Hindawi
Security and Communication Networks
Volume 2017, Article ID 2761486, 16 pages
https://doi.org/10.1155/2017/2761486

Research Article

A Security and Efficient Routing Scheme with Misbehavior Detection in Delay-Tolerant Networks

Feng Li,[1] Yali Si,[2,3] Ning Lu,[1] Zhen Chen,[2] and Limin Shen[2]

[1] *Computer and Communication Engineering College, Northeastern University at Qinhuangdao, Qinhuangdao, China*
[2] *School of Information Science and Engineering, Yanshan University, Qinhuangdao, China*
[3] *School of Liren, Yanshan University, Qinhuangdao, China*

Correspondence should be addressed to Ning Lu; luning@neuq.edu.cn

Received 19 August 2017; Accepted 15 October 2017; Published 26 November 2017

Academic Editor: Lianyong Qi

Due to the unique network characteristics, the security and efficient routing in DTNs are considered as two great challenges. In this paper, we design a security and efficient routing scheme, called SER, which integrates the routing decision and the attacks detection mechanisms. In SER scheme, each DTNs node locally maintains a one-dimensional vector table to record the summary information about the contact with other nodes and the trust degree of other nodes. To obtain the global status and the contact relationship among all nodes, the trusted routing table consisting of vectors of all nodes is built in each DTNs node. The method for detecting malicious nodes and selfish nodes is proposed, which exploits the global summary information to analyze the history forwarding behavior of node and judge whether it is a malicious node or selfish node. The routing decision method is proposed based on trust degree of forwarding messages between nodes, which adopts trust degree as relay node selection strategy. Simulation results show that compared with existing schemes SER scheme could detect the attacks behavior of malicious nodes and selfish nodes, at the same time, with higher delivery rate and lower average delivery delay.

1. Introduction

Delay-tolerant networks (DTNs) refer to a new form of self-organizing networks that is envisioned to support communication in case of failure or no preexisting infrastructure, such as interplanetary communication networks in space areas, high-speed vehicular networks that disseminate the city traffic information, and sensor networks in extreme environment [1, 2]. Different from the traditional wireless networks, DTNs are the challenging networks characterized by open medium, long delay and frequent disruption, and lack of fixed and guaranteed end-to-end communication links [3]. DTNs make use of the store-carry-and-forward strategy to forward the message packets (also named bundle) when two nodes appear with contact opportunity. This routing strategy requires that each node in DTNs can cooperate with other nodes and is willing to help with forwarding.

However, DTNs are threatened by various attacks, because some nodes will behave selfishly and may not be willing to help others forward messages in order to conserve their limited resources (e.g., power and buffer), and even some nodes controlled by adversary will behave maliciously and may launch black hole, grey hole, or DoS attacks against the networks by dropping all or part of the received message packets, maliciously tampering message packets, or producing an enormous number of fake message packets [4–7]. The recent researches show that these selfish or malicious nodes would significantly degrade the routing performance of DTNs, resulting in low delivery rate and poor forwarding efficiency of messages and high average delivery delay. Due to the lack of continuous path and centralized management in DTNs, the detection of these attacks is more difficult. Therefore, how to effectively resolve the selfishness problem and defense against attacks of malicious nodes, improving the routing performance, has become a very challenging issue to design security and efficient routing protocol that combines defense technique.

To achieve the better routing performance of DTNs, many routing protocols in DTNs have been proposed in [8–12].

The purpose of routing in DTNs is to select the proper relay nodes to forward messages and improve the routing performance. Most of the existing routing protocols use historical encounter information or social relations as the decision of predicting relay nodes, which can effectively forward the message to the destination node. However, these routing schemes are inefficient in the DTNs environment with malicious or selfish nodes. To mitigate the impact of selfish or malicious nodes on DTNs, some detection attacks schemes are proposed in [3, 6, 7], which make use of the history forwarding evidences and encounter records of each node to analyze its forwarding behavior. However, these detection schemes are independent of specific routing protocols and require more computing capability, network bandwidth, and storage resources to work well. Due to the limited resources (e.g., power, buffer, and bandwidth) and intermittent connectivity in DTNs, we need an efficient routing scheme with misbehavior detection, which could not only improve the routing performance, but also detect malicious nodes and selfish nodes effectively.

In this paper, we propose a security and efficient routing scheme (SER) to improve message forwarding performance and detect malicious attacks. Different from existing routing schemes and malicious attacks detection schemes that work independently, respectively, we integrate the routing decision and attacks detection mechanisms into the trusted routing table. In SER scheme, each DTNs node maintains a one-dimensional vector to record the summary information about the contact with other nodes. The summary information includes the encounter history evidences with other nodes, the evidences of messages of sending to or receiving from other nodes, and trust degree that represents the ability of other nodes to forward messages to it. To form a global view and obtain the contact relationship among all nodes, SER introduces the trusted routing table that consists of vector of each node. In the initial phase, the trusted routing table of each node has only its own vector; when the two nodes meet, they would exchange the trusted routing table with each other and update it by comparing with the received trusted routing table. Therefore, each node could obtain a global view of the previous network connectivity from its trusted routing table. Based on the trusted routing table, we design the routing decision method and malicious attacks detection mechanisms. The main contributions of this paper include the following three parts.

First, we introduce in detail the method of generating and updating the trusted routing table. The proposed method could not only ensure the security and reliability of trusted routing table, but also make the trusted routing table converge quickly to global consistency.

Second, to accurately evaluate the trust degree of the node, we propose a method of forwarding evidence collection based on layered coin model and digital signature mechanism. The forwarding evidences signed by nodes are bound dynamically on message during the relay processes, and the message carries evidences chain to the destination node. The proposed forwarding evidences collection method greatly improves the timeliness and reliability of the evidences collection and effectively reduces the network overhead.

Third, we propose a routing decision method based on trust degree, which could deliver messages to the destination node along the direction of trust gradient increment and improve effectively routing performance of DTNs. Moreover, the malicious attacks detection method is proposed based on the history evidences of trusted routing table, which could detect selfish nodes and malicious nodes effectively.

The remainder of this paper is organized as follows: In Section 2, we present the system model and design goals and some attacker models. In Section 3, we explained in detail the implementation of SER scheme. In Section 4, simulation results and analysis of SER are introduced. Section 5 highlights related work. Finally, we summarize the conclusions of our works in Section 6.

2. System Model and Design Goals

2.1. System Model. In this paper, we adopt the system model similar to the literature [13, 14]. We consider DTN with no trusted authorization center, which is different from [7] in that it exists with a periodically available TA (trusted authority). Each node is equipped with a wireless communication device to communicate with its one-hop neighbor node when the two nodes contact with each other. Each node has a unique ID identifier N_i and the corresponding public/private key pair sk_i/pk_i that is allocated when it first joins the DTNs. When two nodes first encounter, they would exchange their public key certificate. After a period of time, each node in DTNs may obtain public key certificates of the other nodes. We assume that each node is equipped with an independent capacity-limited buffer B_m to store message packets to be forwarded. Each message consists of message header and message content; message header contains a unique message ID m_i, source node ID, destination node ID, and timestamp t that indicates the time of message generation, finite time to live (TTL), maximum number of copies nc, and the signature information to verify the validity of the message, and message content is encrypted by the public key of the destination node. We assume that all nodes in DTNs are loosely synchronized on the local clock.

We adopt the multicopy message forwarding strategy that allows a message to be copied many times; each time only a copy of the message is forwarded to the next relay node, the max number of messages allowed to be copied is set in the header field of the message. The source node sends a message to the destination node via a sequence of intermediate nodes in a multihop manner. When two nodes contact each other, we would detect the behavior of the encounter node and select the most proper next-hop node as a relay node for each message according to the process of Figure 1. If any one copy of the message is delivered to the destination node, the destination node sends an ACK message packet to the network to indicate that the message has been received. The relay node automatically deletes the message from the buffer when the time to live (TTL) of message is in the end or when it received an ACK message from the destination node. When the node generates a new message or receives a message copy from last hop nodes, it first detects whether the remaining buffer space meets the storage requirements. If the remaining

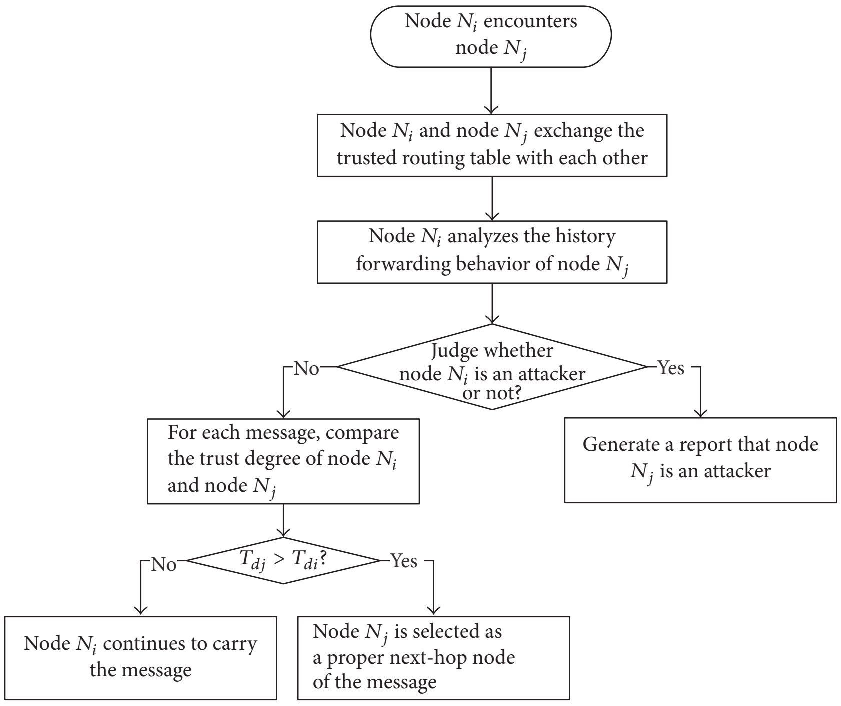

Node N_i encounters node N_j
Node N_i and node N_j exchange the trusted routing table with each other
Node N_i analyzes the history forwarding behavior of node N_j
Judge whether node N_i is an attacker or not?
No
Yes
For each message, compare the trust degree of node N_i and node N_j
Generate a report that node N_j is an attacker
$T_{dj} > T_{di}$?
No
Yes
Node N_i continues to carry the message
Node N_j is selected as a proper next-hop node of the message

Figure 1: A flowchart for SER scheme execution.

buffer space is too small, it will sort the messages according to their TTL and delete the message that its TTL is longest.

2.2. Attacker Model. According to the damage degree of the malicious nodes to the networks, we define two types of attackers as follows.

Definition 1 (selfish attacker). A selfish attacker is a node that often arbitrarily refuses the forwarding message request of the well-behaving nodes, to save the energy, buffer, or computing resources. But selfish attackers may decide to forward the message if they have a good relationship with the source node, the destination node, or last hop relay node. This type of attack is launched by selfish users that only want to profit from network and are not willing to help other users to forward messages.

Definition 2 (malicious attacker). A malicious attacker is a node that often uses the vulnerability of routing scheme to disguise as a relay node to receive a large number of messages from its encounter nodes, then maliciously drops these messages from its buffer, and does not forward these messages to the next-hop node. This type of attack is launched by adversary that wants to degrade or destroy the routing of DTNs.

2.3. Design Goals. Our goals are to design a secure and reliable opportunistic routing protocol that can not only improve the performance of the network, but also effectively restrain the malicious behavior of selfish or malicious node. The specific objectives are as follows.

(1) Improving the Routing Performance of DTNs. The proposed routing scheme should be able to improve the network performance effectively compared with the existing message forwarding methods, that is, higher delivery rate and lower average delivery delay.

(2) Resistance to Malicious and Selfish Attacks. The proposed detection scheme should be able to resist the attacks of malicious and selfish nodes in DTNs. In the process of message delivery, the relay nodes using this scheme could distinguish the malicious nodes and well-behaving nodes and select well-behaving node as the next-hop relay node.

(3) Robustness. The proposed routing scheme should be secure and robust. The formation and evaluation method of node trust degree should be able to resist the attack of malicious nodes. The trusted routing table should not be deleted and modified by malicious nodes.

3. Security and Efficient Routing Scheme Based on Trusted Routing Table

3.1. Basic Idea of Our Scheme. In DTNs, most messages need to be forwarded through multiple intermediate nodes. For example, as shown in Figure 2, the source node N_0 delivers a message to the destination node N_k along the path $N_0 \rightarrow N_1 \rightarrow N_2 \rightarrow N_3 \rightarrow N_k$. Due to those intermediate nodes N_0, N_1, N_2, N_3 successfully participated in forwarding message; according to the regular pattern of periodic movement of nodes, those nodes may deliver messages to the destination node N_k again at some time in the future. If the nodes

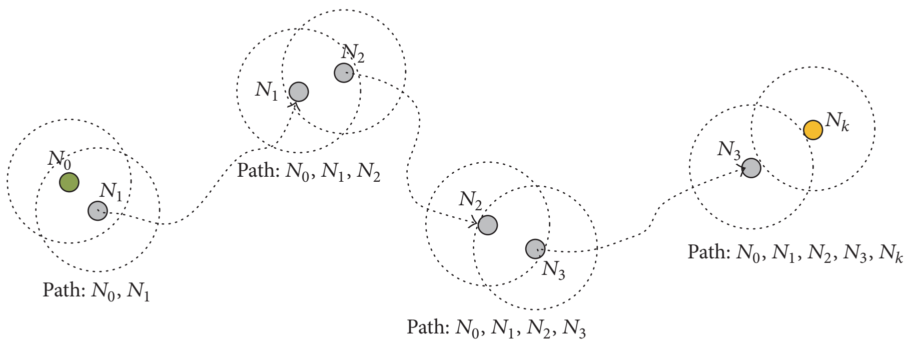


Figure 2: Data transmission path.

N_0, N_1, N_2, N_3 often forward messages to node N_k, those nodes would be the best relay nodes when a node delivers a message to node N_k. Moreover, due to the intermittent connectivity of DTNs, only the destination node could know which nodes have successfully participated in delivering message to it. Therefore, the destination node could reward and evaluate these nodes by using trust mechanisms. This means that the more trusted nodes have higher probability of delivering the message to the destination node.

Based on the above observation, we design a security and efficient routing scheme based on trust mechanism. In our scheme, each DTNs node maintains a trusted routing table (TRT), TRT is $n \times n$ two-dimensional matrix, as shown in (1), and n denotes the number of nodes in DTNs. The tuple $Tr_{ij} = \langle E_{ij}^*, T_{ij} \rangle$ generated by node N_i records the summary information about node N_j based on the encounter histories and message forwarding histories, where $E_{ij}^* = E_{ij}, R_{ij}, S_{ij}, \mathrm{sig}_i, \mathrm{sig}_j$ refers to the encounter evidences, receiving and sending messages evidences between node N_i and node N_j, E_{ij} denotes the number of encounters between node N_i and node N_j, R_{ij} and S_{ij} denote the number of messages that node N_i receives from and sends to node N_j respectively, and $\mathrm{sig}_i, \mathrm{sig}_j$ refer to the signature generated by node N_i and node N_j respectively. We will use E_{ij}, R_{ij}, S_{ij} to detect malicious nodes and selfish nodes. T_{ij} denotes the trust value of node N_j evaluated by node N_i based on history message forwarding evidences, which is a real number in the range of $[0, 1]$; the value of T_{ij} denotes the ability of node N_j to deliver the message to node N_i. The larger the trust value T_{ij}, the stronger the ability to deliver message to node N_i. We will use T_{ij} as routing strategy to determine the proper next-hop relay node.

The row vector of matrix $SR_i = (Tr_{i1}, Tr_{i2}, Tr_{i3}, \ldots, Tr_{in})$ represents the summary information that the node N_i reports about other nodes. The node N_i is responsible for maintaining and updating the vector SR_i and periodically updates the latest SR_i in trusted routing table. The column vector $SC_{*i} = (Tr_{1i}, Tr_{2i}, Tr_{3i}, \ldots, Tr_{ni})^T$ represents the summary information that the other nodes report about node N_i. We can obtain the belief of whether the node N_i is malicious or selfish from the column vector SC_{*i}.

$$\mathrm{TRT} = \begin{bmatrix} Tr_{11} & Tr_{12} & Tr_{13} & \cdots & Tr_{1n} \\ Tr_{21} & Tr_{22} & Tr_{23} & \cdots & Tr_{2n} \\ & & \vdots & & \\ Tr_{i1} & Tr_{i2} & Tr_{i3} & \cdots & Tr_{in} \\ & & \vdots & & \\ Tr_{n1} & Tr_{n2} & Tr_{n3} & \cdots & Tr_{nn} \end{bmatrix}. \tag{1}$$

Therefore, in the routing scheme based on TRT, the relay nodes can determine whether the encountered node is the proper next-hop relay node of the message by using row vector of destination node in the trusted routing table (TRT). The node can determine whether the encounter node is a malicious node or selfish node by using the column vector of encounter node in the trusted routing table.

In Figure 3, we illustrate the message forwarding process based on TRT. Suppose that the node N_1 carries the message m to the destination node N_6, when node N_1 meets node N_2, it first looks up the row vector of node N_2 from TRT to get T_{61}, T_{62}; if $T_{62} > T_{61}$ indicates that the ability of node N_2 to carry messages m to destination node N_6 is greater than that of node N_1, node N_2 is more proper next-hop relay node; therefore node N_1 forwards message m to node N_2. Otherwise, node N_1 continues to carry message m until the destination node is encountered or the next-hop is more reliable. In our scheme, the trust degree of node that it meets is greater than itself as the next-hop relay node. So, the message can be delivered to the destination along the direction of trust gradient increment.

3.2. Collecting Forwarding Evidences. To obtain timely and reliably the forwarding evidences of intermediate nodes, we adopt Captive-Carry mechanism to collect forwarding evidence information. In the message forwarding process, some forwarding evidences information that can prove which nodes have participated in forwarding message is bound

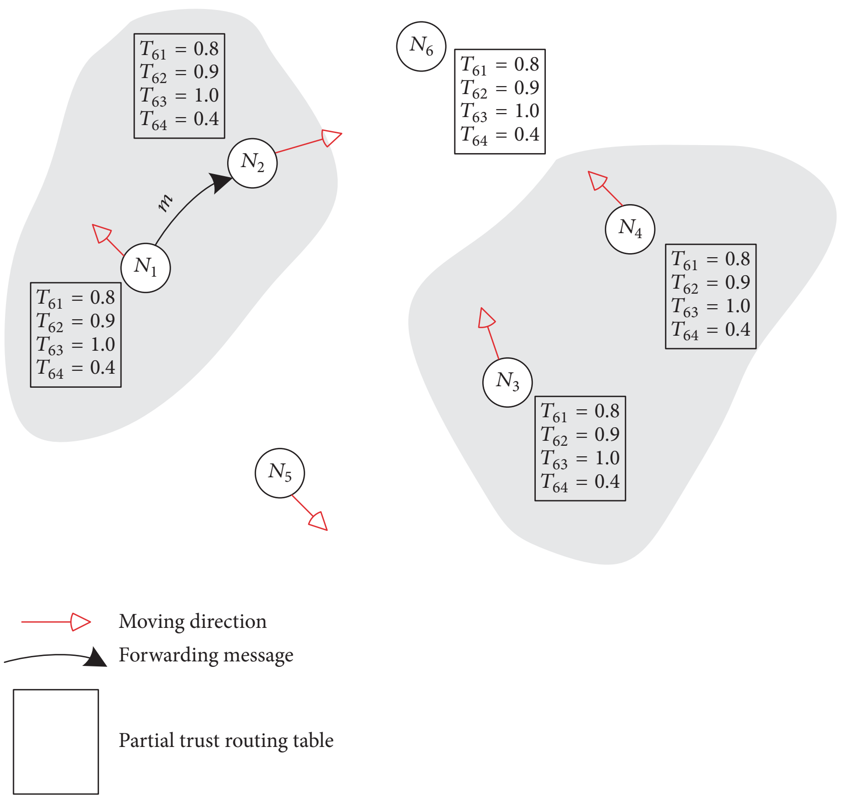


FIGURE 3: Message forwarding process based on trust.

dynamically into message body, carried to the destination node together with the message. After receiving the message, the destination node can obtain a list of intermediate nodes from the forwarding evidence information of message body and validate their authenticity and then reward these intermediate nodes according to the defined evaluation strategy.

To guarantee the security and authenticity of the forwarding evidences and prevent malicious nodes from tampering message and adding fake forwarding evidences, in the implementation, we adopt the layered coin model in the literature [15] to achieve Captive-Carry mechanism and message packet format. A typical layered coin model usually consists of a base layer formed by the source node and multiple endorsed layers formed by the intermediate nodes. If an intermediate node forwards the message to the next-hop relay node, it fists forms an endorsed layer on message and then adds signature information as evidence to the endorsed layer. Figure 3 shows an example of message architecture based on layered coin model; the base layer of message is composed of message header, message content, and endorsed layer 0 formed by the source node, and message header contains six fields: mid, N_0, N_k, t, TTL, nc refer to the identifier of the message, the identifier of the source node, the identifier of the destination node, the timestamp of message created, the time to live, and the max copy number of message, respectively. The message content is only composed of encrypted data; when the source node N_0 wants to send a message to the destination node N_k with the public key pk_k, node N_0 uses the public key pk_k to encrypt the real network data into $E_{\mathrm{pk}_k}(H(N_0 \mid t \mid N_k \mid C))$ to achieve confidentiality, where C denotes real network data and $H(*)$ is a hash function of message properties N_0, t, N_k and network data C to validate message content.

The endorsed layer is formed dynamically by the intermediate node when it wants to forward the message to the next-hop relay node. In Figure 4, for example, when the source node N_0 encounters node N_1 at timestamp ts_0 and has determined that node N_1 is a proper next-hop relay node, node N_0 creates an endorsed layer 0 that contains four fields: $\mathrm{sig}_0, N_1, \mathrm{ts}_0, \mathrm{sig}_1$, where $\mathrm{sig}_0 = \mathrm{SIG}_{\mathrm{sk}_0}(N_0 \mid \mathrm{mid})$ is its signature over the message identifier mid and node identifier N_0 using its corresponding private key sk_0, N_1 is the identifier of the next-hop relay node, ts_0 is a timestamp indicating the time of message forwarding, $\mathrm{sig}_1 = \mathrm{SIG}_{\mathrm{sk}_1}(H'(N_0 \mid \mathrm{ts}_0 \mid N_1))$ is the signature of node N_1 over the content $H'(N_0 \mid \mathrm{ts}_0 \mid N_1)$ using its corresponding private key sk_1, and $H'(*)$ is a hash function for generating summary of $N_0 \mid \mathrm{ts}_0 \mid N_1$. These four fields used as the forwarding evidence proved that node N_0 has forwarded a message to node N_1 at timestamp ts_0, and the signature sig_1 can prove that node N_1 is willing to receive the message from node N_0 at timestamp ts_0. Other endorsed layers created by relay nodes only include the identifier of the forwarding nodes, timestamp of message forwarding, the identifier of the receiving node, and the signature of the receiving node; for example, in Figure 4, endorsed layer 1 contains $N_1, \mathrm{ts}_1, N_2, \mathrm{sig}_2$. The above information is used as forwarding evidence which proved that

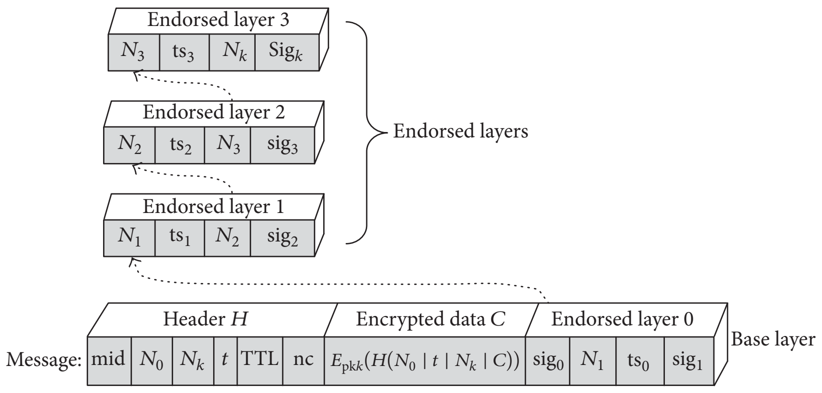

Endorsed layer 3
N_3 ts$_3$ N_k Sig$_k$
Endorsed layer 2
N_2 ts$_2$ N_3 sig$_3$
Endorsed layer 1
N_1 ts$_1$ N_2 sig$_2$
Endorsed layers
Header H
Encrypted data C
Endorsed layer 0
Message: mid N_0 N_k t TTL nc $E_{pkk}(H(N_0 \mid t \mid N_k \mid C))$ sig$_0$ N_1 ts$_0$ sig$_1$
Base layer

Figure 4: Layered coin model based message format.

the message is forwarded to the next-hop relay node. The signature $\text{sig}_1, \text{sig}_2, \ldots$ can witness these nodes that are willing to receive the message.

Overhead of the message is based on layered coin mode. Because the message is added multilayer of evidences information in the forwarding process, the message length is slightly larger than the basic message. Except the signature fields, we assume each field is 2-byte length; then the message header with six fields is 12 bytes, the length of endorsed layer 0 is around $4 + 2 \cdot |\text{sig}|$ bytes, another each endorsed layer is around $6 + |\text{sig}|$ bytes, and then the overhead of a k layered message $\text{Length}_k(m)$ is calculated as follows.

$$\begin{aligned}\text{Length}_k(m) &= 12 + \left|E_{\text{pk}_k}\left(H\left(N_0 \mid t \mid N_k \mid C\right)\right)\right| + 4 \\ &\quad + 2 \cdot |\text{sig}| + (k-1) \cdot \left(6 + |\text{sig}|\right) \text{ bytes} \\ &= 10 + \left|E_{\text{pk}_k}\left(H\left(N_0 \mid t \mid N_k \mid C\right)\right)\right| + 6 \\ &\quad \cdot k + (k+1) \cdot |\text{sig}| \text{ bytes},\end{aligned} \tag{2}$$

where $|E_{\text{pk}_k}(H(N_0 \mid t \mid N_k \mid C))|$ denotes the length of encrypted message content; $|\text{sig}|$ denotes the length of signature. Thus, the overhead of a message is mainly composed of message content and additional evidence information. We assume the $|\text{sig}|$ is 20-byte length; then the length of additional evidence is around $L = 26k + 18$ bytes, when $k = 10$, $L \approx 0.25$ kb. Therefore, the additional evidence information is very small, relative to the message content; the overhead of bandwidth and storage is only a little more than the traditional methods.

3.3. Building and Updating Trusted Routing Table. The tuple $Tr_{ij} = \langle E_{ij}^*, T_{ij}\rangle$ is generated and updated by two processes: encounter process and message receiving process. In encounter process, suppose that node N_i and node N_j meet each other at time t_s; the number of encounters between the two nodes E_{ij}, E_{ji} will be incremented by 1, respectively. Without loss of generality, we assume that node N_i is carrying message that needs to be forwarded to the next-hop nodes. If node N_j is chosen as the next-hop relay of message, node N_i will follow SER routing protocol to forward a message copy to node N_j. After the message copy is successfully forwarded, S_{ij} will be incremented by 1 to record the number of messages that have been sent to node N_j. In addition, after the node N_j received a message from the node N_i, R_{ji} will also be incremented by 1 to record the number of messages received from node N_i. Similarly, if the node N_i received a message from the node N_j, R_{ij} will be incremented by 1 to record the number of messages received from node N_j, and S_{ji} will also be incremented by 1 to record the number of messages that have been sent to node N_i. The initial value of $E_{ij}, S_{ij}, R_{ij}, E_{ji}, S_{ji}, R_{ji}$ is set to 0. To prevent malicious nodes from forging $E_{ij}, S_{ij}, R_{ij}, E_{ji}, S_{ji}, R_{ji}$ and ensure $E_{ij} = E_{ji}, S_{ij} = R_{ji}, R_{ij} = S_{ji}$, the new evidences $E_{ij}^* = E_{ij}, R_{ij}, S_{ij}, \text{sig}_i, \text{sig}_j$ and $E_{ji}^* = E_{ji}, R_{ji}, S_{ji}, \text{sig}_i, \text{sig}_j$ will be generated by node N_i and node N_j respectively, where $\text{sig}_i = \text{SIG}_{\text{sk}_i}(H'(E_{ij} \mid R_{ij} \mid S_{ij} \mid t_s))$ and $\text{sig}_j = \text{SIG}_{\text{sk}_j}(H'(E_{ji} \mid R_{ji} \mid S_{ji} \mid t_s))$ refer to the signatures generated by node N_i and node N_j, respectively, to show that node N_i and node N_j have accepted these evidences $E_{ij}, S_{ij}, R_{ij}, E_{ji}, S_{ji}, R_{ji}$. Node N_i and node N_j can judge whether $E_{ij} = E_{ji}, S_{ij} = R_{ji}, R_{ij} = S_{ji}$ are established by verifying the signature of the other party. Consequently, malicious nodes have difficulty forging the encounter and forwarding evidences unilaterally.

In message receiving process, if node N_i has received a message m and is the destination node of m, node N_i extracts the forwarding evidences from the multiple endorsed layer of the message m and obtains the intermediate nodes and the message forwarding path as an evidence chain path: $N_0 \xrightarrow{\text{ts}_0} N_1 \xrightarrow{\text{ts}_1} N_2 \xrightarrow{\text{ts}_2} \cdots \rightarrow N_j \xrightarrow{\text{ts}_j} \cdots \xrightarrow{\text{ts}_i} N_i$. For each node in the evidence chain, the destination node N_i verifies the validity of their signature $\text{sig}_0, \text{sig}_1, \text{sig}_2, \ldots, \text{sig}_j, \ldots$, and if signature verification for all nodes is correct and valid, those that successfully helped forwarding will be rewarded and trusted. The design of trust reward calculation is the pivot of an efficient routing scheme, which should reflect the ability of the intermediate nodes to forward message to the destination node and the fairness and incentive of trust evaluation. Therefore, we will calculate the trust reward of intermediate nodes based on the principles of reliability and delay.

(1) Reliability Principle. The position of intermediate node in the evidence chain path(N_j) is closer to the destination node N_i, and the trust reward of this intermediate node should be higher. This is because the larger path(N_j), the higher the reliability of node N_j to carry a message to the destination node N_i.

(2) Delay Principle. For the message of same link length, if the message delay time $\Delta t = ts_i - t$ is smaller, the intermediate node should get the higher trust reward from the destination node N_i. This is because the smaller Δt is, the quicker those intermediate nodes can deliver the message to the destination node N_i.

Assume that $T_{ij}^{(m)}$ is the trust reward of node N_j evaluated by the destination node N_i based on the received message m. Based on the above principle, we define $T_{ij}^{(m)}$ as

$$T_{ij}^{(m)} = \frac{1}{2}\left(\rho\left(\frac{\Delta t}{\mathrm{TTL}}\right) + f\left(\frac{\mathrm{path}\left(N_j\right)}{|\mathrm{path}\,(m)|}\right)\right), \quad (3)$$

where $|\mathrm{path}(m)|$ denotes the length of message m forwarding path in evidence chain; $\rho(x) = e^{-\lambda x}$, $0 < x \leq 1$, is the delay reward function of evidence chain, which has monotonic decreasing character with delay time Δt of the message. The value range of function is $e^{-\lambda} \leq \rho(x) < 1$, $\lambda > 0$ is regulatory factor for the minimum value of the delayed reward, and the larger the parameter λ, the smaller the minimum value of the delayed reward. path(N_j) denotes the position of intermediate node N_j in the evidence chain, the range is $1 \leq \mathrm{path}(N_j) \leq |\mathrm{path}(m)|$, and $f(y) = \phi + (1+\phi)y^2$, $0 < y \leq 1$, is a reliability reward function for intermediate node, which has monotone increasing character with the parameter value path(N_j). The value range of function is $f(y) \in (\phi, 1]$, $0 \leq \phi < 1$ is a regulatory factor for the minimum value of the reliability reward, and the larger the parameter ϕ, the larger the minimum value of the reliability reward. Therefore, the range of $T_{ij}^{(m)}$ is $(e^{-\lambda} + \phi)/2 < T_{ij}^{(m)} < 1$.

The trust degree T_{ij} of node N_i toward node N_j is calculated based on all the messages forwarding evidences in history. The following trust degree calculation is exercised: if no new trust reward is gained in time window Tw, then T_{ij} will decrease with the time; otherwise, T_{ij} will increase based on trust reward $T_{ij}^{(m)}$; that is,

$$T_{ij} = \begin{cases} T'_{ij} \times \zeta\left(t_n, t_o\right) & \text{if } t_n - t_o > \mathrm{Tw}\ \&\ N_j \notin \mathfrak{R} \\ T'_{ij} + \left(1 - T'_{ij}\right) \times T_{ij}^{(m)} & \text{otherwise,} \end{cases} \quad (4)$$

where T'_{ij} denotes the old trust degree of node N_i toward node N_j, Tw represents the length of trust update window, and R is the set of nodes that deliver successfully messages to node N_i within the current window, that is, the collection of nodes in the evidence chain received in the current time window. $\zeta \in (1-\gamma, 1)$ is a time decay function, where t_n, t_0 denote the current time and the latest trust update time, respectively; as shown in (5), $0 < \gamma \leq 1$ is a factor of decay rate and minimum; the larger the parameter γ is, the quicker the trust degree value decreases. Therefore, if a node can keep good trust degree continuously, it will have a strong ability to forward message to the node N_i.

$$\zeta\left(t_n, t_o\right) = 1 - \frac{\left(t_n - t_o\right)\gamma}{t_n}. \quad (5)$$

When the encounter process and message receiving process are performed, node N_i will generate or update the tuple $Tr_i j = \langle E_{ij}^*, T_{ij}\rangle$ to record summary information about node N_j. Without loss of generality, node N_i generates the tuple Tr for each DTNs node after multiple cycles of the network operation. Therefore, we adopt vector table $SR_i = (Tr_{i1}, Tr_{i2}, Tr_{i3}, \ldots, Tr_{in})$ to store the summary information that the node N_i reports about other nodes. The trusted routing table (TRT) consists of vector table of each node, but in initial phase, each node's trusted routing table has only its own row vector. To quickly build and update trusted routing table in each DTNs node, when the two nodes meet, they first exchange trusted routing table with each other. When a node received the encounter node's trusted routing table, it compares the received trusted routing table to itself trusted routing table. If there is a new row vector in the received trusted routing table, this node will update its trusted routing table. To prevent malicious nodes from tampering row vector of trusted routing tables, node N_i generates the record information of the row vector $SR_i^* = SR_i, \mathrm{TwID}, \mathrm{Sig}_i$ periodically to update and verify the trusted routing table, where $\mathrm{Sig}_i = \mathrm{SIG}_{\mathrm{sk}_i}(H''(SR_i \mid \mathrm{TwID}))$ refers to the signature generated by node N_i on vector table SR_i, $H''(*)$ is a hash function for generating summary of $SR_i \mid$ TwID, and TwID denotes the latest update window of record information SR_i^*.

We use an example to illustrate the update process of the trusted routing table. As shown in Tables 1 and 2, the trusted routing table TRT$_i$ of node N_i contains three row vectors SR_1, SR_2, SR_i, and the trusted routing table TRT$_j$ of node N_j contains three row vectors SR_1, SR_3, SR_j, and when node N_i and node N_j meet, they exchange the trusted routing tables TRT$_i$, TRT$_j$ with each other. After the update operation, as shown in Table 3, the trusted routing tables TRT$_i$, TRT$_j$ of node N_i and node N_j contain five row vectors $SR_1, SR_2, SR_3, SR_i, SR_j$, and SR_1 is the latest vector table in

Table 1: Trusted routing table TRT$_i$ of node N_i before update.

Destination node	Vector table	TwID	Signature information
N_1	SR_1	1	Sig$_1$
N_2	SR_2	1	Sig$_2$
N_i	SR_i	1	Sig$_i$

Table 2: Trusted routing table TRT$_j$ of node N_j before update.

Destination node	Vector table	TwID	Signature information
N_1	SR_1	2	Sig$_1$
N_3	SR_3	1	Sig$_3$
N_j	SR_j	1	Sig$_j$

TABLE 3: Trusted routing table of node N_i and node N_j after update.

destination node	Vector table	TwID	Signature information
N_1	SR_1	2	Sig_1
N_2	SR_2	1	Sig_2
N_3	SR_3	1	Sig_3
N_i	SR_i	1	Sig_i
N_j	SR_j	1	Sig_j

```
Input: node $N_i$ maintains the vector table
       $SR_i = (Tr_{i1}, Tr_{i2}, Tr_{i3}, \cdots, Tr_{in})$; The initial
       value of trusted routing table $\mathrm{TRT}_i$ contains
       only row vectors $SR_i$;
(1)  if node $N_i$ updated the vector table $SR_i$ then
(2)      generate the new row vector
         $SR_i^* = \{SR_i, \mathrm{TwID}, \mathrm{Sig}_i\}$;
(3)      update row vector $SR_i$ in trusted routing table;
(4)      TwID + +;
(5)  end
(6)  if node $N_i$ and node $N_j$ meet each other then
(7)      send trusted routing table $\mathrm{TRT}_i$ to node $N_j$;
(8)      receive trusted routing table $\mathrm{TRT}_j$ from node $N_j$;
(9)  end
(10) while $\forall SR_k \in \mathrm{TRT}_i \cap \mathrm{TRT}_j$ do
(11)     if $\mathrm{TRT}_j \cdot SR_k \cdot \mathrm{TwID} > \mathrm{TRT}_i \cdot SR_k \cdot \mathrm{TwID}$ then
(12)         verify the validity of the signature $\mathrm{Sig}_k$;
(13)         if $\mathrm{Sig}_k$ is valid then
(14)             update row vector $SR_k$ in trusted routing
                   table $\mathrm{TRT}_i$;
(15)             $\mathrm{TRT}_i \cdot SR_k = \mathrm{TRT}_j \cdot SR_k$;
(16)             $\mathrm{TRT}_i \cdot SR_k \cdot \mathrm{TwID} = \mathrm{TRT}_j \cdot SR_k \cdot \mathrm{TwID}$;
(17)             $\mathrm{TRT}_i \cdot \mathrm{Sig}_k = \mathrm{TRT}_j \cdot \mathrm{Sig}_k$;
(18)         end
(19)     end
(20) end
(21) while $\forall SR_k \in \mathrm{TRT}_j$ and $SR_k \notin \mathrm{TRT}_j$ do
(22)     verify the validity of the signature $\mathrm{Sig}_k$;
(23)     if $\mathrm{Sig}_k$ is valid then
(24)         insert $N_k, SR_k, \mathrm{TwID}, \mathrm{Sig}_k$ into trusted routing
               table $\mathrm{TRT}_i$;
(25)     end
(26) end
```

ALGORITHM 1: Building and updating TRT.

node N_j. The detailed update process of trusted routing table is shown in Algorithm 1.

Robustness Analysis. In the building and updating process of the trusted routing table, the malicious node may modify the trusted routing table to forge high trust value and message forwarding ratio. However, this attack can be thwarted in our scheme, since the number of encounters and forwarding message number of malicious node can be verified by well-behaving nodes that encountered malicious node in the past, so the malicious node cannot forge message forwarding ratio. Because each row vector in the trusted routing table has the signature of the corresponding node, if the malicious node forged high trust value in row vector, then this forged row vector cannot be updated to the trusted routing table of other nodes, because the signature of the node in this forged row vector is incorrect. As a result, the proposed trusted routing table has robustness and nonrepudiation.

3.4. Detecting Malicious Nodes and Selfish Nodes

By analyzing and observing the characteristics of the attacker in Section 2.2, we have the strong belief that can distinguish between well-behaving nodes and malicious nodes through their historical forwarding behavior and trust value, because if a well-behaving node has a high number of encounters with other nodes, it might receive a lot of messages and forward a larger portion of them or all of them; that is, it has higher ratio between forwarded messages and received messages. However, malicious nodes often have high number of encounters and receive a lot of messages from other nodes but only forward a small portion of them or even do not forward any of them, so malicious nodes have lower ratio of forwarded messages over received messages. Different from malicious nodes, selfish nodes receive only a few of messages even if they have high number of encounters with other nodes, but they forward a lot of messages generated by themselves, so selfish nodes have abnormal high ratio between forwarded messages and received messages. Based on the above analysis, we use the column vector in the trusted routing table to define the metrics named malicious behavior ratio, MBR, and selfish behavior ratio, SBR, that can effectively detect malicious nodes and selfish nodes. Suppose the column vector of node N_j is $SC_{*j} = (Tr_{1j}, Tr_{2j}, Tr_{3j}, \ldots, Tr_{nj})^T$, malicious behavior ratio MBR of node N_j can be formulated as

$$\mathrm{MBR} = \frac{\left(\sum_{i=1}^{n} Tr_{ij} \cdot R_{ij}\right)^2}{\sum_{i=1}^{n} Tr_{ij} \cdot S_{ij} \times \sum_{i=1}^{n} Tr_{ij} \cdot E_{ij}}, \quad (6)$$

where $\sum_{i=1}^{n} Tr_{ij} \cdot R_{ij}$ is the total number of messages that all DTNs nodes received from node N_j, that is, the total number of messages forwarded by node N_j. $\sum_{i=1}^{n} Tr_{ij} \cdot S_{ij}$ is the total number of messages that all DTNs nodes send to node N_j, that is, the total number of messages received by node N_j. $\sum_{i=1}^{n} Tr_{ij} \cdot E_{ij}$ is the total number of encounters where all DTNs nodes meet node N_j. MBR can potentially reveal malicious behavior of malicious nodes dropping packets frequently, because $\sum_{i=1}^{n} Tr_{ij} \cdot R_{ij}$ is far less than $\sum_{i=1}^{n} Tr_{ij} \cdot S_{ij}$ and $\sum_{i=1}^{n} Tr_i j \cdot E_i j$, so malicious nodes have lower MBR than the well-behaving nodes.

To effectively detect selfish nodes, selfish behavior ratio SBR can be formulated as

$$\mathrm{SBR} = \frac{\sum_{i=1}^{n} Tr_{ij} \cdot R_{ij} \times \sum_{i=1}^{n} Tr_{ij} \cdot E_{ij}}{\left(\sum_{i=1}^{n} Tr_{ij} \cdot S_{ij}\right)^2}, \quad (7)$$

where the meaning of each value and expression is the same as (6), and SBR can potentially reveal behavior of selfish nodes that frequently refuse the request of forwarding message packets, because $\sum_{i=1}^{n} Tr_{ij} \cdot R_{ij}$ and $\sum_{i=1}^{n} Tr_{ij} \cdot E_{ij}$ are greater than $\sum_{i=1}^{n} Tr_{ij} \cdot S_{ij}$, so selfish nodes have abnormal higher SBR than the well-behaving nodes.

```
Input: Detection node $N_j$, $\text{Th}_{\text{MBR}}$, $\text{Th}_{\text{SBR}}$
Output: detection result of node $N_j$
(1)  get the column vector
       $SC_{*i} = (Tr_{1i}, Tr_{2i}, Tr_{3i}, \ldots, Tr_{ni})^T$ of node $N_j$ from
       trusted routing table;
(2)  calculate malicious behavior ratio MBR of node $N_j$
       using Eq. (6);
(3)  calculate selfish behavior ratio SBR of node $N_j$ using
       Eq. (7);
(4)  if MBR < $\text{Th}_{\text{MBR}}$ then
(5)      return node $N_j$ is a malicious node;
(6)  else
(7)      if SBR > $\text{Th}_{\text{SBR}}$ then
(8)          return node $N_j$ is a selfish node;
(9)      else
(10)         return node $N_j$ is a well-behaving node;
(11)     end
(12) end
```

Algorithm 2: Detecting malicious and selfish nodes.

Therefore, after obtaining MBR and SBR of node N_j, we compare them with predefined thresholds to judge whether the node N_j is malicious node or not. Th_{MBR} and Th_{SBR} denote threshold of malicious behavior ratio and selfish behavior ratio, respectively, and their value is chosen empirically using simulation. The detailed detection process of malicious and selfish nodes is shown in Algorithm 2.

3.5. Security and Efficient Routing Based on TRT. When node N_I meets node N_j, it triggers Algorithm 3 to perform the following routing steps. Step (1): node N_I first uses Algorithm 2 to judge whether node N_j is a well-behaving node or not; if node N_j is a well-behaving node, it runs to next step. Step (2): node N_i queries messages in its buffer B_m; if there are messages that need to be forwarded, it stores the messages in a temporary set M and then goes to next step. Step (3): for each message $m \in M$, node N_i gets the identity N_d of the destination node from head field of message m and obtains row vector $SR_d = (Tr_{d1}, Tr_{d2}, Tr_{d3}, \ldots, Tr_{dn})$ of node N_d in trusted routing table TRT, then queries the trust value $Tr_{di} \cdot T_{di}, Tr_{dj} \cdot T_{dj}$, of node N_i and node N_j evaluated by node N_d, and goes to next step. Step (4): node N_i forwards the message m to node N_j according to the following strategies.

(1) If $Tr_{di} \cdot T_{di} > Tr_{dj} \cdot T_{dj}$, node N_i will forward a copy of message m with copy number nc_j to node N_j and then updates copy number $\text{nc}_i \leftarrow \text{nc}_i - \text{nc}_j$; otherwise node N_i continues to carry message m until it meets a node with greater trust degree. The copy number of message is divided according to the proportion of trust value, as shown in (8). If the node has higher trust value, it has larger copy number of message.

$$\text{nc}_j \longleftarrow \left\lceil \frac{Tr_{dj} \cdot T_{dj}}{Tr_{di} \cdot T_{di} + Tr_{dj} \cdot T_{dj}} \cdot \text{nc}_i \right\rceil . \tag{8}$$

```
(1)  if node $N_i$ meets node $N_j$ then
(2)      node $N_i$ triggers Algorithm 2 and detects behavior of
           node $N_j$;
(3)      get detection result of node $N_j$ from Algorithm 2;
(4)  end
(5)  if node $N_j$ is a well-behavior node and $B_m \neq \phi$ then
(6)      sorts messages by the remaining TTL;
(7)      for each message $m \in B_m$ do
(8)          get destination node $N_d$ from head field of $m$;
(9)          get $Tr_{di} \cdot T_{di}, Tr_{dj} \cdot T_{dj}$ of node $N_i$ and node $N_j$
               from $SR_d = (Tr_{d1}, Tr_{d2}, Tr_{d3}, \ldots, Tr_{dn})$;
(10)         if $Tr_{di} \cdot T_{di} > Tr_{dj} \cdot T_{dj}$ then
(11)             get copy number $\text{nc}_i$ from head field of $m$;
(12)             calculate $\text{nc}_j$ using Eq. (8);
(13)             forwards a copy of $m$ with copy number $\text{nc}_j$
                   to node $N_j$;
(14)             updates $\text{nc}_i \leftarrow \text{nc}_i - \text{nc}_j$;
(15)         else
(16)             if $Tr_{di} \cdot T_{di} == 0$ and $Tr_{dj} \cdot T_{dj} == 0$ then
(17)                 calculate $\overline{T_i}, \overline{T_j}$ of node $N_i$ and node $N_j$;
(18)                 if $\overline{T_i} == 0$ and $\overline{T_j} == 0$ then
(19)                     forwards a copy of $m$ to node $N_j$;
(20)                 else
(21)                     if $\overline{T_j} > \overline{T_i}$ then
(22)                         get copy number $\text{nc}_i$ from head
                               field of $m$;
(23)                         calculate $\text{nc}_j$ using Eq. (9);
(24)                         forwards a copy of $m$ with copy
                               number $\text{nc}_j$ to node $N_j$;
(25)                         updates $\text{nc}_i \leftarrow \text{nc}_i - \text{nc}_j$;
(26)                     end
(27)                 end
(28)             else
(29)         end
(30)     end
(31) end
```

Algorithm 3: Security and efficient routing algorithm.

(2) In the initial phase, if $Tr_{di} \cdot T_{di}$ and $Tr_{dj} \cdot T_{dj}$ are null values, we use average trust values $\overline{T_i}, \overline{T_j}$ of node N_i and node N_j as the decision for message forwarding, $\overline{T_i} = (1/n) \sum_{x=1}^{n} Tr_{xi} \cdot T_{xi}$ and $\overline{T_j} = (1/n) \sum_{x=1}^{n} Tr_{xj} \cdot T_{xj}$. If $\overline{T_j} = \overline{T_i} = 0$, it shows that the network is in the cold start phase. Therefore, node N_i adopts epidemic algorithm to forward the copy of message m to node N_j and does not divide the copy number of message m.

(3) If $\overline{T_j} > \overline{T_i}$, node N_i will forward a copy of message m with copy number nc_j to node N_j and then updates copy number $\text{nc}_i \leftarrow \text{nc}_i - \text{nc}_j$; otherwise node N_i continues to carry message m until it meets a node with greater trust degree. In this case, the copy number of message is divided as shown in

$$\text{nc}_j \longleftarrow \left\lceil \frac{\overline{T_j}}{\overline{T_i} + \overline{T_j}} \cdot \text{nc}_i \right\rceil . \tag{9}$$

The proposed routing algorithm only uses the row vectors and column vectors in the local trusted routing table to judge the behavior of encountered node and make forwarding decision. The overhead of the algorithm is low; the maximum time complexity is equal to $O(|M| \times n)$. The algorithm of detecting malicious nodes and selfish nodes can detect the behavior of the encountered node and effectively resist the attacks of malicious and selfish nodes in DTNs. Therefore, the proposed scheme improves the security and reliability of DTNs effectively and achieves the design goal 2 in Section 2.3. In the proposed scheme, a message is delivered to the destination node along the direction of trust gradient increment; only when the condition $Tr_{di} \cdot T_{di} > Tr_{dj} \cdot T_{dj}$ is met, the relay nodes forward the message to the next-hop relay nodes. This scheme makes the probability of the message reaching the destination node get higher and higher and significantly improves the network routing performance, that is, higher delivery ratio and lower average delivery delay and achieves design goal 1 in Section 2.3.

4. Performance Evaluation

4.1. Simulation Setup. We set up the experiment environment with the ONE (opportunistic network environment) simulator, in which we implement our proposed routing algorithm. ONE simulator is designed for evaluating and verifying DTNs routing protocols and includes a variety of movement models, map of Helsinki city, and some typical routing algorithms such as Epidemic, Spray and Wait (SAW), Prophet, and MaxProp. In our experiment, we adopt the map of Helsinki city as the experiment environment and deploy 200 nodes on the map with size of 4500 m to 3400 m. The well-behaving nodes and selfish nodes use shortest path map based movement model to simulate the movement at speed of 0.5 m/s to 1.5 m/s, and malicious nodes move at speed of 2.7 m/s to 13.9 m/s. Messages are generated at the rate of one per 25 to 35 seconds. The simulation time is set to 24 hours, during which 2900 messages are generated. The size of message is 512 kB. Time to live (TTL) is in the range of 30 to 240 minutes. The buffer B_m of each node is in the range of 5 M to 60 M. Delay of reward regulation factor λ is set to 10. Reliability reward regulation factor ϕ is set to 0.3. Time decay factor γ is set to 0.2. The maximum copy number of message nc is set to 10.

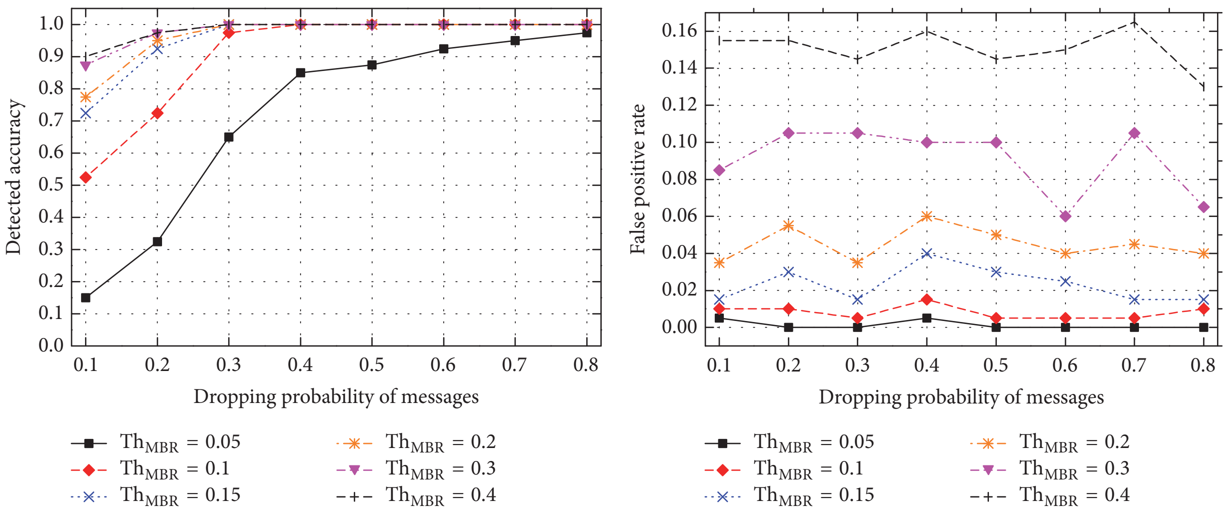
We evaluate our scheme in two aspects: effectiveness of malicious attack detection and routing performance. The performance metrics used in the evaluation are (i) detected accuracy, which is the percentage of malicious nodes and selfish nodes that can be detected; (ii) false positive rate, which is the percentage of well-behaving nodes that are falsely judged as malicious nodes and selfish nodes; (iii) delivery rate, which is the percentage of generated messages that are successfully delivered to destination nodes within time to live; (iv) average delivery delay, which is the average time taken for the messages to be delivered from the source nodes to the destination nodes; (v) overhead rate, which is the proportion between the number of relayed messages (excluding the successfully delivered messages) and the number of successfully delivered messages. Both detected accuracy and false positive rate are used to measure effectiveness of malicious attack detection. Delivery rate, average delivery delay, and overhead rate are used to measure routing performance.

4.2. Simulation Results and Analysis

4.2.1. The Impact of Choosing Different Threshold. First, we evaluate the impact of choosing different threshold Th_{MBR} on malicious behavior detection of SER. The number of malicious nodes is set to 40. The dropping probability of messages is varied from 0.1 to 0.8, which indicates the level of malicious nodes. The threshold of malicious behavior ratio Th_{MBR} is varied from 0.05 to 0.4. Time to live (TTL) of each message is fixed to 30 minutes. The buffer B_m of each node is fixed to 5 M.

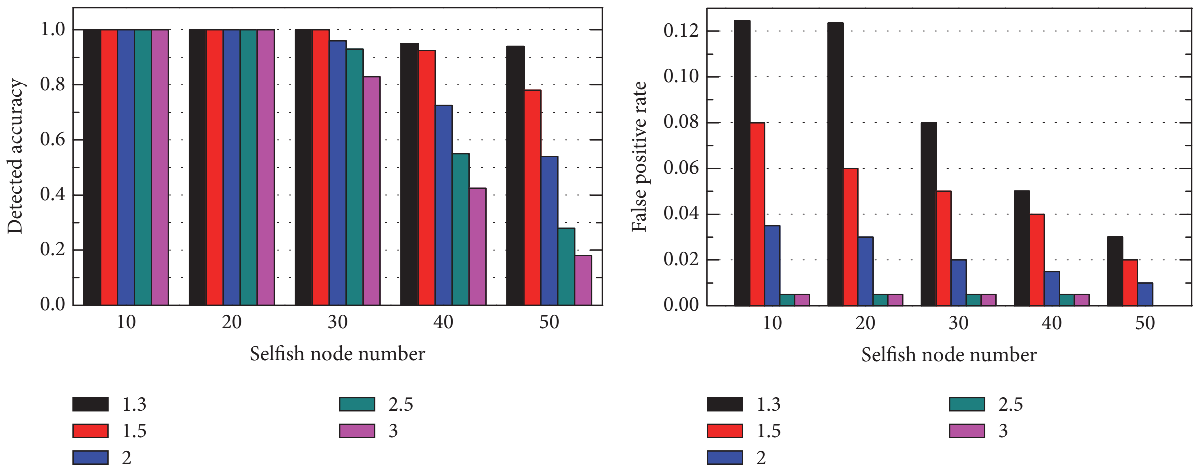
Figure 5 presents the detected accuracy and false positive rate of SER with varying thresholds and dropping probability. Figure 5(a) shows that six curves have similar trends, which indicate that the malicious nodes are more likely to be detected when their dropping probability increases. When the dropping probability of messages increases to 0.3, the detected accuracy of SER reaches to 100 percent using four varying thresholds. Even though the dropping probability of messages is lower than 0.1, the detected accuracy of SER is still higher than 70 percent when the threshold is greater than 0.15. This shows that SER could achieve a better detected accuracy. Furthermore, Figure 5(a) also shows that the greater the threshold, the higher the detected accuracy of SER. However, from Figure 5(b), we can obviously find that the greater the threshold, the higher the false positive rate of SER. When the threshold is equal to 0.4, the false positive rate of SER exceeds 14 percent, which is obviously unacceptable even with higher detected accuracy. Therefore, we need to set a threshold tradeoff between detected accuracy and false positive rate. As seen in Figures 5(a) and 5(b), if the threshold is set to 0.1 or 0.15, SER not only has higher detection accuracy, but also has lower false positive rate. That means SER has little effect on well-behaving nodes when we choose the appropriate threshold.

Similarly, we evaluate the impact of choosing different threshold Th_{SBR} on selfish behavior detection of SER. The total number of selfish nodes is varied from 10 to 50. The threshold Th_{SBR} is varied from 1.3 to 3. Figure 6(a) shows that the detected accuracy of SER reaches to 100 percent under all thresholds, when the total number of selfish nodes is less than 20, which implies that the less selfish nodes are easier to be detected. When the number of selfish nodes exceeds 30, the detected accuracy of SER has the drop trends, and the greater the threshold, the more obvious the drop trends. But correspondingly, as shown in Figure 6(b), the false positive rate of SER also has more significant drop trends, when the number of selfish nodes increases. This is because the selfish nodes have more friends and receive more messages from their friends. The result is that some selfish nodes have lower SBR and do not violate the large thresholds. Even though the number of selfish nodes is increased to 50, the detected accuracy of SER is still higher than 94 percent, but the false positive rate of SER is lower than 2 percent when the threshold is equal to 1.3. This means that SER could detect



(a) Detected accuracy (b) False positive rate

FIGURE 5: SER's malicious behavior detection results under varying thresholds Th_{MBR}.



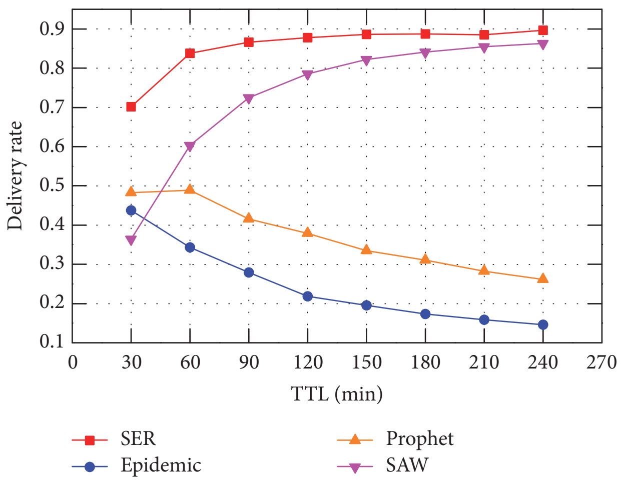
(a) Detected accuracy (b) False positive rate

FIGURE 6: SER's selfish behavior detection results under varying thresholds Th_{SBR}.

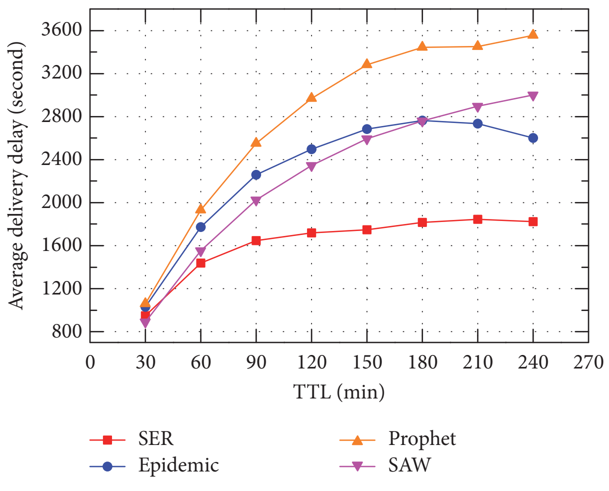
selfish behavior effectively and has a little effect on well-behaving nodes. Therefore, we conclude that if the number of selfish nodes is less than 30, we use the large threshold to detect false positive rate; on the contrary, we use a small threshold.

4.2.2. The Impact of TTL on the Routing Performance. In this section, we compare SER with three classic routing schemes Epidemic, Prophet, and SAW in the routing performance. The buffer B_m of each node is fixed to 5 M. Time to live (TTL) of message is varied from 30 minutes to 240 minutes. Suppose there are no malicious nodes and selfish nodes in DTNs.

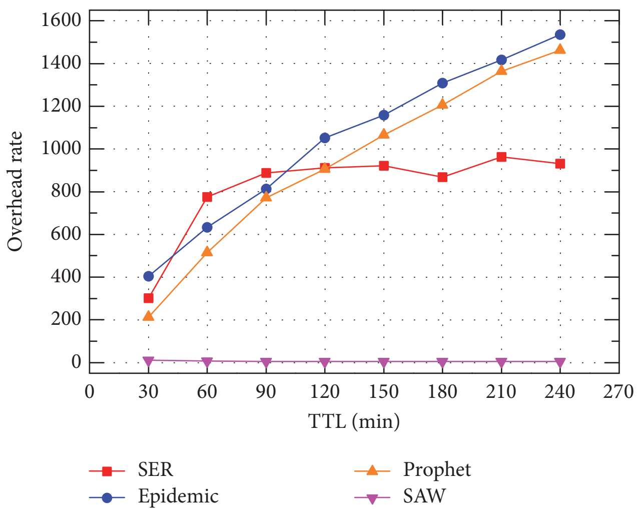
Figure 7 shows the performance of four routing schemes under varying TTL. As seen in Figures 7(a) and 7(b), our scheme SER has obvious advantages in delivery rate and average delivery delay compared with other three schemes. From the figure, TTL has little effect on SER. Even though the TTL is equal to 30 minutes, the delivery rate of SER still reached to 70 percent. When TTL is greater than 120 minutes, the delivery rate of SER reached to 88.5 percent and tends to a steady state. This is because SER adopts the following message forwarding strategy: (1) selecting the more trusted nodes as the next-hop relay nodes and (2) each message having multiple finite copies that are forwarded concurrently along different paths. However, the delivery rates of Epidemic and Prophet have the drop trends when the TTL increases. This is because these two schemes adopt the infinite message copies forwarding strategy; there are many messages that are not forwarded in time, which are deleted by nodes due to buffer capacity limitation and receiving the new messages. Although the delivery rate of SAW approaches SER when the TTL is increased to 240 minutes, SER has greater advantage in the average delivery delay; the other schemes have obvious growth trend with TTL increased. When the TTL is increased



(a) Delivery rate



(b) Average delivery delay

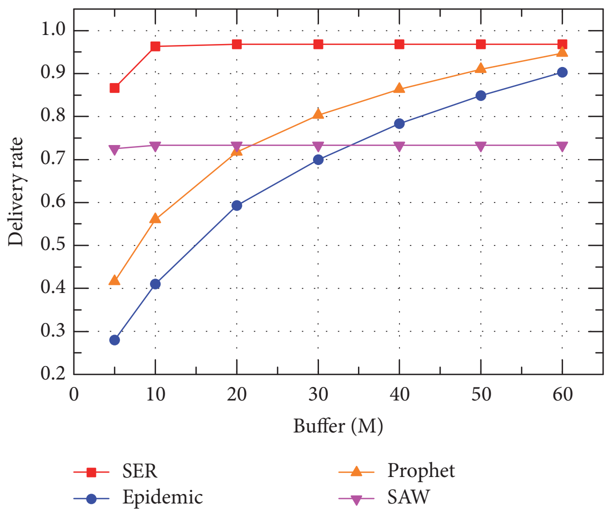


(c) Overheat rate

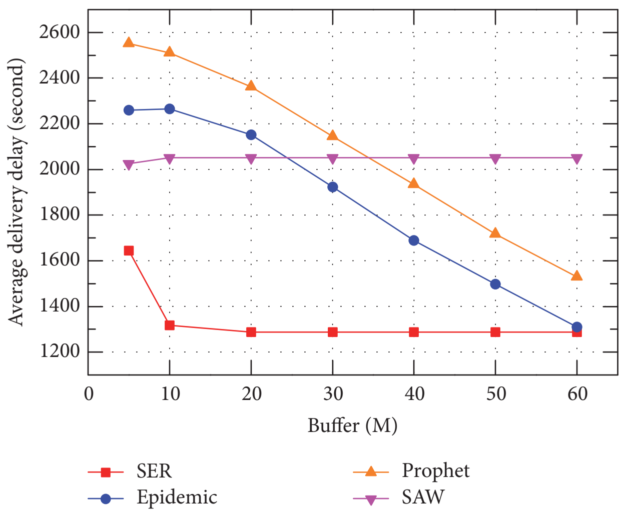
FIGURE 7: Routing performance under varying TTL.

to 240 minutes, the average delivery delay of SAW exceeds 3000 seconds while SER is only about 1800 seconds. This means that our SER scheme could find a trusted forwarding path with a short delay when the trust mechanism is adopted to forward the message. Figure 7(c) shows that SAW has the most obvious advantage in the overhead rate; this is because SAW adopts the single copy message forwarding strategy. The overheat rate of SER is a little higher than other schemes when TTL is less than 110 minutes.

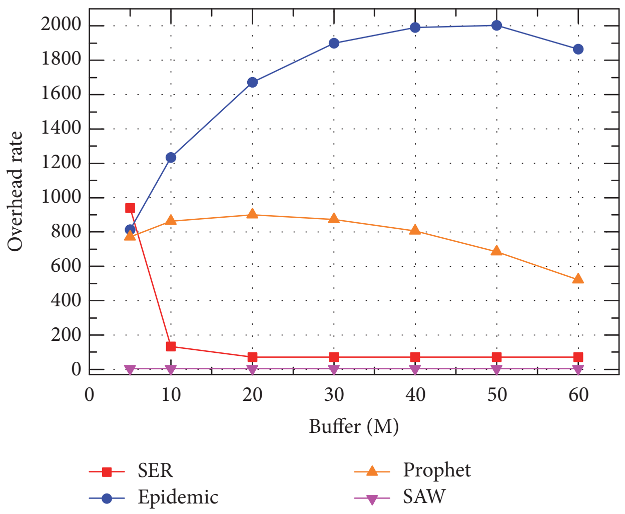
4.2.3. The Impact of Buffer Capacity on the Routing Performance. Time to live (TTL) of message is fixed to 90 minutes. The buffer B_m of node is varied from 5 M to 60 M. Figure 8 shows the performance of four routing schemes under varying buffer capacity. As seen in Figures 8(a) and 8(b), SER achieves the better performance in the delivery rate and average delivery delay by requiring small buffer. When the buffer capacity is greater than 10 M, the delivery rate of SER exceeds 96 percent and tends to have a steady state, but, correspondingly, the average delivery delay of SER is less than 1300 seconds. This result indicates that SER has low requirements of the buffer capacity and is suitable for buffer limited DTNs. The delivery rate of Epidemic and Prophet has obvious increasing trends, but, correspondingly, the average delivery delay of Epidemic and Prophet has the dropping trends when the buffer capacity increases. This is because Epidemic and Prophet have enough space to receive the new message and store the messages for long time. This means that Epidemic and Prophet have strong dependence on the buffer capacity and have lower efficiency in buffer limited DTNs. SAW has lower dependency on the buffer capacity than the other three schemes, but the routing performance of SAW is less efficient than SER. Figure 8(c) shows that the overhead rate of SER has the dropping trends with buffer capacity increases. When the buffer capacity is equal to 10 M, the overhead rate of SER is lower than 100 and tends to be in stable state, which indicates that SER could achieve the stable routing performance with varying buffer capacity.



(a) Delivery rate



(b) Average delivery delay

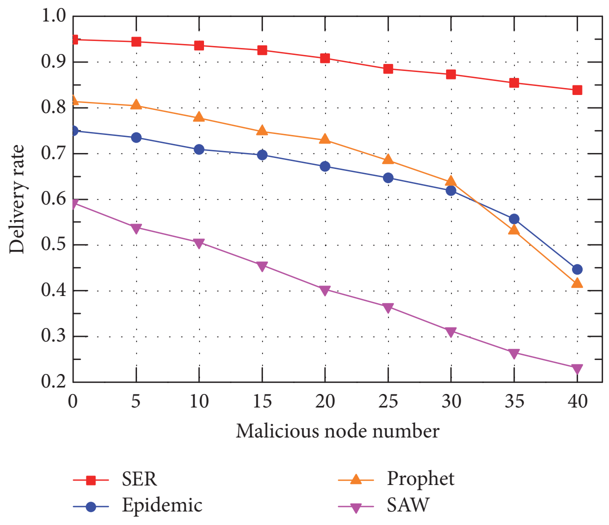


(c) Overheat rate

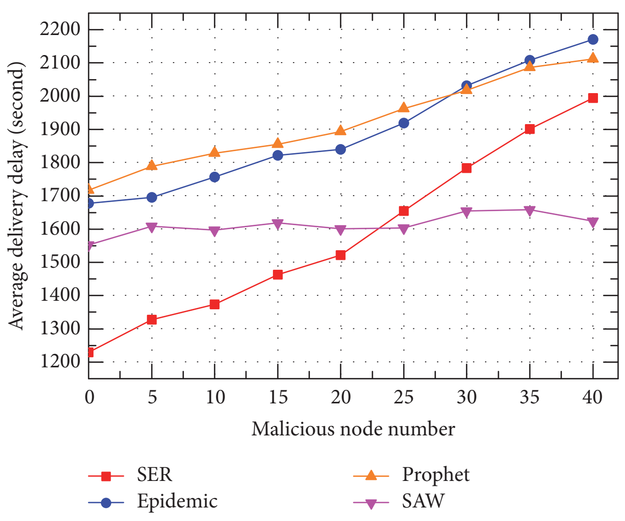
FIGURE 8: Routing performance under varying buffer capacity.

4.2.4. The Impact of the Number of Malicious Nodes on the Routing Performance. In this experiment, we assume that there are malicious nodes in DTNs. The number of malicious nodes is varied from 0 to 40. The dropping message probability of malicious nodes is fixed to 0.3. The threshold Th_{MBR} on malicious behavior detection of SER is set to 0.15. The buffer B_m of each node is fixed to 10 M. Time to live (TTL) of message is fixed to 60 minutes. We evaluate the routing performance of four schemes under varying number of malicious nodes.

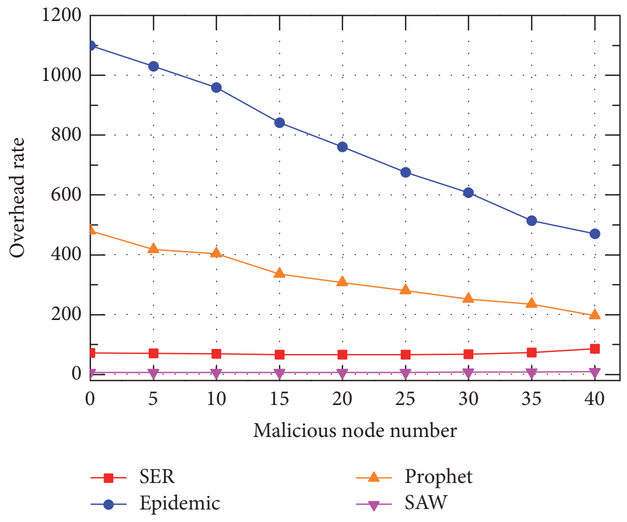
Figure 9(a) shows that the delivery rate of SER has the slight dropping trends, but the delivery rate of other schemes has the obvious dropping trends, when the number of malicious nodes increases. Even though the number of malicious nodes is increased to 40, the delivery rate of SER still exceeds 84 percent that is obviously higher than other schemes. This result indicates that SER has better effect of resisting malicious behavior by using malicious nodes detection mechanism and the trusted forwarding strategy. However, the average delivery delay of four schemes has obvious rising trends in Figure 9(b); this is because the total number of nodes in the experiment is fixed to 200; if the number of malicious nodes increases, the number of well-behaving nodes would decrease to lead to the increase of average delivery delay. The delivery rate of SAW has the obvious linear dropping trend, which indicates that the malicious nodes have big effect on the single copy strategy. As seen in Figure 9(c), the overhead rate of SER is lower than 100 and close to SAW, which indicates that SER could not only detect the malicious nodes, but also forward the message to the most proper next-hop relay node. The overhead rate of Epidemic and Prophet has the dropping trends, when the number of malicious nodes increases; this is because the total number of copies of the message in DTNs is decreased. As



(a) Delivery rate



(b) Average delivery delay



(c) Overheat rate

Figure 9: Routing performance under varying of the number of malicious nodes.

a result of the two-hop routing strategy, the overhead rate of SAW is very small; this is because fewer copies of messages are generated in DTNs.

5. Related Work

In recent years, many research works on misbehavior detection and routing in DTNs have been proposed, which are closely related to our SER scheme. In Prophet [11], Lindgren et al. first propose a probabilistic routing protocol for DTNs, which calculates the probability of a node contacting the destination node as the next-hop relay node selection strategies. A node will forward a message to the next-hop node only when the next-hop node has a higher probability of contacting the destination node. MaxProp [8] exploits priority of the transmitted path to schedule the messages to be forwarded and the messages to be dropped and stores a list of previous intermediaries to prevent message from forwarding twice to the same node. In ERB [12], to minimize overhead in terms of both extra traffic injected into the network and control overhead, ERB adopts historical encounter-based metric for optimization of message forwarding, where each node is only responsible for maintaining the past rate of encounter average to predict future encounter rates. SMART [16], SSAR [10], Bubble Rap [9], and SGBR [17] exploit the various social metrics to select the appropriate next-hop relay node, such as history of interaction, betweenness centrality, and community. Although the routing schemes mentioned above are very effective for improving route performance, they cannot address the security problems in DTNs.

To detect colluding blackhole and greyhole attacks, Pham and Yeo [6] designs a statistical-based detection scheme (SDBG) in which each node locally maintains the encounter records and the meeting summary with other nodes. When the nodes meet each other, they are required to exchange their encounter record histories, based on which other nodes can evaluate their forwarding behaviors. Alajeely et al. [3] present the detection attack and trace back mechanisms based on the Merkle tree, where the legitimate nodes can detect attack based on the received packets and then trace back and identify the malicious nodes. Zhu et al. [7] exploit a trusted authority (TA) to judge the forwarding behavior of nodes based on the collected delegation task evidence, forwarding history evidence, and contact history evidence. Chen et al. [13], Lu et al. [15], Ayday and Fekri [18], Li and Cao [14], Zhao et al. [19], and Chen and Chan [20] adopt the incentive mechanism to motivate selfish nodes to forward messages, which use reputation or credit to represent the forwarding behavior of nodes.

Different from existing routing protocols and misbehavior detection schemes, our proposed SER scheme introduces a trusted routing table (TRT) that contains the behavior history information of each node and the trust degree of forwarding the message to other nodes. We use the trusted routing table not only to analyze the behavior of nodes, but also to make effective routing decisions. Therefore, SER can achieve both the routing performance and the misbehavior detection and only cost the extra resource overhead that maintains the trusted routing table.

6. Conclusions

In this paper, we proposed a security and efficient routing scheme (SER), which has the dual functions of routing decision and malicious attacks detection. Based on the layered coin model and digital signature mechanism, the proposed forwarding evidences collection mechanism can effectively guarantee the security and authenticity of the forwarding evidence. Exploiting the forwarding evidence and historical contact information, we described in detail the build and update process of trusted routing table. By adopting the trusted routing table, the proposed SER scheme can obtain the global view about the contact relationship among all nodes in DTNs. The detailed analysis has shown that the trusted routing table not only is secure and reliable, but also quickly converges to global consistency. The simulation results show that SER could accurately detect the attacks behavior of malicious nodes or selfish nodes by analyzing the history forwarding behavior of node from the global view. In addition, the simulation results also demonstrate that SER has better routing performance compared with the existing algorithms, such as higher delivery rate and lower average delivery delay. For our future work, we will design the hierarchical trusted routing table and further reduce the network resource overhead.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by National Natural Science Foundation of China (Grant nos. 61300193, 61272125, and 61601107), the Hebei Natural Science Foundation (Grant no. F2015501105, no. F2015501122), and the Fundamental Research Funds for the Central Universities (Grant no. N120323012).

References

[1] N. Chakchouk, "A Survey on Opportunistic Routing in Wireless Communication Networks," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 4, pp. 2214–2241, 2015.

[2] S. Cc, V. Raychoudhury, G. Marfia, and A. Singla, "A survey of routing and data dissemination in Delay Tolerant Networks," *Journal of Network and Computer Applications*, vol. 67, pp. 128–146, 2016.

[3] M. Alajeely, R. Doss, A. Ahmad, and V. Mak-Hau, "Defense against packet collusion attacks in opportunistic networks," *Computers and Security*, vol. 65, pp. 269–282, 2017.

[4] J. Burgess, G. D. Bissias, M. D. Corner, and B. N. Levine, "Surviving attacks on disruption-tolerant networks without authentication," in *Proceedings of the MobiHoc'07: 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 61–70, Canada, September 2007.

[5] F. Li, J. Wu, and A. Srinivasan, "Thwarting blackhole attacks in distruption-tolerant networks using encounter tickets," in *Proceedings of the 28th Conference on Computer Communications, IEEE INFOCOM 2009*, pp. 2428–2436, bra, April 2009.

[6] D. Pham and C. K. Yeo, "Detecting colluding blackhole and greyhole attack in delay tolerant networks," *IEEE Transactions on Mobile Computing*, pp. 1–15, 2015.

[7] H. Zhu, S. Du, Z. Gao, M. Dong, and Z. Cao, "A probabilistic misbehavior detection scheme toward efficient trust establishment in delay-tolerant networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 22–32, 2014.

[8] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "MaxProp: routing for vehicle-based disruption-tolerant networks," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*, April 2006.

[9] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE Rap: social-based forwarding in delay-tolerant networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 11, pp. 1576–1589, 2011.

[10] Q. Li, W. Gao, S. Zhu, and G. Cao, "A routing protocol for socially selfish delay tolerant networks," *Ad Hoc Networks*, vol. 10, no. 8, pp. 1619–1632, 2012.

[11] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19-20, 2003.

[12] S. C. Nelson, M. Bakht, and R. Kravets, "Encounter-based routing in DTNs," in *Proceedings of the 28th Conference on Computer Communications (INFOCOM '09)*, pp. 846–854, IEEE, April 2009.

[13] I.-R. Chen, F. Bao, M. Chang, and J.-H. Cho, "Dynamic trust management for delay tolerant networks and its application to secure routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 5, pp. 1200–1210, 2014.

[14] Q. Li and G. Cao, "Mitigating routing misbehavior in disruption tolerant networks," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 664–675, 2012.

[15] R. Lu, X. Lin, H. Zhu, X. Shen, and B. Preiss, "Pi: a practical incentive protocol for delay tolerant networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 4, pp. 1483–1493, 2010.

[16] T. Zhou, R. R. Choudhury, and K. Chakrabarty, "Diverse routing: Exploiting social behavior for routing in delay-tolerant networks," in *Proceedings of the 2009 IEEE International Conference on Social Computing, SocialCom 2009*, pp. 1115–1122, Canada, August 2009.

[17] T. Abdelkader, K. Naik, A. Nayak, N. Goel, and V. Srivastava, "SGBR: A routing protocol for delay tolerant networks using social grouping," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 12, pp. 2472–2481, 2013.

[18] E. Ayday and F. Fekri, "An Iterative Algorithm for Trust Management and Adversary Detection for Delay-Tolerant Networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 9, pp. 1514–1531, 2012.

[19] H. Zhao, X. Yang, and X. Li, "CTrust: trust management in cyclic mobile Ad Hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 6, pp. 2792–2806, 2013.

[20] B. B. Chen and M. C. Chan, "MobiCent: a credit-based incentive system for disruption tolerant network," in *Proceedings of the 2010 IEEE INFOCOM*, pp. 1–9, March 2010.