Wireless Netw (2007) 13:569–582
DOI 10.1007/s11276-006-6220-3

A secure incentive protocol for mobile ad hoc networks

Yanchao Zhang · Wenjing Lou · Wei Liu · Yuguang Fang

Published online: 8 May 2006

Abstract The proper functioning of mobile ad hoc networks depends on the hypothesis that each individual node is ready to forward packets for others. This common assumption, however, might be undermined by the existence of selfish users who are reluctant to act as packet relays in order to save their own resources. Such non-cooperative behavior would cause the sharp degradation of network throughput. To address this problem, we propose a credit-based Secure Incentive Protocol (SIP) to stimulate cooperation among mobile nodes with individual interests. SIP can be implemented in a fully distributed way and does not require any pre-deployed infrastructure. In addition, SIP is immune to a wide range of attacks and is of low communication overhead by using a Bloom filter. Detailed simulation studies have confirmed the efficacy and efficiency of SIP.

Keywords Mobile ad hoc networks · Selfishness · Incentive · Cooperation · Security

This work was supported in part by the U.S. Office of Naval Research under Young Investigator Award N000140210464 and under grant N000140210554.

Y. Zhang · W. Liu · Y. Fang, Ph.D. (✉) ·
Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611
e-mail: yczhang@ufl.edu

W. Lou, Ph.D.
Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA 01609
e-mail: wjlou@ece.wpi.edu

Y. Fang
e-mail: fang@ece.ufl.edu

1. Introduction

Mobile ad hoc networks (MANETs) are finding ever-increasing applications in both military and civilian scenarios due to their self-organizing, self-configuring capabilities. The proper functioning of a MANET depends on the common hypothesis that network nodes are willing to forward others' packets to enable otherwise impossible multi-hop communications. This assumption may be valid in emergency and military situations in which all the nodes belong to a single authority and are naturally motivated to cooperate. However, it might not hold in general civilian applications because of the possible presence of selfish users, which are reluctant to act as packet relays to save their own resources such as battery life, CPU cycles, or available bandwidth. Such non-cooperative behavior would result in the sharp degradation of network throughput, as reported in [1]. It is, therefore, necessary to design effective, efficient mechanisms to stimulate cooperation in packet forwarding among possibly selfish mobile nodes.

In this paper, we propose a Secure Incentive Protocol (SIP) to motivate packet forwarding in totally self-organizing MANETs without relying on any centralized infrastructure. The basic idea of SIP is simple: each node imprints a non-forged "stamp" on each packet forwarded as the proof of forwarding, based on which packet relays are remunerated, while packet sources and destinations are charged with appropriate credits. It is, however, by no means an easy task to implement SIP in a secure, efficient manner. For example, the introduction of credits may serve not only as an incentive for cooperation, but also as a stimulus for cheating. In addition, as an add-on, any incentive scheme like SIP should be efficient and lightweight enough not to disturb other normal network functions such as routing.

Our SIP has a number of nice properties that make it an appealing solution to node selfishness. First of all, SIP does not require any pre-deployed infrastructure and is independent of, but can be well integrated with, any underlying routing protocol. Second, SIP features an efficient pairing-based [2] method to establish various session keys based on node identifiers instead of conventional public-key certificates. Third, SIP can withstand a wide range of cheating actions by using hash functions intelligently. Fourth, SIP employs a space-efficient data structure known as a Bloom filter [3] to greatly reduce the communication overhead. Last, SIP is rather flexible and can well adapt to the dynamically changing nature of MANETs. The effectiveness and efficiency of SIP are justified and validated through extensive simulations.

The rest of this paper is structured as follows. Section 2 introduces the network, node, and payment models as well as the pairing technique. Next, we detail the SIP design in Section 3, followed by some enhancements in Section 4. Then we evaluate the performance of SIP in Section 5, review the related work in Section 6, and end with concluding remarks.

2. System models and the pairing technique

2.1. Network model

We consider a general MANET formed on the fly by a set of mobile devices owned by individual users. For brevity only, we use the term "node" to indicate both a mobile device and its owner when no confusion is caused. Each node i is assumed to have a unique non-zero identifier ID_i, which can be the identifier of its secure module (cf. Section 2.4). We will use node i or ID_i interchangeably hereafter.

We also assume that each node has limited transmission and reception capabilities so that two nodes outside the transmission range of each other can only communicate via a sequence of intermediate nodes in a multi-hop manner. Wireless links are assumed to be bidirectional, which means that if node i can hear another node j's transmission, j can hear i's transmission as well. The extension of SIP to unidirectional link scenarios is left as our future work. We further postulate that nodes may freely roam in the network, but do not continuously move so rapidly as to make the flooding of every data packet the only possible routing protocol. This is a common assumption made about node mobility by almost all MANET routing protocols such as AODV [4] and DSR [5].

2.2. Node model

Mobile nodes under individual control typically have constrained resources such as battery life, CPU cycles, and available network bandwidth. As a result, they are assumed to be *selfish* in nature in the sense that they are reluctant to forward packets destined for other nodes without gain. For instance, they can do this by simply shutting down their devices if not used or setting their wireless network interfaces to be ignorant of others' routing requests through some simple program. It is a natural idea to stimulate cooperation among selfish nodes by rewarding them with some notional credits paid by packet sources and destinations. Our SIP is proposed exactly for this purpose. As everything has two sides, the introduction of credits also provides incentives for various cheating actions. We assume that mobile nodes are *greedy* so that they will try to bypass SIP and cheat for credits, either by paying less or gaining more. In particular, a greedy node may carry out the following cheating actions:

- **Credit fraudulence**: If possible, a greedy node will attempt to be rewarded or reward itself for the work they did not do or more than it has done.
- **Repudiation**: The source or destination may deny previous communications realized through intermediate nodes so as not to pay for them.
- **Node collusion**: Greedy nodes may collude with each other if they can benefit from doing so. For example, two colluding nodes may launch the *free riding* attack by piggybacking data on the packets sent between two well-behaved nodes in order to escape being charged.

However, greedy nodes are further assumed to be *rational*, which means that they only attempt to cheat if the expected benefit of doing so is greater than that of acting honestly. We believe that such definitions as selfishness, greediness, and rationality can well characterize mobile nodes with individual interests. For simplicity, we generally call them *misbehaving* nodes throughout the remainder of this paper.

In addition to misbehaving nodes, there might exist *malicious* nodes whose sole objectives are to interrupt the proper network operations without considering their own gains. Many nice solutions have been proposed to deal with malicious nodes, such as Ariadne [6], SPREAD [7], and MASK [8]. Though important, this issue is beyond the scope of this paper.

2.3. Payment model

In most previous proposals such as [9–11], node remuneration is enabled by charging packet sources and rewarding intermediate nodes. We argue that a more fair payment approach is to charge both packet sources and destinations simply because both of them benefit. The payment proportion between them is adjustable and can be negotiated during the session initialization phase, which will be explained shortly.

As is well known, packet loss may occur in MANETs due to node mobility, collision, channel impairment, or other reasons. Ideally, any node which has ever tried to forward a packet should be rewarded because forwarding a packet will incur a cost to the node, no matter whether the packet eventually reaches its final destination or not. It is, however, difficult to corroborate an intermediate forwarding action in a trustable, distributed manner without involving too complicated design. Considering this situation, we adopt the same rationale as [11] that the source and the destination just need to pay for the packets successfully delivered to the destination. We will present in Section 4.3 a measure to motivate packet forwarding even in the face of frequent packet loss.

The next question is how much to pay well-behaved intermediate nodes. Some solutions like [12] consider that each node has a different forwarding cost and propose to remunerate an intermediate node with a payment corresponding to its incurred cost. Though ideal, this model is difficult to implement in practice without involving a complicated least-forwarding-cost route discovery process and the calculation of en-route individual payments [12], both of which are computationally and communicationally expensive, and are vulnerable to node collusion (as mentioned in [12]). In view of this, we assume that all the nodes have the same charging rate, say λ credits per unit-sized packet, similar to the method used in [9]. Our principle is to make SIP simple and efficient enough against node selfishness, while reducing as much as possible its negative impact on normal network functions.

Motivated by real life scenarios, we also have two payment models: the "debit-card" and "credit-card" models. With the former, only when a node has enough credits in hand could it transmit its own packets. By contrast, the latter allows a node to transmit a certain number of extra packets when it has not enough credits, as long as it agrees to pay more or some "interest" for them later (cf. Section 4.1).

2.4. Tamper-proof secure module

Any payment-based approach demands some sort of tamper-proofness indispensable for guaranteeing the security of the payment process. This requirement was previously satisfied either by a centralized authority [10, 11], or by a (relatively) tamper-proof secure module in each individual node [9]. Since the use of a centralized authority would undermine the self-organizing, decentralized nature of MANETs, we opt for the latter method. We assume that each node i has a relatively tamper-proof secure module, denoted by SM_i. User i might be able to manipulate other sub-units of its mobile device, e.g., intercepting and modifying both the input and output of the wireless network interface, but never to access and modify the contents of SM_i. In practice, SM_i can be part of the medium access control (MAC) hardware or an independent smart card such as the SIM cards in GSM phones. Each SM_i has a unique identifier ID_i (similar to the MAC address) that can be used to uniquely identify and address node i. Notice that, besides stimulating packet forwarding, SM_i might as well be used for other important purposes like billing in future commercialized civilian MANETs.

One may argue that it might be difficult to realize fully tamper-proof secure modules [13], but manufacturers are steadily improving the secure modules[1]. It might be true that no module can be designed to withstand adversaries who can invest millions and hire the right specialists. This is, however, normally beyond the capabilities of selfish yet rational nodes we are dealing with. Similar to GSM users in daily life, such nodes would not struggle to modify or subvert their own secure modules because doing so might cause them to loss much more than they could gain. For this reason, we believe that the assumption about tamper-proof secure modules is very reasonable.

2.5. Pairing technique

In this paper, we use ID-based cryptography (IBC) to secure the charging and rewarding process. First introduced by Shamir in 1984 [14], IBC allows public keys of entities to be directly derived from their known identifiers, thus eliminating the need for conventional public-key certificates. This inbred feature makes IBC more suitable for the resource-constrained wireless arena [15], because nodes no longer need to expend scarce network bandwidth and computational resources in exchanging and verifying certificates. However, only recently has the rapid development of IBC taken place due to the application of the following pairing technique.

Let $\mathbb{G}_1, \mathbb{G}_2$ be two groups of the same prime order q. We view $\mathbb{G}_1$ as an additive group and $\mathbb{G}_2$ as a multiplicative group throughout the paper. Assume that the discrete logarithm problem (DLP) is hard[2] in both $\mathbb{G}_1$ and $\mathbb{G}_2$. For us, a pairing is a computable bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ such that, [3]for all $P, Q, R, S \in \mathbb{G}_1$,

$$\hat{e}(P+Q, R+S) = \hat{e}(P,R)\hat{e}(P,S)\hat{e}(Q,R)\hat{e}(Q,S). \qquad (1)$$

Modified Weil [2] and Tate pairings [16] on supersingular elliptic curves are examples of such bilinear maps, for which

[1] IBM has challenged the argument in [13] by developing the 4758 secure cryptoprocessor, which includes defenses against numerous mechanical, chemical, electrical, and radiological attacks, and is believed to be impenetrable to adversaries with limited time or resources.

[2] It is computationally infeasible to extract the integer $x \in \mathbb{Z}_q^* = \{i \mid 1 \le i \le q-1\}$, given $P, Q \in \mathbb{G}_1$ (respectively, $P, Q \in \mathbb{G}_2$) such that $Q = xP$ (respectively, $Q = P^x$).

[3] In particular, $\forall\, P, Q \in \mathbb{G}_1,\ \forall\, a, b \in \mathbb{Z}_q^*,\ \hat{e}(aP, bQ) = \hat{e}(aP, Q)^b = \hat{e}(P, bQ)^a = \hat{e}(P, Q)^{ab}$ etc.

the *Bilinear Diffie-Hellman Problem* (BDHP) is believed to be hard, i.e., it is believed that, given $< P, xP, yP, zP >$ for $P \in \mathbb{G}_1$ and random $x, y, z \in \mathbb{Z}_q^*$, there is no algorithm running in expected polynomial time, which can compute $\hat{e}(P, P)^{xyz} \in \mathbb{G}_2$ with non-negligible probability. We refer to [2, 16] for a more comprehensive description of how the pairing parameters should be chosen in practice for both efficiency and security.

3. Secure incentive protocol design

In this section, we elaborate the SIP design and show how to employ a Bloom filter to reduce the communicational overhead of SIP.

3.1. Pre-shipment

To facilitate the presentation, we assume that all the secure modules are produced by the same manufacturer. However, it should be noted that SIP can be easily extended to the multi-manufacturer case. Let $\vartheta \in \mathbb{Z}^+$ indicate a given security parameter determining the size of prime q and $\mathcal{BG}$ represent some BDH parameter generator [2] with ϑ as input. The manufacturer executes the following initialization algorithm:

1. Run $\mathcal{BG}$ on input ϑ to generate the pairing parameters $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e})$.
2. Select a random $g \in \mathbb{Z}_q^*$ as the manufacturer *master-key*.
3. Choose a cryptographic hash function H_1, mapping arbitrary strings to non-zero elements in $\mathbb{G}_1$.
4. Calculate a private key $sk_i = gH_1(ID_i) \in \mathbb{G}_1$ for each secure module SM_i with identifier ID_i.

Each SM_i is preloaded with the parameter vector $<q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, H_1, ID_i, sk_i>$ along with other information if needed. Since the DLP is difficult in $\mathbb{G}_1$, the master-key g cannot be deduced from any given (ID_i, sk_i) pair with non-negligible probability.

SIP uses notional credits as incentives to stimulate packet forwarding. For this purpose, each secure module has an inner credit counter (CC), pre-charged with a certain amount of credits before shipped out. The charging and rewarding on a node is done by decreasing or increasing the CC in its secure module. Note that the CC will retain its value even when a node is power-off, and therefore that node could still use the accumulated credits while it is power-on again.

3.2. SIP overview

SIP is implemented in the secure module of each node. We require that, whenever initiating or forwarding a packet, a node first pass the packet to its secure module for SIP processing. Although a misbehaving node might directly send a packet by manipulating its wireless network interface, the packet will not carry correct SIP information and thus will be dropped by other well-behaved nodes.

SIP takes a source-controlled session-based approach, which consists of four phases. During the first *Session Initialization* phase, the source negotiates session traffic information with the destination and intermediate nodes en route. Subsequently, various session keys for securing SIP operations are established during the *Session-Key Establishment* phase. In the next *Packet Forwarding* phase, each intermediate node imprints a non-forged *stamp* on each data packet forwarded, and the source and the destination collect the stamps for later rewarding actions. The final phase is the *Rewarding* phase in which each intermediate node is awarded a certain number of credits commensurate with the service they provided to the source and the destination. In what follows, we will dwell on the operations of each phase one by one.

3.3. Session initialization

For lack of space, we are only concerned with unicast routing in this paper and will report the extension of SIP to multicast routing in a separate paper. When the source (ID_s) intends to communicate with the destination (ID_d), it first needs to establish an end-to-end session with the destination. The purpose is to inform the destination and intermediate nodes en route about the subsequent traffic. Obviously, the session initialization process can be well integrated with the route discovery process of any MANET on-demand routing protocol. For clarity only, we assume that, before the session initialization phase, an end-to-end path has been discovered between the source and the destination with the underlying routing protocol. It is possible that misbehaving nodes attempt to manipulate the route discovery process for various motives, e.g., earning more credits by creating a path longer than normal. To prevent this, we assume the use of secure routing protocols such as Ariadne [6] or ARAN [17].

To set up an end-to-end session, the source unicasts a *session request* of format <*type* = SIP-REQ, ID_s, SN_s, ID_d, *node-list*, *trafficInfo*> over the found path. Here SN_s is a sequence number locally maintained by the source and, along with ID_s, can uniquely identify a session initiated by the source. The *trafficInfo* field contains subsequent session traffic information, such as the expected traffic amount and session duration, the packet arrival rate, the rewarding frequency, and the payment splitting method between the source and the destination. The *node-list* field is initially empty, to which each intermediate node will append its identifier. It is used to notify the source of the identifiers of intermediate nodes. If the underlying routing protocol like Ariadne [6] can provide the source such information, this field is not necessary.

Each intermediate node, once receiving a session request, has a chance to choose not to serve this session based on its current resource availability such as residual energy and the session traffic information described in the session request. It can do so by not propagating the session request without bearing any punishment. Otherwise, if agreeing to serve this session, it appends its identifier to the end of the *node-list* and forwards the session request to the next hop. This process repeats until the session request finally reaches the destination. Notice that this agreement process might not involve the real participation of mobile users. Instead, it can be easily automated by each individual user setting some default parameters. If some intermediate node chooses not to serve this session or a path breaks due to node movement or other reasons, the source has to re-initialize the session through another path to the destination. If secure single-path routing protocols such as Ariadne [6] and ARAN [17] are used, the session re-initialization process involves a new route discovery process; if secure multipath routing protocols like SecMR [18] are used instead, the session re-initialization can be performed via any available redundant path.

Upon receiving a session request and if agreeing to communicate with the source according to *trafficInfo*, the destination unicasts back to the source a session response including *node-list*. Without loss of generality, we assume that *node-list* consists of N intermediate nodes with identifiers ID_i ($1 \le i \le N$) sequentially arranged from the source to the destination.

3.4. Session-key establishment

With the aforementioned pairing technique, the source and each intermediate node can establish a shared master key by just knowing the identifier of each other. For example, the source and an intermediate node i can calculate the shared master key as

$$\begin{aligned}\Gamma_{s,i} &= \hat{e}(sk_s, H_1(ID_i)) \\ &= \hat{e}(gH_1(ID_s), H_1(ID_i)) \\ &= \hat{e}(H_1(ID_s), gH_1(ID_i)) \quad \text{(due to Eq. 1)} \\ &= \hat{e}(H_1(ID_s), sk_i).\end{aligned} \tag{2}$$

Here the source (respectively, node i) derives $\Gamma_{s,i}$ using the first-line equation (respectively, the fourth-line equation). Due to the hardness of the *BDHP*, $\Gamma_{s,i}$ is exclusively available to the source and node i. Then they can calculate a session key $K_{s,i} = h(\Gamma_{s,i}||ID_s||SN_s)$, where $||$ denotes message concatenation and h indicates a fast hash function such as SHA-1 [19] preloaded to each node. In fact, the source and node i can further derive various session keys from $K_{s,i}$ for different security purposes. As an example, they can use $h(K_{s,i}||1)$ for message encryption, while $h(K_{s,i}||2)$ for message authentication. To simplify our presentation, however, henceforth we simply say that a packet is encrypted and authenticated with key $K_{s,i}$, though the packet is actually encrypted with $h(K_{s,i}||1)$ and authenticated with $h(K_{s,i}||2)$, respectively.

Similarly, the source and the destination can establish a shared master key $\Gamma_{s,d} = \hat{e}(H_1(ID_s), H_1(ID_d))^g$ and a session key $K_{s,d} = h(\Gamma_{s,d}||ID_s||SN_s)$. In addition, each pair of adjacent intermediate nodes (ID_i, ID_{i+1}) are required to establish a shared master key $\Gamma_{i,i+1} = \hat{e}(H_1(ID_i), H_1(ID_{i+1}))^g$ and a shared session key $K_{i,i+1} = h(\Gamma_{i,i+1}||ID_s||SN_s)$. Furthermore, the destination and node ID_N should derive a shared master key $\Gamma_{N,d} = \hat{e}(H_1(ID_N), H_1(ID_d))^g$ and a shared session key $K_{N,d} = h(\Gamma_{N,d}||ID_s||SN_s)$. The uses of such session keys will be explained later.

The next step is for the source to unicast a KAGREE packet, including $< \{K_{comm}\}_{K_{s,1}}, \ldots, \{K_{comm}\}_{K_{s,N}}, \{K_{comm}\}_{K_{s,d}} >$, along the path to the destination. Here K_{comm} is a random common key chosen by the source and $\{M\}_K$ means encrypting message M with a symmetric key K. Upon receiving the KAGREE packet, each intermediate node and the destination can decrypt their respective portion to get K_{comm} for later use. To reduce the communication overhead, KAGREE can be sent as part of the first data packet.

It is worth pointing out that all the shared master/session keys and K_{comm} are calculated and stored in tamper-proof secure modules. The nodes themselves have no knowledge about these keys at all. In addition, two nodes establishing a shared session key do not need to prove to each other the knowledge of the shared key, as any future message encrypted and/or authenticated with the shared key can implicitly achieve the same effect.

The above process requires each node to perform one relatively costly pairing operation for deriving a shared master key. The computational overhead can be reduced by letting each node cache the recently calculated shared master keys at the cost of slightly increased space overhead. Later on, if needing to establish shared session keys with a recently encountered node (possibly for different source-destination pairs), a node can avoid re-evaluating the pairing and so does the other node.

Also, note that traditional shared key establishment based on certificate-based cryptography (e.g., RSA [20]) inevitably involves the exchange and verification of public-key certificates. It is, therefore, both communicationally and computationally inefficient as compared to our IBC-based approach. In this sense, in addition to stimulating packet forwarding, we indeed provide an efficient method to establish shared keys used for other security purposes such as secure routing in MANETs. The great potential of our approach in securing MANETs remains to be further explored.

3.5. Packet forwarding

Once shared session keys are established, the source and the destination can begin normal communications. For simplicity, we assume that all the data packets are unit-sized, though SIP can be easily extended to the case with various packet sizes.

A SIP data packet is of format <*type* = SIP-DATA, ID_s, ID_d, SN_s, PN_s, *payload*, *AUTH*, *STAMP*>, essentially a normal data packet supplemented with a few SIP fields. PN_s denotes a non-decreasing session-related packet sequence number set by the source. Depending on different application requirements, the payload part may be optionally encrypted and/or authenticated with the shared key between the source and the destination. The *AUTH* field is set to h(packet fields before $AUTH||K_{comm}$) by the source, and the *STAMP* field is initialized as $STAMP_0 = h(ID_s \parallel SN_s \parallel PN_s)$. The former will remain unchanged during transmission, while the latter will change at each intermediate node.

When receiving a SIP data packet, each intermediate node, say node *i*, passes the packet to its secure module, which, in turn, does the following operations:

1. Make sure that PN_s is larger than the last packet sequence number it recorded for this session, and otherwise abort the processing and drop the packet.
2. Compute *AUTH*'$= h$(packet fields before $AUTH \parallel K_{comm}$) and, if *AUTH*' is not equal to *AUTH*, abort the processing and dump the packet.
3. Calculate $STAMP_i = h(STAMP_{i-1} \parallel K_{s,i})$.
4. Change the *STAMP* field to $STAMP_i$ and output the packet to be forwarded to the next hop.

The *AUTH* field is introduced to defend against the *free riding* attack in which two misbehaving nodes on the forwarding path attempt to exchange packets without paying for them. Obviously, this attack is only beneficial for two colluding nodes when there is at least one well-behaved node residing between them. To perform this attack, a misbehaving node piggybacks some data on the output of its secure module, essentially a to-be-forwarded packet with correct SIP fields. Subsequently, any downstream conspirator serving the same session can simply extract the piggybacked data before passing the packet to its own secure module for SIP processing. The second operation above is designed to deal with this free riding attack. Since K_{comm} is stored and the calculation and verification of *AUTH* fields are done in tamper-proof secure modules, a free riding packet will not have the correct *AUTH* field and thus will be detected and dropped by the first encountered well-behaved node. Therefore, the free riding attack is effectively defeated with computationally efficient per-hop hash operations. In addition, the updated *STAMP* field value $STAMP_i$ can serve as the proof of node i's forwarding action, which will be discussed shortly.

When the destination receives the packet, its secure module forms a final stamp as $STAMP_d = h(STAMP_N \parallel K_{s,d})$ and then saves $<STAMP_d, PN_s>$, termed a *receipt* hereafter, into an internal data structure called a RECEIPT table. After accumulating a certain number of receipts, the destination sends them to the source in a RECEIPT packet. Note that the RECEIPT packet can piggyback onto any normal packet (if any) from the destination to the source so as to reduce the communication overhead.

Upon receipt of a RECEIPT packet, the source's secure module can verify that each intermediate node indeed forwarded certain packets by re-computing the final stamps via a sequence of keyed hash operations and then checking their equality to the received ones (cf. Section 3.6). This is possible because the source's secure module have all the keys of intermediate nodes and the destination that were used to generate the stamps. It also explains the reason that the source needs to establish shared session keys with all the intermediate nodes and the destination during session initialization. The secure module then increases an internal REWARD counter, which records the number of packets for which intermediate nodes have not been remunerated yet, by the number of packets passing the verification.

3.6. Reducing the RECEIPT packet size using Bloom filters

Suppose the destination initiates a RECEIPT packet every β packets. Let each stamp be the first u bits of the 160-bit SHA-1 [19] output and PN_s be a v-bit integer. Without considering the normal packet header, the net payload of a RECEIPT packet is $\beta * (u + v)$ bits, which might be a very large number in some cases. For instance, if $\beta = 10$, $u = 64$, and $v = 16$, each RECEIPT packet is at least 800 bits, which would result in the unfavorable increase of communication overhead and transmission energy consumption. Fortunately, we can greatly alleviate this negative effect by significantly condensing the RECEIPT packet with a Bloom filter [3].

3.6.1. Introduction to Bloom filters

A Bloom filter is a well-known space-efficient data structure for representing a set $\mathcal{S} = \{s_1, s_2, ..., s_\beta\}$ of β elements to support membership checking. The basic idea is to select k independent hash functions $h_1, h_2, ..., h_k$, each with range $\{0, ..., w - 1\}$. For each element $s \in \mathcal{S}$, the bits at positions $h_1(s), h_2(s), ..., h_k(s)$ of a w-bit array BF, which

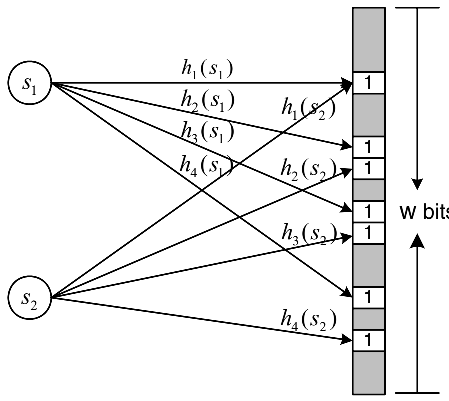


Fig. 1 A Bloom filter with four hash functions

is initialized to all zeros, are set to one. For an element b in question, the bits at positions $h_1(b), h_2(b), ..., h_k(b)$ are checked. If any of them is zero, then b is certainly not in $\mathcal{S}$. Otherwise it is high likely that b belongs to $\mathcal{S}$. For clarity, a Bloom filter with $k = 4$ is shown in Fig. 1, where one may notice that a particular bit might be set multiple times. The Bloom filter may yield a *false positive*, that is, an element is in fact not in $\mathcal{S}$ but all its corresponding bits have been set. Assuming that we have mapped β elements into the array BF and the output of each of k hash functions is uniformly distributed within $\{0, ..., w-1\}$, the probability that a particular bit is not set is exactly $(1 - 1/w)^{k\beta}$. It follows that the probability of a false positive is $(1 - (1 - 1/w)^{k\beta})^k \approx (1 - e^{-k\beta/w})^k$, which is minimized for $k = \ln 2 \times w/\beta$. Since k should be an integer, we might choose a value less than the optimal to strike a good balance between an acceptable false positive probability and computational overhead.

3.6.2. SIP with Bloom filters

To use Bloom filters, the source and the destination should negotiate the values of k, w, and β during session initialization. In particular, w should be the xth power of 2 for an integer x usually much larger than 1. The hash functions are built by first hashing a packet using SHA-1 [19], and then taking the first k groups of $\log_2(w) = x$ bits from the 160-bit output, each corresponding to one hash function. In the following, we assume k to be 4 for ease of presentation.

After accumulating β receipts, the destination's secure module applies the hash functions h_1, h_2, h_3, and h_4 (in fact just one SHA-1 hash operation) to each final stamp $STAMP_d$ and then marks the corresponding bits of the w-bit array. Eventually, it sends to the source a RECEIPT packet consisting of the w-bit array and the largest v-bit packet sequence number among the receipts, denoted by θ. The source's secure module should keep a record θ_l indicating the sequence number of the last verified packet. When receiving a RECEIPT packet, the secure module of the source performs the following operations:

1. Check that θ is greater than θ_l and stop processing the packet otherwise.
2. For each $t \in [\theta_l + 1, \theta]$, first calculate $a_0 = h(ID_s \parallel SN_s \parallel t)$ and then recursively compute $a_{i+1} = h(a_i \parallel K_{s,i+1})$ until getting $a_d = h(a_N \parallel K_{s,d})$.
3. Check the bits at positions $h_1(a_d)$, $h_2(a_d)$, $h_3(a_d)$, and $h_4(a_d)$ of the received w-bit array and increase the REWARD counter by one if the four bits have all been set.
4. Set θ_l to θ.

It is clear that using a Bloom filter can greatly reduce the RECEIPT packet size and thus communication overhead and transmission energy consumption, at the slight risk of some false positives. For example, if $w = 256$, $\beta = 20$, $v = 16$, and $u = 64$, the original RECEIPT packet size is 1600 bits, while the new one is only 272 bits, leading to 83 percent saving. In this case, the false positive probability is around 0.5 %, which is believed to be acceptable.

3.7. Charging and rewarding

As mentioned earlier, SIP stimulates packet forwarding by rewarding each intermediate node for the service they provided, while charging the source and the destination for the service they received. For simplicity, here we assume the use of the debit-card model (cf. Section 2.3) and defer the discussion on the credit-card model to Section 4.1.

In SIP, it is up to the source's secure module to decide when and how frequently to reward intermediate nodes according to its promise in the initial session request. Usually, when the REWARD counter attains a threshold defined in the session request, the source's secure module will output a REWARD packet to remunerate intermediate nodes en route to the destination. The problem here is that, if misbehaving, the source may refuse to pay intermediate nodes for previous communications carried out via them. It can do so by dropping the REWARD packets outputted from its secure module. To urge the source to honestly issue REWARD packets, we decide to charge the source whenever it initiates a data packet.

To simplify our presentation, suppose the source and the destination agreed in the session initialization phase to halve the packet forwarding expense, though our scheme can be used with any other payment splitting method. Recall that we assume a universal charging rate of λ credits per unit-sized packet. Then, for each successfully transmitted unit-sized packet, each of N intermediate nodes should receive λ credits, while both the source and the destination need to pay $\lambda N/2$ credits.

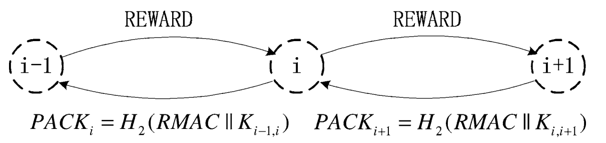

REWARD
REWARD
i-1
i
i+1
$PACK_i = H_2(RMAC \| K_{i-1,i})$
$PACK_{i+1} = H_2(RMAC \| K_{i,i+1})$

Fig. 2 The processing of a REWARD packet, where $1 \leq i \leq N$ and nodes 0 and (N+1) are the source and the destination, respectively

While the source initiates a packet, its secure module first checks whether the internal credit counter has no less than $\lambda N/2$ credits. If so, it decreases the credit counter by λN and outputs the packet with correct SIP fields[4], which is then transmitted to the next hop. Otherwise, the secure module refuses to generate the correct packet fields and the session is suspended. To resume the session, the source has to either wait until earning enough credits by relaying traffic for other nodes or to re-negotiate the payment proportion with the destination. The latter case is possible because the destination may have much more credits than the source and thus would like to contribute more. Upon receipt of a SIP data packet, the secure module of the destination decreases its credit counter by λN credits as well. The purpose of overcharging the source and the destination at this time is to urge the source to send REWARD packets later when the overcharged credits will be refunded. Note that this measure also serves as an incentive for the destination to send RECEIPT packets to the source.

When the REWARD counter attains a threshold μ specified in the initial session request, the source's secure module would output a REWARD packet to remunerate intermediate nodes. A REWARD packet is of format $<$*type* $=$ SIP-REWARD, ID_s, ID_d, SN_s, PN_s, *RMAC*$>$, where $RMAC = h(\text{all previous fields} \parallel K_{comm})$. To ensure that the source indeed sends out the REWARD packet, it does not get a refund of previously overcharged credits until receiving a keyed acknowledgement from node 1. Upon receipt of a REWARD packet, node 1 passes it to its secure module which, in turn, performs the following operations in addition to checking the packet sequence number:

1. Recompute *RMAC* and compare it with what was received. If they are equal,
2. Generate a value $PACK_1 = h(RMAC \| K_{s,1})$.

By the first operation, the secure module can ensure that the REWARD packet is neither a replayed one received previously nor a forged one by node 1 whose purpose is to reward itself for the packets it did not forward. The above verification works because node 1 has no knowledge of K_{comm} or $K_{s,1}$ that are stored in its tamper-proof secure module. In addition, the first operation can effectively prevent an inside-session intermediate node from forwarding the REWARD packet to its out-of-session colluding node in order to increase that peer's credit counter.

$PACK_1$ is used to inform the source's secure module about node 1's receipt of the REWARD packet. If the IEEE 802.11 MAC is used, $PACK_1$ can piggyback onto the MAC-layer acknowledgement. Alternatively, $PACK_1$ can piggyback onto the REWARD packet node 1 forwards to its next hop, i.e., node 2. Since the wireless channel is assumed to be directional, the source will be able to overhear $PACK_1$ by running its wireless network interface in the promiscuous mode. It then inputs $PACK_1$ into its secure module, which would increase the credit counter by the number of overcharged credits, $\lambda\mu N/2$, after verifying the authenticity of $PACK_1$ via a simple hash operation.

Similarly, only when the secure module of node 1 receives an authentic acknowledgement $PACK_2$ from node 2, does it increase the credit counter by $\lambda\mu$. This process continues until the REWARD packet finally reaches the destination. After verifying the REWARD packet, the secure module of the destination immediately increases the credit counter by $\lambda\mu N/2$ and sends an acknowledgement $PACK_d = h(RMAC \| K_{N,d})$ to the last intermediate node N. The secure module of node N can then increase its credit counter by $\lambda\mu$ after verifying $PACK_d$. Here we assume that all the intermediate nodes and the destination would like to send the keyed acknowledgement to their respective previous hop, partly because anyway they have to send the MAC-layer acknowledgements to enable reliable communications. Another reason is to continue earning credits (intermediate nodes) or avoid session interruption (the destination). For example, if the secure module of node i does not receive $PACK_{i+1}$ from node $(i+1)$ for certain times, it may choose to quit the session and thus node $(i+1)$ would be forced to leave the session as well; if the secure module of node N does not receive $PACK_d$ from the destination, it may stop serving the current session, thus leading to unfavorable session interruption.

Notice that a REWARD packet may get lost during its transmission or the route may break due to reasons such as node mobility. If either case occurs, some intermediate nodes will not receive their deserved credits and the source and/or the destination will not get the refund of certain overcharged credits. Fortunately, we can alleviate this problem by letting the source dynamically adjust the frequency of sending REWARD packets, i.e., the threshold value μ. We will dwell on this issue shortly in Section 4.3.

4. SIP enhancements

Up to now, we have described the basic operations of SIP. In this section, we discuss some special measures to make SIP more flexible and adaptable to the dynamically changing nature of MANETs.

[4] A credit counter might have negative credits temporarily.

4.1. Emergent transmission

In the previous section, we require that, when not having enough credits, a node has to first serve others to earn enough credits for its own use. We postulate that there are cases that a node may have emergent information to send, while does not have enough credits for the time being. Motivated by the use of credit cards in real life, SIP allows mobile nodes to have negative credits up to a pre-defined threshold. That is, when the source uses up its credits, it will be charged ξ times the normal rate, i.e., $\lambda\xi$ credits per unit-sized packet, for each packet sent until its negative credits reach a threshold ρ, where $\xi > 1$ is called the *punishing factor* and $\rho < 0$ is called the *maximum deficit*. However, the charging and rewarding rates for other nodes remain unchanged in order to make this method easily implementable.

4.2. Unbalanced credit distribution

For any credit-based stimulation scheme including our SIP, there is an unfairness problem that some nodes like those on the network edges cannot gain as many credits as their peers at other locations. This is not because they are reluctant to serve others, but because they are less frequently selected by the underlying routing protocol and thus cannot participate in enough sessions. This problem can be partially alleviated by node mobility: the higher mobility, the less notable the unfairness is. Another way to cope with such unfairness is to combine SIP with underlying routing protocols, such as some energy-aware routing protocols or others with load balancing features, to select routes in a way that the traffic load can be distributed more evenly onto network nodes. We also notice that such unfairness may motivate nodes to swarm to some locations within the network where they can forward more traffic and thus earn more credits. Currently, we are investigating the feasibility of such intelligent, strategic movement and its impact on a credit-based stimulation scheme like SIP.

4.3. Total credit decline

In a MANET, SIP data/REWARD/RECEIPT packets may get lost on the way due to various reasons such as node mobility, the MAC-layer collision, and wireless channel errors. Consequently, some intermediate nodes might not be correctly remunerated and the source and the destination might not be refunded the overcharged credits. This would cause the gradual decline of the overall credits in the network. Below we propose two approaches to address this problem and leave the performance evaluation to the next section.

One method is to let the source and the destination dynamically adjust the frequency of sending RECEIPT and REWARD packets based on the sequence numbers of arriving packets. That is, RECEIPT/REWARD packets are sent more frequently when much packet loss happens at short intervals and less frequently otherwise. The purpose is to remunerate intermediate nodes and refund the source and the destination in a timely manner before the session topology changes.

The other is to introduce an *asymmetric* payment model as compared to the symmetric one used previously. Since SIP does not aim at perfect billing in MANETs, the traditional symmetric payment model with zero net gain for a whole network is not necessary for our case. We propose appending to the REWARD packet a *risk factor* τ which is normally set to one and dynamically adjusted by the secure module of the source. For example, if packet loss happens frequently, τ is increased and intermediate nodes will be rewarded with a higher rate $\lambda\tau$ credits per unit-sized packet, while the source and the destination are always charged with the constant rate λ. This method will further alleviate the overall credit decline of the network and motivate mobile nodes to participate in packet forwarding even in a highly "risky" situation. How to securely determine the risk factor is part of our ongoing work.

5. Performance evaluation

In this section, we evaluate the performance of SIP using simulations.

5.1. Simulation setup

We implemented SIP in GloMoSim [21], a popular network simulator for MANETs. The bilinear map $\hat{e}$ we use is the Tate pairing, with some of the modifications and performance improvements described in [2, 16]. The security parameters we use are the lengths of two primes, q and p, where q is a 160-bit Solinas prime $2^{159} + 2^{17} + 1$ and p is a 512-bit prime equal to $12qr - 1$ (for some r large enough to make p the correct size). The elliptic curve E we use is $y^2 = x^3 + x$ defined over the finite field $\mathbb{F}_p$ (denoted by $E(\mathbb{F}_p)$). Then $\mathbb{G}_1$ is a q-order subgroup of the additive group of points over $E(\mathbb{F}_p)$, while $\mathbb{G}_2$ is a q-order subgroup of the multiplicative group of the finite field $\mathbb{F}^*_{p^2}$. According to [22], such bit-length configurations of p and q can deliver a comparable level of security to 1024-bit RSA [20]. We base the pairing implementation on MIRACL library [23]. In addition, the hash function used is SHA-1 [19].

The physical-layer path loss model is the two-ray model. The MAC layer protocol is the Distributed Coordination Function (DCF) of the IEEE 802.11. The radio propagation range for each node is 250 meters and the channel capacity is 2 Mb/s. For simplicity of simulations, we select AODV [4] as the underlying routing protocol by assuming that there are no routing attacks.

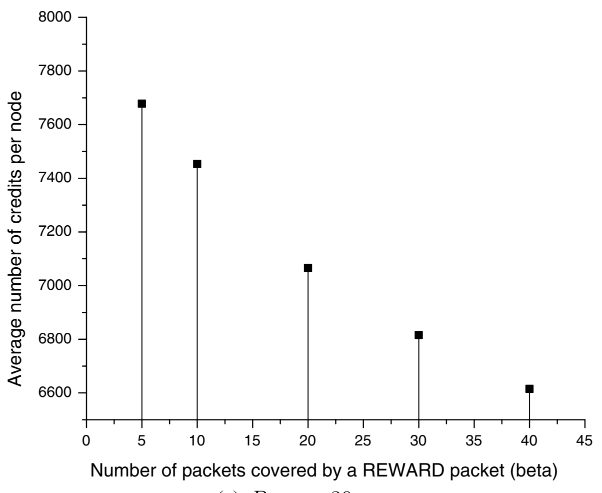
We simulate a MANET with 50 nodes uniformly deployed in a 1500×300 m^2 rectangular field. To emulate node mobility, we adopt the modified random waypoint model [24]. Specifically, a node travels towards a random destination uniformly selected within the network field; upon reaching the destination, it pauses for some time; and the process repeats itself afterwards. A node chooses its initial speed and pause time according to the methods given in [24], while subsequent speeds uniformly from the range [1, 20] m/s and pause times uniformly from the range $[0, P_{max}]$ s. We simulate different network mobility levels by varying the maximum pause time P_{max}.

The traffic used is 25 CBR connections with randomly selected source-destination pairs. All the data packets are 512 bytes and sent at a speed of 4 packets/s. Each simulation is executed for 15 simulated minutes and each data point represents an average of ten runs with identical traffic models, but differently generated mobility scenarios.

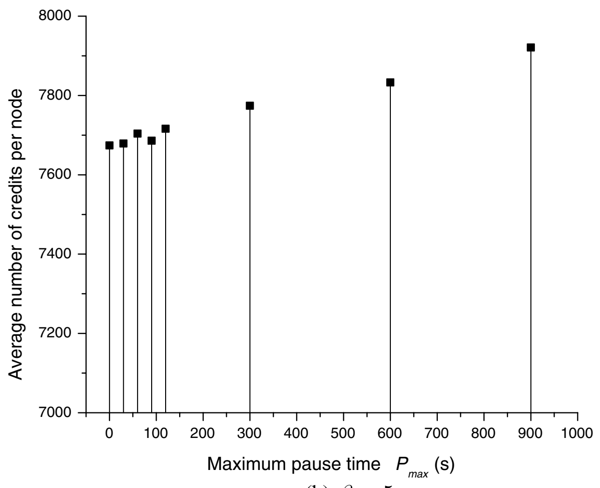
5.2. Simulation results

In their seminal paper [1], Marti *et al.* report that if 10 to 40 percent of network nodes are selfish, the average network throughput could degrade by 16 to 32 percent, which is confirmed by our simulation results. With SIP in place, nodes are naturally motivated to participate in packet forwarding to earn as many credits as possible for their own traffic. The network throughput would return closely to the normal level as if there were no selfish nodes. This result is quite intuitive, so we do not intend to report the related simulation results here. Instead, we focus on some most important aspects of any credit-based stimulation mechanism like SIP: the average network credit level, credit distribution, and control signalling overhead.

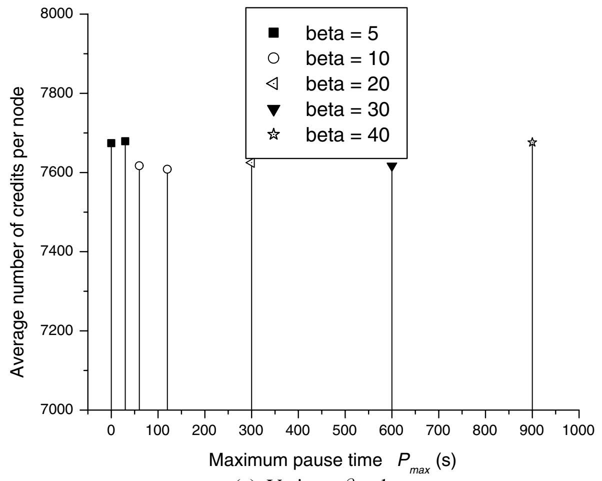
In our simulations, each node is preloaded with 8000 credits. For simplicity, each node has the same charging rate $\lambda = 1$ credit per unit-sized packet. In addition, the source will send out a REWARD packet for each received



(a) $P_{max} = 30$ s.



(b) $\beta = 5$.



(c) Various β values.

Fig. 3 Average number of credits per node

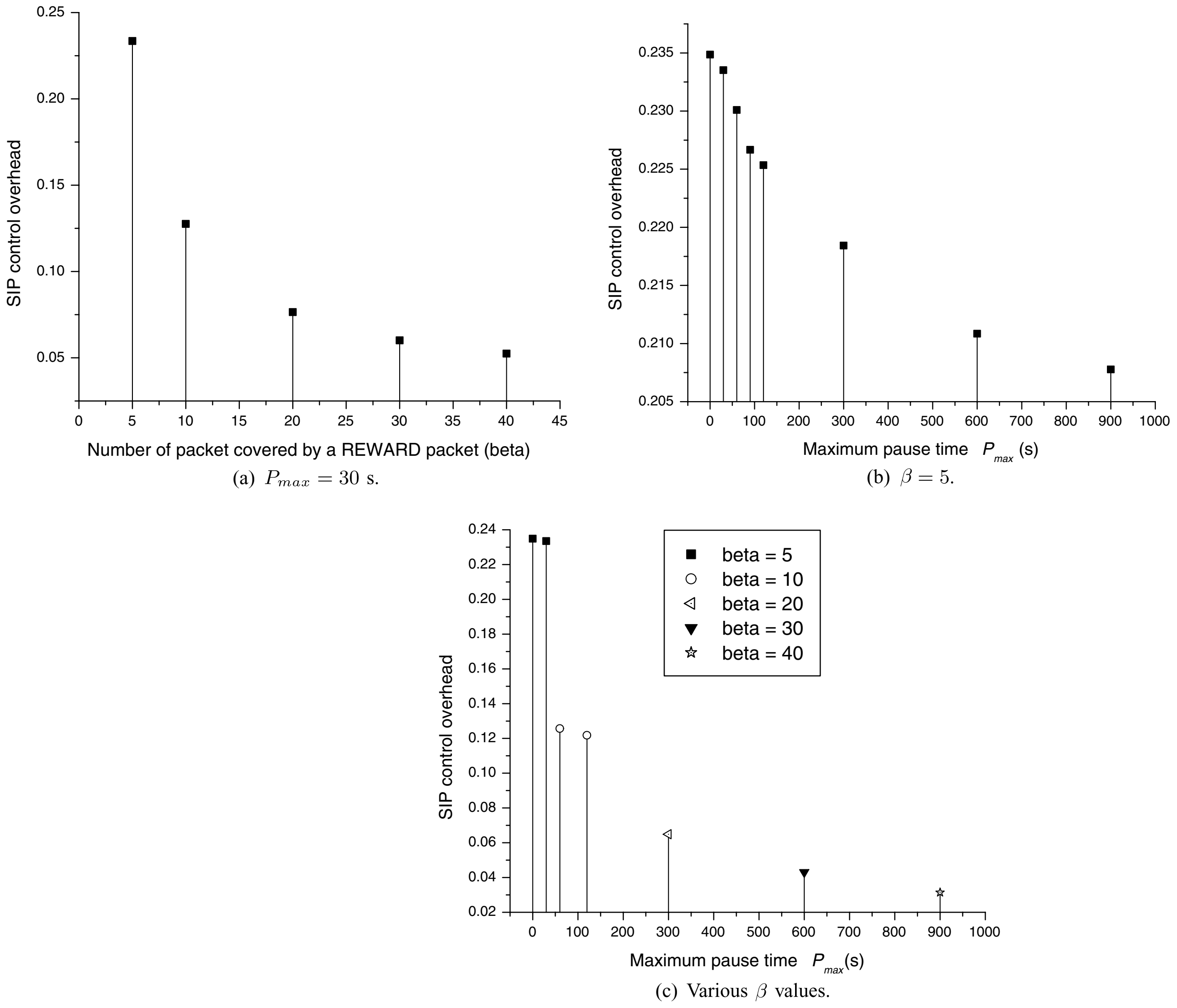


Fig. 4 SIP control overhead

RECEIPT packet. Therefore, we could realize different sending frequencies for REWARD/RECEIPT packets by merely adjusting the threshold β of the RECEIPT counter.

Figure 3 shows the average number of credits per node, i.e., the average network credit level, for different scenarios after 15 simulated minutes. In Fig. 3(a), the maximum pause time P_{max} is fixed to 30 s. As we can see, the number of data packets covered by a REWARD packet, i.e., β, has a large impact on the average network credit level under high network mobility: the larger β, the lower rewarding frequency, the lower average credit level is. This result is of no surprise because high mobility would result in more data/REWARD/RECEIPT packet loss and thus the total credit decline, as analyzed in Section 4.3. Therefore, we should dynamically adjust the rewarding frequency by setting β smaller in the face of high network mobility. The effect of this measure is depicted in Fig. 3(b), where β is fixed to five. It is obvious that this small β value can help maintain a high average network credit level under all mobility scenarios. The case of using different β values is plotted in Fig. 3(c), from which we can observe an expected rather stable average network credit level.

Figure 4 demonstrates SIP control overhead, defined as the ratio of all the SIP control packets over the total number of data packets transmitted during the simulation time. It is clear that, although a smaller β value would cause a higher network credit level, it will result in the increase of SIP control overhead due to the increased number of RECEIPT/REWARD packets. However, SIP control overhead is still at a very low level, especially under medium to low mobility scenarios. For example, when $P_{max} = 300/600/900$ s, SIP only incurs an overhead of 0.064/0.043/0.031 in order to maintain a stable, high average network credit level. It is also worth mentioning that in our simulations each SIP control packet is sent individually without piggybacking onto any data packet. In this sense, the shown SIP control overhead represents the worst scenario. In practice, because of their small sizes, SIP control packets could piggyback

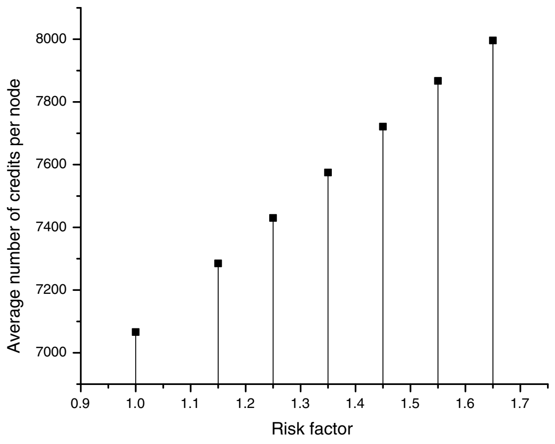


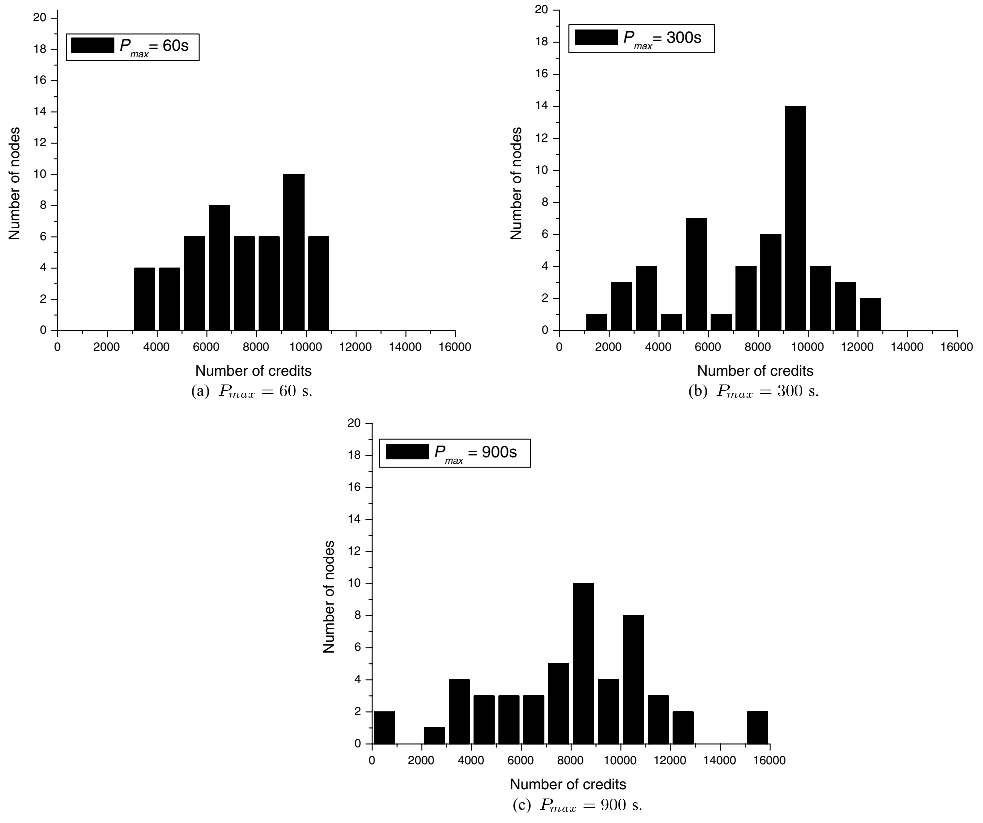
Fig. 5 Average number of credits per node for $P_{max} = 30$ s and $\beta = 20$

onto normal data packets whenever possible, in which case SIP control overhead can be reduced to a large extent.

Figure 5 illustrates the effectiveness of the asymmetric payment model proposed in Section 4.3 to deal with total credit decline. This figure is generated with $P_{max} = 30$ s and $\beta = 20$. It is not surprising to see that the larger the risk factor τ, the more stable the average network credit level is. The nice thing with this approach is that it does not increase SIP control overhead as compared to the above approach. The difficulty, however, might lie in the determination of an appropriate risk factor τ. On the one hand, if it is too small, it does not help much in maintaining a stable network credit level. On the other hand, if it is too large, selfish users might earn too many credits in a short time window and have no incentives to participate in subsequent sessions. It remains an open topic to further study the impact of this asymmetric payment model on a SIP-like stimulation scheme for MANETs.

Figure 6 shows the network credit distribution at the end of simulations. To be meaningful, the 25 CBR connections used in generating this figure involve all the nodes, that is, each node is either a source or a destination. We can clearly see that high network mobility would result in a much fairer credit distribution because each node would have many chances of participating in packet forwarding and thus earning credits. However, if network mobility is relatively low so that the network topology is relatively stable, some edge nodes will have lower chances of accumulating credits than their peers in the middle, as they are less frequently selected by the underlying routing protocol. For the latter scenario, some routing protocols with inherent load

Fig. 6 Network credit distribution



balancing features would help for a fair credit distribution. Due to space constraints, we do not report the rather intuitive results here.

To summarize, the above simulation results demonstrate that SIP is indeed a viable, lightweight solution to stimulating packet forwarding in totally self-organizing MANETs.

6. Related work

Recent years have witnessed a growing body of work on addressing node selfishness, which can be classified into two categories: *reactive* approaches and *preventive* approaches. The former are intended to enforce the cooperation by first detecting the selfish nodes, avoiding routing through them, and then punishing them via spreading their bad reputations and thus isolating them [1, 25, 26]. The major concern, however, is that it seems difficult (if not impossible) to prevent the propagation of incorrect reputations, either bad [25] and good [26], in the presence of node collusion.

As for the latter, most of the proposals are concerned with providing some kinds of incentives for selfish nodes. Buttyan and Hubaux [9] propose to stimulate packet forwarding by remunerating intermediate forwarding nodes with some credits paid by the source. SIP differs from [9] in many aspects such as the source-controlled session-based approach, the novel identifier-based session key establishment, and its flexibility and adaptability to network dynamics (e.g., the asymmetric payment model). Zhong *et al.* [10] develop another credit-based collusion-resistant scheme to address node selfishness, but their approach requires a centralized Credit Clearance Service on the backbone network, which may undermine the self-organizing, decentralized nature of MANETs. Due to the same reason, the base-station-based charging and rewarding scheme for motivating packet forwarding in multi-hop cellular networks [11] is less suitable in infrastructureless MANETs.

In addition to incentive-based preventive approaches, Srinivasan *et al.* [27] use game theory to model the rational yet non-cooperative behavior of nodes and to analyze the optimal trade-off between throughput and lifetime of energy-constrained nodes. The drawback is that it requires each node to keep track of the individual behavior of all the other nodes, which may be unrealistic in most cases. Furthermore, Felegyhazi *et al.* [28] prove in a game theoretic framework the existence of a cooperative equilibrium of packet forwarding strategies, but they neither consider node mobility nor suggest how to attain the equilibrium. More recently, Anderegg and Eidenbenz [12] and Wang *et al.* [29] apply mechanism design to the node selfishness issue, but both schemes are vulnerable to node collusion, as mentioned in [12, 29]. Moreover, Sundaramurthy and Belding-Royer [30] propose to address node selfishness using anonymous routing. Though interesting, their work is only loosely related to SIP.

7. Conclusion

In this paper, we propose a credit-based Secure Incentive Protocol (SIP) to stimulate cooperation in packet forwarding for infrastructureless MANETs. SIP is carefully designed to be a secure yet lightweight charging and remuneration protocol and can withstand a wide range of cheating actions. SIP is also of low communication overhead by using a space-efficient Bloom filter. In addition, it is flexible and well adaptable to the network dynamics of MANETs. The effectiveness of SIP is validated through extensive simulations. As the future research, we will first extend SIP to multicast routing. We also plan to combine SIP with reputation-based approaches to provide a unified solution against node selfishness.

References

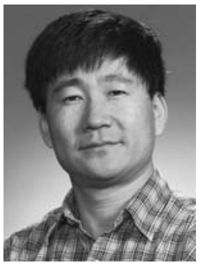
1. S. Marti, T. Giuli, K. Lai, and M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in: *Proceedings of ACM MobiCom* (Boston, Massachusetts, August 2000).
2. D. Boneh and M. Franklin, Identify-based encryption from the weil pairing, in *Proceedings of CRYPTO'01, ser. LNCS* 2139 (Springer-Verlag, 2001) pp. 213–229.
3. B. Bloom, Space/time trade-offs in hash coding with allowable errors, Communications of the ACM 13(7) (July 1970).
4. C. Perkins, E. Belding-Royer, and S. Das, Ad hoc on-demand distance vector (AODV) routing, RFC 3561 (July 2003).
5. D. Johnson and D. Maltz, *Dynamic Source Routing in Ad Hoc Wireless Networks* (Kluwer Academic Publishers, Vol. 353, 1996) pp. 153–181.
6. Y.-C. Hu, A. Perrig, and D. B. Johnson, Ariadne: A secure on-demand routing protocol for ad hoc networks, in: *Proc. ACM MobiCom* (Atlanta, GA, Sept. 2002).
7. W. Lou, W. Liu, and Y. Fang, SPREAD: Enhancing data confidentiality in mobile ad hoc networks, in: *Proc. IEEE INFOCOM'04* (Hong Kong, China, March 2004).
8. Y. Zhang, W. Liu, and W. Lou, Anonymous communications in mobile ad hoc networks, in: *Proc. IEEE INFOCOM'05* (Miami, FL, March 2005).
9. L. Buttyan and J. Hubaux, Stimulating cooperation in self-organizing mobile ad hoc networks, ACM Journal for Mobile Networks and Applications (MONET) 8(5) (October 2003).
10. S. Zhong, J. Chen, and Y. Yang, Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks, in: *Proc. IEEE INFOCOM* (San Francisco, CA, April 2003).
11. N. Salem, L. Buttyan, J. Hubaux, and M. Jakobsson, A charging and rewarding scheme for packet forwarding in multi-hop cellular networks, in: *Proc. ACM MobiHoc* (Annapolis, Maryland, June 2003).
12. L. Anderegg and S. Eidenbenz, Ad hoc-vcg: A trustful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents, in: *Proc. ACM MobiCom* (San Diego, CA, Sep. 2003).
13. R. Anderson and M. Kuhn, Tamper resistance—a cautionary note, in: *Proc. 2nd USENIX Workshop on Electronic Commerce* (Oakland, CA, Nov. 1996).
14. A. Shamir, Identity based cryptosystems and signature schemes, in: *Proc. CRYPTO'84*, ser. LNCS, vol. 196 (Springer-Verlag, 1984) pp. 47–53.

15. Y. Zhang, W. Liu, W. Lou, Y. Fang, and Y. Kwon, AC-PKI: Anonymous and certificateless public-key infrastructure for mobile ad hoc networks, in: *Proc. IEEE ICC'05* (Seoul, Korea, May 2005).
16. P. Barreto, H. Kim, B. Bynn, and M. Scott, Efficient algorithms for pairing-based cryptosystems, in *Proc. CRYPTO'02*, ser. LNCS, vol. 2442 (Springer-Verlag, 2002) pp. 354–368.
17. K. Sanzgiri, D. LaFlamme, B. Dahill, B. Levine, C. Shields, and E. Belding-Royer, Authenticated routing for ad hoc networks, IEEE J. Select. Areas Commun 23(3) (March 2005) 598–610.
18. P. Kotzanikolaou, R. Mavropodi, and C. Douligeris, Secure multipath routing for mobile ad hoc networks, in: *Proc. Second Annual Conference on Wireless On-demand Network Systems and Services (WONS'05)* (St. Moritz, Switzerland, Jan. 2005).
19. NIST, Digital hash standard, Federal Information Processing Standards PUBlication 180-1 (April 1995).
20. R. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public key cryptosystems, Communications of the ACM 21(2) (Feb. 1978) 120–126.
21. X. Zeng, R. Bagrodia, and M. Gerla, GloMoSim: A library for parallel simulation of large scale wireless networks, in *Proc. 12 Workshop on Parallel and Distributed Simulations (PADS'98)* (Banff, Alberta, Canada, May 1998) pp. 154–161.
22. D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon, and H.-C. Wong, Secure handshakes from pairing-based key agreements, in *Proc. IEEE Symposium on Security & Privacy* (Oakland, CA, May 2003).
23. Shamus Software Ltd., Miracl library. [Online]. Available: http://indigo.ie/ mscott/.
24. J. Yoon, M. Liu, and B. Nobles, Sound mobility models, in *Proc. ACM MobiCom* (San Diego, CA, Sept. 2003).
25. S. Buchegger and J. Boudec, Performance analysis of the confidant protocol: Cooperation of nodes-fairness in distributed ad-hoc networks, in: *Proc. IEEE/ACM MobiHoc* (Lausanne, Switzerland, June 2002).
26. P. Michiardi and R. Molva, Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks, in *Proc. 6th IFIP Comm. Multimedia Security Conf.* (Portorosz, Slovenia, Sep. 2002).
27. V. Srinivasan, P. Nuggehalli, C. Chiasserini, and R. Rao, Cooperation in wireless ad hoc networks, in: *Proc. IEEE INFOCOM* (San Francisco, CA, April 2003).
28. M. Felegyhazi, L. Buttyan, and J. Hubaux, Equilibrium analysis of packet forwarding strategies in wireless ad hoc networks-the static case, in: *Proc. Personal Wireless Communication (PWC)* (Venice, Italy, Sept. 2003).
29. W. Wang, X. Li, and Y. Wang, Truthful multicast routing in selfish wireless networks, in: *ACM MobiCom* (Philadelphia, Pennsylvania, Sept. 2004).
30. S. Sundaramurthy and E. M. Belding-Royer, The ad-mix protocol for encouraging participation in mobile ad hoc networks, in: *IEEE ICNP* (Atlanta, GA, Nov. 2003).

Yanchao Zhang received the B.E. degree in Computer Communications from Nanjing University of Posts and Telecommunications, Nanjing, China, in July 1999, and the M.E. degree in Computer Applications from Beijing University of Posts and Telecommunications, Beijing, China, in April 2002. Since September 2002, he has been working towards the Ph.D. degree in the Department of Electrical and Computer Engineering at the University of Florida, Gainesville, Florida, USA. His research interests are network and distributed system security, wireless networking, and mobile computing, with emphasis on mobile ad hoc networks, wireless sensor networks, wireless mesh networks, and heterogeneous wired/wireless networks.

Wenjing Lou is an assistant professor in the Electrical and Computer Engineering department at Worcester Polytechnic Institute. She obtained her Ph.D degree in Electrical and Computer Engineering from University of Florida in 2003. She received the M.A.Sc degree from Nanyang Technological University, Singapore, in 1998, the M.E degree and the B.E degree in Computer Science and Engineering from Xi'an Jiaotong University, China, in 1996 and 1993 respectively. From Dec 1997 to Jul 1999, she worked as a Research Engineer in Network Technology Research Center, Nanyang Technological University. Her current research interests are in the areas of ad hoc and sensor networks, with emphases on network security and routing issues.

Wei Liu received his B.E. and M.E. in Electrical and Information Engineering from Huazhong University of Science and Technology, Wuhan, China, in 1998 and 2001. In August 2005, he received his PhD in Electrical and Computer Engineering from University of Florida. Currently, he is a senior technical member with Scalable Network Technologies. His research interest includes cross-layer design, and communication protocols for mobile ad hoc networks, wireless sensor networks and cellular networks.



Yuguang Fang received a Ph.D. degree in Systems Engineering from Case Western Reserve University in January 1994 and a Ph.D degree in Electrical Engineering from Boston University in May 1997. He was an assistant professor in the Department of Electrical and Computer Engineering at New Jersey Institute of Technology from July 1998 to May 2000. He then joined the Department of Electrical and Computer Engineering at University of Florida in May 2000 as an assistant professor, got an early promotion to an associate professor with tenure in August 2003 and a professor in August 2005. He has published over 150 papers in refereed professional journals and conferences. He received the National Science Foundation Faculty Early Career Award in 2001 and the Office of Naval Research Young Investigator Award in 2002. He has served on many editorial boards of technical journals including IEEE Transactions on Communications, IEEE Transactions on Wireless Communications, IEEE Transactions on Mobile Computing and ACM Wireless Networks. He is a senior member of the IEEE.