A Privacy-Preserving and Secure Framework for Opportunistic Routing in DTNs

Lei Zhang, *Member, IEEE*, Jun Song, *Member, IEEE*, and Jianping Pan, *Senior Member, IEEE*

***Abstract*—Opportunistic routing has been extensively studied and utilized in delay/disruption-tolerant networks. The extensive use of nodes' local information, e.g., the distance to the destination or the contact frequency with the destination, in such routing schemes can cause severe security and privacy problems. Existing solutions of anonymous routing can introduce undesired overhead and fail to provide the confidentiality of the routing metric. In this paper, we propose an advanced framework for opportunistic routing schemes, providing the following properties: confidentiality of the nodes' routing metric, anonymous authentication, and efficient key agreement for pairwise communication. A comprehensive evaluation, including security analysis, efficiency analysis, and simulation evaluation, is presented to show the security and feasibility of the proposed framework.**

***Index Terms*—Delay/disruption-tolerant networks (DTNs), opportunistic routing, privacy, security.**

I. Introduction

In delay/disruption-tolerant networks (DTNs), message propagation is usually conducted in a multihop fashion with the help of store–carry–forward routing techniques. In the literature, different routing approaches can be divided into two categories, namely, topology-based and opportunity-based. Different from topology-based routing schemes, opportunistic routing schemes make routing decisions based on nodes' local information, making them more applicable for networks of large scale and with high dynamics [1]. Opportunistic routing has been extensively studied in DTNs, e.g., [2]. In most opportunistic routing algorithms, messages are forwarded to the nodes with a higher chance of delivery to the destination. Nodes in opportunistic routing schemes need to broadcast, exchange, and compare their local or individual information, e.g., the distance or visit frequency to the destination. In this paper, we call such information the *routing metric*.

However, from privacy and security perspectives, opportunistic routing can raise critical issues. A serious threat is the traffic analysis, where the network traffic can be observed by a malicious node, and then, it uses the information gathered to launch attacks. Moreover, the routing metrics, e.g., geographic location or contact history, are highly privacy sensitive. Without proper protection, severe privacy problems may occur.

Although the routing metric information is very privacy sensitive, most of the current work on the security and privacy of DTNs has neglected to protect it effectively. Much of such work [3]–[5], [9] focuses on node identity anonymity, with the help of techniques such as pseudonyms [9], group signature, and identity-based encryption [25]. On the other hand, some recent studies [10], [31] take the privacy issue of the "metric" information into consideration in social-based DTNs. However, because of their social relationship-based nature, such studies do not provide node identity anonymity.

To address these concerns, in this paper, we propose an advanced secure and privacy-preserving framework particularly for opportunistic routing schemes, integrating the following three properties. 1) The first property is the confidentiality of the routing metric. Protected by cryptographic tools, the routing metric is known only to its owner. However, to perform message routing, the framework allows a node to compare its own routing metric with others' without knowing the exact values of the others' routing metrics. This is achieved by integrating a solution to "Yao's millionaire problem" [11], which belongs to the secure multiparty computation problem. The protection of the routing metric, thus enhancing the node privacy, is the key feature that distinguishes our design from others. 2) The second property is anonymous authentication. Authentication is the fundamental mechanism for various security properties, i.e., data integrity, authenticity, and nonrepudiation. For the strong requirement of identity privacy [32], [33] in DTNs, anonymity is another essential property that must be provided. In this paper, we adopt a group-signature-based scheme to achieve anonymous authentication. 3) The third property is efficient key agreement. In DTNs, particularly in some mobile scenarios, e.g., mobile ad hoc networks, it is desirable for each pair of nodes to share a unique session key to achieve pairwise confidentiality. Considering the total number of session keys and the lack of central control in such distributed systems, efficient key management is crucial. In this paper, we adopt an efficient pairing-based key agreement scheme and integrate it seamlessly into the message routing process without creating much overhead.

A comprehensive evaluation of the proposed framework is provided. We first analyze the security of our design and

Manuscript received January 28, 2015; revised June 15, 2015 and August 16, 2015; accepted September 2, 2015. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, by the Canada Foundation for Innovation, and by the British Columbia Knowledge Development Fund. A preliminary version of this paper appeared at the 11th IEEE International Conference on Sensing, Communication, and Networking (SECON'14), Singapore. The review of this paper was coordinated by Prof. Y. Qian.

L. Zhang and J. Pan are with the Department of Computer Science, University of Victoria, Victoria, BC V8P 5C2, Canada (e-mail: leiz@uvic.ca; pan@uvic.ca).

J. Song is with the Department of Computer Science, China University of Geosciences, Wuhan 430074, China (e-mail: songjun@cug.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2015.2480761

then evaluate the performance with both cryptographic implementation specifications and event-driven simulations. These evaluations show the security and feasibility of the framework. Moreover, our framework can be applied to many opportunistic routing scenarios, e.g., mobile ad hoc networks or vehicular ad hoc networks (VANETs).

The rest of this paper is outlined as follows. Related work, security and privacy background, and related cryptographic techniques are introduced in Section II. Section III gives the detailed description of the proposed framework, including the system setup, the different algorithms involved, and security analysis. The performance evaluation is presented in Section IV. Section VI concludes this paper.

II. Background and Related Work

A. Related Work

Ranging from the physical layer to the application layer [6], security and privacy are always hot topics in DTN systems, i.e., VANETs [7], [9], [13], [14] and wireless sensor networks [8]. In [13] and [15], Papadimitratos *et al.* give comprehensive introductions on the basic assumptions, requirements, system models, adversary models, design principles, and a spectrum of VANET (which is a typical DTN system) security mechanisms.

With the special focus on anonymous routing, Cadger *et al.* [3] proposed a solution to separate the routing metric from a node's true identity, so that the attackers cannot link the privacy-sensitive routing metric to a specific node. In [4], Zhi and Choong utilized an anonymous table that stores pseudonyms along with the routing metric (position data in that paper) for the routing process. In [5], [9], and [31], extra servers or DTN gateway nodes or "roadside units" are deployed to manage the anonymous nodes, leading to the extra management overhead and security risk. None of such solutions provide the confidentiality of the routing metric, making it possible for attackers to spoil user privacy. Le *et al.* [34] and Shi *et al.* [35] adapted onion routing [36] to opportunistic networks for anonymity purposes; however, they require that the encryption keys of nodes are known to the message source node, which implies a complicated key management.

In terms of security, Patra *et al.* [14] used hierarchical identity-based cryptography to achieve authentication and key management. With a similar technique but by introducing pseudonyms, Kate *et al.* [9] achieved identity anonymity. The authors in [38], [39], and [45] preserved the location privacy of the sender using trusted social contacts. Recent studies [10], [40] took the privacy of the "metric" information into consideration. However, these papers did not provide user identity anonymity. Moreover, this work focused on a specific field, i.e., social-based DTN, where the strong social relationship (e.g., community) among nodes (e.g., cellphones) was utilized and is not applicable to more general DTN schemes, since the social relationship among nodes is not always sufficiently strong and explicit in some DTNs, particularly for mobile DTNs. Another direction of the DTN security study focuses on the detection and prevention of the attacks from the internal malicious node, e.g., black hole [31], [41]–[45] and Sybil attacks [46], [47]. This direction is from a different perspective and, thus, less related to the main focus of our paper.

Zhang *et al.* in [48] have looked into both anonymous routing and security. However, the scope of that work was only limited to VANETs, and the simulation evaluation was also limited to VANETs. In this paper, we expand our scope from VANETs to a more general network scenario, i.e., DTNs, so that our proposed framework can be applied to a wider range of applications. The simulation has also been redesigned to reflect the properties of DTNs and demonstrate the effectiveness of the proposed framework.

B. Security and Privacy Goals

We now introduce the general security and privacy properties that our design can provide.

- **Authentication**: Valid users must be authorized by a Certificate Authority (CA), and they can verify each other.
- **Data integrity**: A user should be able to detect the message change or damage, which is caused by either intentional or unexpected factors during its transmission.
- **Data confidentiality**: The secret data are only visible to eligible users.
- **Nonrepudiation**: No user can deny their past behaviors, e.g., signing, relaying a message, etc. Every node should be responsible for its behaviors.

Different from other schemes, when taking the routing metric issue into consideration, our scheme can provide privacy preservation in the following two aspects.

- **Identity anonymity**: The true identity of a user should not be exposed during any networking activity, including authentication, safety beacon broadcasting, etc.
- **Users' routing metric confidentiality**: As mentioned, the routing metric information has been extensively utilized in opportunistic routing. The protection of such information is essential to preserve the users' privacy.

Other requirements related to security management are revocation and traceability. Since they are less related to the routing process, we do not have them discussed in this work. However, we believe that, with the anonymous authentication in our framework, those properties are also achievable [19].

C. Threats and Adversaries

On the other hand, we review the possible threats and adversaries in the routing process, on which we focus.

1) Threats: Threats in mobile network systems can be categorized into two types: active and passive. In active attacks, the adversaries take active actions to incur damages to the network. Typical active attacks include the following.

- **Message forging/cheating**: The attackers send fake messages for malicious purposes. They can cheat on their identities, using fake identities to broadcast messages, e.g., Sybil attack, or they use their real identities but send messages containing fake information, e.g., dishonest routing metric, for malicious purposes.

- **Message modification/dropping**: Attackers may modify or damage the messages they received and forward them to other nodes, causing disorder. Attackers may even deliberately drop the messages to conduct a black-hole attack.
- **Message replay attack**: The adversary replays the messages previously sent to disturb the network.

For passive attacks, adversaries are usually referred to as "curious but honest," which means that they intend to peek at others' secret or private information but do not conduct active actions to spoil the system. Typical passive attacks include the following.

- **Message eavesdropping**: Because of the openly shared medium of wireless communications, "curious" attackers can easily eavesdrop on the conversation of others, causing damage to user confidentiality and privacy.
- **Privacy digging**: With the eavesdropped information, the attackers dig up more private information of others. For example, once the attacker intercepts the routing metric (e.g., the visiting frequency to certain locations) of a node, he may learn the node's mobility patterns.

2) Adversaries: Adversaries can be divided into external and internal adversaries. External adversaries are those who are not authorized by the CA or whose certificates are revoked by the CA. With the authentication scheme proposed in this paper, our framework can resist both passive and active attacks of the external adversaries, because the nodes who fail the authentication verification will be simply ignored by the authorized adversaries. In contrast, the internal adversaries are those who are authorized but malicious. They can conduct attacks until they are discovered, and then, they will be revoked from the trusted group. In this paper, we consider the internal adversaries to be passive attackers, which are "curious but honest." The discovery and resistance of internal active adversaries can be very complicated, and different attacks usually need very different solutions. We do not cover them in this paper.

D. Cryptographic Tools

Cryptographic tools are important for security scheme designs. Here, we briefly introduce the cryptographic tools used in our framework. First, as mentioned in Section I, our anonymous authentication function is achieved by a group signature scheme. Second, the protection of the routing metric confidentiality is essentially "Yao's millionaire problem," where homomorphic encryption is used as a main support of the solution. Finally, the pairing-based Sakai–Ohgishi–Kasahara (SOK) key agreement serves as the basis of the session key distribution.

1) Group Signature: Group signature is an efficient solution to achieve anonymity authentication. In group signature [20], network nodes are organized in groups, and each group has a group manager to represent the members. The main feature of the group signature scheme is that it provides anonymous authentication to the group members. A verifier can determine whether a signer is authorized by a group without knowing or linking the true identity of the signer. Different from the other anonymity techniques, group signature reduces the workload of the public key and certificate distribution and verification operations. As an authentication scheme, group signature can satisfy other basic security requirements, such as message integrity and nonrepudiation.

In this paper, we choose one of the group signature schemes as our anonymous authentication scheme. It is a bilinear-map-based authentication scheme, which is also adopted in an enhanced version [21] of the Directed Anonymous Attestation (DAA) [22]. The original DAA was adopted by the Trusted Computing Group for anonymous authentication purposes. It is essentially a group signature scheme.

2) Yao's Millionaire Problem: In [11], Yao first introduced a problem that is analogous to a more general problem, where there are two numbers a and b, and the goal is to verify the inequality $a \geq b$ without revealing the actual values of a and b. To achieve routing metric confidentiality, it is expected that a node can compare its routing metrics with others' without knowing the values of the others' routing metrics. In this paper, we integrate the solution proposed in [23] into our security framework, as the main idea explained below.

Let all the routing metrics be expressed in a binary form with a fixed length n. For each binary-form routing metric, two sets of its substrings can be constructed, i.e., *0-encoding* and *1-encoding*. For a binary-form routing metric $r = r_n r_{n-1}, \ldots, r_1$, its 0-encoding set S_r^0 is defined as

$$S_r^0 = \{r_n r_{n-1}, \ldots, r_{i+1} 1 | r_i = 0, 1 \leq r \leq n\} \qquad (1)$$

whereas its 1-encoding set S_r^1 is defined as

$$S_r^1 = \{r_n r_{n-1}, \ldots, r_{i+1} r_i | r_i = 1, 1 \leq r \leq n\}. \qquad (2)$$

A very important conclusion is that for two routing metric values x and y, $x > y$ if there is one common element in both S_x^1 and S_y^0 [23]. This is easy to prove. If $x > y$, there must be a position i so that the substring $r_n^x r_{n-1}^x, \ldots, r_{i+1}^x$ is the same as $r_n^y r_{n-1}^y, \ldots, r_{i+1}^y$; however, $r_i^x = 1$, and $r_i^y = 0$. Thus, with the construction of 0-encoding and 1-encoding sets previously described, for S_x^1, it must contain an element $r_n^x r_{n-1}^x, \ldots, r_i^x$; for S_y^0, it must contain an element $r_n^y r_{n-1}^y, \ldots, r_{i+1}^y 1$, which is identical to $r_n^x r_{n-1}^x, \ldots, r_i^x$.

3) Homomorphic Encryption: In the implementation of the solution to Yao's millionaire problem, the homomorphism property of ElGamal encryption is utilized. Encryption schemes with the homomorphism property are referred to as homomorphic encryption. The homomorphism property allows a specific type of operation, e.g., $\otimes$, to be applied directly on two ciphertexts, e.g., $Enc(p_1)$ and $Enc(p_2)$, to obtain a result $R = Enc(p_1) \otimes Enc(p_2)$, which can be decrypted. The decryption of R is a result obtained from applying another operation, e.g., $\odot$, on the corresponding plaintexts, which means $D(R) = p_1 \odot p_2$. The operation $\odot$ can be either multiplication or addition, corresponding to multiplication homomorphic and addition homomorphic, respectively. The homomorphism property is a desirable feature since it can operate directly on the ciphertexts, without exposing the plaintexts to the parties performing the operations.

4) SOK Key Agreement: To achieve data confidentiality and for efficiency consideration, the secret messages are usually

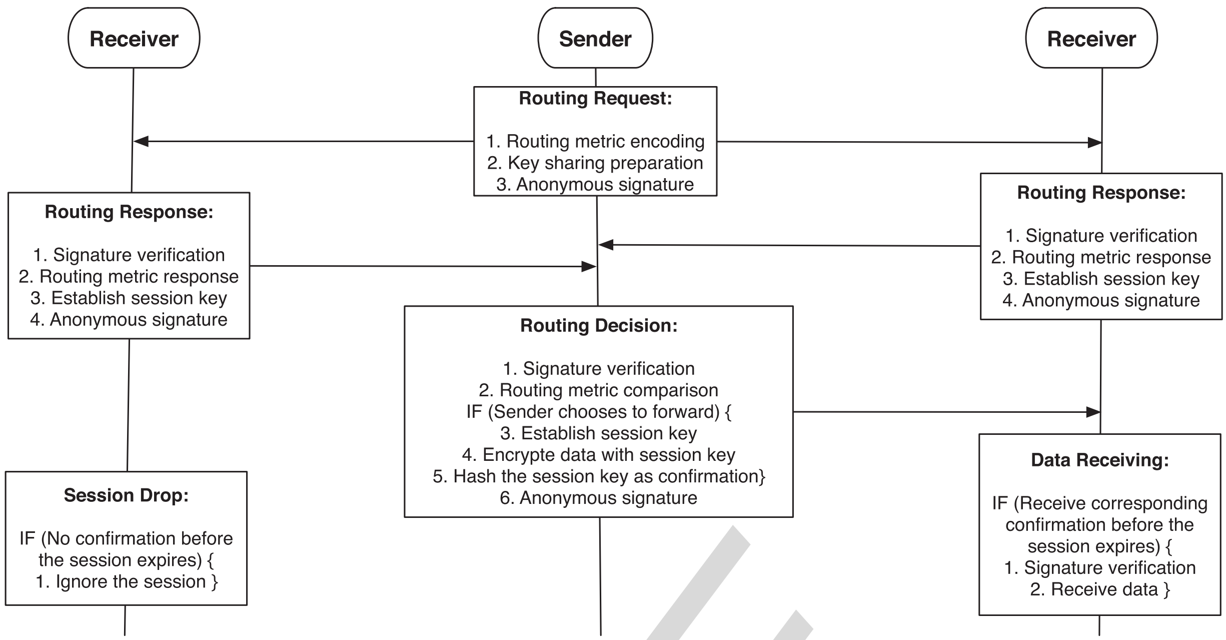


Fig. 1. Protocol flow.

encrypted by symmetric encryption schemes, such as AES. Considering the ad hoc environment, it is crucial to have an efficient and lightweight key agreement scheme to manage the huge number of session keys since each pair of users should share a distinct session key. We deploy a key agreement scheme similar to the SOK scheme [24], which has been also utilized in DTNs [9]. In the SOK key agreement, there are two groups $\mathbb{G}$ (written additively) and $\mathbb{G}_T$ (written multiplicatively) of order p (a large prime number) and an efficiently computable bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Assume that the CA possesses a master secret key $s \in \mathbb{Z}_q$ and that each user possesses an identity ID. The CA constructs each user i's secret key by calculating $d_i = sH(ID_i) \in \mathbb{G}$, where $H(\cdot)$ is a public hash function mapping an input to an element in $\mathbb{G}$. Under such a scheme, two users authorized by the same CA can noninteractively compute a shared session key with the identity of the other participant and their own private keys. For example, for users a and b, we have

$$\begin{aligned} Key_{ab} &= \hat{e}\left(H(ID_a), d_b\right) = \hat{e}\left(d_a, H(ID_b)\right) \\ &= \hat{e}(H(ID_a), H(ID_b))^s. \end{aligned} \tag{3}$$

Dupont and Enge [26] proved that this key agreement is secure in the random oracle model under the bilinear Diffie–Hellman (BDH) assumption in $\langle \mathbb{G}, \mathbb{G}_T, \hat{e} \rangle$.

III. Framework Design

Here, we provide the detailed framework design toward a privacy-preserving and secure opportunistic routing in DTNs. Without the topology information and route maintenance processes, routing decision in opportunistic routing is made by exchanging and comparing the routing metrics among individuals; hence, the nodes that have a larger chance at delivering the message, i.e., nodes with larger routing metric values, are chosen as the relays. Under such a scenario, any one-hop routing follows the protocol flow shown in Fig. 1. Four main algorithms are involved in the routing, namely, Routing Request (see Algorithm 3), Routing Response (see Algorithm 5), Routing Decision (see Algorithm 7), and Decision Confirm (see Algorithm 8).

Anonymous authentication is mainly provided in Sign and Verify algorithms, i.e., Algorithm 1 and Algorithm 2, respectively. For security concerns, every message sent should be signed first by the sender. The messages that failed to pass the verification will be automatically dropped by the receivers. To achieve the confidentiality of the routing metrics during the routing, Algorithm 4 and Algorithm 6 are embedded in the Routing Request and Routing Response algorithms, respectively. Some necessary processing of the routing metric information is also performed by these two algorithms.

A. Protocol Setup

The notations of our framework are listed in Table I. Our design is based on the finite-field cryptography, and the cryptographic setup is presented as follows. First, three cyclic groups are chosen: $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, of sufficiently large prime order q. Two random generators are selected such that $\mathbb{G}_1 = \langle P_1 \rangle$ and $\mathbb{G}_2 = \langle P_2 \rangle$ along with a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We write $\mathbb{G}_1$, $\mathbb{G}_2$ additively, and $\mathbb{G}_T$ multiplicatively. The pairing $\hat{e}$ is a map [19] with the following properties.

1) $\hat{e}$ is bilinear, which means $\hat{e}(aP_1, bP_2) = \hat{e}(P_1, P_2)^{ab}$ for any two integers a and $b \in \mathbb{Z}_q$.
2) $\hat{e}$ is nondegenerate, which means $\hat{e}(P_1, P_2) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is the identity element of $\mathbb{G}_T$.
3) $\hat{e}$ is computable, i.e., there is a polynomial-time algorithm for computing $\hat{e}(P, Q)$ for any $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.

Second, two hash functions are selected, i.e., $H_1 : \{0,1\}^* \to \mathbb{Z}_q$ and $H_2 : \{0,1\}^* \to \mathbb{G}_1$, mapping an arbitrary-length binary string to an integer and a $\mathbb{G}_1$ element, respectively.

TABLE I
NOTATIONS

Notation	Explanation
$\mathbb{G}_1, \mathbb{G}_2$	Two additive cyclic groups with order q
$\mathbb{G}_T$	A multiplicative cyclic group with order q
$\mathbb{Z}_q$	A integer cyclic group with order q
P_1, P_2	Generators for $\mathbb{G}_1$ and $\mathbb{G}_2$
$\hat{e}$	A bilinear map: $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$
n_t	A timestamp
s, r	Sender and receiver node, respectively
rq, rp	Routing request and routing response, respectively
mtr	Routing metric of a node
(pk, sk)	A key pair: (public key, secret key)
$\mathcal{SL}$	Sending list, containing the chosen relays
EH, DE	Encryption and decryption with ElGamal
Eec, Dec	Encryption and decryption of a symmetric cryptosystem, e.g., AES
n	The fixed length of the binary form of the routing metric
TB	A table of ciphertexts with 2 columns and n rows
CR	A list of ciphertexts with size n
S^0, S^1	0-encoding and 1-encoding sets of a binary string

Third, each node has a true and secret identity $f \in \mathbb{Z}_q$. The CA, who issues the certificates, has a secret key (x, y), where $x, y \leftarrow \mathbb{Z}_q$, and a public key (X, Y), where $X = x \cdot P_2 \in \mathbb{G}_2$, $Y = y \cdot P_2 \in \mathbb{G}_2$. The CA manages the true identities of all nodes and issues a certificate to each node. The certificate is a triplet (A, B, C), where $A \leftarrow r \cdot P_1$, $B \leftarrow y \cdot A$, and $C \leftarrow (x \cdot A + fxy \cdot A)$. The number r is randomly chosen from $\mathbb{Z}_q$; hence, for a specific node with identity f, its certificate is not deterministic. As we can see, the certificate is constructed with the secret key of the CA, i.e., (x, y), which is the main proof of the CA's attestation. It is also constructed with the true ID of the corresponding node, i.e., f, so that each certificate is specifically created for that specific node.

B. Signing and Verification

The signing and verification protocols are used to achieve anonymous authentication. The authentication is required for all messages, which means every message has to be signed before they are sent out. For every message received from other nodes, its signature needs to be verified by the receiver. In this paper, we deploy a scheme similar to DAA [19], [22], which is a group signature scheme.

Algorithm 1 performs the signing on the message and generates a signature σ. It contains a triplet (R, S, T), which can be seen as a shuffle of the true certificate, i.e., (A, B, C), so that every message is signed with an anonymous certificate. The calculation of (J, K, L, c, s) is used to provide the proof of the connection between the certificate and the node's true identity f. n_t is a timestamp providing time information, which is embedded into the message signature to resist the replay attack. n_c is a nonce that should be used in the same request, response, decision, and confirmation session. Different from the schemes in [19] and [22], there are two versions of the Sign algorithm. Version 1 is for the normal usage, and version 2 is only used when the sender wants to establish session keys with the possible relays, where a key agreement process will be executed with the help of P and Q. The details of the key agreement process will be discussed in the Routing Response algorithm, i.e., Algorithm 5.

Algorithm 1 Sign

1: **procedure** SIGN (Message msg)
2: $\quad a \leftarrow \mathbb{Z}_q;\ z \leftarrow \mathbb{Z}_q$
3: $\quad J \leftarrow H_2(msg);\ K = f \cdot J;\ L \leftarrow z \cdot J$
4: $\quad R \leftarrow a \cdot A;\ S \leftarrow a \cdot B;\ T \leftarrow a \cdot C;\ \tau \leftarrow \hat{e}(S, X)^z$
5: $\quad c \leftarrow H_1(R \| S \| T \| \tau \| J \| K \| L \| n_t \| n_c \| msg)$
6: $\quad s \leftarrow z + c \cdot f \pmod{q}$
7: **If** Version 1 **then**
8: $\quad \sigma \leftarrow (R, S, T, J, K, c, s, n_t, n_c, TTL)$
9: **else if** Version 2 **then**
10: $\quad b \leftarrow \mathbb{Z}_q$
11: $\quad P \leftarrow b \cdot A;\ Q \leftarrow b \cdot B$
12: $\quad \sigma \leftarrow (R, S, T, J, K, c, s, P, n_t, n_c, TTL)$
13: **end if**
14: **return** σ
15: **end procedure**

Verification of the signature is described in Algorithm 2. At the beginning, a few inspections are performed for a quick verification. First, data integrity of the message is provided by checking whether $J \neq H_2(msg)$, so that any corruption of the message can be detected. Second, by a quick comparison of $\hat{e}(R, Y)$ and $\hat{e}(S, P_2)$, it checks the internal relationship between R, S, and Y, i.e., $S = a \cdot B = ay \cdot A = y \cdot R$, so that $\hat{e}(R, Y) = \hat{e}(A, P_2)^{ay} \equiv \hat{e}(S, P_2)$.

Algorithm 2 Verify

1: **procedure** VERIFY(Message msg, Signature σ)
2: $\quad$ **if** TTL has elapsed or $J \neq H_2(msg)$ or $\hat{e}(R, Y) \neq \hat{e}(S, P_2)$ **then**
3: $\quad\quad$ **return** Reject
4: $\quad$ **end if**
5: $\quad \rho_a^\dagger \leftarrow \hat{e}(R, X);\ \rho_b^\dagger \leftarrow \hat{e}(S, X);\ \rho_c^\dagger \leftarrow \hat{e}(T, P_2)$
6: $\quad \tau^\dagger \leftarrow (\rho_b^\dagger)^s \cdot (\rho_c^\dagger / \rho_a^\dagger)^{-c}$
7: $\quad L^\dagger \leftarrow s \cdot J - c \cdot K$
8: $\quad$ **if** $c \neq H_1(R \| S \| T \| \tau^\dagger \| J \| K \| L^\dagger \| n_t \| n_c \| msg)$ **then**
9: $\quad\quad$ **return** Reject
10: $\quad$ **end if**
11: **return** Accept
12: **end procedure**

The following verification, i.e., lines 5–7, is a recovering process of τ and L. If signature σ is correctly generated by the signer and is successfully transmitted without any corruption, τ and L should be recovered by calculating $\tau^\dagger$ and $L^\dagger$. The correctness is shown as follows: First

$$L^\dagger = s \cdot J - c \cdot K = (s - cf) \cdot J \equiv L \tag{4}$$

second

$$\begin{aligned}\tau^\dagger &= \left(\rho_b^\dagger\right)^s \cdot \left(\frac{\rho_c^\dagger}{\rho_a^\dagger}\right)^{-c} = \hat{e}(S, X)^s \cdot \hat{e}(T, P_2)^{-c} \cdot \hat{e}(R, X)^c \\ &= \hat{e}(S, X)^s \cdot \hat{e}(P_1, P_2)^{-acxr(1+fy)+acxr} \\ &= \hat{e}(S, X)^{s+cf} = \hat{e}(S, X)^z \equiv \tau.\end{aligned} \tag{5}$$

If τ and L are successfully recovered and other fields, e.g., n_t, msg, etc., are successfully transmitted, the verifier should be able to recover c in line 8 to finish the verification.

C. Routing

The routing procedure is shown in Fig. 1. Before the data transmission, the sender first broadcasts a request message, i.e., rq in Algorithm 3, asking other nodes for their routing metrics. Once a neighbor node receives a request, it broadcasts a response, i.e., rp in Algorithm 5, which contains its routing metric. Based on the received responses, the sender makes the routing decision in Algorithm 3 and chooses those that have larger metrics as the relays. By checking the decision announcement of the sender, the receiver decides on its next action, as shown in Algorithm 8: receiving the data if it is chosen as the relay or ignoring the data otherwise. During the request and response processes, the sender and each chosen relay also finish the key agreement process to establish a unique pairwise key, so that they can secretly communicate for the following data transmissions.

Algorithm 3 Routing Request

1: **procedure** ROUTINGREQUEST
2: $\quad \{pk_s, sk_s\} \leftarrow \mathbb{Z}_q$
3: $\quad T \leftarrow Encoding(mtr_s, pk_s)$
4: $\quad msg.\text{data} = T \| pk_s$
5: $\quad \sigma \leftarrow \text{Sign}_{v2}(msg)$
6: $\quad$ Keep track of $\sigma.P$ and Q
7: $\quad$ Keep track of the request TTL TTL_{rq}
8: $\quad$ **return** $rq \leftarrow (msg, \sigma)$
9: **end procedure**

The routing procedure is straightforward; hence, we focus on the implementation of the two main security properties: routing metric confidentiality and key agreement.

1) Routing Metric Confidentiality: During the routing "request-response" phase, the sender inquires, obtains, and compares other nodes' routing metrics. Then, it chooses those that have a higher chance than itself to deliver the message to be the next relay. To keep the confidentiality of the routing metrics, we require that the sender has no access to the plaintext of the routing metrics; instead, it performs the comparison without revealing the actual value of others' metric information, which is known as Yao's millionaire problem. In this paper, we choose and integrate a solution from [23], which is based on the homomorphic encryption, into our framework. Recall the homomorphism property mentioned in Section II. To be specific, the multiplicative homomorphism of the ElGamal encryption system, which is denoted as $EH(\cdot)$, is utilized, i.e., line 9 in Algorithm 6, so that $EH(x_1) \otimes EH(x_2) = EH(x_1 \cdot x_2)$. The ciphertext of the ElGamal encryption is a pair of values, e.g., (a, b), and the operation $\otimes$ is defined as $EH(x_1) \otimes EH(x_2) = (a_1, b_1) \otimes (a_2, b_2) = (a_1 \cdot a_2, b_1 \cdot b_2)$.

The main idea of the solution to Yao's millionaire problem is described in Section II, i.e., by checking whether there is a common element in the sender's 1-encoding set S_s^1 and the receiver's 0-encoding set S_r^0, the sender can determine whether its routing metric mtr_s is larger than that of the receiver mtr_r. The implementation details are described as follows. During the routing request phase, the sender performs the Encoding algorithm, i.e., Algorithm 4, using its own routing metric mtr_s to construct a $2 \times n$ table TB, i.e., lines 3–7. Essentially, TB integrates the S_s^1 of the sender metric in an anonymous way, since each element in the table is in a ciphertext form. TB is then included in the routing request message rq and broadcast to the potential relay nodes.

Algorithm 4 Encoding

1: **procedure** ENCODING (mtr, pk)
2: $\quad$ Convert mtr to binary form $c_n c_{n-1}, \ldots, c_1 \in \{0, 1\}^n$
3: $\quad$ Initialize T as a $2 \times n$ table
4: $\quad$ **for** k from n to 1 **do**
5: $\qquad TB[c_k, k] = EH_{pk}(1)$
6: $\qquad TB[\bar{c_k}, k] = EH_{pk}(r)$ for a random r
7: $\quad$ **end for**
8 $\quad$ **return** TB
9: **end procedure**

Upon receiving the request, the receiver performs the Routing Response algorithm, i.e., Algorithm 5, to make a response to the request. The Coding Response algorithm, i.e., Algorithm 6, is called at this point. The algorithm first derives the 0-encoding set S_r^0 of the receiver's mtr_r, i.e., lines 3–7. Then, along with the table TB from the sender, it generates CR, i.e., lines 8–11, where each element c_t is the result of applying $\otimes$ on the ciphertexts in TB following some rules defined by the element of S_r^0, i.e., t and S_r^0. Hence, S_r^0 is integrated in CR. Because of the homomorphism of the ElGamal encryption EH, each c_t is essentially a ciphertext encrypted by EH. Up to line 11, the size of CR is determined by the number of elements in S_r^0, i.e., $|S_r^0|$. Extra $n - |S^0|$ random ciphertexts are padded into CR for security considerations, i.e., lines 12–14. Details will be provided in the security analysis in Section IV.

Algorithm 5 Routing Response

1: **procedure** ROUTINGRESPONSE(Request rq)
2: $\quad$ **if** Verify(rq) fails **then**
3: $\qquad$ **return** Ignore
4: $\quad$ **end if**
5: $\quad CR \leftarrow CodingResponse(mtr_r, rq.T, rq.pk_s)$
6: $\quad msg.\text{data} = CR$
7: $\quad n_c = rq.n_c$
8: $\quad \sigma \leftarrow \text{Sign}_{v2}(msg)$
9: $\quad$ Keep track of $\sigma.P$ and Q
10: $\quad Key \leftarrow \hat{e}(rq.\sigma.P, Q)$
11: $\quad$ Keep track of the response TTL TTL_{rp}
12: $\quad$ **return** $rp \leftarrow (msg, \sigma)$
13: **end procedure**

Algorithm 6 Coding Response

1: **procedure** CODINGRESPONSE(mtr, TB, pk)
2: Convert mtr into binary form $c_n c_{n-1}, \ldots, c_1 \in \{0,1\}^n$
3: **for** k from n to 1 **do**
4: **if** $c_k == 0$ **then**
5: Add binary string $c_n c_{n-1}, \ldots, c_{k-1}1$ into set S^0
6: **end if**
7: **end for**
8: **for** each $t = t_n t_{n-1}, \ldots, t_k$ in S^0 **do**
9: $c_t = TB[t_n, n] \otimes TB[t_{n-1}, n-1] \otimes, \ldots, \otimes TB[t_i, i]$
10: Add c_t to set CR
11: **end for**
12: **for** k from 1 to $n - |S^0|$ **do**
13: Add $EH_{pk}(r)$ to set CR for a random r
14: **end for**
15: **return** CR
16: **end procedure**

Then, the receiver sends CR back to the sender. In Algorithm 7, when a sender receives CR's, it decrypts the ciphertexts contained in each CR_i from receiver i. If there is a result equal to 1, it means there is a common element between S_s^1 and $S_{r_i}^0$, and the sender's metric is larger than that of receiver i. Thus, the sender will not choose i as its next relay. This conclusion is due to the ingenious constructions of TB and CR. However, if all ciphertexts in CR_i are not decrypted to 1, it means that the metric of receiver i is larger than that of the sender and that receiver i can be chosen as the next relay.

Algorithm 7 Routing Decision

1: **procedure** ROUTINGDECISION(Response $\{rp_1, rp_2, \ldots\}$)
2: **if** TTL$_{rq}$ has elapsed **then**
3: Ignore all responses
4: **end if**
5: **for** Any rp_i in $\{rp_1, rp_2, \ldots\}$ **do**
6: **if** Verify(rp_i) fail or $rp_i.n_c \neq rq.n_c$ **then**
7: **return** Ignore
8: **end if**
9: $CR \leftarrow rp_i.msg$.data
10: **for** Any t in set CR **do**
11: $k = DE_{sk_s}(t)$
12: **if** $k == 1$ **then**
13: Go to line 2 and try another rp
14: **end if**
15: **end for**
16: $Key_i \leftarrow \hat{e}(rp_i.\sigma.P, Q)$
17: Add (rp_i, Key_i) to $\mathcal{SL}$
18: **end for**
19: **for** Any (rp, Key) in $\mathcal{SL}$ **do**
20: msg.data $\leftarrow Enc_{Key}(Message)$
21: Send announcement $anc \leftarrow H(Key)$
22: $n_c = rp.n_c$
23: $\sigma \leftarrow$ Sign$_{v1}(msg)$
24: Send data $\leftarrow (msg, \sigma)$
25: **end for**
26: **end procedure**

Algorithm 8 Decision Confirm

1: **procedure** DECISIONCONFIRM(Announcement $\{anc\}$)
2: **if** $H(Key) == anc$ and TTL$_{rp}$ has not elapsed **then**
3: Receive data
4: **if** Verify(data) fails **then**
5: **return** Reject
6: **else**
7: $Message \leftarrow Dec_{Key}$(data)
8: **return** Accept
9: **end if**
10: **else**
11: **return** Ignore
12: **end if**
13: **end procedure**

2) Key Agreement: During the routing request and response processes, the sender and each of the chosen relays establish a unique secret session key, with which the data can be encrypted so that pairwise confidentiality can be achieved. As mentioned, the second version of the **Sign** algorithm is used for the key agreement purpose. Assume that b_s and b_r are two random numbers generated by the sender and the receiver, respectively. In the second version of Algorithm 1, when sending a request, the sender calculates $P_s = b_s \cdot A_s$ and $Q_s = b_s \cdot B_s$ and broadcasts P_s. In Algorithm 5, when a receiver receives P_s, it first generates $P_r = b_r \cdot A_r$ and $Q_r = b_r \cdot B_r$ and then obtains a session key $Key_{rs} = \hat{e}(P_s, Q_r) = \hat{e}(b_s \cdot A_s, b_r \cdot B_r) = \hat{e}(A_s, B_r)^{b_s b_r} = \hat{e}(P_1, P_1)^{r_s b_s r_r y b_r}$. Note that this session key is only valid when the receiver is chosen by the sender as a relay. The receiver includes its P_r in its response rp to the sender. According to the responses received, the sender chooses the proper receiver as the relay and establishes the session key $Key_{rs} = \hat{e}(P_r, Q_s) = \hat{e}(b_r \cdot A_r, b_s \cdot B_s) = \hat{e}(A_r, B_s)^{b_r b_s} = \hat{e}(P_1, P_1)^{r_r b_r r_s y b_s}$.

D. Traceability

For privacy concerns, it is not desired to trace back the signer's true identity from its signature. This is also the reason for introducing anonymous authentication. However, we still reserve the tracing ability of the CA for management purposes. The tracing can be only conducted by the CA since it is trusted by anyone else. Algorithm 9 shows the tracing process. Since the CA knows the true identities of every user and system secret key x, it can verify the ownership of the certificate (i.e., R, T, and S), which is attached in the signature, by performing a matching with the internal relationship of (R, S, T), i.e., line 4. The correctness is shown as

$$\begin{aligned} \sigma.T &= a \cdot \sigma.C \\ &= a \cdot x \cdot \sigma.A + a \cdot fxy \cdot \sigma.A \\ &\equiv x \cdot \sigma.R + x \cdot f \cdot \sigma.S. \end{aligned} \tag{6}$$

Algorithm 9 Traceability

1: **procedure** TRACE (Signature σ)
2:   **if** Verify(σ) regardless of its TTL successfully **then**
3:     **for** Any f in $\{f_1, f_2, \dots\}$ **do**
4:       **if** $\sigma.T == x \cdot \sigma.R + x \cdot f \cdot \sigma.S$ **then**
5:         The signature is traced to identity f
6:         **return** Found
7:       **end if**
8:     **end for**
9:     **return** Not Found
10:   **Else**
11:     **return** Ignore
12:   **end if**
13: **end procedure**

E. Security Analysis

1) Security of the Signature: The security of the signature is guaranteed by the hardness of the **LRSW Assumption** [27]: Suppose that a *Setup*(1^k) algorithm generates a multiplicative group $\mathbb{G}$ with a generator g and an order q, where k is a parameter related the security level. There exist $X, Y \in \mathbb{G}$, $X = g^x$, and $Y = g^y$. Let $O_{X,Y}(\cdot)$ be an oracle that, with an input value of $m \in \mathbb{Z}_q$, outputs a triplet (a, a^y, a^{x+mxy}) for a randomly chosen $a \in \mathbb{G}$. Then, for all probabilistic polynomial-time adversaries $\mathcal{A}$, $v(k)$ is a negligible function defined as follows:

$$\Pr\Big[(q,\mathbb{G},g) \leftarrow Setup(1^k); x \leftarrow \mathbb{Z}_q; y \leftarrow \mathbb{Z}_q \\ X = g^x; Y = g^y; (m,a,b,c) \leftarrow \mathcal{A}^{O_{X,Y}}(q,\mathbb{G},g) \\ a \in \mathbb{G} \wedge b = a^y \wedge c = a^{x+mxy}\Big] = v(k). \tag{7}$$

This means that given the group setup $(q, \mathbb{G}, g)$ and the system public key (X, Y), it is impossible for a polynomial-time adversary to construct a triplet (a, a^y, a^{x+mxy}) without knowing the secret key (x, y), where a and m are random numbers. This assumption guarantees the effectiveness of our authentication scheme. Only the CA who possesses the secret key x and y can construct valid certificates to users. Without knowing the secret key, a malicious node can hardly forge a valid certificate $(A, B = y \cdot A, C = x \cdot A + fxy \cdot A)$, where the group in our scheme, i.e., $\mathbb{G}_1$, is additively written but isomorphic to the multiplicative form in the given assumption.

2) Security of the Key Agreement Process: The security of the key agreement is guaranteed by the **BDH assumption** [25]. Suppose that $\mathbb{G}_1$ is an additive group with generator g, $\mathbb{G}_2$ is a multiplicative group, and $\hat{e}$ is a bilinear map of $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, as described in Section II. Let $P \in \mathbb{G}_1$, $a, b, c \leftarrow \mathbb{Z}_q$, then $a \cdot P, b \cdot P, c \cdot P \in \mathbb{G}_1$. Let $O_{a \cdot P, b \cdot P, c \cdot P}(\cdot)$ be an oracle that outputs $r = \hat{e}(P, P)^{abc} \in \mathbb{G}_2$. Then, for all probabilistic polynomial-time adversaries $\mathcal{A}$, $v(k)$ is a negligible function defined as follows:

$$\Pr\Big[(q,\mathbb{G}_1,g,\mathbb{G}_2,\hat{e}) \leftarrow Setup(1^k); a \leftarrow \mathbb{Z}_q; b \leftarrow \mathbb{Z}_q \\ c \leftarrow \mathbb{Z}_q; P \leftarrow \mathbb{G}_1; r \leftarrow \mathcal{A}^{O_{a\cdot P, b\cdot P, c\cdot P}}(q,\mathbb{G}_1,g,\mathbb{G}_2,\hat{e}) \\ r = \hat{e}(P,P)^{abc}\Big] = v(k). \tag{8}$$

In the key agreement process mentioned in Section III, the session key established by any two nodes a and b can be expressed as $\hat{e}(A_s, B_r)^{b_s b_r} = \hat{e}(A_r, B_s)^{b_s b_r} = \hat{e}(P_1, P_1)^{b_a r_a b_b r_b y}$, where random numbers r_a and r_b are introduced in the certificate construction process and random numbers b_a and b_b are introduced in the second version of the Sign algorithm, i.e., Algorithm 1. During the wireless transmission, an adversary can easily eavesdrop on $P_a = b_a r_a \cdot P_1$ and $P_b = b_b r_b \cdot P_1$. Assume that it also knows the system public key $Y = y \cdot P_1$. Because of the hardness of the BDH assumption, the probability of the adversary being able to recover the session key $\hat{e}(P_1, P_1)^{b_a r_a b_b r_b y}$, with P_a, P_b, and Y known, is negligible.

3) Security of the Routing Metric Comparison: The correctness has been explained in Session III. As the way how S^0 and S^1 are constructed, the confidentiality of the routing metric lies in the confidentiality of these two sets. The 1-encoding set S_s^1 of the sender's routing metric is embedded into the table TB, which is broadcast during the routing request phase. From the adversary point of view, the table TB reveals no effective information on S_s^1 to attackers because this table contains only ciphertexts encrypted by the sender's public key, and thus, only the sender can decrypt them. According to Algorithm 4, although all $T[x_i, i] = E(1)$ where $1 \leq i \leq n$, these $E(1)$'s are different from each other because ElGamal encryption is probabilistic. Because of the security of ElGamal encryption, it is also unfeasible for the adversaries or the receiver to distinguish $E(1)$ and $E(r)$, where r is a random number. Therefore, the secrecy of the sender's routing metric is preserved.

For each receiver, the 0-encoding set of the routing metric S_r^0 is embedded in its CR list, which is sent back to the sender in the routing response phase. During the calculation of CR, the multiplicative homomorphism of ElGamal encryption is applied, and each element in the 0-encoding set corresponds to an element in CR. However, the size of the 0-encoding set, i.e., $|S^0|$, is determined by the number of 0's in the binary form of the receiver's routing metric and can be smaller than n. Thus, to conceal the value of $|S^0|$, extra $n - |S^0|$ random encryptions are padded into CR, so that the size of CR is always n. Even with the CR eavesdropped, the adversary cannot obtain any effective information on the receiver's routing metric since it contains n ciphertexts. Because of the homomorphism operations and the padding, even the sender will not be able to obtain extra information on mtr_r except for the comparison result between mtr_s and mtr_r. Hence, the secrecy of the receiver's routing metric is preserved.

IV. Performance Evaluation

Here, we perform a comprehensive evaluation of the proposed framework. By considering the implementation details, we conduct the efficiency analysis on the overheads. Then, we apply the proposed framework on an existing opportunistic routing algorithm in simulations with realistic network settings, where we show the feasibility of our framework in the VANET environment.

A. Efficiency Analysis

1) Computation Overhead: Since all messages are signed before being sent out and verified after being received, the signing and verification processes introduce some computation overhead. According to the existing implementation results from [28], the most expensive operations are the scalar multiplication in $\mathbb{G}_1$, exponentiation in $\mathbb{G}_T$, and pairing evaluation. In comparison, the overhead of the hash functions and arithmetic operations in $\mathbb{Z}_q$ is very small. Because of the bilinear property of the mapping $\hat{e}$, we can transform some exponentiations in $\mathbb{G}_T$ into scalar multiplications in $\mathbb{G}_1$ for faster implementation. For example, to calculate $\hat{e}(S, X)^x$ in the Sign algorithm, we can first compute $x \cdot S$ and then get the value of $\hat{e}(S, X)^x$ by computing $\hat{e}(x \cdot S, X)$. This trick also applies to the Verify algorithm, i.e., $(\rho_b^{\dagger})^s = \hat{e}(s \cdot S, X)$, $(\rho_c^{\dagger}/\rho_a^{\dagger})^{-c} = \hat{e}(-c \cdot T, P_2) \cdot \hat{e}(c \cdot R, X)$. If we let $n \cdot \mathbb{G}_1$ denote n scalar multiplications in $\mathbb{G}_1$ and $m \cdot P$ denote m pairing operations, then by applying the given trick, we can obtain the following computation overhead for signing: Sign v1, $6 \cdot \mathbb{G}_1 + 1 \cdot P$; Sign v2, $8 \cdot \mathbb{G}_1 + 1 \cdot P$; Verify, $5 \cdot \mathbb{G}_1 + 5 \cdot P$. Here, we evaluate these operations with the implementation results from [28] obtained on a Pentinum IV 3.0-GHz machine. To achieve an 80-bit security level, approximately the same level as a standard 1024-bit RSA signature, a 512-bit prime number q, and a group $\mathbb{G}_1$, where each element is 160 bits long are chosen. The experimental results show that the average time required for a scalar multiplication in $\mathbb{G}_1$ and an $E(\mathbb{F}_p)$ Tate paring are 3.08 and 2.97 ms, respectively. Hence, the computation overheads for signing and verification are 21.45 ms (Sign v1), 27.61 ms (Sign v2), and 30.25 ms, respectively.

The secret routing metric comparison also introduces extra computation. In the Routing Request phase, i.e., Algorithm 3, the sender encrypts n 1 s and n random numbers to fill the table TB with size $2 \times n$. Because these encryptions can be precalculated, it will not introduce extra computation overhead in real time. Once receiving TB, the receiver calculates CR, which contains n ciphertexts. $n - |S_r^0|$ of them are the results of random-number encryptions, which can be precalculated, and $|S_r^0|$ of them are calculated by applying arithmetic multiplication on the elements in the table TB, whose computation overhead can be neglected. After the sender receives a CR, it decrypts the elements in the list. If one of the elements is decrypted to 1, the rest of the elements in CR are ignored. Only when all the elements are decrypted to values that are not 1 will the corresponding node be chosen as the relay. With n elements in the CR, a sender performs decryption, at most, n times. Because each ElGamal decryption takes approximately 0.54 ms,[1] for each receiver whose CR is received by the sender, the sender will spend, at most, $n \cdot 0.54 = 3.78$ ms to make a decision when we use 7 bits to represent a metric value, i.e., $n = 7$.

When the sender chooses a relay, they establish a session key. For each node, it performs two scalar multiplications and one pairing, and thus, the overhead introduced is $2 \cdot \mathbb{G} + 1 \cdot P$ with roughly 9.13 ms on the benchmark platform. Note that the two scalar multiplications have also been counted in the second version of the Sign algorithm.

[1]The decryption of ElGamal ciphertexts takes one exponentiation operation in $\mathbb{G}_T$ and one arithmetic multiplication. Because one exponentiation operation in $\mathbb{G}_T$ takes 0.54 ms [28] and arithmetic multiplications can be neglected, one decryption takes 0.54 ms.

2) Communication Overhead: As mentioned, to achieve an 80-bit security level, we choose a prime q that is 512 bits long, i.e., $|q| = 512$ bits, and groups with an element length of 160 bits, i.e., $|\mathbb{G}| = 160$ bits. Because the signature needs to be included in each broadcast message, the communication overhead of the authentication is determined by the signature size, which is approximately $5|\mathbb{G}| + 4|q| = 2.848$ Kb for version 1 and $6|\mathbb{G}| + 4|q| = 3.008$ Kb for version 2. If we consider that n_T and TTL do not require as many as 512 bits, the overhead can be even smaller.

For the routing metric comparison, the sender needs to broadcast its TB table along with its public key in the routing request phase, which has size $2n|\mathbb{G}| + |q|$. If we let $n = 7$, then the communication overhead in the routing request message is around 2.752 Kb. In the routing response phase, each receiver sends back the CR with size $n|\mathbb{G}| = 1.12$ Kb.

For the key agreement, the only communication overhead is the transmission of P_s from the sender to the receiver and P_r from a receiver to the sender, with the size of $|\mathbb{G}| = 0.16$ Kb each. Again, this has already been counted in the overhead of the second version of the signature.

B. Simulation Evaluation

The similar scheme is evaluated in the VANET scenario in [48]. The simulation results in [48] do not show an obvious impact of the security framework overhead. In fact, in a non-saturated network, the framework impact is hard to be detected since both the computation and the communication overheads are relatively small. To better understand the impact of the security framework and make such impacts obvious, we have to push the network toward its capacity limit. This can be achieved by applying the following approaches: 1) increasing the message generation rate to increase the total network traffic workload and 2) densifying the node contacts to accelerate the message transfers. We use an abstract scenario to model the opportunistic message-forwarding process, with higher message generation rates and denser node contacts. The simulation still reflects the necessity of DTN routing, i.e., opportunistic contact and routing. The change is only made to help us better understand the impact of the security framework under the extreme condition.

Our simulation is conducted using network simulator, i.e., OMNeT++, which provides finer granularity and better flexibility for simulation settings. In many DTN systems, message forwarding only happens when nodes encounter each other opportunistically. In the simulation, we use opportunistic links between nodes to model the opportunistic node contacts, so that at any time instance, a link between two neighbor nodes can be either active (i.e., nodes encounter each other) or inactive (i.e., no encounter happens).

In terms of routing, whenever a message carrier has an active link to a neighbor (i.e., encountering the neighbor node), it forwards the messages only when the neighbor is "closer" (routing metrics depending on different routing algorithms) to the destinations. In our simulation, we use the hop distance to the destination as the routing metric. In this sense, we are simulating the routing-metric-based opportunistic routing.

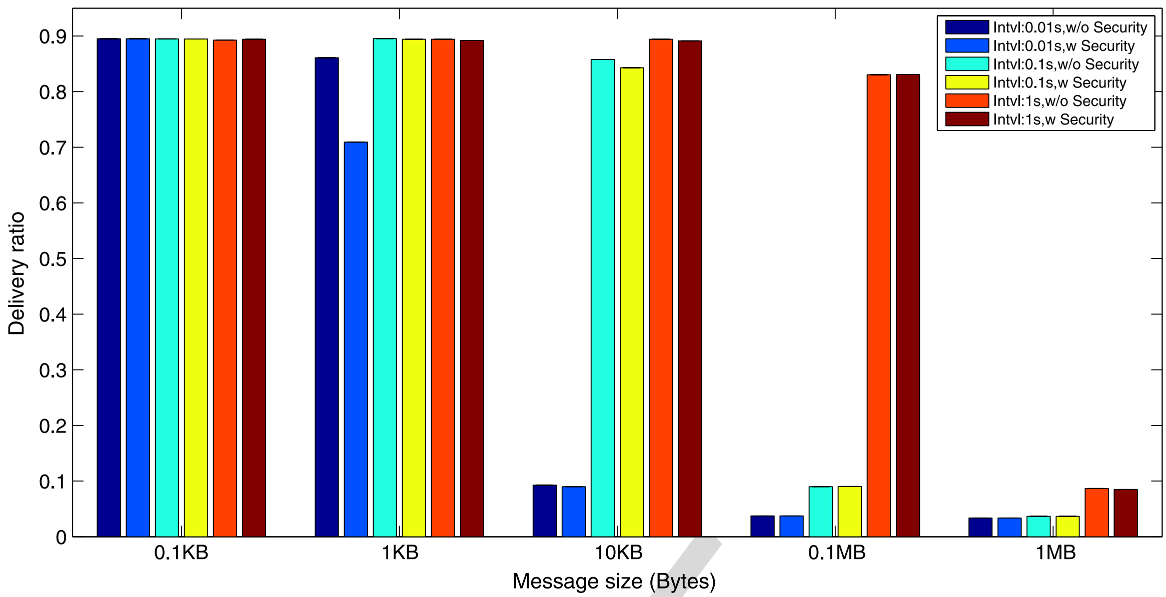


Fig. 2. Delivery ratio comparison.

The simulation is conducted on 30 nodes with a random static topology, and each node shares an opportunistic link with its nearby node. In our simulation, we set each node to have three opportunistic connections (i.e., three neighbors). Note that this number just implies the activeness of the node. Depending on different network scenarios, a more active (in terms of having contacts with other nodes) node can have more contact "neighbors," whereas a less active node has fewer contact "neighbors." We use three for our simulation. Each node generates messages following a Poisson process. In different simulation settings, the mean value of the message general interval varies among 0.01, 0.1, and 1 s, which leads to different message generation rates. To achieve denser node contacts, we assume that the intercontact time of any pair of neighboring nodes is ten times the complete message transmission time (including the message transmission and security overhead). Such intercontact time is much smaller than that in [48], indicating a very frequent node contact. We let the message size vary among 0.1 KB, 1 KB, 10 KB, 0.1 MB, and 1 MB. The message source and destination are randomly chosen among all nodes. We assume that each node carries a buffer with size 30 MB (smaller buffer size also makes it easier to reach to network capacity). The total simulation time for each parameter setting is 500 s.

We compare the performance results of different scenarios, i.e., with or without security framework, with different message sizes and different message generation intervals. We investigate the results with four performance metrics: delivery ratio, which is calculated as (N_D/N_G), where N_D is the total number of delivered messages, and N_G is the total number of generated messages; overhead ratio, which is calculated as $((N_R - N_D)/N_D)$, where N_R is the total number of message relays; average latency, which is the average delay for successful deliveries; and average hop count, which is the average hop count for the delivered messages.

In Fig. 2, it is shown that with the increase in the message size, the delivery ratio decreases. This is mainly due to the limited buffer size. With a larger message size, the storage competition of the buffer at each node is more severe, leading to a lower delivery ratio. However, when the message size is much smaller than the buffer size, e.g., 0.1 or 1 KB, the message size impact is not very apparent with given message generation intervals. We can also observe that, with a shorter message generation interval, the delivery ratio is lower. This is because the smaller the message generation interval, the more messages are generated, leading to a more severe buffer competition. However, if the message size is too small, e.g., 0.1 KB, compared with the buffer size, the impact of the traffic intensity is less apparent.

In terms of the security framework, its impact is more apparent when the message size is equal to 1 KB and the message generation interval is equal to 0.01 s. With the security framework, the size of the signature is, at most, 3.008 Kb (i.e., 0.376 KB). For message sizes of 10 KB, 0.1 MB, and 1 MB, the overhead is too small to make an apparent impact. The security framework is supposed to have a great impact on the messages with small sizes, i.e., 0.1 and 1 KB. However, because messages with size 0.1 KB are too small, even with the signature overhead, the size 0.476 KB is still too small to make obvious performance difference. The difference is shown with the messages with size 1 KB. The effect is also particularly obvious when the message generation interval is short (i.e., 0.01 s), indicating that the traffic intensity is high.

Fig. 3 shows some interesting results for the average latency. As we can see, for each message generation interval, the result (the bars with the same color) fluctuates. For most cases (except those whose message generation interval is 1 s), the delay first increases and then decreases with the increase in the message size. This is a mixed consequence of two factors: the message size and the buffer size. When the message is very small, all transmissions are smooth without much buffer competition, leading to a high delivery ratio and short delay. However, with the increase in the message size, the buffer competition gets fierce, leading to the decrease in the delivery ratio and the increase in the delay, mostly because of the message retransmissions. As the message size keeps increasing, the buffer resource

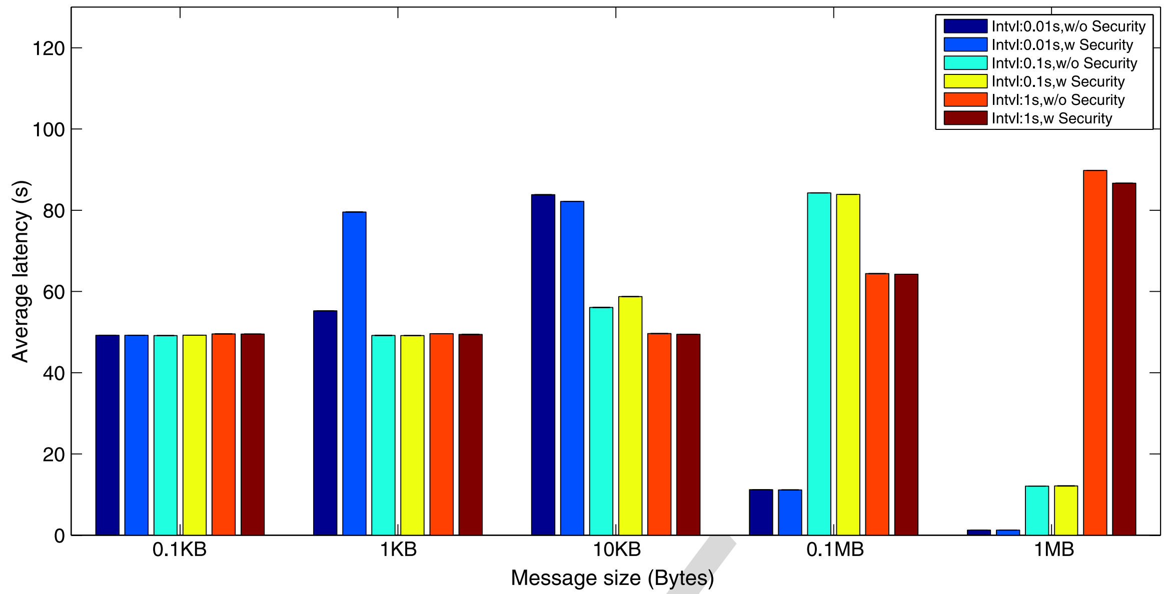


Fig. 3. Average latency comparison.

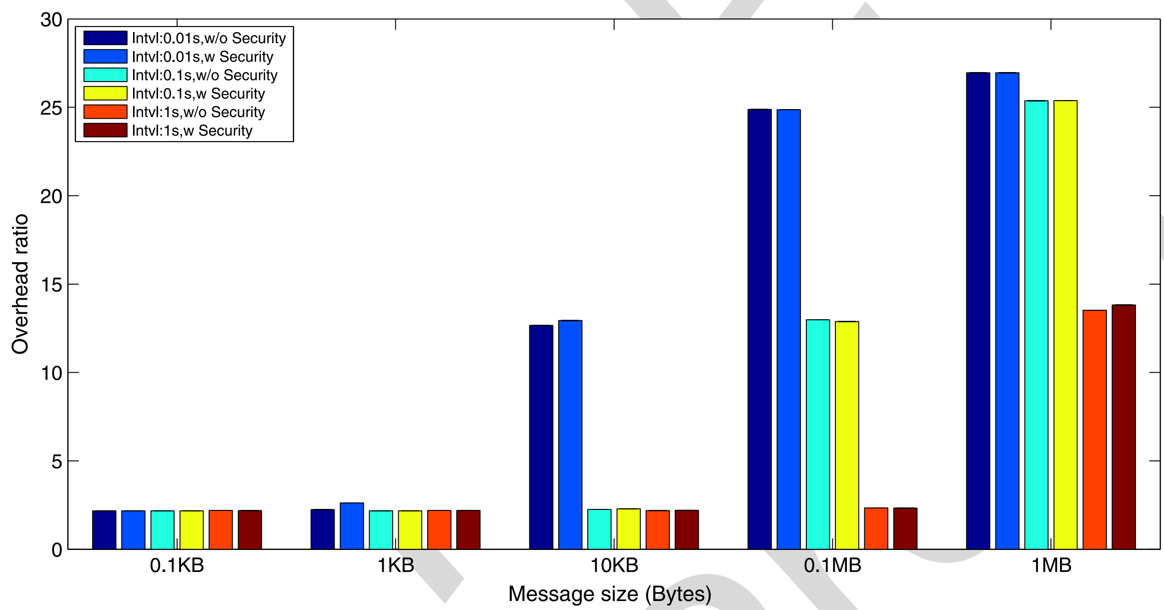


Fig. 4. Overhead ratio comparison.

becomes too limited to support the majority of the message transmissions, leading to a very low message delivery ratio. In such cases, the successfully transmitted messages are usually those whose sources are close to their destinations. This explains the short average latency. This can also be shown with the small average hop count for larger-message-size cases shown in Fig. 5. The cases with an interval of 1 s only show the increase phase. The impact of the security overhead is also more obvious when the message size is equal to 1 KB with an interval of 0.01 s.

Fig. 4 shows the performance results for the overhead ratio. With the same message generation interval, when the message size increases, the buffer competition becomes fierce, leading to a larger amount of message retransmission, i.e., increase in the overhead. The security framework increases the original message size, leading to a larger overhead, particularly for messages with original sizes of 1 and 10 KB. For the same message size, with a larger message generation interval, fewer messages are generated in the network, leading to a lower resource competition and less overhead.

Fig. 5 shows the performance results of the average hop count. As mentioned, for the same message generation interval, with the increase in the message size, the delivery ratio decreases. Moreover, the delivered messages are those whose sources and destinations are close. This explains the decrease in the average hop count. The security framework increases the original message size, leading to a smaller average hop count. For the same message size, with a larger message generation interval, fewer messages are generated in the network, leading to the lower resource competition and higher delivery ratio;

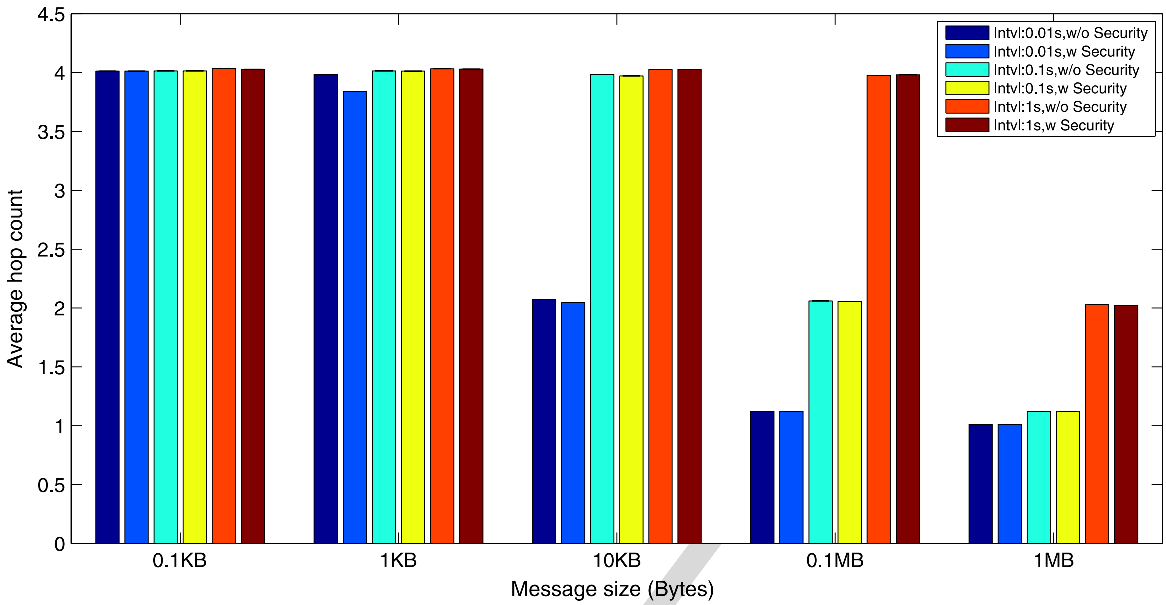


Fig. 5. Average hop count comparison.

furthermore, messages have a larger chance at being transmitted farther away from the sources with a larger average hop count.

As a conclusion, we see that the overhead introduced by the security framework is limited. In our simulations, the negative impact of the security framework becomes obvious when the network traffic is intense (i.e., with a message generation interval of 0.01 s per node), and the storage overhead (in terms of the extra size increment per message) is relatively large compared with the original size. For more general simulation settings, e.g., that in [48], the impact is almost unnoticeable. In terms of scalability, by comparing with the results obtained from 200 nodes in [48], we can see that the performance is mainly determined by the network traffic load and node contact density, instead of the total number of nodes. This is because intense network traffic and node contact intensity can happen in all network scenarios, regardless of the total number of nodes.

V. Further Discussions and Future Work

For further improvements, there are some issues worth exploring.

One possible concern is about the routing metric protection process. In our current solution, the sender initiates the routing metric comparison process by sending out its encoded routing metric in the routing request phase. Once receiving the routing request, each receiver makes a response based on the request, without knowing the comparison results. Based on the received responses, the sender can perform the comparisons, reveal the results, and choose the proper receivers as relays. However, one may suggest shifting the routing metric comparison workload to the receivers and letting the receivers make the routing decision so that only the proper receivers continue the conversation with the sender to reduce the computation and communication overhead. However, such an approach will not help much. If we let receivers perform the comparison, according to the process, the receivers become those to first send out the encoded routing metric. This will lead to extra interactions between the sender and receivers if we assume that the sender always initiates the routing request. Moreover, in the suggested case, although the routing metric comparison workload is distributed in the receivers from the sender, the routing metric response workload is accumulated to the sender from the receivers. In addition, more routing metric encoding workload will be introduced on the receivers' side.

Another feature, which is nice to have, is that the sender is able to perform the privacy-preserving comparison of the routing metrics of other receivers so that the best receiver or a few receivers can be chosen as the relay. However, such a feature can potentially invade the routing metric privacy. This is because if the sender can compare any two encoded routing metrics from different receivers, it can forge an encoded routing metric and perform the comparison with the real routing metric from a receiver. It can further repeat the comparisons with different forged routing metric values until it finds one value that is close enough to the real value of the other receiver's routing metric. However, it will be our interest to investigate proper security tools to enable the sender to perform secure comparisons with no privacy invasion risk.

Second, although the proposed scheme can defend most external attackers with the proposed authentication approaches, it does not integrate mechanisms resisting the attacks from internal attackers, e.g., black-hole attacks and Sybil attacks. We assume that the proposed scheme is operated on trustable and honest internal users. If a user is trusted by the CA, it is supposed to be honest and willing to help forward messages when possible. However, if this assumption does not hold, we should integrate other secure mechanisms to achieve corresponding protection. Although not the main focus of this paper, such mechanisms are well studied in the literature [31], [41]–[47], and they are relatively independent from our proposed scheme since they achieve different functions. However, we believe that efficient integration is feasible and will be of interest to us for future work.

VI. Conclusion

Opportunistic routing is widely employed in many mobile networks, e.g., DTNs, VANETs, and mobile sensor networks. Considering that the nodes' local and private information (i.e., routing metric) is extensively utilized in opportunistic routing, in this work, we have focused on its security and privacy concerns and proposed an advanced framework for opportunistic routing, providing various security and privacy preservation properties. A comprehensive evaluation was conducted to show the security and feasibility of the proposed framework.

References

[1] L. Zhang, B. Yu, and J. Pan, "GeoMob: A mobility-aware geocast scheme in metropolitans via taxicabs and buses," in *Proc. IEEE INFOCOM*, 2014, pp. 1779–1787.

[2] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 7, no. 3, pp. 19–20, Jul. 2003.

[3] F. Cadger, K. Curran, J. Santos, and S. Moffett, "A survey of geographical routing in wireless ad-hoc networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 621–653, 2nd Quart. 2013.

[4] Z. Zhi and Y. K. Choong, "Anonymizing geographic ad hoc routing for preserving location privacy," in *Proc. IEEE ICDCS Workshop*, 2005, pp. 646–651.

[5] M. Rahman, M. Mambo, A. Inomata, and E. Okamoto, "An anonymous on-demand position-based routing in mobile ad hoc networks," in *Proc. IEEE SAINT*, 2006, pp. 300–306.

[6] L. Zhou, D. Wu, B. Zheng, and M. Guizani, "Joint physical-application layer security for wireless multimedia delivery," *IEEE Commun. Mag.*, vol. 52, no. 3, pp. 66–72, Mar. 2014.

[7] Y. Qian and N. Moayeri, "Design of secure and application-oriented VANETs," in *Proc. IEEE VTC Spring*, 2008, pp. 2794–2799.

[8] Y. Qian, K. Lu, and D. Tipper, "A design for secure and survivable wireless sensor networks," *IEEE Wireless Commun.*, vol. 14, no. 5, pp. 30–37, Oct. 2007.

[9] A. Kate, G. Zaverucha, and U. Hengartner, "Anonymity and security in delay tolerant networks," in *Proc. IEEE SecureComm*, 2007, pp. 504–513.

[10] L. Guo, C. Zhang, H. Yue, and Y. Fang, "A privacy-preserving social-assisted mobile content dissemination scheme in DTNs," in *Proc. IEEE INFOCOM*, 2013, pp. 2301–2309.

[11] A. Yao, "Protocols for secure computations," in *Proc. FOCS*, 1982, pp. 160–164.

[12] M. Raya, P. Papadimitratos, and J.-P. Hubaux, "Securing vehicular communications," *IEEE Wireless Commun.*, vol. 13, no. 5, pp. 8–15, Oct. 2006.

[13] P. Papadimitratos *et al.*, "Secure vehicular communication systems: Design and architecture," *IEEE Wireless Commun.*, vol. 46, no. 11, pp. 100–109, Nov. 2008.

[14] R. Patra, S. Surana, and S. Nedevschi, "Hierarchical identity based cryptography for end-to-end security in DTNs," in *Proc. ICCP*, 2008, pp. 223–230.

[15] P. Papadimitratos, V. Gligor, and J. Hubaux, "Securing vehicular communications—Assumptions, requirements, and principles," in *Proc. ESCAR Workshops*, 2006, pp. 5–14.

[16] G. Calandriello, P. Papadimitratos, J. Hubaux, and A. Lioy, "On the performance of secure vehicular communication systems," *IEEE Trans. Dependable Secure Comput.*, vol. 8, no. 6, pp. 898–912, Nov./Dec. 2011.

[17] U. Shevade, H. H. Song, L. Qiu, and Y. Zhang, "Incentive-aware routing in DTNs," in *Proc. IEEE ICNP*, 2008, pp. 238–247.

[18] R. Lu, X. Lin, H. Zhu, X. Shen, and B. Preiss, "Pi: A practical incentive protocol for delay tolerant networks," *IEEE Trans. Wireless Commun.*, vol. 9, no. 4, pp. 1483–1493, Apr. 2010.

[19] L. Chen, S. Ng, and G. Wang, "Threshold anonymous announcement in VANETs," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 3, pp. 605–615, Mar. 2011.

[20] D. Chaum and E. Heyst, "Group signatures," in *Advances in Cryptology-EUROCRYPT*. Berlin, Germany: Springer-Verlag, 1991, pp. 257–265.

[21] E. Brickell, L. Chen, and J. Li, "Simplified security notions of direct anonymous attestation and a concrete scheme from pairings," *Int. J. Inf. Security*, vol. 8, no. 5, pp. 315–330, Oct. 2009.

[22] E. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in *Proc. ACM CCS*, 2004, pp. 132–145.

[23] H. Lin and W. Tzeng, "An efficient solution to the millionaires problem based on homomorphic encryption," in *Applied Cryptography and Network Security*. Berlin, Germany: Springer-Verlag, 2005, pp. 456–466.

[24] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," in *Proc. SCIS*, 2000, p. 26–28.

[25] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology-CRYPTO*. Berlin, Germany: Springer-Verlag, 2001, pp. 213–229.

[26] R. Dupont and A. Enge, "Provably secure non-interactive key distribution based on pairings," *Discr. Appl. Math.*, vol. 154, no. 2, pp. 270–276, Feb. 2006.

[27] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Advances in Cryptology-CRYPTO*. Berlin, Germany: Springer-Verlag, 2004, pp. 56–72.

[28] M. Scott and N. Costigan, "Implementing cryptographic pairings on smartcards," in *Cryptographic Hardware and Embedded Systems*. Berlin, Germany: Springer-Verlag, 2007, pp. 177–196, ser. Lecture Notes in Computer Science.

[29] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE simulator for DTN protocol evaluation," in *Proc. ICST SIMUtools*, 2009, p. 55.

[30] P. Papadimitratos, A. La Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *IEEE Commun. Mag.*, vol. 47, no. 11, pp. 84–95, Nov. 2009.

[31] R. Lu, X. Lin, and X. Shen, "Spring: A social-based privacy-preserving packet forwarding protocol for vehicular delay tolerant networks," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[32] L. Dóra and T. Holczer, "Hide-and-Lie: Enhancing application-level privacy in opportunistic networks," in *Proc. ACM Int. Workshop Mobile Opportunistic Netw.*, 2010, pp. 135–142.

[33] M. Radenkovic, I. Vaghi, S. Zakhary, and A. Benslimane, "AdaptAnon: Adaptive anonymity for service queries in mobile opportunistic networks," in *Proc. IEEE ICC*, 2013, pp. 1839–1844.

[34] Z. Le, G. Vakde, and M. Wright, "PEON: Privacy-enhanced opportunistic networks with applications in assistive environments," in *Proc. ACM PETRA*, 2009, pp. 76–84.

[35] C. Shi, X. Luo, P. Traynor, M. Ammar, and E. Zegura, "ARDEN: Anonymous networking in delay tolerant networks," *Ad Hoc Netw.*, vol. 10, no. 6, pp. 918–930, Aug. 2012.

[36] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *ACM Commun. ACM*, vol. 42, no. 2, pp. 39–41, Feb. 1999.

[37] S. Zakhary and M. Radenkovic, "Utilizing social links for location privacy in opportunistic delay-tolerant networks," in *Proc. IEEE ICC*, 2012, pp. 1059–1063.

[38] S. Zakhary, M. Radenkovic, and A. Benslimane, "The quest for location-privacy in opportunistic mobile social networks," in *Proc. IEEE IWCMC*, 2013, pp. 667–673.

[39] A. Benslimane, M. Radenkovic, and S. Zakhary, "Efficient location privacy-aware forwarding in opportunistic mobile networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 2, pp. 893–906, Feb. 2014.

[40] O. Hasan, J. Miao, S. Mokhtar, and L. Brunie, "A privacy preserving prediction-based routing protocol for mobile delay tolerant networks," in *Proc. IEEE AINA*, 2013, pp. 546–553.

[41] F. Li, J. Wu, and A. Srinivasan, "Thwarting blackhole attacks in disruption-tolerant networks using encounter tickets," in *Proc. IEEE INFOCOM*, 2009, pp. 2428–2436.

[42] M. Chuah, P. Yang, and J. Han, "A ferry-based intrusion detection scheme for sparsely connected ad hoc networks," in *Proc. IEEE MobiQuitous*, 2007, pp. 1–8.

[43] M. Uddin *et al.*, "Making DTNs robust against spoofing attacks with localized countermeasures," in *Proc. IEEE SECON*, 2011, pp. 332–340.

[44] A. Al-Hinai, H. Zhang, Y. Chen, and Y. Li, "TB-SnW: Trust-based Spray-and-Wait routing for delay-tolerant networks," *J. Supercomput.*, vol. 69, no. 2, pp. 593–609, Aug. 2014.

[45] S. Zakhary and M. Radenkovic, "Erasure coding with replication to defend against malicious attacks in DTN," in *Proc. IEEE WiMob*, 2011, pp. 357–364.

[46] S. Trifunovic, M. Kurant, K. Hummel, and F. Legendre, "Preventing spam in opportunistic networks," *Comput. Commun.*, vol. 41, pp. 31–42, Mar. 2014.

[47] S. Trifunovic, F. Legendre, and C. Anastasiades, "Social trust in opportunistic networks," in *Proc. IEEE INFOCOM Workshops*, 2010, pp. 1–6.

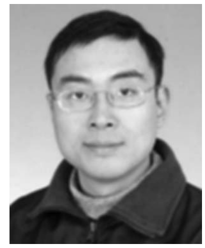
[48] L. Zhang, J. Song, and J. Pan, "Towards privacy-preserving and secure opportunistic routings in VANETs," in *Proc. IEEE SECON*, 2014, pp. 627–635.

Lei Zhang (M'14) received the Bachelor's degree in information security from China University of Geosciences, Wuhan, China, in 2010 and the Ph.D. degree from the Department of Computer Science, University of Victoria, Victoria, BC, Canada, in 2015.

His main research interest is in advanced wireless networks, including user mobility modeling, social characteristics, and security and privacy concerns.

AQ1 **Jun Song** (M'10) received the Bachelor's and Master's degrees from the China University of Geosciences, Wuhan, China, and the Ph.D. degree from Wuhan University, all in computer science.

He is currently an Associate Professor of computer science with the China University of Geosciences. His area of specialization is cryptography application and information security, and his current research interests include security analysis of cryptography application in wireless networks, applied network security, and cryptography security for big data.



Jianping Pan (SM'08) received the Bachelor's and Ph.D. degrees in computer science from Southeast University, Nanjing, China. AQ2

He was a Postdoctoral Researcher with the University of Waterloo, Waterloo, ON, Canada. He is currently a Professor of computer science with the University of Victoria, Victoria, BC, Canada. He has also been with Fujitsu Labs and NTT Labs. His area of specialization is computer networks and distributed systems, and his current research interests include protocols for advanced networking, performance analysis of networked systems, and applied network security.

Dr. Pan has been serving on the Technical Program Committees of major computer communications and networking conferences, including the IEEE International Conference on Computer Communications, the IEEE International Conference on Communications, the IEEE Global Telecommunications Conference (Globecom), the IEEE Wireless Communications and Networking Conference, and the IEEE Consumer Communications and Networking Conference. He is the Ad Hoc and Sensor Networking Symposium Cochair of the 2012 IEEE Globecom and an Associate Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

AUTHOR QUERIES

AUTHOR PLEASE ANSWER ALL QUERIES

AQ1 = Please provide the year when the degrees were received by author “J. Song.”
AQ2 = Please provide the year when the degrees were received by author “J. Pan.”

END OF ALL QUERIES

A Privacy-Preserving and Secure Framework for Opportunistic Routing in DTNs

Lei Zhang, *Member, IEEE*, Jun Song, *Member, IEEE*, and Jianping Pan, *Senior Member, IEEE*

***Abstract*—Opportunistic routing has been extensively studied and utilized in delay/disruption-tolerant networks. The extensive use of nodes' local information, e.g., the distance to the destination or the contact frequency with the destination, in such routing schemes can cause severe security and privacy problems. Existing solutions of anonymous routing can introduce undesired overhead and fail to provide the confidentiality of the routing metric. In this paper, we propose an advanced framework for opportunistic routing schemes, providing the following properties: confidentiality of the nodes' routing metric, anonymous authentication, and efficient key agreement for pairwise communication. A comprehensive evaluation, including security analysis, efficiency analysis, and simulation evaluation, is presented to show the security and feasibility of the proposed framework.**

***Index Terms*—Delay/disruption-tolerant networks (DTNs), opportunistic routing, privacy, security.**

I. Introduction

In delay/disruption-tolerant networks (DTNs), message propagation is usually conducted in a multihop fashion with the help of store–carry–forward routing techniques. In the literature, different routing approaches can be divided into two categories, namely, topology-based and opportunity-based. Different from topology-based routing schemes, opportunistic routing schemes make routing decisions based on nodes' local information, making them more applicable for networks of large scale and with high dynamics [1]. Opportunistic routing has been extensively studied in DTNs, e.g., [2]. In most opportunistic routing algorithms, messages are forwarded to the nodes with a higher chance of delivery to the destination. Nodes in opportunistic routing schemes need to broadcast, exchange, and compare their local or individual information, e.g., the distance or visit frequency to the destination. In this paper, we call such information the *routing metric*.

However, from privacy and security perspectives, opportunistic routing can raise critical issues. A serious threat is the traffic analysis, where the network traffic can be observed by a malicious node, and then, it uses the information gathered to launch attacks. Moreover, the routing metrics, e.g., geographic location or contact history, are highly privacy sensitive. Without proper protection, severe privacy problems may occur.

Although the routing metric information is very privacy sensitive, most of the current work on the security and privacy of DTNs has neglected to protect it effectively. Much of such work [3]–[5], [9] focuses on node identity anonymity, with the help of techniques such as pseudonyms [9], group signature, and identity-based encryption [25]. On the other hand, some recent studies [10], [31] take the privacy issue of the "metric" information into consideration in social-based DTNs. However, because of their social relationship-based nature, such studies do not provide node identity anonymity.

To address these concerns, in this paper, we propose an advanced secure and privacy-preserving framework particularly for opportunistic routing schemes, integrating the following three properties. 1) The first property is the confidentiality of the routing metric. Protected by cryptographic tools, the routing metric is known only to its owner. However, to perform message routing, the framework allows a node to compare its own routing metric with others' without knowing the exact values of the others' routing metrics. This is achieved by integrating a solution to "Yao's millionaire problem" [11], which belongs to the secure multiparty computation problem. The protection of the routing metric, thus enhancing the node privacy, is the key feature that distinguishes our design from others. 2) The second property is anonymous authentication. Authentication is the fundamental mechanism for various security properties, i.e., data integrity, authenticity, and nonrepudiation. For the strong requirement of identity privacy [32], [33] in DTNs, anonymity is another essential property that must be provided. In this paper, we adopt a group-signature-based scheme to achieve anonymous authentication. 3) The third property is efficient key agreement. In DTNs, particularly in some mobile scenarios, e.g., mobile ad hoc networks, it is desirable for each pair of nodes to share a unique session key to achieve pairwise confidentiality. Considering the total number of session keys and the lack of central control in such distributed systems, efficient key management is crucial. In this paper, we adopt an efficient pairing-based key agreement scheme and integrate it seamlessly into the message routing process without creating much overhead.

A comprehensive evaluation of the proposed framework is provided. We first analyze the security of our design and

Manuscript received January 28, 2015; revised June 15, 2015 and August 16, 2015; accepted September 2, 2015. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, by the Canada Foundation for Innovation, and by the British Columbia Knowledge Development Fund. A preliminary version of this paper appeared at the 11th IEEE International Conference on Sensing, Communication, and Networking (SECON'14), Singapore. The review of this paper was coordinated by Prof. Y. Qian.

L. Zhang and J. Pan are with the Department of Computer Science, University of Victoria, Victoria, BC V8P 5C2, Canada (e-mail: leiz@uvic.ca; pan@uvic.ca).

J. Song is with the Department of Computer Science, China University of Geosciences, Wuhan 430074, China (e-mail: songjun@cug.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2015.2480761

then evaluate the performance with both cryptographic implementation specifications and event-driven simulations. These evaluations show the security and feasibility of the framework. Moreover, our framework can be applied to many opportunistic routing scenarios, e.g., mobile ad hoc networks or vehicular ad hoc networks (VANETs).

The rest of this paper is outlined as follows. Related work, security and privacy background, and related cryptographic techniques are introduced in Section II. Section III gives the detailed description of the proposed framework, including the system setup, the different algorithms involved, and security analysis. The performance evaluation is presented in Section IV. Section VI concludes this paper.

II. Background and Related Work

A. Related Work

Ranging from the physical layer to the application layer [6], security and privacy are always hot topics in DTN systems, i.e., VANETs [7], [9], [13], [14] and wireless sensor networks [8]. In [13] and [15], Papadimitratos *et al.* give comprehensive introductions on the basic assumptions, requirements, system models, adversary models, design principles, and a spectrum of VANET (which is a typical DTN system) security mechanisms.

With the special focus on anonymous routing, Cadger *et al.* [3] proposed a solution to separate the routing metric from a node's true identity, so that the attackers cannot link the privacy-sensitive routing metric to a specific node. In [4], Zhi and Choong utilized an anonymous table that stores pseudonyms along with the routing metric (position data in that paper) for the routing process. In [5], [9], and [31], extra servers or DTN gateway nodes or "roadside units" are deployed to manage the anonymous nodes, leading to the extra management overhead and security risk. None of such solutions provide the confidentiality of the routing metric, making it possible for attackers to spoil user privacy. Le *et al.* [34] and Shi *et al.* [35] adapted onion routing [36] to opportunistic networks for anonymity purposes; however, they require that the encryption keys of nodes are known to the message source node, which implies a complicated key management.

In terms of security, Patra *et al.* [14] used hierarchical identity-based cryptography to achieve authentication and key management. With a similar technique but by introducing pseudonyms, Kate *et al.* [9] achieved identity anonymity. The authors in [38], [39], and [45] preserved the location privacy of the sender using trusted social contacts. Recent studies [10], [40] took the privacy of the "metric" information into consideration. However, these papers did not provide user identity anonymity. Moreover, this work focused on a specific field, i.e., social-based DTN, where the strong social relationship (e.g., community) among nodes (e.g., cellphones) was utilized and is not applicable to more general DTN schemes, since the social relationship among nodes is not always sufficiently strong and explicit in some DTNs, particularly for mobile DTNs. Another direction of the DTN security study focuses on the detection and prevention of the attacks from the internal malicious node, e.g., black hole [31], [41]–[45] and Sybil attacks [46], [47]. This direction is from a different perspective and, thus, less related to the main focus of our paper.

Zhang *et al.* in [48] have looked into both anonymous routing and security. However, the scope of that work was only limited to VANETs, and the simulation evaluation was also limited to VANETs. In this paper, we expand our scope from VANETs to a more general network scenario, i.e., DTNs, so that our proposed framework can be applied to a wider range of applications. The simulation has also been redesigned to reflect the properties of DTNs and demonstrate the effectiveness of the proposed framework.

B. Security and Privacy Goals

We now introduce the general security and privacy properties that our design can provide.

- **Authentication**: Valid users must be authorized by a Certificate Authority (CA), and they can verify each other.
- **Data integrity**: A user should be able to detect the message change or damage, which is caused by either intentional or unexpected factors during its transmission.
- **Data confidentiality**: The secret data are only visible to eligible users.
- **Nonrepudiation**: No user can deny their past behaviors, e.g., signing, relaying a message, etc. Every node should be responsible for its behaviors.

Different from other schemes, when taking the routing metric issue into consideration, our scheme can provide privacy preservation in the following two aspects.

- **Identity anonymity**: The true identity of a user should not be exposed during any networking activity, including authentication, safety beacon broadcasting, etc.
- **Users' routing metric confidentiality**: As mentioned, the routing metric information has been extensively utilized in opportunistic routing. The protection of such information is essential to preserve the users' privacy.

Other requirements related to security management are revocation and traceability. Since they are less related to the routing process, we do not have them discussed in this work. However, we believe that, with the anonymous authentication in our framework, those properties are also achievable [19].

C. Threats and Adversaries

On the other hand, we review the possible threats and adversaries in the routing process, on which we focus.

1) Threats: Threats in mobile network systems can be categorized into two types: active and passive. In active attacks, the adversaries take active actions to incur damages to the network. Typical active attacks include the following.

- **Message forging/cheating**: The attackers send fake messages for malicious purposes. They can cheat on their identities, using fake identities to broadcast messages, e.g., Sybil attack, or they use their real identities but send messages containing fake information, e.g., dishonest routing metric, for malicious purposes.

- **Message modification/dropping**: Attackers may modify or damage the messages they received and forward them to other nodes, causing disorder. Attackers may even deliberately drop the messages to conduct a black-hole attack.
- **Message replay attack**: The adversary replays the messages previously sent to disturb the network.

For passive attacks, adversaries are usually referred to as "curious but honest," which means that they intend to peek at others' secret or private information but do not conduct active actions to spoil the system. Typical passive attacks include the following.

- **Message eavesdropping**: Because of the openly shared medium of wireless communications, "curious" attackers can easily eavesdrop on the conversation of others, causing damage to user confidentiality and privacy.
- **Privacy digging**: With the eavesdropped information, the attackers dig up more private information of others. For example, once the attacker intercepts the routing metric (e.g., the visiting frequency to certain locations) of a node, he may learn the node's mobility patterns.

2) Adversaries: Adversaries can be divided into external and internal adversaries. External adversaries are those who are not authorized by the CA or whose certificates are revoked by the CA. With the authentication scheme proposed in this paper, our framework can resist both passive and active attacks of the external adversaries, because the nodes who fail the authentication verification will be simply ignored by the authorized adversaries. In contrast, the internal adversaries are those who are authorized but malicious. They can conduct attacks until they are discovered, and then, they will be revoked from the trusted group. In this paper, we consider the internal adversaries to be passive attackers, which are "curious but honest." The discovery and resistance of internal active adversaries can be very complicated, and different attacks usually need very different solutions. We do not cover them in this paper.

D. Cryptographic Tools

Cryptographic tools are important for security scheme designs. Here, we briefly introduce the cryptographic tools used in our framework. First, as mentioned in Section I, our anonymous authentication function is achieved by a group signature scheme. Second, the protection of the routing metric confidentiality is essentially "Yao's millionaire problem," where homomorphic encryption is used as a main support of the solution. Finally, the pairing-based Sakai–Ohgishi–Kasahara (SOK) key agreement serves as the basis of the session key distribution.

1) Group Signature: Group signature is an efficient solution to achieve anonymity authentication. In group signature [20], network nodes are organized in groups, and each group has a group manager to represent the members. The main feature of the group signature scheme is that it provides anonymous authentication to the group members. A verifier can determine whether a signer is authorized by a group without knowing or linking the true identity of the signer. Different from the other anonymity techniques, group signature reduces the workload of the public key and certificate distribution and verification operations. As an authentication scheme, group signature can satisfy other basic security requirements, such as message integrity and nonrepudiation.

In this paper, we choose one of the group signature schemes as our anonymous authentication scheme. It is a bilinear-map-based authentication scheme, which is also adopted in an enhanced version [21] of the Directed Anonymous Attestation (DAA) [22]. The original DAA was adopted by the Trusted Computing Group for anonymous authentication purposes. It is essentially a group signature scheme.

2) Yao's Millionaire Problem: In [11], Yao first introduced a problem that is analogous to a more general problem, where there are two numbers a and b, and the goal is to verify the inequality $a \ge b$ without revealing the actual values of a and b. To achieve routing metric confidentiality, it is expected that a node can compare its routing metrics with others' without knowing the values of the others' routing metrics. In this paper, we integrate the solution proposed in [23] into our security framework, as the main idea explained below.

Let all the routing metrics be expressed in a binary form with a fixed length n. For each binary-form routing metric, two sets of its substrings can be constructed, i.e., *0-encoding* and *1-encoding*. For a binary-form routing metric $r = r_n r_{n-1}, \ldots, r_1$, its 0-encoding set S_r^0 is defined as

$$S_r^0 = \{r_n r_{n-1}, \ldots, r_{i+1} 1 | r_i = 0, 1 \le r \le n\} \tag{1}$$

whereas its 1-encoding set S_r^1 is defined as

$$S_r^1 = \{r_n r_{n-1}, \ldots, r_{i+1} r_i | r_i = 1, 1 \le r \le n\}. \tag{2}$$

A very important conclusion is that for two routing metric values x and y, $x > y$ if there is one common element in both S_x^1 and S_y^0 [23]. This is easy to prove. If $x > y$, there must be a position i so that the substring $r_n^x r_{n-1}^x, \ldots, r_{i+1}^x$ is the same as $r_n^y r_{n-1}^y, \ldots, r_{i+1}^y$; however, $r_i^x = 1$, and $r_i^y = 0$. Thus, with the construction of 0-encoding and 1-encoding sets previously described, for S_x^1, it must contain an element $r_n^x r_{n-1}^x, \ldots, r_i^x$; for S_y^0, it must contain an element $r_n^y r_{n-1}^y, \ldots, r_{i+1}^y 1$, which is identical to $r_n^x r_{n-1}^x, \ldots, r_i^x$.

3) Homomorphic Encryption: In the implementation of the solution to Yao's millionaire problem, the homomorphism property of ElGamal encryption is utilized. Encryption schemes with the homomorphism property are referred to as homomorphic encryption. The homomorphism property allows a specific type of operation, e.g., $\otimes$, to be applied directly on two ciphertexts, e.g., $Enc(p_1)$ and $Enc(p_2)$, to obtain a result $R = Enc(p_1) \otimes Enc(p_2)$, which can be decrypted. The decryption of R is a result obtained from applying another operation, e.g., $\odot$, on the corresponding plaintexts, which means $D(R) = p_1 \odot p_2$. The operation $\odot$ can be either multiplication or addition, corresponding to multiplication homomorphic and addition homomorphic, respectively. The homomorphism property is a desirable feature since it can operate directly on the ciphertexts, without exposing the plaintexts to the parties performing the operations.

4) SOK Key Agreement: To achieve data confidentiality and for efficiency consideration, the secret messages are usually

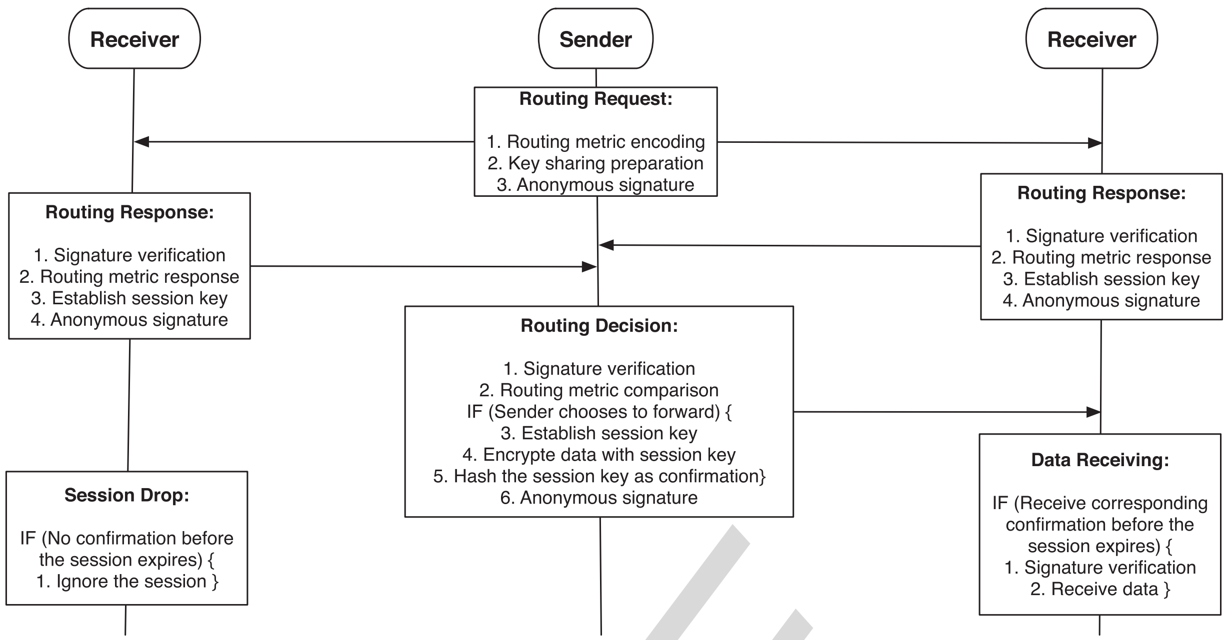


Fig. 1. Protocol flow.

encrypted by symmetric encryption schemes, such as AES. Considering the ad hoc environment, it is crucial to have an efficient and lightweight key agreement scheme to manage the huge number of session keys since each pair of users should share a distinct session key. We deploy a key agreement scheme similar to the SOK scheme [24], which has been also utilized in DTNs [9]. In the SOK key agreement, there are two groups $\mathbb{G}$ (written additively) and $\mathbb{G}_T$ (written multiplicatively) of order p (a large prime number) and an efficiently computable bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Assume that the CA possesses a master secret key $s \in \mathbb{Z}_q$ and that each user possesses an identity ID. The CA constructs each user i's secret key by calculating $d_i = sH(ID_i) \in \mathbb{G}$, where $H(\cdot)$ is a public hash function mapping an input to an element in $\mathbb{G}$. Under such a scheme, two users authorized by the same CA can noninteractively compute a shared session key with the identity of the other participant and their own private keys. For example, for users a and b, we have

$$\begin{aligned} Key_{ab} &= \hat{e}\left(H(ID_a), d_b\right) = \hat{e}\left(d_a, H(ID_b)\right) \\ &= \hat{e}(H(ID_a), H(ID_b))^s. \end{aligned} \tag{3}$$

Dupont and Enge [26] proved that this key agreement is secure in the random oracle model under the bilinear Diffie–Hellman (BDH) assumption in $\langle \mathbb{G}, \mathbb{G}_T, \hat{e} \rangle$.

III. Framework Design

Here, we provide the detailed framework design toward a privacy-preserving and secure opportunistic routing in DTNs. Without the topology information and route maintenance processes, routing decision in opportunistic routing is made by exchanging and comparing the routing metrics among individuals; hence, the nodes that have a larger chance at delivering the message, i.e., nodes with larger routing metric values, are chosen as the relays. Under such a scenario, any one-hop routing follows the protocol flow shown in Fig. 1. Four main algorithms are involved in the routing, namely, Routing Request (see Algorithm 3), Routing Response (see Algorithm 5), Routing Decision (see Algorithm 7), and Decision Confirm (see Algorithm 8).

Anonymous authentication is mainly provided in Sign and Verify algorithms, i.e., Algorithm 1 and Algorithm 2, respectively. For security concerns, every message sent should be signed first by the sender. The messages that failed to pass the verification will be automatically dropped by the receivers. To achieve the confidentiality of the routing metrics during the routing, Algorithm 4 and Algorithm 6 are embedded in the Routing Request and Routing Response algorithms, respectively. Some necessary processing of the routing metric information is also performed by these two algorithms.

A. *Protocol Setup*

The notations of our framework are listed in Table I. Our design is based on the finite-field cryptography, and the cryptographic setup is presented as follows. First, three cyclic groups are chosen: $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, of sufficiently large prime order q. Two random generators are selected such that $\mathbb{G}_1 = \langle P_1 \rangle$ and $\mathbb{G}_2 = \langle P_2 \rangle$ along with a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We write $\mathbb{G}_1$, $\mathbb{G}_2$ additively, and $\mathbb{G}_T$ multiplicatively. The pairing $\hat{e}$ is a map [19] with the following properties.

1) $\hat{e}$ is bilinear, which means $\hat{e}(aP_1, bP_2) = \hat{e}(P_1, P_2)^{ab}$ for any two integers a and $b \in \mathbb{Z}_q$.
2) $\hat{e}$ is nondegenerate, which means $\hat{e}(P_1, P_2) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is the identity element of $\mathbb{G}_T$.
3) $\hat{e}$ is computable, i.e., there is a polynomial-time algorithm for computing $\hat{e}(P, Q)$ for any $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.

Second, two hash functions are selected, i.e., $H_1 : \{0,1\}^* \to \mathbb{Z}_q$ and $H_2 : \{0,1\}^* \to \mathbb{G}_1$, mapping an arbitrary-length binary string to an integer and a $\mathbb{G}_1$ element, respectively.

TABLE I
NOTATIONS

Notation	Explanation
$\mathbb{G}_1, \mathbb{G}_2$	Two additive cyclic groups with order q
$\mathbb{G}_T$	A multiplicative cyclic group with order q
$\mathbb{Z}_q$	A integer cyclic group with order q
P_1, P_2	Generators for $\mathbb{G}_1$ and $\mathbb{G}_2$
$\hat{e}$	A bilinear map: $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$
n_t	A timestamp
s, r	Sender and receiver node, respectively
rq, rp	Routing request and routing response, respectively
mtr	Routing metric of a node
(pk, sk)	A key pair: (public key, secret key)
$\mathcal{SL}$	Sending list, containing the chosen relays
EH, DE	Encryption and decryption with ElGamal
Eec, Dec	Encryption and decryption of a symmetric cryptosystem, e.g., AES
n	The fixed length of the binary form of the routing metric
TB	A table of ciphertexts with 2 columns and n rows
CR	A list of ciphertexts with size n
S^0, S^1	0-encoding and 1-encoding sets of a binary string

Third, each node has a true and secret identity $f \in \mathbb{Z}_q$. The CA, who issues the certificates, has a secret key (x, y), where $x, y \leftarrow \mathbb{Z}_q$, and a public key (X, Y), where $X = x \cdot P_2 \in \mathbb{G}_2$, $Y = y \cdot P_2 \in \mathbb{G}_2$. The CA manages the true identities of all nodes and issues a certificate to each node. The certificate is a triplet (A, B, C), where $A \leftarrow r \cdot P_1$, $B \leftarrow y \cdot A$, and $C \leftarrow (x \cdot A + fxy \cdot A)$. The number r is randomly chosen from $\mathbb{Z}_q$; hence, for a specific node with identity f, its certificate is not deterministic. As we can see, the certificate is constructed with the secret key of the CA, i.e., (x, y), which is the main proof of the CA's attestation. It is also constructed with the true ID of the corresponding node, i.e., f, so that each certificate is specifically created for that specific node.

B. Signing and Verification

The signing and verification protocols are used to achieve anonymous authentication. The authentication is required for all messages, which means every message has to be signed before they are sent out. For every message received from other nodes, its signature needs to be verified by the receiver. In this paper, we deploy a scheme similar to DAA [19], [22], which is a group signature scheme.

Algorithm 1 performs the signing on the message and generates a signature σ. It contains a triplet (R, S, T), which can be seen as a shuffle of the true certificate, i.e., (A, B, C), so that every message is signed with an anonymous certificate. The calculation of (J, K, L, c, s) is used to provide the proof of the connection between the certificate and the node's true identity f. n_t is a timestamp providing time information, which is embedded into the message signature to resist the replay attack. n_c is a nonce that should be used in the same request, response, decision, and confirmation session. Different from the schemes in [19] and [22], there are two versions of the Sign algorithm. Version 1 is for the normal usage, and version 2 is only used when the sender wants to establish session keys with the possible relays, where a key agreement process will be executed with the help of P and Q. The details of the key agreement process will be discussed in the Routing Response algorithm, i.e., Algorithm 5.

Algorithm 1 Sign

procedure SIGN (Message msg)
  $a \leftarrow \mathbb{Z}_q$; $z \leftarrow \mathbb{Z}_q$
  $J \leftarrow H_2(msg)$; $K = f \cdot J$; $L \leftarrow z \cdot J$
  $R \leftarrow a \cdot A$; $S \leftarrow a \cdot B$; $T \leftarrow a \cdot C$; $\tau \leftarrow \hat{e}(S, X)^z$
  $c \leftarrow H_1(R\|S\|T\|\tau\|J\|K\|L\|n_t\|n_c\|msg)$
  $s \leftarrow z + c \cdot f \pmod{q}$
  If Version 1 **then**
    $\sigma \leftarrow (R, S, T, J, K, c, s, n_t, n_c, TTL)$
  else if Version 2 **then**
    $b \leftarrow \mathbb{Z}_q$
    $P \leftarrow b \cdot A$; $Q \leftarrow b \cdot B$
    $\sigma \leftarrow (R, S, T, J, K, c, s, P, n_t, n_c, TTL)$
  end if
return σ
end procedure

Verification of the signature is described in Algorithm 2. At the beginning, a few inspections are performed for a quick verification. First, data integrity of the message is provided by checking whether $J \neq H_2(msg)$, so that any corruption of the message can be detected. Second, by a quick comparison of $\hat{e}(R, Y)$ and $\hat{e}(S, P_2)$, it checks the internal relationship between R, S, and Y, i.e., $S = a \cdot B = ay \cdot A = y \cdot R$, so that $\hat{e}(R, Y) = \hat{e}(A, P_2)^{ay} \equiv \hat{e}(S, P_2)$.

Algorithm 2 Verify

1: **procedure** VERIFY(Message msg, Signature σ)
2:   **if** TTL has elapsed or $J \neq H_2(msg)$ or $\hat{e}(R, Y) \neq \hat{e}(S, P_2)$ **then**
3:     **return** Reject
4:   **end if**
5:   $\rho_a^\dagger \leftarrow \hat{e}(R, X)$; $\rho_b^\dagger \leftarrow \hat{e}(S, X)$; $\rho_c^\dagger \leftarrow \hat{e}(T, P_2)$
6:   $\tau^\dagger \leftarrow (\rho_b^\dagger)^s \cdot (\rho_c^\dagger / \rho_a^\dagger)^{-c}$
7:   $L^\dagger \leftarrow s \cdot J - c \cdot K$
8:   **if** $c \neq H_1(R\|S\|T\|\tau^\dagger\|J\|K\|L^\dagger\|n_t\|n_c\|msg)$ **then**
9:     **return** Reject
10:   **end if**
11: **return** Accept
12: **end procedure**

The following verification, i.e., lines 5–7, is a recovering process of τ and L. If signature σ is correctly generated by the signer and is successfully transmitted without any corruption, τ and L should be recovered by calculating $\tau^\dagger$ and $L^\dagger$. The correctness is shown as follows: First

$$L^\dagger = s \cdot J - c \cdot K = (s - cf) \cdot J \equiv L \tag{4}$$

second

$$\begin{aligned} \tau^\dagger &= \left(\rho_b^\dagger\right)^s \cdot \left(\frac{\rho_c^\dagger}{\rho_a^\dagger}\right)^{-c} = \hat{e}(S, X)^s \cdot \hat{e}(T, P_2)^{-c} \cdot \hat{e}(R, X)^c \\ &= \hat{e}(S, X)^s \cdot \hat{e}(P_1, P_2)^{-acxr(1+fy)+acxr} \\ &= \hat{e}(S, X)^{s+cf} = \hat{e}(S, X)^z \equiv \tau. \end{aligned} \tag{5}$$

If τ and L are successfully recovered and other fields, e.g., n_t, msg, etc., are successfully transmitted, the verifier should be able to recover c in line 8 to finish the verification.

C. Routing

The routing procedure is shown in Fig. 1. Before the data transmission, the sender first broadcasts a request message, i.e., rq in Algorithm 3, asking other nodes for their routing metrics. Once a neighbor node receives a request, it broadcasts a response, i.e., rp in Algorithm 5, which contains its routing metric. Based on the received responses, the sender makes the routing decision in Algorithm 3 and chooses those that have larger metrics as the relays. By checking the decision announcement of the sender, the receiver decides on its next action, as shown in Algorithm 8: receiving the data if it is chosen as the relay or ignoring the data otherwise. During the request and response processes, the sender and each chosen relay also finish the key agreement process to establish a unique pairwise key, so that they can secretly communicate for the following data transmissions.

Algorithm 3 Routing Request

1: **procedure** ROUTINGREQUEST
2: $\{pk_s, sk_s\} \leftarrow \mathbb{Z}_q$
3: $T \leftarrow Encoding(mtr_s, pk_s)$
4: msg.data $= T \| pk_s$
5: $\sigma \leftarrow \text{Sign}_{v2}(msg)$
6: Keep track of $\sigma.P$ and Q
7: Keep track of the request TTL TTL_{rq}
8: **return** $rq \leftarrow (msg, \sigma)$
9: **end procedure**

The routing procedure is straightforward; hence, we focus on the implementation of the two main security properties: routing metric confidentiality and key agreement.

1) Routing Metric Confidentiality: During the routing "request-response" phase, the sender inquires, obtains, and compares other nodes' routing metrics. Then, it chooses those that have a higher chance than itself to deliver the message to be the next relay. To keep the confidentiality of the routing metrics, we require that the sender has no access to the plaintext of the routing metrics; instead, it performs the comparison without revealing the actual value of others' metric information, which is known as Yao's millionaire problem. In this paper, we choose and integrate a solution from [23], which is based on the homomorphic encryption, into our framework. Recall the homomorphism property mentioned in Section II. To be specific, the multiplicative homomorphism of the ElGamal encryption system, which is denoted as $EH(\cdot)$, is utilized, i.e., line 9 in Algorithm 6, so that $EH(x_1) \otimes EH(x_2) = EH(x_1 \cdot x_2)$. The ciphertext of the ElGamal encryption is a pair of values, e.g., (a, b), and the operation $\otimes$ is defined as $EH(x_1) \otimes EH(x_2) = (a_1, b_1) \otimes (a_2, b_2) = (a_1 \cdot a_2, b_1 \cdot b_2)$.

The main idea of the solution to Yao's millionaire problem is described in Section II, i.e., by checking whether there is a common element in the sender's 1-encoding set S_s^1 and the receiver's 0-encoding set S_r^0, the sender can determine whether its routing metric mtr_s is larger than that of the receiver mtr_r. The implementation details are described as follows. During the routing request phase, the sender performs the Encoding algorithm, i.e., Algorithm 4, using its own routing metric mtr_s to construct a $2 \times n$ table TB, i.e., lines 3–7. Essentially, TB integrates the S_s^1 of the sender metric in an anonymous way, since each element in the table is in a ciphertext form. TB is then included in the routing request message rq and broadcast to the potential relay nodes.

Algorithm 4 Encoding

1: **procedure** ENCODING (mtr, pk)
2: Convert mtr to binary form $c_n c_{n-1}, \ldots, c_1 \in \{0, 1\}^n$
3: Initialize T as a $2 \times n$ table
4: **for** k from n to 1 **do**
5: $TB[c_k, k] = EH_{pk}(1)$
6: $TB[\bar{c_k}, k] = EH_{pk}(r)$ for a random r
7: **end for**
8 **return** TB
9: **end procedure**

Upon receiving the request, the receiver performs the Routing Response algorithm, i.e., Algorithm 5, to make a response to the request. The Coding Response algorithm, i.e., Algorithm 6, is called at this point. The algorithm first derives the 0-encoding set S_r^0 of the receiver's mtr_r, i.e., lines 3–7. Then, along with the table TB from the sender, it generates CR, i.e., lines 8–11, where each element c_t is the result of applying $\otimes$ on the ciphertexts in TB following some rules defined by the element of S_r^0, i.e., t and S_r^0. Hence, S_r^0 is integrated in CR. Because of the homomorphism of the ElGamal encryption EH, each c_t is essentially a ciphertext encrypted by EH. Up to line 11, the size of CR is determined by the number of elements in S_r^0, i.e., $|S_r^0|$. Extra $n - |S^0|$ random ciphertexts are padded into CR for security considerations, i.e., lines 12–14. Details will be provided in the security analysis in Section IV.

Algorithm 5 Routing Response

1: **procedure** ROUTINGRESPONSE(Request rq)
2: **if** Verify(rq) fails **then**
3: **return** Ignore
4: **end if**
5: $CR \leftarrow CodingResponse(mtr_r, rq.T, rq.pk_s)$
6: msg.data $= CR$
7: $n_c = rq.n_c$
8: $\sigma \leftarrow \text{Sign}_{v2}(msg)$
9: Keep track of $\sigma.P$ and Q
10: $Key \leftarrow \hat{e}(rq.\sigma.P, Q)$
11: Keep track of the response TTL TTL_{rp}
12: **return** $rp \leftarrow (msg, \sigma)$
13: **end procedure**

Algorithm 6 Coding Response

1: **procedure** CODINGRESPONSE(mtr, TB, pk)
2: Convert mtr into binary form $c_n c_{n-1}, \ldots, c_1 \in \{0,1\}^n$
3: **for** k from n to 1 **do**
4: **if** $c_k == 0$ **then**
5: Add binary string $c_n c_{n-1}, \ldots, c_{k-1}1$ into set S^0
6: **end if**
7: **end for**
8: **for** each $t = t_n t_{n-1}, \ldots, t_k$ in S^0 **do**
9: $c_t = TB[t_n, n] \otimes TB[t_{n-1}, n-1] \otimes, \ldots, \otimes TB[t_i, i]$
10: Add c_t to set CR
11: **end for**
12: **for** k from 1 to $n - |S^0|$ **do**
13: Add $EH_{pk}(r)$ to set CR for a random r
14: **end for**
15: **return** CR
16: **end procedure**

Then, the receiver sends CR back to the sender. In Algorithm 7, when a sender receives CR's, it decrypts the ciphertexts contained in each CR_i from receiver i. If there is a result equal to 1, it means there is a common element between S_s^1 and $S_{r_i}^0$, and the sender's metric is larger than that of receiver i. Thus, the sender will not choose i as its next relay. This conclusion is due to the ingenious constructions of TB and CR. However, if all ciphertexts in CR_i are not decrypted to 1, it means that the metric of receiver i is larger than that of the sender and that receiver i can be chosen as the next relay.

Algorithm 7 Routing Decision

1: **procedure** ROUTINGDECISION(Response $\{rp_1, rp_2, \ldots\}$)
2: **if** TTL$_{rq}$ has elapsed **then**
3: Ignore all responses
4: **end if**
5: **for** Any rp_i in $\{rp_1, rp_2, \ldots\}$ **do**
6: **if** Verify(rp_i) fail or $rp_i.n_c \neq rq.n_c$ **then**
7: **return** Ignore
8: **end if**
9: $CR \leftarrow rp_i.msg$.data
10: **for** Any t in set CR **do**
11: $k = DE_{sk_s}(t)$
12: **if** $k == 1$ **then**
13: Go to line 2 and try another rp
14: **end if**
15: **end for**
16: $Key_i \leftarrow \hat{e}(rp_i.\sigma.P, Q)$
17: Add (rp_i, Key_i) to $\mathcal{SL}$
18: **end for**
19: **for** Any (rp, Key) in $\mathcal{SL}$ **do**
20: msg.data $\leftarrow Enc_{Key}(Message)$
21: Send announcement $anc \leftarrow H(Key)$
22: $n_c = rp.n_c$
23: $\sigma \leftarrow \text{Sign}_{v1}(msg)$
24: Send data $\leftarrow (msg, \sigma)$
25: **end for**
26: **end procedure**

Algorithm 8 Decision Confirm

1: **procedure** DECISIONCONFIRM(Announcement $\{anc\}$)
2: **if** $H(Key) == anc$ and TTL$_{rp}$ has not elapsed **then**
3: Receive data
4: **if** Verify(data) fails **then**
5: **return** Reject
6: **else**
7: $Message \leftarrow Dec_{Key}$(data)
8: **return** Accept
9: **end if**
10: **else**
11: **return** Ignore
12: **end if**
13: **end procedure**

2) Key Agreement: During the routing request and response processes, the sender and each of the chosen relays establish a unique secret session key, with which the data can be encrypted so that pairwise confidentiality can be achieved. As mentioned, the second version of the **Sign** algorithm is used for the key agreement purpose. Assume that b_s and b_r are two random numbers generated by the sender and the receiver, respectively. In the second version of Algorithm 1, when sending a request, the sender calculates $P_s = b_s \cdot A_s$ and $Q_s = b_s \cdot B_s$ and broadcasts P_s. In Algorithm 5, when a receiver receives P_s, it first generates $P_r = b_r \cdot A_r$ and $Q_r = b_r \cdot B_r$ and then obtains a session key $Key_{rs} = \hat{e}(P_s, Q_r) = \hat{e}(b_s \cdot A_s, b_r \cdot B_r) = \hat{e}(A_s, B_r)^{b_s b_r} = \hat{e}(P_1, P_1)^{r_s b_s r_r y b_r}$. Note that this session key is only valid when the receiver is chosen by the sender as a relay. The receiver includes its P_r in its response rp to the sender. According to the responses received, the sender chooses the proper receiver as the relay and establishes the session key $Key_{rs} = \hat{e}(P_r, Q_s) = \hat{e}(b_r \cdot A_r, b_s \cdot B_s) = \hat{e}(A_r, B_s)^{b_r b_s} = \hat{e}(P_1, P_1)^{r_r b_r r_s y b_s}$.

D. Traceability

For privacy concerns, it is not desired to trace back the signer's true identity from its signature. This is also the reason for introducing anonymous authentication. However, we still reserve the tracing ability of the CA for management purposes. The tracing can be only conducted by the CA since it is trusted by anyone else. Algorithm 9 shows the tracing process. Since the CA knows the true identities of every user and system secret key x, it can verify the ownership of the certificate (i.e., R, T, and S), which is attached in the signature, by performing a matching with the internal relationship of (R, S, T), i.e., line 4. The correctness is shown as

$$\begin{aligned} \sigma.T &= a \cdot \sigma.C \\ &= a \cdot x \cdot \sigma.A + a \cdot fxy \cdot \sigma.A \\ &\equiv x \cdot \sigma.R + x \cdot f \cdot \sigma.S. \end{aligned} \tag{6}$$

Algorithm 9 Traceability

```
procedure TRACE (Signature σ)
    if Verify(σ) regardless of its TTL successfully then
        for Any f in {f_1, f_2, ...} do
            if σ.T == x · σ.R + x · f · σ.S then
                The signature is traced to identity f
                return Found
            end if
        end for
        return Not Found
    Else
        return Ignore
    end if
end procedure
```

E. Security Analysis

1) Security of the Signature: The security of the signature is guaranteed by the hardness of the **LRSW Assumption** [27]: Suppose that a *Setup*(1^k) algorithm generates a multiplicative group $\mathbb{G}$ with a generator g and an order q, where k is a parameter related the security level. There exist $X, Y \in \mathbb{G}$, $X = g^x$, and $Y = g^y$. Let $O_{X,Y}(\cdot)$ be an oracle that, with an input value of $m \in \mathbb{Z}_q$, outputs a triplet (a, a^y, a^{x+mxy}) for a randomly chosen $a \in \mathbb{G}$. Then, for all probabilistic polynomial-time adversaries $\mathcal{A}$, $v(k)$ is a negligible function defined as follows:

$$\begin{aligned}\Pr\big[&(q,\mathbb{G},g) \leftarrow Setup(1^k); x \leftarrow \mathbb{Z}_q; y \leftarrow \mathbb{Z}_q\\ &X = g^x; Y = g^y; (m,a,b,c) \leftarrow \mathcal{A}^{O_{X,Y}}(q,\mathbb{G},g)\\ &a \in \mathbb{G} \wedge b = a^y \wedge c = a^{x+mxy}\big] = v(k). \end{aligned} \tag{7}$$

This means that given the group setup $(q, \mathbb{G}, g)$ and the system public key (X, Y), it is impossible for a polynomial-time adversary to construct a triplet (a, a^y, a^{x+mxy}) without knowing the secret key (x, y), where a and m are random numbers. This assumption guarantees the effectiveness of our authentication scheme. Only the CA who possesses the secret key x and y can construct valid certificates to users. Without knowing the secret key, a malicious node can hardly forge a valid certificate $(A, B = y \cdot A, C = x \cdot A + fxy \cdot A)$, where the group in our scheme, i.e., $\mathbb{G}_1$, is additively written but isomorphic to the multiplicative form in the given assumption.

2) Security of the Key Agreement Process: The security of the key agreement is guaranteed by the **BDH assumption** [25]. Suppose that $\mathbb{G}_1$ is an additive group with generator g, $\mathbb{G}_2$ is a multiplicative group, and $\hat{e}$ is a bilinear map of $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, as described in Section II. Let $P \in \mathbb{G}_1$, $a, b, c \leftarrow \mathbb{Z}_q$, then $a \cdot P, b \cdot P, c \cdot P \in \mathbb{G}_1$. Let $O_{a\cdot P, b\cdot P, c\cdot P}(\cdot)$ be an oracle that outputs $r = \hat{e}(P,P)^{abc} \in \mathbb{G}_2$. Then, for all probabilistic polynomial-time adversaries $\mathcal{A}$, $v(k)$ is a negligible function defined as follows:

$$\begin{aligned}\Pr\big[&(q,\mathbb{G}_1,g,\mathbb{G}_2,\hat{e}) \leftarrow Setup(1^k); a \leftarrow \mathbb{Z}_q; b \leftarrow \mathbb{Z}_q\\ &c \leftarrow \mathbb{Z}_q; P \leftarrow \mathbb{G}_1; r \leftarrow \mathcal{A}^{O_{a\cdot P, b\cdot P, c\cdot P}}(q,\mathbb{G}_1,g,\mathbb{G}_2,\hat{e})\\ &r = \hat{e}(P,P)^{abc}\big] = v(k). \end{aligned} \tag{8}$$

In the key agreement process mentioned in Section III, the session key established by any two nodes a and b can be expressed as $\hat{e}(A_s, B_r)^{b_s b_r} = \hat{e}(A_r, B_s)^{b_s b_r} = \hat{e}(P_1, P_1)^{b_a r_a b_b r_b y}$, where random numbers r_a and r_b are introduced in the certificate construction process and random numbers b_a and b_b are introduced in the second version of the Sign algorithm, i.e., Algorithm 1. During the wireless transmission, an adversary can easily eavesdrop on $P_a = b_a r_a \cdot P_1$ and $P_b = b_b r_b \cdot P_1$. Assume that it also knows the system public key $Y = y \cdot P_1$. Because of the hardness of the BDH assumption, the probability of the adversary being able to recover the session key $\hat{e}(P_1, P_1)^{b_a r_a b_b r_b y}$, with P_a, P_b, and Y known, is negligible.

3) Security of the Routing Metric Comparison: The correctness has been explained in Session III. As the way how S^0 and S^1 are constructed, the confidentiality of the routing metric lies in the confidentiality of these two sets. The 1-encoding set S_s^1 of the sender's routing metric is embedded into the table TB, which is broadcast during the routing request phase. From the adversary point of view, the table TB reveals no effective information on S_s^1 to attackers because this table contains only ciphertexts encrypted by the sender's public key, and thus, only the sender can decrypt them. According to Algorithm 4, although all $T[x_i, i] = E(1)$ where $1 \leq i \leq n$, these $E(1)$'s are different from each other because ElGamal encryption is probabilistic. Because of the security of ElGamal encryption, it is also unfeasible for the adversaries or the receiver to distinguish $E(1)$ and $E(r)$, where r is a random number. Therefore, the secrecy of the sender's routing metric is preserved.

For each receiver, the 0-encoding set of the routing metric S_r^0 is embedded in its CR list, which is sent back to the sender in the routing response phase. During the calculation of CR, the multiplicative homomorphism of ElGamal encryption is applied, and each element in the 0-encoding set corresponds to an element in CR. However, the size of the 0-encoding set, i.e., $|S^0|$, is determined by the number of 0's in the binary form of the receiver's routing metric and can be smaller than n. Thus, to conceal the value of $|S^0|$, extra $n - |S^0|$ random encryptions are padded into CR, so that the size of CR is always n. Even with the CR eavesdropped, the adversary cannot obtain any effective information on the receiver's routing metric since it contains n ciphertexts. Because of the homomorphism operations and the padding, even the sender will not be able to obtain extra information on mtr_r except for the comparison result between mtr_s and mtr_r. Hence, the secrecy of the receiver's routing metric is preserved.

IV. Performance Evaluation

Here, we perform a comprehensive evaluation of the proposed framework. By considering the implementation details, we conduct the efficiency analysis on the overheads. Then, we apply the proposed framework on an existing opportunistic routing algorithm in simulations with realistic network settings, where we show the feasibility of our framework in the VANET environment.

A. Efficiency Analysis

1) Computation Overhead: Since all messages are signed before being sent out and verified after being received, the signing and verification processes introduce some computation overhead. According to the existing implementation results from [28], the most expensive operations are the scalar multiplication in $\mathbb{G}_1$, exponentiation in $\mathbb{G}_T$, and pairing evaluation. In comparison, the overhead of the hash functions and arithmetic operations in $\mathbb{Z}_q$ is very small. Because of the bilinear property of the mapping $\hat{e}$, we can transform some exponentiations in $\mathbb{G}_T$ into scalar multiplications in $\mathbb{G}_1$ for faster implementation. For example, to calculate $\hat{e}(S,X)^x$ in the Sign algorithm, we can first compute $x \cdot S$ and then get the value of $\hat{e}(S,X)^x$ by computing $\hat{e}(x \cdot S, X)$. This trick also applies to the Verify algorithm, i.e., $(\rho_b^\dagger)^s = \hat{e}(s \cdot S, X)$, $(\rho_c^\dagger/\rho_a^\dagger)^{-c} = \hat{e}(-c \cdot T, P_2) \cdot \hat{e}(c \cdot R, X)$. If we let $n \cdot \mathbb{G}_1$ denote n scalar multiplications in $\mathbb{G}_1$ and $m \cdot P$ denote m pairing operations, then by applying the given trick, we can obtain the following computation overhead for signing: Sign v1, $6 \cdot \mathbb{G}_1 + 1 \cdot P$; Sign v2, $8 \cdot \mathbb{G}_1 + 1 \cdot P$; Verify, $5 \cdot \mathbb{G}_1 + 5 \cdot P$. Here, we evaluate these operations with the implementation results from [28] obtained on a Pentinum IV 3.0-GHz machine. To achieve an 80-bit security level, approximately the same level as a standard 1024-bit RSA signature, a 512-bit prime number q, and a group $\mathbb{G}_1$, where each element is 160 bits long are chosen. The experimental results show that the average time required for a scalar multiplication in $\mathbb{G}_1$ and an $E(\mathbb{F}_p)$ Tate paring are 3.08 and 2.97 ms, respectively. Hence, the computation overheads for signing and verification are 21.45 ms (Sign v1), 27.61 ms (Sign v2), and 30.25 ms, respectively.

The secret routing metric comparison also introduces extra computation. In the Routing Request phase, i.e., Algorithm 3, the sender encrypts n 1 s and n random numbers to fill the table TB with size $2 \times n$. Because these encryptions can be precalculated, it will not introduce extra computation overhead in real time. Once receiving TB, the receiver calculates CR, which contains n ciphertexts. $n - |S_r^0|$ of them are the results of random-number encryptions, which can be precalculated, and $|S_r^0|$ of them are calculated by applying arithmetic multiplication on the elements in the table TB, whose computation overhead can be neglected. After the sender receives a CR, it decrypts the elements in the list. If one of the elements is decrypted to 1, the rest of the elements in CR are ignored. Only when all the elements are decrypted to values that are not 1 will the corresponding node be chosen as the relay. With n elements in the CR, a sender performs decryption, at most, n times. Because each ElGamal decryption takes approximately 0.54 ms,[1] for each receiver whose CR is received by the sender, the sender will spend, at most, $n \cdot 0.54 = 3.78$ ms to make a decision when we use 7 bits to represent a metric value, i.e., $n = 7$.

When the sender chooses a relay, they establish a session key. For each node, it performs two scalar multiplications and one pairing, and thus, the overhead introduced is $2 \cdot \mathbb{G} + 1 \cdot P$ with roughly 9.13 ms on the benchmark platform. Note that the two scalar multiplications have also been counted in the second version of the Sign algorithm.

[1]The decryption of ElGamal ciphertexts takes one exponentiation operation in $\mathbb{G}_T$ and one arithmetic multiplication. Because one exponentiation operation in $\mathbb{G}_T$ takes 0.54 ms [28] and arithmetic multiplications can be neglected, one decryption takes 0.54 ms.

2) Communication Overhead: As mentioned, to achieve an 80-bit security level, we choose a prime q that is 512 bits long, i.e., $|q| = 512$ bits, and groups with an element length of 160 bits, i.e., $|\mathbb{G}| = 160$ bits. Because the signature needs to be included in each broadcast message, the communication overhead of the authentication is determined by the signature size, which is approximately $5|\mathbb{G}| + 4|q| = 2.848$ Kb for version 1 and $6|\mathbb{G}| + 4|q| = 3.008$ Kb for version 2. If we consider that n_T and TTL do not require as many as 512 bits, the overhead can be even smaller.

For the routing metric comparison, the sender needs to broadcast its TB table along with its public key in the routing request phase, which has size $2n|\mathbb{G}| + |q|$. If we let $n = 7$, then the communication overhead in the routing request message is around 2.752 Kb. In the routing response phase, each receiver sends back the CR with size $n|\mathbb{G}| = 1.12$ Kb.

For the key agreement, the only communication overhead is the transmission of P_s from the sender to the receiver and P_r from a receiver to the sender, with the size of $|\mathbb{G}| = 0.16$ Kb each. Again, this has already been counted in the overhead of the second version of the signature.

B. Simulation Evaluation

The similar scheme is evaluated in the VANET scenario in [48]. The simulation results in [48] do not show an obvious impact of the security framework overhead. In fact, in a non-saturated network, the framework impact is hard to be detected since both the computation and the communication overheads are relatively small. To better understand the impact of the security framework and make such impacts obvious, we have to push the network toward its capacity limit. This can be achieved by applying the following approaches: 1) increasing the message generation rate to increase the total network traffic workload and 2) densifying the node contacts to accelerate the message transfers. We use an abstract scenario to model the opportunistic message-forwarding process, with higher message generation rates and denser node contacts. The simulation still reflects the necessity of DTN routing, i.e., opportunistic contact and routing. The change is only made to help us better understand the impact of the security framework under the extreme condition.

Our simulation is conducted using network simulator, i.e., OMNeT++, which provides finer granularity and better flexibility for simulation settings. In many DTN systems, message forwarding only happens when nodes encounter each other opportunistically. In the simulation, we use opportunistic links between nodes to model the opportunistic node contacts, so that at any time instance, a link between two neighbor nodes can be either active (i.e., nodes encounter each other) or inactive (i.e., no encounter happens).

In terms of routing, whenever a message carrier has an active link to a neighbor (i.e., encountering the neighbor node), it forwards the messages only when the neighbor is "closer" (routing metrics depending on different routing algorithms) to the destinations. In our simulation, we use the hop distance to the destination as the routing metric. In this sense, we are simulating the routing-metric-based opportunistic routing.

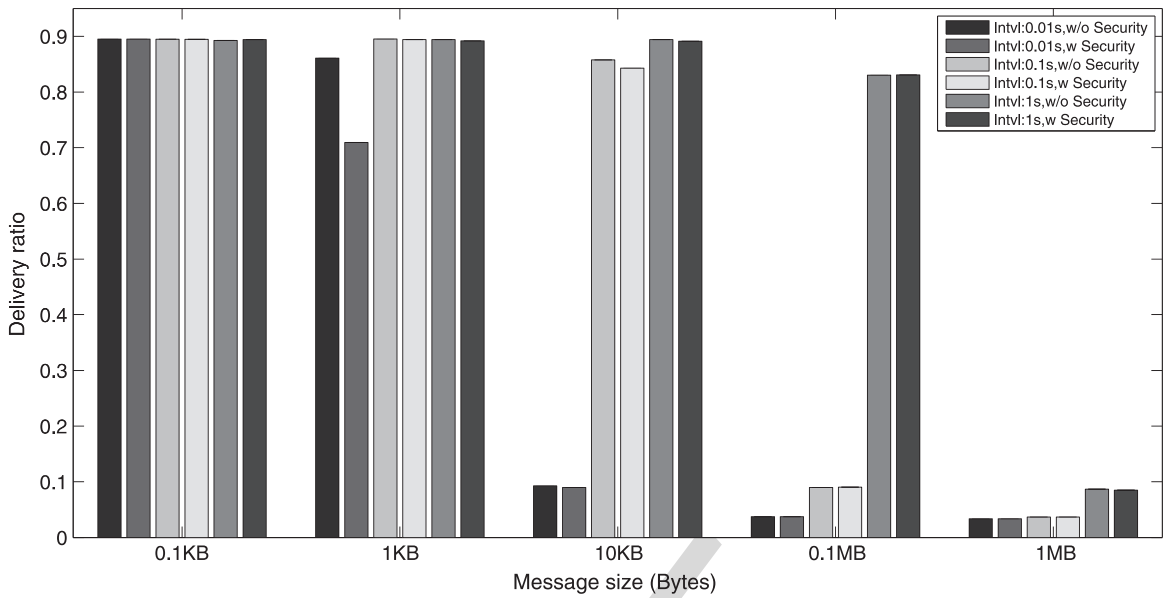


Fig. 2. Delivery ratio comparison.

The simulation is conducted on 30 nodes with a random static topology, and each node shares an opportunistic link with its nearby node. In our simulation, we set each node to have three opportunistic connections (i.e., three neighbors). Note that this number just implies the activeness of the node. Depending on different network scenarios, a more active (in terms of having contacts with other nodes) node can have more contact "neighbors," whereas a less active node has fewer contact "neighbors." We use three for our simulation. Each node generates messages following a Poisson process. In different simulation settings, the mean value of the message general interval varies among 0.01, 0.1, and 1 s, which leads to different message generation rates. To achieve denser node contacts, we assume that the intercontact time of any pair of neighboring nodes is ten times the complete message transmission time (including the message transmission and security overhead). Such intercontact time is much smaller than that in [48], indicating a very frequent node contact. We let the message size vary among 0.1 KB, 1 KB, 10 KB, 0.1 MB, and 1 MB. The message source and destination are randomly chosen among all nodes. We assume that each node carries a buffer with size 30 MB (smaller buffer size also makes it easier to reach to network capacity). The total simulation time for each parameter setting is 500 s.

We compare the performance results of different scenarios, i.e., with or without security framework, with different message sizes and different message generation intervals. We investigate the results with four performance metrics: delivery ratio, which is calculated as (N_D/N_G), where N_D is the total number of delivered messages, and N_G is the total number of generated messages; overhead ratio, which is calculated as $((N_R - N_D)/N_D)$, where N_R is the total number of message relays; average latency, which is the average delay for successful deliveries; and average hop count, which is the average hop count for the delivered messages.

In Fig. 2, it is shown that with the increase in the message size, the delivery ratio decreases. This is mainly due to the limited buffer size. With a larger message size, the storage competition of the buffer at each node is more severe, leading to a lower delivery ratio. However, when the message size is much smaller than the buffer size, e.g., 0.1 or 1 KB, the message size impact is not very apparent with given message generation intervals. We can also observe that, with a shorter message generation interval, the delivery ratio is lower. This is because the smaller the message generation interval, the more messages are generated, leading to a more severe buffer competition. However, if the message size is too small, e.g., 0.1 KB, compared with the buffer size, the impact of the traffic intensity is less apparent.

In terms of the security framework, its impact is more apparent when the message size is equal to 1 KB and the message generation interval is equal to 0.01 s. With the security framework, the size of the signature is, at most, 3.008 Kb (i.e., 0.376 KB). For message sizes of 10 KB, 0.1 MB, and 1 MB, the overhead is too small to make an apparent impact. The security framework is supposed to have a great impact on the messages with small sizes, i.e., 0.1 and 1 KB. However, because messages with size 0.1 KB are too small, even with the signature overhead, the size 0.476 KB is still too small to make obvious performance difference. The difference is shown with the messages with size 1 KB. The effect is also particularly obvious when the message generation interval is short (i.e., 0.01 s), indicating that the traffic intensity is high.

Fig. 3 shows some interesting results for the average latency. As we can see, for each message generation interval, the result (the bars with the same color) fluctuates. For most cases (except those whose message generation interval is 1 s), the delay first increases and then decreases with the increase in the message size. This is a mixed consequence of two factors: the message size and the buffer size. When the message is very small, all transmissions are smooth without much buffer competition, leading to a high delivery ratio and short delay. However, with the increase in the message size, the buffer competition gets fierce, leading to the decrease in the delivery ratio and the increase in the delay, mostly because of the message retransmissions. As the message size keeps increasing, the buffer resource

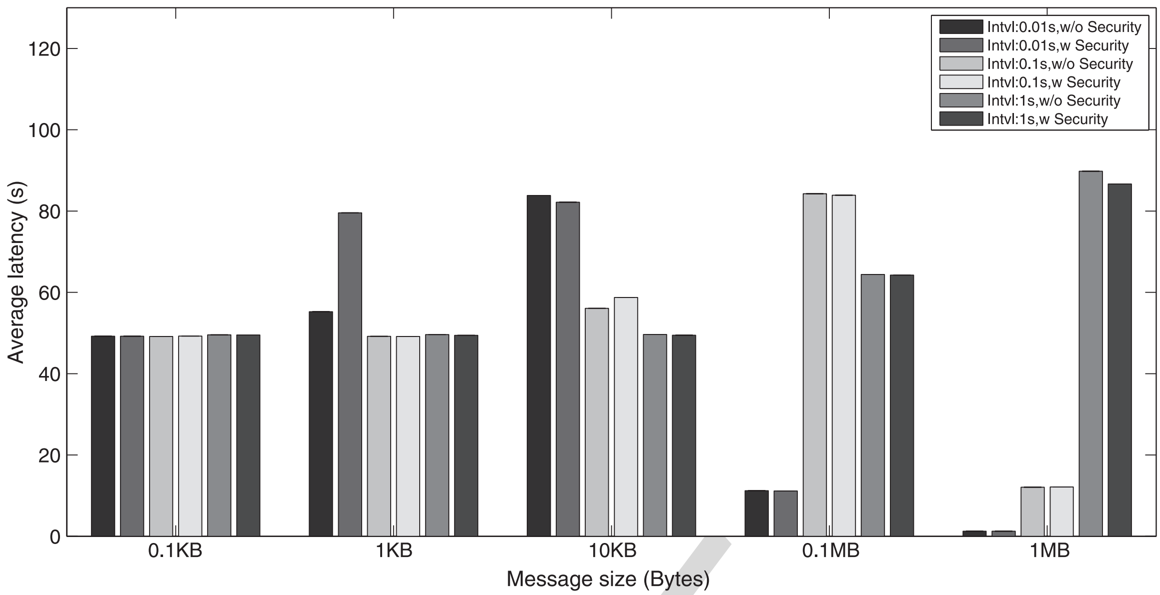


Fig. 3. Average latency comparison.

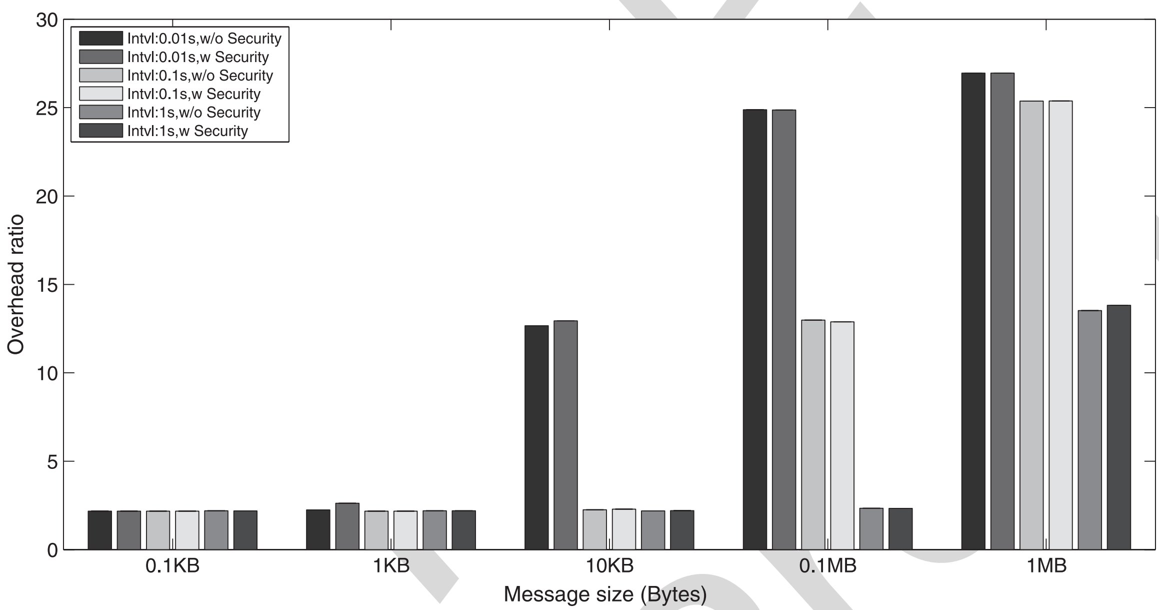


Fig. 4. Overhead ratio comparison.

becomes too limited to support the majority of the message transmissions, leading to a very low message delivery ratio. In such cases, the successfully transmitted messages are usually those whose sources are close to their destinations. This explains the short average latency. This can also be shown with the small average hop count for larger-message-size cases shown in Fig. 5. The cases with an interval of 1 s only show the increase phase. The impact of the security overhead is also more obvious when the message size is equal to 1 KB with an interval of 0.01 s.

Fig. 4 shows the performance results for the overhead ratio. With the same message generation interval, when the message size increases, the buffer competition becomes fierce, leading to a larger amount of message retransmission, i.e., increase in the overhead. The security framework increases the original message size, leading to a larger overhead, particularly for messages with original sizes of 1 and 10 KB. For the same message size, with a larger message generation interval, fewer messages are generated in the network, leading to a lower resource competition and less overhead.

Fig. 5 shows the performance results of the average hop count. As mentioned, for the same message generation interval, with the increase in the message size, the delivery ratio decreases. Moreover, the delivered messages are those whose sources and destinations are close. This explains the decrease in the average hop count. The security framework increases the original message size, leading to a smaller average hop count. For the same message size, with a larger message generation interval, fewer messages are generated in the network, leading to the lower resource competition and higher delivery ratio;

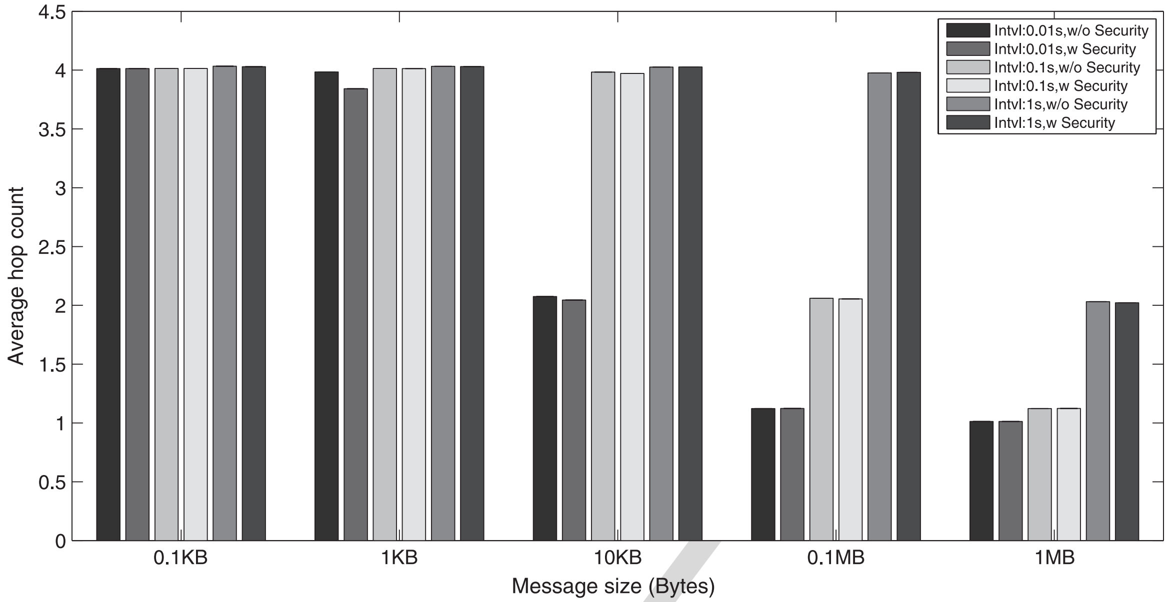


Fig. 5. Average hop count comparison.

furthermore, messages have a larger chance at being transmitted farther away from the sources with a larger average hop count.

As a conclusion, we see that the overhead introduced by the security framework is limited. In our simulations, the negative impact of the security framework becomes obvious when the network traffic is intense (i.e., with a message generation interval of 0.01 s per node), and the storage overhead (in terms of the extra size increment per message) is relatively large compared with the original size. For more general simulation settings, e.g., that in [48], the impact is almost unnoticeable. In terms of scalability, by comparing with the results obtained from 200 nodes in [48], we can see that the performance is mainly determined by the network traffic load and node contact density, instead of the total number of nodes. This is because intense network traffic and node contact intensity can happen in all network scenarios, regardless of the total number of nodes.

V. Further Discussions and Future Work

For further improvements, there are some issues worth exploring.

One possible concern is about the routing metric protection process. In our current solution, the sender initiates the routing metric comparison process by sending out its encoded routing metric in the routing request phase. Once receiving the routing request, each receiver makes a response based on the request, without knowing the comparison results. Based on the received responses, the sender can perform the comparisons, reveal the results, and choose the proper receivers as relays. However, one may suggest shifting the routing metric comparison workload to the receivers and letting the receivers make the routing decision so that only the proper receivers continue the conversation with the sender to reduce the computation and communication overhead. However, such an approach will not help much. If we let receivers perform the comparison, according to the process, the receivers become those to first send out the encoded routing metric. This will lead to extra interactions between the sender and receivers if we assume that the sender always initiates the routing request. Moreover, in the suggested case, although the routing metric comparison workload is distributed in the receivers from the sender, the routing metric response workload is accumulated to the sender from the receivers. In addition, more routing metric encoding workload will be introduced on the receivers' side.

Another feature, which is nice to have, is that the sender is able to perform the privacy-preserving comparison of the routing metrics of other receivers so that the best receiver or a few receivers can be chosen as the relay. However, such a feature can potentially invade the routing metric privacy. This is because if the sender can compare any two encoded routing metrics from different receivers, it can forge an encoded routing metric and perform the comparison with the real routing metric from a receiver. It can further repeat the comparisons with different forged routing metric values until it finds one value that is close enough to the real value of the other receiver's routing metric. However, it will be our interest to investigate proper security tools to enable the sender to perform secure comparisons with no privacy invasion risk.

Second, although the proposed scheme can defend most external attackers with the proposed authentication approaches, it does not integrate mechanisms resisting the attacks from internal attackers, e.g., black-hole attacks and Sybil attacks. We assume that the proposed scheme is operated on trustable and honest internal users. If a user is trusted by the CA, it is supposed to be honest and willing to help forward messages when possible. However, if this assumption does not hold, we should integrate other secure mechanisms to achieve corresponding protection. Although not the main focus of this paper, such mechanisms are well studied in the literature [31], [41]–[47], and they are relatively independent from our proposed scheme since they achieve different functions. However, we believe that efficient integration is feasible and will be of interest to us for future work.

VI. Conclusion

Opportunistic routing is widely employed in many mobile networks, e.g., DTNs, VANETs, and mobile sensor networks. Considering that the nodes' local and private information (i.e., routing metric) is extensively utilized in opportunistic routing, in this work, we have focused on its security and privacy concerns and proposed an advanced framework for opportunistic routing, providing various security and privacy preservation properties. A comprehensive evaluation was conducted to show the security and feasibility of the proposed framework.

References

[1] L. Zhang, B. Yu, and J. Pan, "GeoMob: A mobility-aware geocast scheme in metropolitans via taxicabs and buses," in *Proc. IEEE INFOCOM*, 2014, pp. 1779–1787.

[2] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 7, no. 3, pp. 19–20, Jul. 2003.

[3] F. Cadger, K. Curran, J. Santos, and S. Moffett, "A survey of geographical routing in wireless ad-hoc networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 621–653, 2nd Quart. 2013.

[4] Z. Zhi and Y. K. Choong, "Anonymizing geographic ad hoc routing for preserving location privacy," in *Proc. IEEE ICDCS Workshop*, 2005, pp. 646–651.

[5] M. Rahman, M. Mambo, A. Inomata, and E. Okamoto, "An anonymous on-demand position-based routing in mobile ad hoc networks," in *Proc. IEEE SAINT*, 2006, pp. 300–306.

[6] L. Zhou, D. Wu, B. Zheng, and M. Guizani, "Joint physical-application layer security for wireless multimedia delivery," *IEEE Commun. Mag.*, vol. 52, no. 3, pp. 66–72, Mar. 2014.

[7] Y. Qian and N. Moayeri, "Design of secure and application-oriented VANETs," in *Proc. IEEE VTC Spring*, 2008, pp. 2794–2799.

[8] Y. Qian, K. Lu, and D. Tipper, "A design for secure and survivable wireless sensor networks," *IEEE Wireless Commun.*, vol. 14, no. 5, pp. 30–37, Oct. 2007.

[9] A. Kate, G. Zaverucha, and U. Hengartner, "Anonymity and security in delay tolerant networks," in *Proc. IEEE SecureComm*, 2007, pp. 504–513.

[10] L. Guo, C. Zhang, H. Yue, and Y. Fang, "A privacy-preserving social-assisted mobile content dissemination scheme in DTNs," in *Proc. IEEE INFOCOM*, 2013, pp. 2301–2309.

[11] A. Yao, "Protocols for secure computations," in *Proc. FOCS*, 1982, pp. 160–164.

[12] M. Raya, P. Papadimitratos, and J.-P. Hubaux, "Securing vehicular communications," *IEEE Wireless Commun.*, vol. 13, no. 5, pp. 8–15, Oct. 2006.

[13] P. Papadimitratos *et al.*, "Secure vehicular communication systems: Design and architecture," *IEEE Wireless Commun.*, vol. 46, no. 11, pp. 100–109, Nov. 2008.

[14] R. Patra, S. Surana, and S. Nedevschi, "Hierarchical identity based cryptography for end-to-end security in DTNs," in *Proc. ICCP*, 2008, pp. 223–230.

[15] P. Papadimitratos, V. Gligor, and J. Hubaux, "Securing vehicular communications—Assumptions, requirements, and principles," in *Proc. ESCAR Workshops*, 2006, pp. 5–14.

[16] G. Calandriello, P. Papadimitratos, J. Hubaux, and A. Lioy, "On the performance of secure vehicular communication systems," *IEEE Trans. Dependable Secure Comput.*, vol. 8, no. 6, pp. 898–912, Nov./Dec. 2011.

[17] U. Shevade, H. H. Song, L. Qiu, and Y. Zhang, "Incentive-aware routing in DTNs," in *Proc. IEEE ICNP*, 2008, pp. 238–247.

[18] R. Lu, X. Lin, H. Zhu, X. Shen, and B. Preiss, "Pi: A practical incentive protocol for delay tolerant networks," *IEEE Trans. Wireless Commun.*, vol. 9, no. 4, pp. 1483–1493, Apr. 2010.

[19] L. Chen, S. Ng, and G. Wang, "Threshold anonymous announcement in VANETs," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 3, pp. 605–615, Mar. 2011.

[20] D. Chaum and E. Heyst, "Group signatures," in *Advances in Cryptology-EUROCRYPT*. Berlin, Germany: Springer-Verlag, 1991, pp. 257–265.

[21] E. Brickell, L. Chen, and J. Li, "Simplified security notions of direct anonymous attestation and a concrete scheme from pairings," *Int. J. Inf. Security*, vol. 8, no. 5, pp. 315–330, Oct. 2009.

[22] E. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in *Proc. ACM CCS*, 2004, pp. 132–145.

[23] H. Lin and W. Tzeng, "An efficient solution to the millionaires problem based on homomorphic encryption," in *Applied Cryptography and Network Security*. Berlin, Germany: Springer-Verlag, 2005, pp. 456–466.

[24] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," in *Proc. SCIS*, 2000, p. 26–28.

[25] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology-CRYPTO*. Berlin, Germany: Springer-Verlag, 2001, pp. 213–229.

[26] R. Dupont and A. Enge, "Provably secure non-interactive key distribution based on pairings," *Discr. Appl. Math.*, vol. 154, no. 2, pp. 270–276, Feb. 2006.

[27] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Advances in Cryptology-CRYPTO*. Berlin, Germany: Springer-Verlag, 2004, pp. 56–72.

[28] M. Scott and N. Costigan, "Implementing cryptographic pairings on smartcards," in *Cryptographic Hardware and Embedded Systems*. Berlin, Germany: Springer-Verlag, 2007, pp. 177–196, ser. Lecture Notes in Computer Science.

[29] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE simulator for DTN protocol evaluation," in *Proc. ICST SIMUtools*, 2009, p. 55.

[30] P. Papadimitratos, A. La Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *IEEE Commun. Mag.*, vol. 47, no. 11, pp. 84–95, Nov. 2009.

[31] R. Lu, X. Lin, and X. Shen, "Spring: A social-based privacy-preserving packet forwarding protocol for vehicular delay tolerant networks," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[32] L. Dóra and T. Holczer, "Hide-and-Lie: Enhancing application-level privacy in opportunistic networks," in *Proc. ACM Int. Workshop Mobile Opportunistic Netw.*, 2010, pp. 135–142.

[33] M. Radenkovic, I. Vaghi, S. Zakhary, and A. Benslimane, "AdaptAnon: Adaptive anonymity for service queries in mobile opportunistic networks," in *Proc. IEEE ICC*, 2013, pp. 1839–1844.

[34] Z. Le, G. Vakde, and M. Wright, "PEON: Privacy-enhanced opportunistic networks with applications in assistive environments," in *Proc. ACM PETRA*, 2009, pp. 76–84.

[35] C. Shi, X. Luo, P. Traynor, M. Ammar, and E. Zegura, "ARDEN: Anonymous networking in delay tolerant networks," *Ad Hoc Netw.*, vol. 10, no. 6, pp. 918–930, Aug. 2012.

[36] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *ACM Commun. ACM*, vol. 42, no. 2, pp. 39–41, Feb. 1999.

[37] S. Zakhary and M. Radenkovic, "Utilizing social links for location privacy in opportunistic delay-tolerant networks," in *Proc. IEEE ICC*, 2012, pp. 1059–1063.

[38] S. Zakhary, M. Radenkovic, and A. Benslimane, "The quest for location-privacy in opportunistic mobile social networks," in *Proc. IEEE IWCMC*, 2013, pp. 667–673.

[39] A. Benslimane, M. Radenkovic, and S. Zakhary, "Efficient location privacy-aware forwarding in opportunistic mobile networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 2, pp. 893–906, Feb. 2014.

[40] O. Hasan, J. Miao, S. Mokhtar, and L. Brunie, "A privacy preserving prediction-based routing protocol for mobile delay tolerant networks," in *Proc. IEEE AINA*, 2013, pp. 546–553.

[41] F. Li, J. Wu, and A. Srinivasan, "Thwarting blackhole attacks in disruption-tolerant networks using encounter tickets," in *Proc. IEEE INFOCOM*, 2009, pp. 2428–2436.

[42] M. Chuah, P. Yang, and J. Han, "A ferry-based intrusion detection scheme for sparsely connected ad hoc networks," in *Proc. IEEE MobiQuitous*, 2007, pp. 1–8.

[43] M. Uddin *et al.*, "Making DTNs robust against spoofing attacks with localized countermeasures," in *Proc. IEEE SECON*, 2011, pp. 332–340.

[44] A. Al-Hinai, H. Zhang, Y. Chen, and Y. Li, "TB-SnW: Trust-based Spray-and-Wait routing for delay-tolerant networks," *J. Supercomput.*, vol. 69, no. 2, pp. 593–609, Aug. 2014.

[45] S. Zakhary and M. Radenkovic, "Erasure coding with replication to defend against malicious attacks in DTN," in *Proc. IEEE WiMob*, 2011, pp. 357–364.

[46] S. Trifunovic, M. Kurant, K. Hummel, and F. Legendre, "Preventing spam in opportunistic networks," *Comput. Commun.*, vol. 41, pp. 31–42, Mar. 2014.

[47] S. Trifunovic, F. Legendre, and C. Anastasiades, "Social trust in opportunistic networks," in *Proc. IEEE INFOCOM Workshops*, 2010, pp. 1–6.

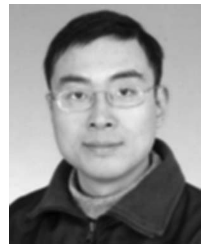
[48] L. Zhang, J. Song, and J. Pan, "Towards privacy-preserving and secure opportunistic routings in VANETs," in *Proc. IEEE SECON*, 2014, pp. 627–635.

Lei Zhang (M'14) received the Bachelor's degree in information security from China University of Geosciences, Wuhan, China, in 2010 and the Ph.D. degree from the Department of Computer Science, University of Victoria, Victoria, BC, Canada, in 2015.

His main research interest is in advanced wireless networks, including user mobility modeling, social characteristics, and security and privacy concerns.

AQ1 **Jun Song** (M'10) received the Bachelor's and Master's degrees from the China University of Geosciences, Wuhan, China, and the Ph.D. degree from Wuhan University, all in computer science.

He is currently an Associate Professor of computer science with the China University of Geosciences. His area of specialization is cryptography application and information security, and his current research interests include security analysis of cryptography application in wireless networks, applied network security, and cryptography security for big data.



Jianping Pan (SM'08) received the Bachelor's and Ph.D. degrees in computer science from Southeast University, Nanjing, China. AQ2

He was a Postdoctoral Researcher with the University of Waterloo, Waterloo, ON, Canada. He is currently a Professor of computer science with the University of Victoria, Victoria, BC, Canada. He has also been with Fujitsu Labs and NTT Labs. His area of specialization is computer networks and distributed systems, and his current research interests include protocols for advanced networking, performance analysis of networked systems, and applied network security.

Dr. Pan has been serving on the Technical Program Committees of major computer communications and networking conferences, including the IEEE International Conference on Computer Communications, the IEEE International Conference on Communications, the IEEE Global Telecommunications Conference (Globecom), the IEEE Wireless Communications and Networking Conference, and the IEEE Consumer Communications and Networking Conference. He is the Ad Hoc and Sensor Networking Symposium Cochair of the 2012 IEEE Globecom and an Associate Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

AUTHOR QUERIES

AUTHOR PLEASE ANSWER ALL QUERIES

AQ1 = Please provide the year when the degrees were received by author “J. Song.”
AQ2 = Please provide the year when the degrees were received by author “J. Pan.”

END OF ALL QUERIES