A Secure Message Transaction Protocol for Delay Tolerant Networks

Zhongtian Jia[a b], Lixiang Li[b], Zhuoran Yu[c], Shudong Li[b], Yixian Yang[b]
[a] School of Information Science and Engineering, Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan 250022, China
Email: jiazht@bupt.edu.cn
[b] Information Security Center, Beijing University of Posts and Telecommunications, Beijing 100876, China
Email: {jiazht, lixiang, lsd, yxyang}@bupt.edu.cn
[c] School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China
Email: charnugagoo@gmail.com

Abstract— **Delay/disruption tolerant network adopts a store-carry-and-forward mechanism, of which all the participants are assumed to cooperate with one another in message delivery, to overcome the challenges of the intermittent connection and the time-varying network topology. Unfortunately, there are always some nodes deviating from the rules in order to save their own precious resources. To address the selfishness and the security problems, we propose a secure message transaction protocol for delay tolerant networks, in which the messages are encrypted by the source nodes and authorized by the TTP before they are propagated in the networks. The signatures are aggregated sequentially by the source node and the subsequent forwarders. The aggregated signatures record the message delivery paths. By checking the signatures aggregated so far, the intermediate nodes can be sure of the message authenticity and decide whether to deliver the messages to the next hops or not. Thus, the free riding attacks and path forging attacks are prevented. Furthermore, the payment mechanism of the proposed scheme makes the participants have no incentive to launch the collusion attacks in the path disclosure. In addition, the hash binary tree is harnessed to decrease the number of authorization times when fragmentation is invoked at the source, and the relationship between the delivery overhead and the authorization times is presented. In the proposed scheme, the cipher computation and bundle envelopment can be performed off-line, without the information of next hops, while the DTN nodes are driving freely without any DTN connections. Performance analysis and simulation results prove that this off-line computing design is efficient in terms of delivery ratio and delivery speed.**

Index Terms— **delay tolerant networks, incentive, secure, message transaction, aggregate signature**

I. INTRODUCTION

DELAY/disruption tolerant network (DTN) is a network architecture working as an overlay above the transport layers of the networks it interconnects, aiming to solve the problems merged in challenging wireless environments, such as inter-planetary networks, disaster relief team networks and military Ad hoc networks [1]. Now, it is often deployed in other environments, such as social networks [2] and vehicular networks etc. [3]. The common characteristics of these environments are 1) mobile-end, 2) contacts unpredictable, 3)be short of or limited network infrastructures, 4)lacking continuous end-to-end connectivity, and 5)possible long round trip time between the source and the destination [4].

In terms of the features mentioned above, the DTN message transmission adopts a store-carry-and-forward mechanism [5]. Messages are exchanged while there is an opportunistic connection between two candidate nodes. If it is not the destination of the messages, the message receiver will store and carry the received messages until the destination or the next nodes that may deliver the messages to the destination are met. Then, the messages are forwarded to the new encounters. Every node in DTN plays roles not only as a terminal but also as a router that is responsible for routing the messages to the next hops. This kind of transmission mechanism is based on the hypothesis that all the participants must be credible and cooperate with one another. In other words, every DTN node should act as a data router, which stores, carries and forwards the messages for others in the process of its movement. Unfortunately, there will always be some selfish nodes or even malicious ones who download messages from the networks only but refuse to serve as routers in order to save their own precious wireless network resources, like energy, storage space and computation power [6], [7], which leads to the sparsity of the intermediate forwarders and a worse delivery ratio. Considering the worst case, all of the participants do not deliver messages for others. And in this case, the only way to transmit the messages will be face-to-face with the source node and the destination node themselves, of which the probability actually can

Manuscript received August 10, 2011; revised January 2, 2012; accepted April 16, 2012. © 2005 IEEE.

This work was supported in part by the Foundation for the Author of National Excellent Doctoral Dissertation of PR China (FANEDD) (GrantNo.200951), the National Natural Science Foundation of China (GrantNo.61070209), the Specialized Research Fund for the Doctoral Program of Higher Education (GrantNo.20100005110002), and National S&T Major Program (GrantNo.2011ZX03002-005-01), the Project of Jinan City Science and Technology Program (GrantNo.201202014), the Project of Shandong Province Higher Educational Science and Technology Program (GrantNo.J12LN18).

doi:10.4304/jcm.7.8.622-633

be very small due to the intrinsic characteristics of DTN. Hence, measures must be taken to overcome the sparsity of the volunteer forwarders when the selfish or malicious nodes exist. In other words, the free riding problem must be solved before DTNs can be deployed commercially. The straightforward method is to take the mandatory or incentive schemes by giving the contributors some rewards to encourage the nodes more collaborative on the message delivery. Another feasible measure to overcome the sparsity of the intermediate forwarders is to adopt the epidemic routing algorithm which is robust to selfishness and more reducing of delivery cost according to [8], especially when the DTN nodes are deployed sparsely in the networks. Furthermore, security is another important issue in DTNs, since information transmitted over the DTN could be sensitive [4] and the privacy should be protected in some applications [9] [10]. Many approaches have been proposed to address the aforementioned issues, but don't use the current existing heterogeneous wireless network environment effectively. Nowadays, it is very common that in DTNs, a DTN node is equipped with multiple interfaces using different wireless technologies. For example, in VANETs, a vehicle could be equipped with both satellite communication system and dedicated short range communication (DSRC) system, where the DSRC system provides high data transmission ratio but with a relative small communication range while satellite communication can cover a much large area.

In this paper, we propose a secure transaction protocol to solve the selfishness and security problems in a practical DTN. In this environment (see Fig.1), we assume there exists two kind of wireless channels, the long range narrow band (LRNB) channels between the nodes and the base station (BP) or the access point (AP) (e.g., cellular interface), and the short range broad band (SRBB) channels between the node pairs (e.g., bluetooth). The LRNB channels can be accessed from the nearby access point (AP) and the SRBB channels are transient and stochastic, existing while the DTN nodes are in a very near area and can communicate with one another through the SRBB signal. Authentication messages and payment redeem messages are transmitted over the LRNB channels and large volume data are propagated over the SRBB channels hop by hop. We further assume the epidemic routing algorithm is adopted to overcome the forwarder sparsity problem. For example, we intend to transmit a large volume of photos and videos to another place without going there by ourselves. How can we complete this task? One of the feasible methods is to propagate the messages through the DTNs. We can transmit or copy the messages to the persons who are equipped with the DTN device and have the ability to deliver the messages towards the destination subsequently. Before sending the messages to the intermediate deliverer, some sort of credential should be appended to the messages, by which the intermediate nodes can identify the authenticity of the received messages and decide whether to deliver the received messages to the next hop or not. In addition, we want to keep the messages confidential to the arbitrator on the delivery path. Thus we must encrypt the messages before sending them out.

In the proposed protocol, the messages are treated as packages of post services. Before being propagated over the networks, the messages are enveloped at the source end and attached the post stamps signed by the third trust party (TTP). On receiving the messages, the intermediate nodes check their credentials, then they decide whether to serve as a router or not. The delivery paths are recorded by the sequential aggregated signatures. On the delivery path, no one can read the content of the message as plain text. After the messages get to the destination, the last intermediate node is responsible for requesting the receipt from the destination node and disclosing the delivery path to TTP. Then, TTP can allocate rewards to the nodes on the delivery path in terms of the volume of the messages. In summary, the contributions of this paper are as follows:

1) We propose a secure transaction protocol for message transmission in DTNs, which takes the aggregate signature approach to record the delivery path.
2) We find the relationship between the number of authorization rounds and the transmission overhead when fragmentation is invoked at the source.
3) We present an incentive-compatible payment mechanism to stimulate cooperation and block free riding attacks and collusion attacks. Further, the payment mechanism is fair, where all DTN nodes participating in data forwarding are ensured to receive their rewards due to the fact that TTP first checks the sender's credits and charges the sender before authorizing the message sent.

The rest of the paper is organized as follows. In section II, we describe the related works about incentive schemes. Section III reviews the preliminaries that are referred to in this paper. In section IV, we propose our transaction protocol. Security is analyzed in section V. Performance evaluation and conclusions are presented in section VI and section VII, respectively.

II. Related Works

Recently, the free-riding problem in DTN has been discussed widely. In 2009, Karaliopoulos, M. modelled the message delivery and analyzed the performance of two popular routing alternatives, the unrestricted and the two-hop relay schemes, under the condition of nodes selfishness [11]. It showed that the selfish behavior in data forwarding can heavily affect the performance of the networks, and suggested that some measures should be taken to coordinate the cooperation of the nodes. In 2011, [12] investigated the impact of node selfishness on multicasting. By modelling the message delivery process with social selfishness as a two dimensional continuous time Markov chain, [12] showed that different selfish behaviors may have different impacts on different performance metrics. In order to achieve a better network performance, DTN message forwarding should be teamwork. However, the monitoring and coordinating of the

forwarding behavior is a challenged problem due to the inherent characteristics of DTN. In the very recent years, a number of schemes and protocols have been proposed intending to address this problem, which either try to prevent the selfish behavior or to stimulate the cooperation in data forwarding.

The selfishness-preventing methods have been investigated in literature [13]–[15]. [13] studied both the quality-of-service (QoS) trust properties (connectivity) and the social trust property (honesty and unselfishness) in DTN. By incorporating the two kinds of trust (QoS trust and social trust) into the trust management for routing decision, higher delivery ratio and shorter message delay were able to be achieved without incurring high message overhead. [14] proposed a modular solution, by which the misbehaving nodes that did not forward packets to save their own resources could be monitored, detected, and isolated. In order to get better services from others and not to be isolated, the DTN nodes had to behave cooperatively. To drive DTN nodes to cooperate one another for their own interest optimization, [15] proposed Pay-for-Gain (PFG) method, which harnessed the game theory and loan-credit theory to investigate an equilibrium point that optimized the nodes' interest.

However, it is very difficult to monitor and detect the selfish behaviors due to a lack of network infrastructure and opportunistic connectivity. Therefore, a large number of incentive schemes, intending to stimulate the participants to cooperate, have been proposed. In 2007, Buttyan et al. put forward a barter based method to discourage selfish behavior and stimulate cooperation among nodes in the applications of personal wireless communications [16]. In their scheme, the message holder asked a certain number of rewards from the next receivers before delivering the held message to them. On receiving the message at last, the destination node recovered the rewards from TTP, which was responsible for the rewards allocation and arbitration. A similar barter scheme was proposed in [17], in which a DTN node received a bundle from another DTN node only if it also provided a bundle in return at the same time. With the help of these barter mechanisms, the baleful impact of selfish behavior on the DTN performance can be eliminated.

In [18], an incentive mechanism of tit-for-tat was discussed. By incorporating generosity and contrition, the bootstrapping and exploitation problems were addressed and an incentive-aware routing protocol was presented, which allowed selfish users to adaptively optimize their individual performance subjecting to TFT constraints. Srinivasan [19] studied the selfish problem in publish-subscribe framework and proposed a cost model based incentive scheme encouraging data forwarding in publish-subscribe framework, in which the receivers took the message on if they were willing to carry it. It suggested that the cost model based strategy could maximize total utility by encouraging selfish nodes to participate in carrying the message.

The layered-coin based incentive schemes were researched in [7], [20]. In [7], Zhu et al. proposed a multilayer credit based incentive scheme. In their scheme, the information of the intermediate nodes on the delivery path was injected into a multilayered coin capable of being recovered by the destination node. Thus the intermediate relay nodes were rewarded. By attaching an incentive on the sending bundle, the incentive mechanism of [20] could stimulate the selfish nodes to cooperate in message delivery. In these schemes, the information of the next hop must be recorded in the current layer, thus the layered coins have to be generated online, which maybe affect the network performance, especially when concurrent connections occur.

Li et al. [21] studied the routing socially in DTNs, and proposed a social selfishness aware routing (SSAR) algorithm. By considering both of the willingness and the connection opportunity of a forwarder, a forwarding strategy that is better than purely contact-based approaches could be made. Leveraging the game theory, [3] and [22] designed the incentive data forwarding scheme for mobile wireless networks of selfish individuals. After several rounds of games, a Nash equilibria among selfish nodes could be established, thus they had no interest to deviate the rules.

Different from the previous works, we proposed a end-to-end secure transaction protocol for message delivery in DTN. The proposed method treats the message as the package of the post services. The candidate messages are enveloped and attached stamps. In this way, the intermediate forwarders are able to verify the authenticity of the messages but not able to read enveloped messages. The delivery paths are recorded by aggregated signatures and disclosed by the last mediate forwarders. Incentive rewards are allocated by the TTP. Path forging attacks and free riding attacks are thwarted by the sequential aggregated signatures. The payment mechanism proposed in this paper make the participants have no incentive to launch the collusion attacks.

III. Preliminaries

In this section, we review bilinear maps and the sequential aggregate signature briefly. For more detail about these preliminaries, see [23].

A. Bilinear maps

Let $\mathbb{G}$ and $\mathbb{G}_T$ are multiplicative cyclic groups of prime order p, and g is the generator of $\mathbb{G}$. We say $e \quad \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is an efficiently computable map, and $\mathbb{G}$ is bilinear group, if they satisfy the following conditions:

$\mathbb{G}$ and $\mathbb{G}_T$ are both equipped with an efficiently computable multiplicative operation.
Bilinear: $\forall\ u,\ v \in \mathbb{G}$ and $\forall\ a,\ b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$;
Non-degennerate: $e(g, g) \neq 1$.
Symmetry: $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$

B. Aggregate Signature

An aggregate signature scheme is a digital signature which enables us to compress a list of distinct signatures into one signature. Any verifier can be convinced of the correctness of all the signatures by verified only once. One aggregate mechanism is general aggregation, in which n distinct signatures σ_1, σ_2, $\cdots$, σ_n on the same message m are aggregated into one signature σ or they are aggregated incrementally, such that σ_1 and σ_2 are aggregated into σ_{12}, σ_{12} and σ_3 are aggregated into σ_{123}, and so on. The other aggregate mechanism is sequential aggregation. Different from the general aggregate signature, the sequential aggregate signature can only be performed during the signing process. Thus, user 1 generates σ_1 by signing message m_1, user 2 aggregates σ_1 and message m_2 into signature σ_2, user 3 aggregate signature σ_2 and message m into signature σ_3, etc. Hence, the sequential signature has an explicit signing order of the signers. Here, we only briefly review the sequential aggregate signature scheme [23] that is used in this paper in the following paragraphs of this subsection.

A sequential aggregate signature scheme (see [23]) works as follows:

1) **Initialization**: Let $\mathbb{G}$ and $\mathbb{G}_T$ are bilinear groups with the common prime order of p. $e \quad \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$ is a bilinear map. u, v and g are generators of group $\mathbb{G}$. H_1:$\{0,1\}^* \to \mathbb{G}$ and H_2:$\{0,1\}^* \to \mathbb{Z}_p^*$ are hash functions. The system chooses a random $\alpha \in \mathbb{Z}_P$ and computes g^α. The system publishes the parameters $(p, \mathbb{G}, \mathbb{G}_T, e, u, v, g, g^\alpha, H_1, H_2)$ as its public key, and keeps α secret as its private key.
2) **Registration**: On receiving the registration request of user i with the input ID_i, system computes $H_1(ID_i)^\alpha$ and issues it to user i securely. Here, $H_1(ID_i)^\alpha$ serves as the private key of node i corresponding to its public key, ID_i.
3) **Signing**: On inputs private key sk_{ID}, m_i, $((ID_1, m_1), (ID_2, m_2), \cdots, (ID_{i\ 1}, m_{i\ 1}))$, σ (σ is the signature aggregated so far.), σ is parsed as (X, Y, Z), (For the first signer, σ is defined as $(1_G, 1_G, 1_G)$). Then, the signing algorithm chooses a random $r \in \mathbb{Z}_p$. Let s_j denotes $ID_1||m_1||ID_2||m_2||\cdots||ID_j||m_j$, $j \geq 1$, the signing process is as follows:

   a) $X \leftarrow X \cdot u^r \prod_{j=1} {}^{H_2(s_j)} \cdot H_1(ID_i)^\alpha$
   b) $Y \leftarrow Y^{H_2(s\ )^{-1}} \cdot v^r \cdot H_1(ID_i)^\alpha$
   c) $Z \leftarrow Z^{H_2(s\ )^{-1}} \cdot g^r$

   Finally, the tuple (X, Y, Z) is outputed as the new aggregated signature. Therefore, the format of an aggregate signature is as follows: $\sigma = (X, Y, Z)$, where,

$$
\begin{aligned}
X &= \prod_i^n u^{r\ \prod_{j=1} {}^{H_2(s_j)}} \cdot H_1(ID_i)^\alpha \\
Y &= \prod_i^n (v^r \cdot H_1(ID_i)^\alpha)^{(\prod_{j=\ +1}^n H_2(s_j))^{-1}} \\
Z &= \prod_i^n g^{r\ \cdot (\prod_{j=\ +1}^n H_2(s_j))^{-1}}
\end{aligned}
$$

4) **Verification**: On inputs pk_T, (ID_1, m_1), $\cdots$, (ID_n, m_n) and the signature σ, the verification algorithm executes as follows:
   a) The algorithm first checks the repeatability of the IDs. It returns 0 if not all the IDs are distinct.
   b) The algorithm parses σ as (X, Y, Z) and computes q= $e(\prod_i^n H_1(ID_i)^{(\prod_{j=\ +1}^n H_2(s_j))^{-1}}, g^\alpha)$
   c) The system checks if

$$e(Y, g) \stackrel{?}{=} e(v, Z) \cdot q. \tag{1}$$

   If not, the algorithm returns 0. Else, goes to the next step.
   d) The algorithm computes $Z' = Z^{\prod_{=1}^n H_2(s\ )}$ and verifies if

$$e(X, g) \stackrel{?}{=} e(Z', u) \cdot e(\prod_i^n H_1(ID_i), g^\alpha)). \tag{2}$$

   If it is true, the algorithm returns 1. Else, it returns 0.

C. Identification Based Encryption

1) **Encrypt:** For a given message M, the encryption process works as follows:
   a) The encryptor chooses a random $r \in \mathbb{Z}_p^*$ and computes

$$C_1 = g^r \tag{3}$$

   b) The encryptor computes g_{ID} = $e(H_1(ID), g^\alpha)$;
   c) The encryptor computes

$$C_2 = M \oplus H_3(g_{ID}^r) \tag{4}$$

   where $H_3 \quad \mathbb{G}_T \to \{0,1\}^*$ is a hash function.
   The ciphertext of M is C=(C_1, C_2).
2) **Decrypt:** On inputs C and sk, the decryption of cipher-text C works as follows:
   a) The decryptor parse C as (C_1, C_2);
   b) The decryptor computes

$$M = C_2 \oplus H_3(e(sk, C_1)) \tag{5}$$

For more detailed information about this encryption scheme, see [24].

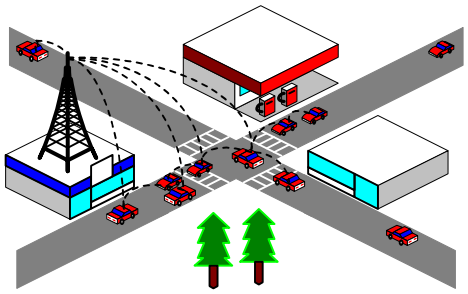

Figure 1. Network model. Dotted line denotes wireless signals.

IV. The Proposed Scheme

In this section, we formally describe the proposed transaction protocol. In Section IV-A, we introduce the network model and assumptions. Section IV-B defines terminologies and notations used in the proposed protocol. In section IV-C, we present the basic transaction protocol without fragmentation. Fragmentation of the proposed transaction protocol is discussed in Section IV-D.

A. Network Model and Assumptions

We envision a DTN composed of vehicles in a city. In this environment there are two kinds of communication channels. One is the LRNB channel between a base station(BS) (or AP) and a vehicle, such as GSM and GPRS, and the other is SRBB channel existing between two vehicles, such as Bluetooth and 802.11b see Fig.1. The former is called B2V communication which always exists and the later is called V2V communication which occurs while one DTN node falls in the other's local broadband wireless communication area. The message is transmitted between two vehicles over the SRBB channels by store-carry-and-forward mechanism. Small volume data of authentication are transmitted between a BS and a vehicle over the LRNB channels once in a while.

We further assume that every DTN node has the process ability to perform cryptography computation and an unique ID as its public key. In practice, the public key may be a transformed format of the ID, for example a hash value of it. Here we use the ID as public key directly for convenience in description in this paper.

B. Terminologies and Notations

We define the following new notation before we discuss the basic transaction protocol.

1) DTN node: refer to the car with a DTN modular;
2) $E(\cdot)$: symmetric algorithm;
3) m: message that will be transmitted;
4) $E_k(m)$: encrypts message m using symmetric algorithm with secret key k;
5) $H_0(\cdot)$: $\{0,1\}^* \rightarrow \{0,1\}^\tau$ and $H_3(\cdot)$ $\mathbb{G}_T \rightarrow \{0,1\}^*$ are hash functions;
6) ID_i: the identity of node i, specifically, ID_0, ID_1 and ID_n denote the identity of the TTP, the identity of the source node and the identity of the destination node, respectively; We use ID_T and ID_0 denote the identity of TTP interchangeably in different context;
7) $\mathbb{K}$: the key space of a symmetric algorithm;
8) ttl: life time of the data to be transacted;
9) $L(m)$: the length of the message m;
10) λ: the transaction price of per unit length of the message;
11) O: transaction number;

C. Transaction Protocol

The proposed transaction protocol has four phases which are Initialization, Registration, Authorization, Propagation and Payment.

1) Initialization: In this phase, system generates the parameters and the master key of TTP. To do this, TTP runs the initialization algorithm of the aggregate signature and outputs the public parameters $(p, \mathbb{G}, \mathbb{G}_T, e, u, v, g, g^\alpha, H_1, H_2)$. The master secret key of TTP is α. Additionally, TTP generates the signing key $H_1(ID_0)^\alpha$ for himself.

2) Registration: Before access the DTN services, every DTN node needs to register to TTP. On receiving the registration request of node i, TTP computes $H_1(ID_i)^\alpha$ and issue it as a private key to node i securely. Correspondingly, ID_i acts as the public key. In practice, the public key may be a variation of ID_i in forms, such as the hash value of it. To facilitate description, we regard ID_i as the public key of node i in this paper. As an optional way, the public parameters $(p, \mathbb{G}, \mathbb{G}_T, e, u, v, g, g^\alpha, H_1, H_2)$ of the system may load to the DTN node in this step.

3) Authorization: This phase performs the authorization of the message to be transmitted. The source node requests TTP for message transaction authorization over the LRNB wireless channel. On receiving the data transaction request, TTP checks the account of the source node or destination node in terms of the specific payment policy. If there is enough money remained on the account of the source/destination for this transaction, TTP will authorize this transaction by issuing an authenticator to the requester. Otherwise, TTP will not accept this request by replying a deficit message. For security and transmitting overhead consideration, only the necessary information of authorization should be passed through the narrow band authentication channel. Specifically, the plain-text should not be disclosed to TTP in the authorization process for privacy. Formal description of this process is as follows.

1) The source node chooses a random number $r_1', r_1'' \in \mathbb{Z}_p^*$, $K \in \mathbb{K}$ and computes

$$c_0 = E_K(m), \tag{6}$$

$$g_{ID} = e(h_1(ID_n), g^\alpha), \tag{7}$$

$$c_1 = g^{r_1''}, \tag{8}$$

$$c_2 = K \oplus H_3(g_{ID}^{r_1''}). \tag{9}$$

where (c_1, c_2) is the cipher text of K. Let c denote $c_0||c_1||c_2$.

2) The source node signs $ID_1||M_1'$, where M_1'=$ID_1||ID_n||H_0(c)||L(c)||ttl$, as follows:
   a) $X \leftarrow u^{r' H_2(ID_1 \ M_1')} \cdot H_1(ID_1)^{\alpha}$
   b) $Y \leftarrow v^{r'} \cdot H_1(ID_1)^{\alpha}$
   c) $Z \leftarrow g^{r'}$

   The signature σ is (X, Y, Z).
3) The source node send the R=(M_1', σ, t_1) to the TTP over the LRNB channel, where t_1 is the time stamp.
4) On receiving the request R, the TTP fist parses R as (M_1, σ, t_1). Then it compares the system time t with t_1. Checks if $t - t_1 < \Delta t$. If it true, goes to next step, else rejects it.
5) TTP parses σ as (X, Y, Z) and computes: q=$e(H_1(ID_1)^{(H_2(ID_1 \ M_1'))^{-1}}, g^{\alpha})$. The system checks
$$e(Y, g) \stackrel{?}{=} e(v, Z) \cdot q. \tag{10}$$
   If not, TTP rejects this request. else, the algorithm computes $Z' = Z^{H_2(ID_1 \ M_1)}$ and verifies
$$e(X, g) \stackrel{?}{=} e(Z', u) \cdot e(H_1(ID_1), g^{\alpha})). \tag{11}$$
   If not, TTP reject this request, else goes to next step.
6) TTP checks if $\omega_{ID} \geq \lambda \cdot L(c)$. If it is true, goes to the next step, else rejects this request. In order to check the account balance, TTP may need to access the bank net work.
7) TTP chooses a random $r_0 \in \mathbb{Z}_p^*$ and generates the authenticator of this transaction as follows:
   a) M_0=$O||ID_1||ID_n||H_0(c)||L(c)||ttl$
   b) $X \leftarrow u^{r_0 H_2(ID_0 \ M_0)} \cdot H_1(ID_0)^{\alpha}$
   c) $Y \leftarrow v^{r_0} \cdot H_1(ID_0)^{\alpha}$
   d) $Z \leftarrow g^{r_0}$

   The authenticator σ_0 is (X, Y, Z).
8) TTP replies the source node with $M_0||\sigma_0||t_0$, where t_0 is a time stamp.
9) On receiving the response, the source node first checks its freshness. If $t - t_0 < \Delta t$. If not, halts this transaction and requests for authorization again, else, goes to next step.
10) The source node verifies the signature of TTP. It parses σ_0 as (X, Y, Z) and computes: q=$e(H_1(ID_0)^{(H_2(ID_0 \ M_0))^{-1}}, g^{\alpha})$. The system checks
$$e(Y, g) \stackrel{?}{=} e(v, Z) \cdot q. \tag{12}$$
    If not, the source node breaks this request. else, computes $Z' = Z^{H_2(ID_1 \ M_1)}$ and verifies
$$e(X, g) \stackrel{?}{=} e(Z', u) \cdot e(H_1(ID_0), g^{\alpha})). \tag{13}$$
    If not, the source node breaks this request, else goes to next step.
11) The source node generates the aggregate signature based on signature aggregated so far, i.e. σ_0. First, the source node choose a random $r_1 \in \mathbb{Z}_p^*$ and parse σ_0 as (X, Y, Z). Then it executes:
    a) $s_j = ID_0||M_0|| \cdots ||ID_j||M_j$, $j = 0, 1$, $M_1 = T_1$, T_1 is the current time.
    b) $X \leftarrow X \cdot u^{r_1 H_1(s_0) H_2(s_1))} \cdot H_1(ID_1)^{\alpha}$
    c) $Y \leftarrow Y^{H_2(s_1)^{-1}} \cdot v^{r_1} \cdot H_1(ID_1)^{\alpha}$
    d) $Z \leftarrow Z^{H_2(s_1)^{-1}} \cdot g^{r_1}$

    The aggregate signature σ_1 is (X, Y, Z).

4) Propagation:

1) The source node delivers the message $ID_0||M_0||ID_1||M_1||\sigma_1||c$ to the networks. After several rounds propagation and signature aggregation, the message takes the form as $ID_0||M_0|| \cdots ||ID_{i \ 1}||M_{i \ 1}||\sigma_{i \ 1}||c$.
2) On receiving the message, ith node parses the message as $(ID_0, M_0, \cdots, ID_{n \ 1}, M_{n \ 1}, \sigma_{n \ 1}, c')$, then parses M_0 as $(O, ID_1, ID_n, H_0(c), L(c), ttl)$. It compares the destination ID with its own. If they are identical, goes to step 8), else puts the message into a temporary buffer for the time being. In the interval time before meets other nodes, ith performs the verification. It follows the next steps:
   a) Checks the distinction of the IDs. If they are distinct each other, goes to the next step. Else, breaks and discards the message.
   b) Checks if $H(c') = H_0(c)$, $ttl < T_i$, and $L(c') < L(c)$, T_i is the current time. If not, breaks this verification and discards this message. Else, goes to next step.
   c) Let $s_j = ID_0||M_0 \cdots ||ID_j||M_j$, $j > 0$, q=$e(\prod_{k \ 1}^{i} H_1(ID_k)^{(\prod_{j=k+1}^{-1} H_2(s_j))^{-1}}, g^{\alpha})$. Parses $\sigma_{i \ 1}$ as (X, Y, Z). and checks if
$$e(Y, g) \stackrel{?}{=} e(v, Z) \cdot q. \tag{14}$$
      If not, it breaks and discards the message. Else, it computes $Z' = Z^{\prod_{k=1}^{-1} H_2(s_k)}$ and verifies if
$$e(X, g) \stackrel{?}{=} e(Z', u) \cdot e(\prod_{k}^{i \ 1} H_1(ID_k), g^{\alpha})). \tag{15}$$
      If it is true, goes to the next step to aggregate new signature. Else, breaks and discards this message.
3) On inputs ID_i and M_i, where M_i is the current time. $r_i \in \mathbb{Z}_p^*$ is a random. Let s_j denote $ID_0||M_0||ID_1||M_1|| \cdots ||ID_j||M_j$, $j \geq 0$, the aggregate process of node i is as follows:
   a) $X \leftarrow X \cdot u^{r \ \prod_{j=1} H_2(s_j)} \cdot H_1(ID_i)^{\alpha}$
   b) $Y \leftarrow Y^{H_2(s \ )^{-1}} \cdot v^{r \ } \cdot H_1(ID_i)^{\alpha}$
   c) $Z \leftarrow Z^{H_2(s \ )^{-1}} \cdot g^{r}$

   The aggregated signature σ_i so far is (X, Y, Z). The format of message to be propagate is $ID_0||M_0|| \cdots ||ID_i|M_i||\sigma_i||c$. With the enveloped message in hand, the intermediate forwarder executes the next step.
4) On encountering the next neighbor, the message carrier compare the ID_n in M_0 with the identity of the encountered neighbor, if they are identical, goes to step 5, else propagates the message to the encountered neighbor.

5) The message carrier sends receipt request req to the destination, where $req = O||ID_i||t_r$, t_r is the current time of the system.
6) On receiving the receipt request, the destination compare the system time t and t_r. If $0 < t - t_r < \Delta t$, it choose a random number r_d and signs the receipt rec, where $rec = c_r||ID_i||t_r||t_d$, as follows:
   a) $X_r \leftarrow u^{r_d H_2(ID_n \ rec)} \cdot H_1(ID_n)^{\alpha}$
   b) $Y_r \leftarrow v^{r_d} \cdot H_1(ID_n)^{\alpha}$
   c) $Z_r \leftarrow g^{r_d}$

   Then the destination replies the forwarder with the receipt and the signature $rec||\sigma_r$, where $\sigma_r = (X_r, Y_r, Z_r)$. Else, the destination rejects this request.
7) On receiving the receipt and its signature, the forwarder i verifies the signature. If it is true, the forwarder deliver the carried message to the destination, else, doest not launch message transmission.
8) The destination node accepts the message and performs the following verification.
   It first parses the message as $(ID_0, M_0, \cdots, ID_{n\ 1}, M_{n\ 1}, \sigma_{n\ 1}, c)$, then parses M_0 as $(M_0{=}O, ID_1, ID_n, H_0(c), L(c), ttl)$. let $s_j{=}ID_0||m_0||ID_2||m_2||\cdots||ID_j||m_j$. $q{=}e(\prod_{k}^{n\ 1} H_1(ID_k)^{(\prod_{j=k+1}^{n-1} H_2(s_j))^{-1}}, g^{\alpha})$. Check if

$$e(Y, g) \stackrel{?}{=} e(v, Z) \cdot q. \tag{16}$$

   If not, breaks and discards the message. Else, computes $Z^{'} = Z^{\prod_{k=1}^{n-1} H_2(s_k)}$ and verifies if

$$e(X, g) \stackrel{?}{=} e(Z^{'}, u) \cdot e(\prod_{k}^{n\ 1} H_1(ID_k), g^{\alpha})). \tag{17}$$

   If it is true, goes to the next step. Else, it breaks and discards this message.
9) On inputs ID_n and M_n, where M_i is the current time. $r_n \in \mathbb{Z}_p^*$ is a random. Let s_j denotes $ID_0||m_0||ID_2||m_2||\cdots||ID_j||m_j$, $j \geq 0$. The destination node computes:
   a) $X \leftarrow X \cdot u^{r_n \prod_{j=1}^{n} H_2(s_j)} \cdot H_1(ID_n)^{\alpha}$
   b) $Y \leftarrow Y^{H_2(s_n)^{-1}} \cdot v^{r_n} \cdot H_1(ID_n)^{\alpha}$
   c) $Z \leftarrow Z^{H_2(s_n)^{-1}} \cdot g^{r_n}$.

   The aggregated signature σ_n so far is (X, Y, Z). The destination sends σ_n to the node that just transmitted the message. (σ_n will be submitted to the TTP with the receipt by the last forwarder for its reward.)
10) The destination node parses c as (c_0, c_1, c_2) and decrypts c_0, c_1 and c_2 as follows:

$$K = C_2 \oplus H_3(e(sk_{ID_n}, c_1)) \tag{18}$$

$$m = D_K(c_0). \tag{19}$$

As so far, the destination node has gotten the plain-text sent by the source node. Next, we discuss the payment mechanism.

5) Payment: In this part, we discuss how to reward the relay nodes that has contributed to this message delivery. However, it is not a trivial matter in such a challenged network environment since the rewarding mechanism has a very important impact on the enthusiasm of DTN nodes. In previous works, such as the mechanism of [3], the disclosure of delivery paths depends on the destination node. In the proposed scheme, it is the last delivery node that is responsible for the aggregate signature submitting. The aggregate signature records the message delivery path. As long as a node lies in the paths of message delivery from the source to the destination, which is completed with the life time of the message, ttl, it should be regarded as a contributor that has completed the transfer task on time and should be rewarded. But, in order to encourage the DTN nodes to forward the carried messages as quickly as possible, it is reasonable that only the first finished one or several paths is awarded. However, it is possible for the source node to collude with the destination and the last forwarder. Assume w_0 and n denote the total payables of this transaction and the nodes on the delivery path including the source and the destination, respectively. Each node on the delivery path except the source should receive $\frac{w_0}{n\ 1}$ \$. However, if the source node colludes with the last delivery or the destination by giving them $\frac{w_0}{n\ 1} + \delta$ \$, $\delta > 0$, they may be breaks the protocol by give up aggregate signature submitting or refuse signing the receipt. Accordingly, the source node can get $w_0 - \left(\frac{w_0}{n\ 1} + \delta\right)$ \$ by this collusion.

In order to defeat this kind of collusion attacks, some mechanism should be designed to ensure the money they gotten by collusion is no more than they act honestly. To achieve such a goal, TTP pre-charged the source node Δw_0 \$ and the destination node Δw_1 \$, respectively. In addition, to prevent the destination node signing an obsolete transaction to win a contributing reward, the destination node will be charged w_1 \$ per transaction.

1) To prevent the source node colludes with the destination node, $\Delta w_0 \leq \Delta w_1$ should hold.
2) To stop the destination node from colluding with the last forwarder, $w_0 \leq \Delta w_0$ should hold.
3) To block the destination node sign an obsolete transaction, $\frac{w_0}{n\ 1} + \Delta w_1 \leq w_1$ should be true.

From the above rules, we get the condition of the preventing collusion attacks as follows:

$$\begin{cases} \Delta w_0 \leq \Delta w_0 \leq \Delta w_1 \\ \frac{w_0}{n\ 1} \leq w_1 - \Delta w_1 \end{cases} \tag{20}$$

For example, we assume $w_0 =$, $\Delta w_0 = 10$, $w_1 =$ 1 , $\Delta w_1 = 11$, and n = 4. TTP pre-charges the source node 10 \$ and the destination node 11 \$, respectively. After the delivery path is disclosed, the actual payment of the source node is 9 \$, which will be allocated to the message forwarders and the destination node averagely (Each forwarder and the destination node will be awarded 3 \$.). The overcharged $10 -$ $= 1$ \$ will be return to the source node. The payables of the destination is 14 \$ for

TABLE I.
EXAMPLE OF PAYMENT

	Source	Forwarder	Destination
Pre-charge	10	0	11
Actual Payment	9	0	14
Award	0	3	3
Balance	1	3	0

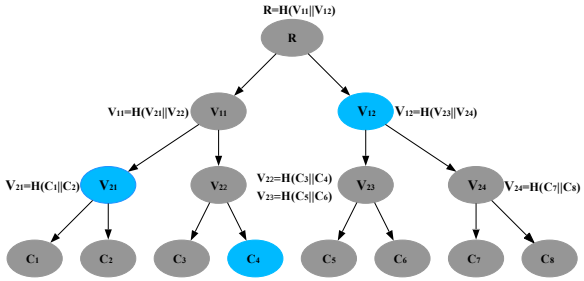

Figure 2. . C_i is the hash value of fragment c_i.

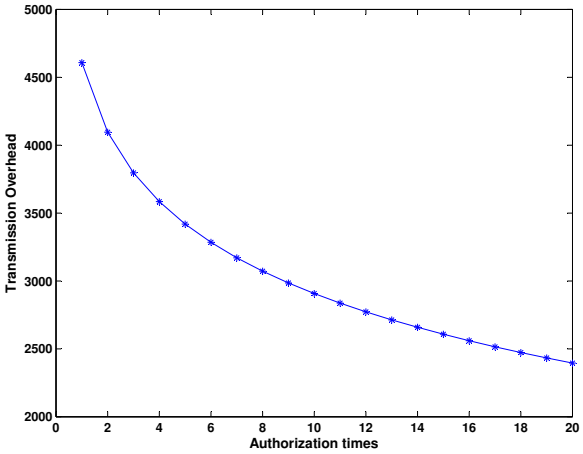

Figure 3. Relationship between the transmission overhead and the authorization times.

receiving its messages. Because it has been pre-charged 11 $ and will be awarded $\frac{9}{3} = 3$ $ for its contribution in path disclosure, the destination node has 11 + 3 = 14 $ holding by the TTP, which just can be used to pay for the receiving message (See TABLE.I).

There are many methods about reward distribution among the relay nodes on a given path. The simplest one is average allocation, such as applied in our scheme. If the holding time and carrying distance are taken into consideration, the problem becomes very challenged and may be left as an open problem.

D. Fragmentation

If the size of the message to be propagated is larger than the average buffer size of DTN nodes, fragmentation will occur at the source side. Every fragment should be authorized by the TTP before being forwarded in the networks. A naive scheme is fragment by fragment authorization. However, this will lead to traffic congestion between the TTP and source nodes. It may be resulted in DDOS attack to TTP in the worst case. Thus, we must try to reduce the connections. In this paper we harness Merkle tree [25] to achieve such a goal.

We assume the size of the cipher text and its every fragment are L and l, respectively. Thus, the cipher text is split into $\lceil \frac{L}{l} \rceil$ fragments, denoted as N. Let $c = c_1||c_2||\cdots||c_N$, where c and c_i are the cipher text and its fragment, respectively. We build our Merkle tree as follows:

1) the value of the leaf nodes $C_i = H(c_i)$, $i = 1, 2, \cdots, N$;
2) the value of the internal nodes $V = H(LeftSon||RightSon)$.

As shown in Fig.2, we built a Merkle tree of height 3. With this binary hash tree, we are able to authorize these 8 fragments by submitting the root, R to TTP for authorization rather than submitting all the hash values of the fragments one by one, which, however, will introduce some transmitting overhead into DTN. For example, in order to verify the authority of c_3, c_3 must be propagated along with C_4, V_{21} and V_{12}. Specifically, we can replace $H_0(c)$ and $L(c)$ with R and $l+\Delta$ in the proposed protocol respectively, where Δ is the size of additional hash values that must be transmitted along with a specific fragment.

Generally, N fragments can generate $\frac{N}{2^h}$ Merckle tree, where h is the height of the tree. The transmitting overhead of a fragment is h hash values. The total transmitting overhead of a Merckle tree is $h * 2^h$. Hence, total transmitting overhead f of N fragments is

$$f = h \cdot 2^h \cdot \frac{N}{2^h} = h \cdot N. \quad (21)$$

On the other side, it needs μ times authorization, where

$$\mu = \frac{N}{2^h}. \quad (22)$$

With equation (21) and (22), we can get relationship of f and μ, see Fig.3.

$$f = h * N = \text{o}_2 \frac{N}{\mu} \cdot N = (log_2 N - log_2 \mu) \cdot N \quad (23)$$

From Fig.3 and equation (23), we can see the transmission overhead increase quickly with the decrease of the number of authorization times. In practice, we can find a balance between the transmission overhead and the number of authorization times in terms of this relational expression (23).

V. SECURITY ANALYSIS

In this section, we analyze the security of the proposed protocol.

Firstly, the messages are encrypted using a symmetric cryptographic algorithm at the source end, $c_0 = E_K(M)$. The encryption key K is encrypted using asymmetric algorithm with the public key ID_n of the destination node, $c_1 = g^{r''_1}$, $c_2 = K \oplus H_3(g_{ID}^{r''_1})$, where $g_{ID} = e(h_1(ID_n), g^{\alpha})$. Therefore, only the destination node of the messages can recover the encrypted messages. Any intermediate node can not get the plain-text content of

the encrypted messages. In addition, only the hash values $h_0(c)$ of the cipher-text are sent to the TTP for authorization. The TTP is blind to what it has signed. Hence, the proposed protocol achieves end-to-end security.

Secondly, the messages are signed by the TTP before they are propagated over the networks. The signatures signed by TTP are aggregated by the source node and the intermediate nodes along the delivery path sequentially. On receiving the messages, intermediate nodes can decide whether to act as a router for the messages in terms of the authority of messages and the remained amount of the payer by once signature verification. Hence, the free riding problems are addressed. In the verification, the order and the distinction of the IDs of the deliverer are checked. If a forwarder injects its own ID more than once in the delivery path by aggregating the same signature repeatedly, it can be detected in the verification. In addition, unless the aggregated signature is cracked, the forwarder can not remove an existing node from the delivery path. Therefore, the authenticity of the delivery path can be hold. No one can forge a path by inserting or removing a node into the legitimate delivery path. Thus, the path forging attacks are thwarted.

Thirdly, the TTP pre-charged the source and the destination some extra money according to the conditions in inequality given in (20), which makes the source node and the destination node have no interest to launch collusion attacks. If the message does not arrive the destination before the time set in ttl, there will not be any legitimate declaration for reward since any right declaration must be followed a receipt signed by the destination. If the last forwarder collude with the source or the destination, the other peer will be deducted the pre-charged extra money. Else if the source, the destination and the last forwarder collude together, the current specific delivery path will be abandoned. However, the destination is blind to the order number of the coming message when it signs the receipt which is encrypted by the last forwarder with a random key. Hence, there will be subsequent last forwarders that can receive the receipts from the destination and disclose the delivery paths along which the same message is propagated. Therefore, the collusion attacks can also be defeated.

In summary, the proposed protocol is a end-to-end secure and can prevent three kinds of attacks, the including free riding attacks, the path forging attacks and the collusion attacks.

VI. Performance Evaluation

In this section, we evaluate our protocol and analyze how the cipher computation duration affects the transmission performance in terms of delivery ratio. To do this evaluation, we modified TheOne [26] simulator by integrating cipher computation duration into it. Let's suppose that the cipher computation duration of the protocol is t_c, message is received at time t_r, and other DTN nodes are encountered at time t_e. The algorithm for this cipher computation is as algorithm 1.

TABLE II.
Pairings and Exponentiation in the Proposed Scheme

Pairing or Exponentiation	Signing	Verification
pairing	0	5
exponentiation of fix point	5	0
exponentiation of random point	3	3

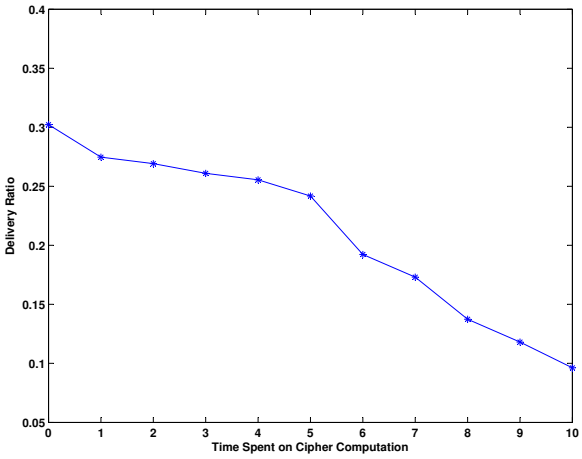

Figure 4. Impact of Cipher Computation.

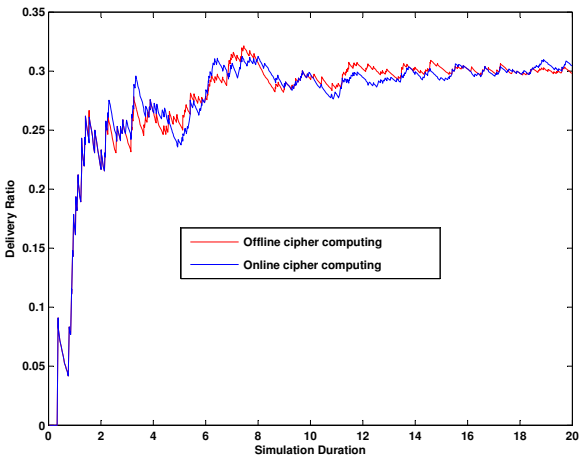

Figure 5. Delivery Ratio.

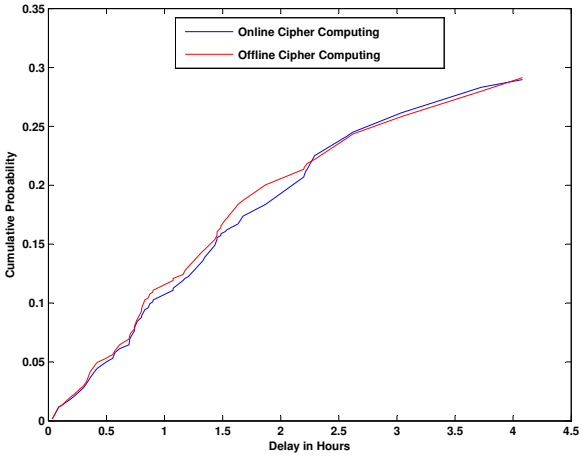

Figure 6. Cumulative Probability of Message Delay.

Algorithm 1 Cipher Computation Overhead Processing

Require: t_c, t_r, t_e

Ensure: Online cipher computation duration t_o

1: t_r = the time at which message was recieved
2: t_e = the time at which other DTN nodes is encountered
3: **if** $(t_c > (t_e - t_r))$ **then**
4: $\quad t_o = t_c - (t_e - t_r)$
5: **else**
6: $\quad t_o = 0$
7: **end if**

From algorithm 1, we can see that online cipher computation duration t_o is decided by how long the DTN node free driving is, during which there is not any suitable node is met to which the holding messages can be forwarded. In the scenario that DTN nodes are sparse, connections are build up when one node falls the communication area of others occasionally. In this case, DTN nodes are free driving without connections with others in most of the time. The free time duration is very suitable for cipher computing including authentication verification and construction of authentication bundles. If a design has the metric by which the authentication and new construction of authentication bundles can be done without the knowledge of the next neighbors, all the cipher related computation can be executed on the free driving time. Otherwise, cipher computation has to be done after connections have been built up. But, this will heavily affect the performance of the message delivery ratio. As message transmitting begins only after the cipher related operations are finished. Let v denote the message transmission speed, $v * t_c$ is amount of data that is delayed in every connection by cipher computation online. Therefore, the parameter t_c heavily affects the message delivery ratio in DTNs. The impact of cipher computation time is illustrated in Fig. 4. From the figure, it can be seen that the delivery ratio decreases with the computation time increasing. Considering t_c will increase linearly with the number of concurrent connections, the message delivery ratio can be heavily affected if concurrency connection occurs frequently. Thus, when a given DTN node encounters several nodes at the same time, the cipher computation time becomes very sensitive to the delivery ratio because the connection time may be very limited and the nodes must exchange messages as soon as possible.

In our scheme, all the computation can be done in the duration in which the DTN nodes drive freely without any connections, since the delivery path can be recorded without the knowledge about the next neighbors. The nodes can using the precious connection time to exchange the prior enveloped messages without extra computation. Online computation occurs only if $t_c > t_e - t_r$ and the online computing time is $t_c - (t_e - t_r)$ rather than t_c. By the algorithm 1, we run our protocol in the modified TheOne simulator. Main parameters are listed in the table III. The simulation results in terms of delivery ratio and delay cumulative probability are shown in Fig.5 and Fig.6, respectively.

TABLE III.
SIMULATION PARAMETERS

Parameters	Values
movementModel	ShortestPathMapBasedMovement
router	EpidemicRouter
btInterface.transmitSpeed	250 Kilobyte
btInterface.transmitRange	10 Meters
High transmitSpeed	10 Megabytes
High transmitRange	100 Meters
bufferSize	5 Megabytes
nrofHosts	50
t_c	0.3 Seconds

According to [27], a Tate Pairing of characteristic two can be computed in 32.5ms if the large prime order $l = 2^{241} - 2^{121} + 1$ on field $\mathbb{F}_{2^{241}}$. A exponentiation of a fixed point and a random point can be achieved in 7.79ms and 26.93ms [28], respectively. The number of pairing and exponentiation is concluded in table II. The main computation time is spent on the calculation of pairing and point exponentiation. From the table II, we can estimate that the main computation time t_c is $32.5 * 5 + 26.\ 3 * 3 +\ .\ \ * 5 \approx 0.3s$.

As shown in Fig.5, the whole delivery ratio comes to be stable after two hours. The delivery ratio is about 30% in the stable state. It can also be seen from the chart that the delivery ratio of the proposed protocol, which has an offline cipher computing, is slightly higher than the protocol with an online cipher computation. Furthermore, if concurrent connections are considered, the greater the number of the concurrent connections is, the more time spent on cipher computation appears. As illustrated in Fig.4 the delivery ratio of the online computing protocol is decline quickly with the increase of the time spent on online computation. Hence, the delivery ratio of the online computing protocol will decrease with the increase of the number of concurrent connections. Thus, the metric of the proposed protocol with offline computing is more obvious.

Fig.6 shows the cumulative probability of message delay in the simulations. It can be seen from the figure that the cumulative probability of offline computation is higher than the cumulative probability of online computation while the delay time is lower than 2.5h. But while the delay time is higher than 2.5h, the cumulative probability of offline computation tends to be lower than the cumulative probability of protocol with online computation. This shows that the average message delay of the protocol with offline computation is smaller than that of the protocol with online computation.

VII. CONCLUSIONS

In this paper, we propose a secure message transaction protocol with an incentive payment mechanism for message transmission in delay tolerant networks. The protocol is end-to-end secure, incentive compatible and off-line computed. Simulation results show that this design increases the delivery ratio and delivery speed. In addition,

three kinds of attacks (the free riding attacks, path forging attacks and the collusion attacks) are prevented.

Moreover, we propose the authorization methods based on authentication hash tree which can decrease the number of communications over the LRNB channels when fragmentation is revoked. The relationship between the delivery overhead and the number of authorization times, proposed in this paper, can be used to make the tradeoff between the delivery overhead and the number of communications over the LRNB channels.

Further more, the messages are encrypted and authorized only once at the source end. In the messages propagation, it does not need to contact to the TTP. Therefore the proposed protocol can be deployed in the network environments in which the BSs or APs are very sparse.

ACKNOWLEDGMENT

The authors are grateful to the anonymous referees for their valuable comments and suggestions to improve the presentation of this paper.

REFERENCES

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *SIGCOMM, 2003 proceedings ACM*, 2003, pp. 27–34.

[2] V. Erramilli, A. Chaintreau, M. Crovella, and C. Diot, "Diversity of forwarding paths in pocket switched networks," in *SIGCOMM, 2007 proceedings ACM*, 2007, pp. 161–174.

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

[4] K. Fall and S. Farrell, "Dtn: an architectural retrospective," *Selected Areas in Communications, IEEE Journal on*, vol. 26, no. 5, pp. 828–836, 2008.

[5] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," *Computer Communication Review*, vol. 34, no. 4, pp. 145–157, 2004, aCM/SIGCOMM 2004 Conference on Computer Communications Aug 30-sep 03, 2004 Portland, OR.

[6] X. Lin, R. Lu, C. Zhang, H. Zhu, P. Ho, and X. Shen, "Security in vehicular ad hoc networks," *Communications Magazine, IEEE*, vol. 46, no. 4, pp. 88–95, 2008.

[7] H. Zhu, X. Lin, R. Lu, Y. Fan, and X. Shen, "Smart: A secure multilayer credit-based incentive scheme for delay-tolerant networks," *Vehicular Technology, IEEE Transactions on*, vol. 58, no. 8, pp. 4628–4639, 2009.

[8] Y. Li, P. Hui, D. Jin, L. Su, and L. Zeng, "Evaluating the impact of social selfishness on the epidemic routing in delay tolerant networks," *Communications Letters, IEEE*, vol. 14, no. 11, pp. 1026–1028, 2010.

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

[10] X. Lin, R. Lu, X. Liang, and X. Shen, "Stap: A social-tier-assisted packet forwarding protocol for achieving receiver-location privacy preservation in vanets," in *INFOCOM, 2011 Proceedings IEEE*, 2011, pp. 2147–2155.

[11] M. Karaliopoulos, "Assessing the vulnerability of dtn data relaying schemes to node selfishness," *Communications Letters, IEEE*, vol. 13, no. 12, pp. 923–925, 2009.

[12] Y. Li, G. Su, D. O. Wu, D. Jin, L. Su, and L. Zeng, "The impact of node selfishness on multicasting in delay tolerant networks," *Vehicular Technology, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2011.

[13] I.-R. Chen, F. Bao, M. Chang, and J.-H. Cho, "Trust management for encounter-based routing in delay tolerant networks," in *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*, 2010, pp. 1–6.

[14] D. Djamel and B. Nadjib, "On eliminating packet droppers in manet: A modular solution," *Ad Hoc Networks*, vol. 7, no. 6, pp. 1243–1258, 2009.

[15] L. Yin, H. mei Lu, Y. da Cao, and J. min Gao, "Cooperation in delay tolerant networks," in *Signal Processing Systems (ICSPS), 2010 2nd International Conference on*, vol. 1, 2010, pp. 202–205.

[16] B. Levente, D. Laszlo, F. Mark, and V. Istvan, "Barter-based cooperation in delay-tolerant personal wireless networks," in *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a*, 2007, pp. 1–6.

[17] B. Levente, D. Lszl, F. Mrk, and V. Istvn, "Barter trade improves message delivery in opportunistic networks," *Ad Hoc Networks*, vol. 8, no. 1, pp. 1–14, 2010.

[18] U. Shevade, S. Han Hee, Q. Lili, and Z. Yin, "Incentive-aware routing in dtns," in *Network Protocols, 2008. ICNP 2008. IEEE International Conference on*, 2008, pp. 238–247.

[19] K. Srinivasan, S. Rajkumar, and P. Ramanathan, "Incentive schemes for data collaboration in disruption tolerant networks," in *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*, 2010, pp. 1–5.

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

[21] Q. Li, S. Zhu, and G. Cao, "Routing in socially selfish delay tolerant networks," in *INFOCOM, 2010 Proceedings IEEE*, 2010, pp. 1–9.

[22] A. Mei and J. Stefa, "Give2get: Forwarding in social mobile wireless networks of selfish individuals," in *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, 2010, pp. 488–497.

[23] B. Alexandra, G. Craig, O. Adam, and Y. D. Hyun, "Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing," in *Proceedings of the 14th ACM conference on Computer and communications security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 276–285. [Online]. Available: http://doi.acm.org/10.1145/1315245.1315280

[24] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," *Siam Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[25] R. C. Merkle, "Protocols for public key cryptosystems," *Security and Privacy, IEEE Symposium on*, pp. 122–133, 1980.

[26] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA, 2009, pp. 1–10.

[27] G. Steven, H. Keith, and S. David, "Implementing the tate pairing," in *Algorithmic Number Theory*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2002, vol. 2369, pp. 69–86.

[28] M. Atsuko, O. Takatoshi, and C. Henri, "Efficient elliptic curve exponentiation," in *Information and Communications Security*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 1997, vol. 1334, pp. 282–290.

Zhongtian Jia received his B.S. degree in applied mathematics from Qingdao University, China in June 1996 and his M.S. degree in computer science from Shandong University, China in June 2005. He is currently working towards his Ph.D. degree in Cryptography at Beijing University of Posts and Telecommunications, China. His current research interest includes wireless network security and applied cryptography.

Lixiang Li received the B.S. and M.S. degree in electronic information technology and Circuit and System from Yanshan University, Qinhuangdao, China, in 2000 and 2003, respectively, and Ph.D. degree in signal & information processing, Beijing University of Posts and Telecommunications, Beijing, China in 2006. She is an associate professor at the School of Computer Science and Technology, Beijing University of Posts and Telecommunications, China. Her research interests include swarm intelligence, synchronization and parameter estimation of dynamical systems and information security.

Zhuoran Yu is currently a college student and working towards his B.S. degree at Beijing University of Posts and Telecommunications, China. His current research interest includes wireless network security and applied cryptography.

Shudong Li received his B.S. degree in applied mathematics from Yantai Normal College, China in June 2002 and his M.S. degree in applied mathematics from Tongji University, China in June 2005. He is currently working towards his Ph.D. degree in Cryptography at Beijing University of Posts and Telecommunications, China. His current research interest includes wireless network security and network science.

Yixian Yang received his Ph.D. degree in signal and information processing from Beijing University of Posts and Telecommunications, Beijing, China in 1988. He is currently a Changjiang Scholar and Distinguished Professor with Beijing University of Posts and Telecommunications, Beijing, China. He is also the director of National Engineering Laboratory for Disaster Backup and Recovery, Key Laboratory of Network and Information Attack & Defence technology of MOE, and Information Security Center, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. His research interests include network and information security theory and applications, network-based computer application technology, Coding Theory and Technology.