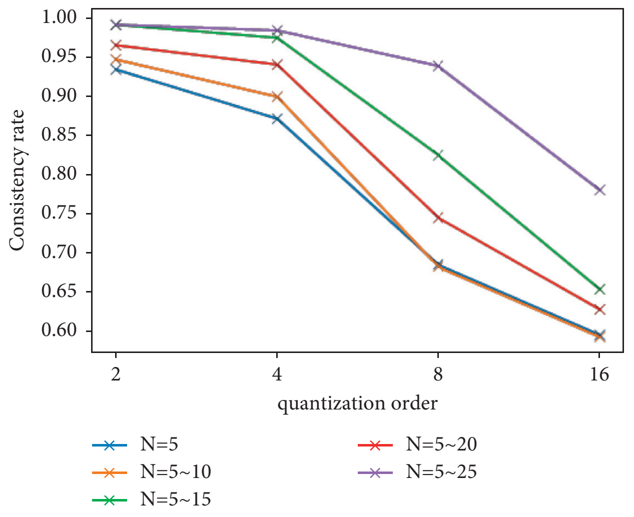

FIGURE 10: Quantization order R versus p with different N.

In summary, the bitstreams of Alice and Bob's transmission delays are inconsistent in some bits, so it is not feasible to exploit these binary bits to generate keys. Therefore, we need to reconcile these different bits, that is, to correct the inconsistent bits in the bitstreams of transmission delays. However, there will be a problem in this process: how to prevent information leakage. So we will use (2, 1, 3) convolutional code, in which input is t bits and output is $2t$. The information reconciliation process is as follows:

(1) Alice and Bob generate the bitstreams of transmission delays K_A and K_B, respectively.

(2) Alice generates a convolutional code c by an encoder.

(3) Alice calculates $T_A = K_A \oplus c$ and transmits T_A to Bob on the public network.

(4) Bob calculates $c' = K_B \oplus K_A \oplus c$ after he receives T_A. Since the inconsistent bits of c' and c are the same as that of K_A and K_B, it can be considered that c' is the result of c being interfered with by the channel. Then, c can be recovered from c' by the error correction performance of the convolutional code. And we can calculate K_A from $K_A \oplus c \oplus c$.

After the bitstreams are reconciled, Alice and Bob generate the same key by them. Then, both sides can use the key to realize secure communication.

5. Key Distribution System

5.1. System Design. In Section 4, we propose a scheme of how to generate random transmission delays by using forwarding nodes. When an equal-length packet is transmitted in the forwarding network, both sides will generate a pair of delays with reciprocity. After the pair of delay is encoded, two binary strings with high consistency probability will be generated. By using channel coding technology, both sides can generate exactly the same key. However, the scheme cannot be used directly. This is because a key must be generated between both sides who trust each other, and the process must be able to resist attacks from the network. So we develop a key distribution system (KDS), as shown in Figure 11.

In Figure 11, $\Longleftrightarrow$ denotes a secure channel; $\cdot - \cdot -$ denotes a state transmission channel, through which the status information of the forwarding nodes is transmitted to the MC; — denotes a key agreement channel, through which users negotiate keys; - - - denotes a link between forwarding nodes.

The system consists of a Management Center (MC), a Packet Forwarding Network (PFN), and a User group. MC is responsible for user registration, processing user's key agreement requests, and collecting the status information from the forwarding nodes. Only users who have registered in MC can negotiate keys with each other. In this system, MC is similar to a trusted third party, so we assume that it is secure. MC can be either a server or an organization. No attacker can obtain secret information from MC. At the same time, a trusted Certificate Authority (CA) issued a digital certificate for MC, through which any entity on the network can verify the identity of MC.

The user can be any smart terminal or device on the network. Each user has an independent IP address through which other users can find it. However, users do not know the location of any forwarding node on the network nor the real-time status of each forwarding node. When two registered users negotiate a key, one of them only submits a key agreement request to MC, and then MC redirects it to the first forwarding node to start the key agreement. If any user applies for a key agreement many times, the first forwarding node may be different each time. Once the key is generated, both sides can communicate confidentially.

PFN consists of forwarding nodes connected in pairs. In order to ensure the randomness of transmission delays, any two forwarding nodes should not be deployed in the same city. The more the number of forwarding nodes, the larger the range of random delays. In this system, only the MC knows the locations of all forwarding nodes and is able to transmit the status information of each forwarding node to other nodes at regular intervals. After the forwarding node

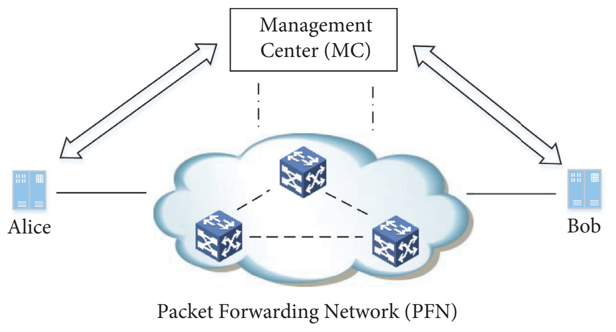

Figure 11: Key distribution system.

obtains this information, it calculates a random forwarding strategy, which contains the probability of forwarding to each possible next hop node and randomly selects the next hop node according to this probability. Although the selection of the next hop node is random, any forwarding node cannot choose itself as the next hop node. In order to trace back, each forwarding node must store the IP address of the previous node instead of that of all forwarding nodes on the forwarding route. If a forwarding node fails to connect to the Internet or to be a malfunction, it will be removed from the PFN. When the node is restored, it can be automatically added to the PFN.

5.2. Key Distribution Protocol. In KDS, only registered users can negotiate keys with each other, so users participating in key negotiation must first register on the MC. As shown in Figure 12, the user registration steps are as follows:

Step 1. The user first verifies MC's identity. MC has a digital certificate, which includes a validity period, MC's subject, and a public key, issued by a trusted CA. After obtaining the certificate from the MC, the user first determines whether the certificate is valid and then verifies whether the subject is consistent with the MC identity. If the identity of the MC is confirmed, then we have the following step.

Step 2. The user submits his identity information to the MC through a secure channel, such as an SSL/TSL secure channel.

Step 3. After receiving the message, the MC must verify whether the information is authentic and matches the identity of the submitter. If yes, then we have the following step.

Step 4. The MC responds to the user with an ACK message.

Step 5. The user submits the login username (UN) and password (PW) to MC through the secure channel. The UN is public, and the registered user can query UN corresponding to other users in the MC. But the password is confidential and must not be disclosed to any third party.

Step 6. The MC generates a matching identification $\mathrm{ID_X}$ for each user, where $_\mathrm{X}$ represents the user. For example, Alice's identification is $\mathrm{ID_A}$. And the identification must be kept secret.

In the KDS system, user registration and key agreement are two completely separate processes. Once the registration is successful, the user can use the KDS system to negotiate the key, but it does not need to be executed immediately. Different from user registration, a key agreement requires four entities, including two users, one MC and one PNF, to participate. In order to generate a shared key between any two users and resist man-in-the-middle attacks, we will adopt the protocol shown in Figure 13.

When Alice wants to communicate with Bob in secret, they must have the same encryption/decryption key, so they will take the following steps to complete the key agreement.

Step 1. Alice submits a request for the key agreement with Bob to the MC. Alice links to the MC by a security protocol such as HTTPS and submits its UN and PW to the MC. The MC verifies this information to determine whether to allow Alice to log in. If the verification is yes, Alice immediately submits the key agreement request with Bob to the MC. And Alice will generate key K_1 by using his own PW.

Step 2. After the MC receives the request from Alice, it will generate a random number N, key K_1, and K_2. K_1 is the symmetric key between MC and Alice, and K_2 is the symmetric key between MC and Bob. The MC uses the key generation function to convert Alice and Bob's PW into symmetric keys K_1 and K_2, respectively. These two symmetric keys can only be shared between the user and the MC, and no third party knows the information of these keys.

Step 3. The MC, respectively, uses K_1 and K_2 to encrypt $\mathrm{ID_A} \oplus \mathrm{ID_B}$, UN, and N, namely, E (K_1, $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_B}$, N) and E (K_2, $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, N), and then transmits them to Alice.

Step 4. Alice decrypts E (K_1, $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, N) and obtains $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, and N after he receives E (K_1, $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_B}$, N) and E (K_2, $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, N). At the same time, she transmits E (K_2, $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, N) to Bob via a public channel.

Step 5. Bob receives E (K_2, $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, N). He will generate key K_2 by using his own PW. Then he decrypts and obtains $\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, and N. If Bob agrees to negotiate the secret key with Alice, he sends an agree-negotiation response to MC; otherwise, he sends a refuse-negotiation response.

Step 6. The MC receives the response from Bob. If Bob agrees with the key negotiation with Alice, the MC will randomly select an entry node of the PFN, send the IP address of the node to Alice, and then go to Step 7. Otherwise, he will disconnect from Alice, and the protocol will be terminated.

Step 7. Alice will be redirected to the entry node of the PFN after she receives the IP address. Then, the PFN starts to forward packets randomly. Assuming that there is no secure channel between Alice and PFN, any attacker can eavesdrop on Alice's input and output information but is not able to eavesdrop on any

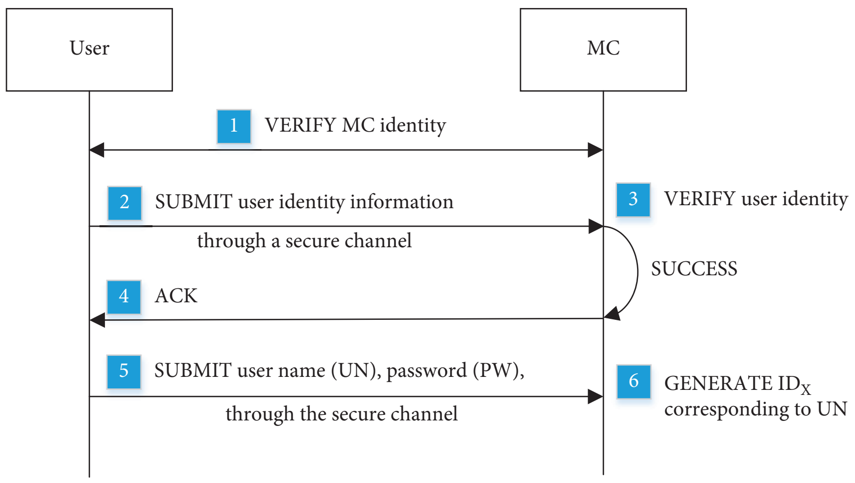
User
MC
1 VERIFY MC identity
2 SUBMIT user identity information through a secure channel
3 VERIFY user identity
SUCCESS
4 ACK
5 SUBMIT user name (UN), password (PW), through the secure channel
6 GENERATE ID_X corresponding to UN

Figure 12: Users registration process.

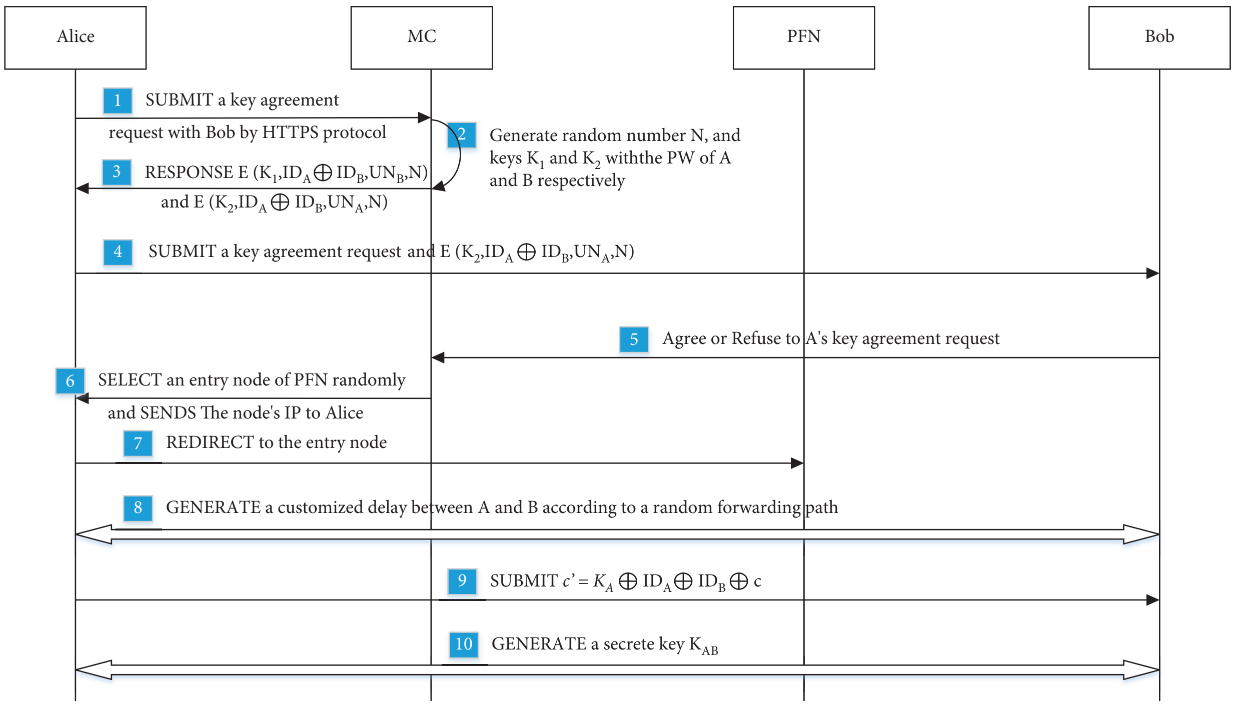
Alice
MC
PFN
Bob
1 SUBMIT a key agreement request with Bob by HTTPS protocol
2 Generate random number N, and keys K1 and K2 withthe PW of A and B respectively
3 RESPONSE E (K1,IDA ⊕ IDB,UNB,N) and E (K2,IDA ⊕ IDB,UNA,N)
4 SUBMIT a key agreement request and E (K2,IDA ⊕ IDB,UNA,N)
5 Agree or Refuse to A's key agreement request
6 SELECT an entry node of PFN randomly and SENDS The node's IP to Alice
7 REDIRECT to the entry node
8 GENERATE a customized delay between A and B according to a random forwarding path
9 SUBMIT c' = KA ⊕ IDA ⊕ IDB ⊕ c
10 GENERATE a secrete key KAB

Figure 13: Key distribution protocol.

information between any two forwarding nodes in the PFN.

Step 8. According to the randomly selected packet forwarding times, the customized transmission delays are generated between Alice and Bob. In the PFN, the next hop node is randomly selected. Even for the same pair of users, the next hop node is different for each key agreement. After the same packet is exchanged between Alice and Bob many times, Alice and Bob, respectively, generate a set of transmission delays. According to the scheme in Section 4, they calculate K_A and K_B, respectively.

Step 9. Alice uses an encoder to generate a convolutional code c and then calculates $c' = K_A \oplus \mathrm{ID}_A \oplus \mathrm{ID}_B \oplus c$, where $\mathrm{ID}_A \oplus \mathrm{ID}_B$ and c are secret. Alice transmits c' to Bob via a public channel.

Step 10. Bob uses the reconciliation scheme in Section 4 to get the value of K_A. Next, a secret key K_{AB} is generated between A and B. Then, they will use K_{AB} to realize secure communication.

In this protocol, although Alice and Bob generate symmetric keys K_1 and K_2 with MC, respectively, they cannot be used for key agreement between Alice and Bob. This is because the session key between them cannot be generated by MC; otherwise, the confidentiality of the session key will be destroyed; that is, the third party knows the session key, which violates the key security principle [9].

6. Security Analysis

6.1. Attack Model. It is assumed that there is an attacker on the Internet who attempts to destroy the confidential communication between Alice and Bob by intercepting the session key between Alice and Bob. We refer to such actions as man-in-the-middle attacks, which are launched by eavesdropping on the communication traffic between Alice and Bob. Accordingly, we propose an attack model as shown in Figure 14, which has the following two assumptions:

(1) The attacker Eve can eavesdrop on all input and output traffic of Alice and Bob but cannot modify these messages. Here, we consider the worst situation where Eve can generate a set of delays by eavesdropping on Alice or Bob's traffic, but it does not guarantee that it is exactly the same as their datasets.

(2) In the PFN, Eve cannot eavesdrop on all forwarding nodes and track the packet forwarding path. If Eve intercepts the forwarded packets in the PFN, the next hop forwarding node will find it immediately and report it to the MC, and the delay data will be deleted from the dataset.

In this paper, we do not consider DoS attack. Because the KDS system will be paralyzed, and the process of the key agreement will be suspended once Eve launches the DoS attack. This is not the problem that we consider and need to solve in this paper.

6.2. Formal Security Analysis. In order to determine whether the key distribution protocol is secure, we will utilize the formal analysis tools. Among them, BAN logic [35, 36] is the most typical one which is a modal logic based on belief. In the process of reasoning using BAN logic, the belief of the specific protocol principles will change gradually with the exchange of messages. When analyzing a specific protocol by using BAN logic, all messages of the protocol will be converted into formulas in BAN logic.

6.2.1. Syntax and Semantics of BAN Logic

P, Q: the interaction principle.

K: the shared key between principles.

X, Y: statements in general.

(X, Y): conjunctions of X and Y.

$P|\equiv X$: P believes X, and the principal P trusts the message X is true.

$P\lhd X$: P sees X, P receives a message containing X, and P can read and repeat X.

$P|\sim X$: P said X. The principal P sometimes sends a message including X. There are two implications: one is that the message X is sent by P; namely, P is the message source; the other is that the principal P can confirm and recognize the message X and interpret the message X correctly.

$P\Rightarrow X$: P has jurisdiction over X.

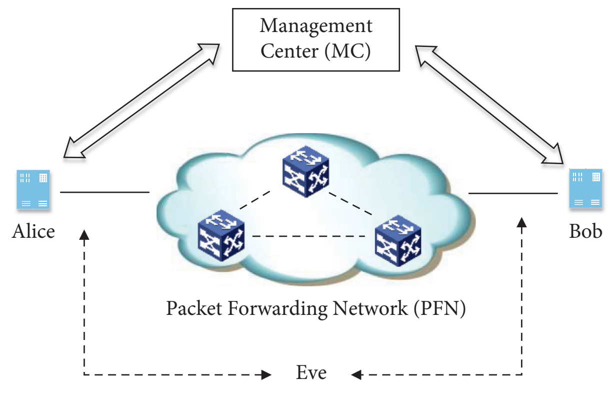

FIGURE 14: Attack model.

$\#(X)$: the statement X is fresh; that is, X has not been sent in a message at any time before the protocol is executed.

$P\overset{K}{\leftrightarrow}Q$: P and Q may use the shared key K to communicate with each other. The key K is unique and will never be discovered by any principle except P or Q or a third principle trusted by either P or Q.

$\{X\}_K$: this represents the statement X encrypted under the key K.

6.2.2. Rules of BAN Logic. Message meaning rules:

Rule 1. If $P|\equiv P\overset{K}{\leftrightarrow}Q$, $P\lhd\{X\}K$, then $P|\equiv Q|\sim X$.

Nonce verification rule:

Rule 2. If $P|\equiv\#(X)$ and $P|\equiv Q|\sim X$, then $P|\equiv Q|\equiv X$.

Jurisdiction rule:

Rule 3. If $P|\equiv Q\Rightarrow X$ and $P|\equiv Q|\equiv X$, then $P|\equiv X$.

Decryption rule:

Rule 4. If $P|\equiv P\overset{K}{\leftrightarrow}Q$ and $P\lhd\{X\}K$, then $P\lhd X$.

Freshness rules:

Rule 5. If $P|\equiv\#(X)$, then $P|\equiv\#(X, Y)$.

6.2.3. Analyzing the Key Distribution Protocol with BAN Logic. The initial belief assumptions are as follows:

Assumption 1: Alice$|\equiv$UN$_B$. The principal Alice trusts the message UN$_B$ is true.

Assumption 2: Bob$|\equiv$UN$_A$. The principal Bob trusts the message UN$_A$ is true.

Assumption 3: Alice$|\equiv$RK$_{AB}$. Alice trusts the message RK$_{AB}$ is true, where RK$_{AB}$ represents a key agreement request which Alice sends to MC.

Assumption 4: Alice$|\equiv\#$(RK$_{AB}$). Alice trusts the message that RK$_{AB}$ is fresh is true.

Assumption 5: Alice$|\equiv$Alice$\overset{K_1}{\leftrightarrow}$MC. Alice trusts the message that Alice and MC may use the shared key K_1 to communicate with each other is true.

Assumption 6: Bob|≡Bob$\overset{K_2}{\leftrightarrow}$MC. Bob trusts the message that Bob and MC may use the shared key K_2 to communicate secretly with each other is true.

Assumption 7: Bob|≡MC⇒$\mathrm{RK_{AB}}$. Bob trusts the message that MC has jurisdiction over $\mathrm{RK_{AB}}$ is true.

Assumption 8: Alice|≡MC⇒ Bob|≡$\mathrm{RK_{AB}}$. Alice trusts the message that MC has jurisdiction over Bob|≡$\mathrm{RK_{AB}}$ is true.

Assumption 9: Alice|≡#(N). Alice trusts the message that N is refreshed is true.

Assumption 10: Bob|≡#(N). Bob trusts the message that N is refreshed is true.

Assumption 11: Bob|≡Alice$\overset{ID_A \oplus ID_B}{\leftrightarrow}$Bob. Bob trusts the message that Alice and Bob may use $\mathrm{ID_A} \oplus \mathrm{ID_B}$ to communicate secretly with each other is true.

Logic reasoning steps are as follows:

Under Step 2 and Step 3 in the key distribution protocol, we have the following:

MC⟶Alice: {IDA ⊕ IDB, UNB, N}K1,{IDA ⊕ IDB, UNA, N}K2.

Under Rule 4 and Assumption 5, we have the following:

Alice⊲(IDA ⊕ IDB, UNB, N).

Under Rule 5 and Assumption 9, we have the following:

Alice|≡#(IDA ⊕ IDB, UNB).

Under Rule 1 and Assumption 1, we have the following:

Alice|≡MC|~ $\mathrm{RK_{AB}}$.

Under Rule 2 and Assumption 4, we have the following:

Alice|≡MC|≡$\mathrm{RK_{AB}}$.

Step 4 in the key distribution protocol is Alice ⟶ Bob: $\{\mathrm{ID_A} \oplus \mathrm{ID_B}, \mathrm{UN_A}, \mathrm{N}\}_{K2}$. Under Rule 4 and Assumption 6, we have the following:

Bob⊲($\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$, N).

Under Rule 1 and Assumption 2, we have the following:

Bob|≡Alice|≡$\mathrm{RK_{AB}}$ and Bob|≡MC|~ $\mathrm{RK_{AB}}$.

Under Rule 2, Rule 5, and Assumption 10, we have the following:

Bob|≡MC|≡$\mathrm{RK_{AB}}$ and Bob|≡#($\mathrm{ID_A} \oplus \mathrm{ID_B}$, $\mathrm{UN_A}$).

Under Step 5 in the key distribution protocol, if Bob agrees to Alice's key agreement request, he sends an agree-negotiation response $\mathrm{ACK_{AB}}$ to MC by HTTPS protocol. Accordingly, under Rule 1, we have the following:

MC|≡Bob|~$\mathrm{ACK_{AB}}$.

Once Alice receives the redirection message under Step 6 in the key distribution protocol, we have the following:

Alice|≡MC|≡Bob|≡$\mathrm{RK_{AB}}$.

Under Rule 3 and Assumption 8, we have the following:

Alice|≡Bob|≡$\mathrm{RK_{AB}}$.

Under Rule 1 and Assumption 11, we have the following:

Bob|≡Alice|~$\mathrm{K_A} \oplus \mathrm{ID_A} \oplus \mathrm{ID_B} \oplus$ c.

As shown in the above analysis, the key agreement protocol proposed in this paper can resist man-in-the-middle attacks in Figure 14.

7. Conclusions

It is an inspiring idea to realize low-cost and secure key distribution on the Internet. In this paper, a novel key distribution method is proposed based on transmission delays. Based on the experimental observation, we find that the statistical characteristics of transmission delays are about the same if the equal-length packets are transmitted between any two principles on the Internet. Accordingly, we deployed 7 devices as forwarding hosts in different cities to generate customized transmission delays between Alice and Bob. We collect 2000 groups of delays by exchanging the same packet between Alice and Bob many times. The Pearson correlation coefficient calculated from these data is 88.4%. Therefore, it is possible to negotiate the key exploiting the transmission delay characteristics. After then, we correct outliers, normalize, quantize, and encode these transmission delays in turn. The coding consistency rate p of the delays is more than 91% when the order of quantization $R = 2$; the receiver can correct the encoded delays with the error correction code. So we use convolutional code to reconcile the encoded delays so as to generate the same binary stream, which can generate a session key for the two sides. Finally, we design the key distribution protocol and the key distribution system and reason that the protocol can resist man-in-the-middle attacks.

Data Availability

All the data are available at https://github.com/wxwled/data-for-HJ_paper.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (no. 2018YFB2100400).

References

[1] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," *Communications of the ACM*, vol. 21, no. 12, pp. 993–999, 1978.

[2] M. E. Hellman, "An overview of public key cryptography," *IEEE Communications Society Magazine*, vol. 16, no. 6, pp. 24–32, 1978.

[3] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[4] S. M. Bellovin and M. Merritt, "Encrypted key exchange: password-based protocols secure against dictionary attack," in *Proceedings of theIEEE Computer Society Conference on Research in Security and Privacy*, pp. 72–84, Oakland, CA, USA, May 1992.

[5] B. C. Neuman and T. Ts'o. Kerberos, "An authentication service for computer networks," *IEEE Communications Magazine*, vol. 32, no. 9, pp. 33–38, 1994.

[6] A. Biryukov, J Lano, and B. Preneel, "Cryptanalysis of the alleged SecurID hash function," in *Proceedings of the 10th Annual International Workshop on Selected Areas in Cryptography*, Ottawa, Canada, August 2003.

[7] N. Sklavos, C. Efstathiou, and S.I. D. authenticator, "On the hardware implementation efficiency," in *Proceedings of the*

IEEE International Conference on Electronics, Circuits, and Systems, pp. 589–592, Marrakech, Morocco, December 2007.

[8] I. V. Anikin, A. Z. Makhmutova, and O. E. Gadelshin, "Symmetric encryption with key distribution based on neural networks," in *Proceedings of the 2nd International Conference on Industrial Engineering, Applications and Manufacturing*, Chelyabinsk, Russia, May 2006.

[9] W Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[10] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[11] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[12] V. S. Miller, "Use of elliptic curves in cryptography," in *Proceedings of theAdvances in Cryptology – CRYPTO'85*, pp. 417–426, New York, NY, USA, 1986.

[13] Nist, *RFC 5280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, IETF, 2008.

[14] G. J. Popek and C. S. Kline, "Encryption and secure computer networks," *Computing Surveys*, vol. 11, no. 4, pp. 331–356, 1979.

[15] M. Beunardeau, F. E. Orche, D. Maimut, D. Naccache, P. B. Rønne, and P. Y. A. Ryan, "Authenticated key distribution: when the coupon collector is your enemy," *Lecture Notes in Computer Science*, pp. 1–20, 2020.

[16] IBM Ces, "IBM q system one is the world's first commercial quantum computer system," 2019, https://indianexpress.com/article/technology/gadgets/ces-2019-ibm-q-system-one-is-the-worlds-first-integrated-quantum-computer-5529844/.

[17] C. H. Bennett and G. Brassard, "Quantum cryptography: public key distribution and coin tossing," in *Proceedings of theProcess International Conference on Computer*, pp. 175–179, System & Signal Processing, Bangalore, India, December 1984.

[18] C. H. Bennett, "Quantum cryptography using any two nonorthogonal states," *Physical Review Letters*, vol. 78, no. 21, pp. 3121–3124, 1992.

[19] A. K. Ekert, "Quantum cryptography based on Bell's theorem," *Physical Review Letters*, vol. 67, no. 6, pp. 661–663, 1991.

[20] M. Hillery, V. Buzek, and A. Berthiaume, "Quantum secret sharing," *Physical Review*, vol. 59, no. 3, pp. 1829–1834, 1998.

[21] S. Kartick and O. Hari, "Efficient quantum secret sharing without a trusted player," *Quantum Information Processing*, vol. 19, no. 2, pp. 1–15, 2020.

[22] T. Y. Wang, X. X. Wang, X. Q. Cai, C. Y. Wei, and R. L. Zhang, "Analysis of efficient and secure dynamic quantum secret sharing protocol based on Bell states," *Quantum Information Processing*, vol. 20, no. 1, pp. 1–10, 2021.

[23] F. Grosshans, G. V. Assche, J. Wenger, R. Brouri, N. J. Cerf, and P. Grangier, "Quantum key distribution using Gaussian-modulated coherent states," *Nature*, vol. 421, pp. 238–241, 2003.

[24] L. Gyongyosi, L. Bacsardi, and S. Imre, "A survey on quantum key distribution," *Infocommunications Journal*, vol. 11, no. 2, pp. 14–21, 2019.

[25] Y. Zhang, Z. Li, Z. Chen et al., "Continuous-variable QKD over 50km commercial fiber," *Quantum Science and Technology*, vol. 4, no. 3, pp. 1–12, 2019.

[26] U. Maurer, "Secret key agreement by public discussion from common information," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 733–742, 1993.

[27] G. Y. Li, J. B. Yu, and A. Q. Hu, "Research on physical-layer security based on device and channel characteristics," *Journal of Cryptologic Research*, vol. 7, no. 2, pp. 224–248, 2020.

[28] J. B. Yu, A. Q. Hu, and C. M. Zhu, "RF Fingerprinting extraction and identification of wireless communication devices," *Journal of Cryptologic Research*, vol. 3, no. 5, pp. 433–446, 2016.

[29] A. C. Polak and D. L. Goeckel, "Wireless device identification based on RF oscillator imperfections[C]," in *Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2679–2683, IEEE, Florence, Italy, May 2014.

[30] A. Mukherjee, S. A. A. Fakoorian, J. Huang, and A. L. Swindlehurst, "Principles of layer security in multiuser wireless networks: a survey," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 3, pp. 1550–1573, 2014.

[31] L. Peng, G. Li, J. Zhang, R. Woods, M. Liu, and A. Hu, "An investigation of using loop-back mechanism for channel reciprocity enhancement in secret key generation," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 507–519, 2019.

[32] H. Koorapaty, A. A. Hassan, and S. Chennakeshu, "Secure information transmission for mobile radio," *IEEE Communications Letters*, vol. 4, no. 2, pp. 52–55, 2000.

[33] A. Sayeed and A. Perrig, "Secure wireless communications: secret keys through multipath," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3013–3016, Las Vegas, NV, USA, April 2008.

[34] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. B. Mandayam, "Information-theoretically secret key generation for fading wireless channels," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 240–254, 2010.

[35] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *Proceedings of the Royal Society, Series A*, vol. 426, no. 1871, pp. 233–271, 1989.

[36] M. Abadi and M. R. Tuttle, "A semantics for a logic of authentication," in *Proceedings of the Annual ACM Symposium on Principles of Distributed Computing*, pp. 201–216, Toronto, Canada, August 1991.