IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 2, Issue 6, Dec-Jan, 2015
**ISSN: 2320 – 8791 (Impact Factor: 1.479)**
**www.ijreat.org**

# Ensuring Account Security In Wireless Networks

**Chaitanya Chowdary Bandlamudi[1], K. Venkata Ramaiah[2]**

**[1]M.Tech Scholar, Department of Computer Science & Engineering, Chebrolu Engineering College, Chebrolu, Andhra Pradesh, India**

**[2]Assoc. Prof & Head of the Department of Computer Science & Engineering, Chebrolu Engineering College, Chebrolu, Andhra Pradesh, India**

**Abstract**

A secure payment scheme, called as the Report based pAyment sChemE (RACE) is used in multi hop wireless networks to stimulate node cooperation, regulate packet transmission and enforce fairness. The nodes submit lightweight payment reports (instead of receipts) to the Trusted Authority (TA) to update their credit accounts and temporarily store the evidences which are undeniable. The report includes the session information. The Trusted Authority verifies the payment by investigating the consistency of the report and clears the fair reports with almost no cryptographic operations or computational overhead. The nodes which do not pass or relay others packets, but it makes use of neighbor or cooperative nodes to relay its packets are called selfish nodes. This degrades the network connectivity and fairness. Such type of nodes also submits reports to the Trusted Authority. But when tested for consistency, it is found to be a cheating node. For such reports, the evidences are requested by the Trusted Authority to identify and evict the cheating nodes or selfish nodes. After evicting the selfish nodes, communication can be efficiently established again with increased throughput and less amount of processing and communication overhead.

RACE is the first payment scheme that uses the concept of evidences to secure the payments. It requires cryptographic operations in clearing the payment only in the case of cheating. Also this is the first system that can verify the payment by investigating the consistency of the node's reports without submitting and processing security tokens and without false accusations. To prevent the multihop communications from failing due to insufficient credits, the source node can borrow credits or they can be purchased with real money from the Trusted Authority.

RACE can secure the payment and precisely identify the cheating nodes without false denunciations. This is done by establishing a route between the source and the destination by sending a route request to the destination and the destination replies with path, a hash element from the hash chain and the signature.

***Keywords*—**Cooperation incentive schemes, network-level security, payment schemes, lightweight payment reports and Evidences

## 1. Introduction

MWNs can be deployed readily at low cost in developing and rural areas. Multihop packet relay can extend the network coverage using limited transmit power, improve area spectral efficiency, and enhance the network throughput and capacity. In multihop wireless networks (MWNs), the traffic originated from a node is usually relayed through the other nodes to the destination for enabling new applications and enhancing the network performance and deployment [1]. MWNs can also implement many useful applications such as data sharing [2] and multimedia data transmission [3]. For example, users in one area (residential neighborhood, university campus, etc.) having different wireless-enabled devices, e.g., PDAs, laptops, tablets, cell phones, etc., can establish a network to communicate, distribute files, and share information. However, the assumption that the nodes are willing to spend their scarce resources, such as battery energy, CPU cycles, and available network bandwidth, to relay others' packets without compensation cannot be held for civilian applications where the nodes are autonomous and aim to maximize their welfare.

The fairness issue arises when the selfish nodes make use of the cooperative nodes to relay their packets without any contribution to them, and thus the cooperative nodes are unfairly overloaded because the network traffic is concentrated through them. This selfish behavior also degrades the network connectivity significantly, which may cause the multihop communication to fail [4].

Payment (or incentive) schemes [5] use credits (or micropayment) to motivate the nodes to cooperate in relaying others packets by making cooperation more beneficial than selfishness. The nodes earn credits for relaying others packets and spend these credits to get their packets relayed by others. In addition to cooperation stimulus, these schemes can enforce fairness, discourage *Message-Flooding* attacks, regulate packet transmission, and efficiently charge for the network services. Fairness can be enforced by rewarding the nodes that relay more packets and charging the nodes that send more packets. For example, the nodes situated at the network center relay more packets than the other nodes because they are more frequently selected by the routing protocol. Since the source nodes pay for relaying their packets, the payment schemes regulate packet transmission and discourage *Message-Flooding* attacks where the attackers send bogus messages to diminish the intermediate nodes' resources. Moreover, since the communication sessions may be held without involving a trusted party (TP) and the nodes may travel among different foreign networks, the payment schemes can charge the nodes efficiently without contacting distant home location registers [6].

For the payment schemes in MWNs, there is usually one customer (the source node) and multiple merchants (the intermediate nodes). The merchants' number is large because any network node can act as a merchant (or packet relay), and a transaction's value is much less than those in credit card payment schemes. The relation between a customer and a merchant is usually short due to the network dynamic topology, and the nodes are involved in low-value transactions very frequently because once a route is broken, a new transaction should be done to reestablish the route. Due to these unique characteristics, MWNs require a specially designed payment scheme.

A good payment scheme should be secure, and require low overhead. However, the existing receipt-based payment schemes impose significant processing and communication overhead and implementation complexity. Since a trusted party may not be involved in communication sessions, the nodes compose proofs of relaying others' packets, called receipts, and submit them to an offline accounting center (AC) to clear the payment. The receipts' size is large because they carry security proofs, e.g., signatures, to secure the payment, which significantly consumes the nodes' resources and the available bandwidth in submitting them. The AC has to apply a large number of cryptographic operations to verify the receipts, which may require impractical computational power and make the practical implementation of these schemes complex or inefficient. Moreover, since a transaction (relaying packets) value may be very low, the scheme uses micropayment, and thus a transaction's overhead in terms of submitting and clearing the receipts should be much less than its value. Therefore, reducing the communication and the payment processing overhead is essential for the effective implementation of the payment scheme and to avoid creating a bottleneck at the AC and exhausting the nodes' resources.

In this paper, we propose RACE, a Report-based pAyment sChemE for MWNs. The nodes submit lightweight payment reports (instead of receipts) to the AC to update their credit accounts, and temporarily store undeniable security tokens called Evidences. The reports contain the apparent charges and rewards of different sessions without security proofs, e.g., signatures. The AC verifies the payment by investigating the consistency of the reports, and clears the payment of the fair reports with almost no cryptographic operations or computational overhead. For cheating reports, the Evidences are requested to identify and evict the cheating nodes that submit incorrect reports, e.g., to steal credits or pay less. In other words, the Evidences are used to resolve disputes when the nodes disagree about the payment. Instead of requesting the Evidences from all the nodes participating in the cheating reports, RACE can identify the cheating nodes with submitting and processing few Evidences. Moreover, Evidence aggregation technique is used to reduce the storage area of the Evidences.

In RACE, Evidences are submitted and the AC applies cryptographic operations to verify them only in case of cheating, but the nodes always submit security tokens, e.g., signatures, and the AC always applies cryptographic operations to verify the payment in the existing receipt-based schemes. RACE can clear the payment nearly without applying cryptographic operations and with submitting lightweight reports when Evidences are not frequently requested. Widespread cheating actions are not expected in civilian applications because the common users do not have the technical knowledge to tamper with their devices. Moreover, cheating nodes are evicted once they commit one cheating action and it is neither easy nor cheap to change identities. Our analytical and simulation results demonstrate that RACE requires much less communication and processing overhead than the existing receipt-based schemes with acceptable payment clearance delay and Evidences' storage area, which is necessary to make the practical implementation of the payment scheme effective. Moreover, RACE can secure the payment and precisely identify the cheating nodes without false accusations or stealing credits.

To the best of our knowledge, RACE is the first payment scheme that can verify the payment by investigating the consistency of the nodes' reports without systematically submitting and processing security tokens and without false accusations. RACE is also the first scheme that uses the concept of Evidence to secure the payment and requires applying cryptographic operations in clearing the payment only in case of cheating.

## 2. Related Work

The existing payment schemes can be classified into **T**amper-**P**roof-**D**evice (TPD)-based and receipt-based schemes. In TPD-based payment schemes [7], [8], [9], [10], a TPD is installed in each node to store and manage its credit account and secure its operation. For receipt-based payment schemes [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], an offline central unit called the accounting center stores and manages the nodes' credit accounts. The nodes usually submit undeniable proofs for relaying packets, called receipts, to the AC to update their credit accounts.

In Nuglets [7], the self-generated and forwarded packets by a node are passed to the TPD to decrease and increase the node's credit account, respectively. Packet purse and packet trade models have been proposed. For the packet purse model, the source node's credit account is charged the full payment before sending a packet, and each intermediate node acquires the payment for relaying the packet. For the packet trade model, each intermediate node runs an auction to sell the packets to the next node in the route, and the destination node pays the total cost of relaying the packets. In SIP [8], after receiving a data packet, the destination node sends a RECEIPT packet to the source node to issue a REWARD packet to increment the credit accounts of the intermediate nodes. In [9], the credit account of the source node is charged and a signature is attached to each data packet. Upon receiving the packet, the credit account of the destination node is also charged, and a digitally signed acknowledgement (ACK) packet is sent back to the source node to increase the credit accounts of the intermediate nodes.

The receipt-based payment schemes impose more over-head than the TPD-based schemes because they require submitting receipts to the AC and processing them. How-ever, the TPD-based payment schemes suffer from the following serious issues. ***First***, the assumption that the TPD cannot be tampered with, cannot be guaranteed because the nodes are autonomous and self-interested, and the attackers can communicate freely in an undetectable way if they could compromise the TPDs. ***Second***, the nodes cannot communicate if they do not have sufficient credits during the communication time. Unfortunately, the nodes at the network border cannot earn as many credits as the other nodes because they are less frequently selected by the routing protocol. ***Finally***, since credits are cleared in real time, the multihop communications fail if the network does not have enough credits circulating around because the nodes do not have sufficient credits to communicate. In [10], it is shown that the overall credits in the network decline gradually with using TPD-based schemes because the total charges may be more than the total rewards. This is because the source node is fully charged after sending a packet but some intermediate nodes may not be rewarded when the route is broken.

In order to eliminate the need for TPDs, an offline central bank called the AC is used to store and manage the nodes' credit accounts. In Sprite [11], for each message, the source node signs the identities of the nodes in the route and the message, and sends the signature as a proof for sending a message. The intermediate nodes verify the signature, compose receipts containing the identities of the nodes in the route and the source node's signature, and submit the receipts to the AC to claim the payment. The AC verifies the source node's signature to make sure that the payment is correct. However, the receipts overwhelm the network because the scheme generates a receipt per message.

Unlike Sprite that charges only the source node, FESCIM [12] adopts fair charging policy by charging both the source and destination nodes when both of them are interested in the communication. In PIS [13], the source node attaches a signature to each message and the destination node replies with a signed ACK packet. PIS can reduce the receipts' number by generating a fixed-size receipt per session regardless of the number of messages instead of generating a receipt per message in Sprite. In order to reduce the communication and processing overhead, CDS [14] uses statistical methods to identify the cheating nodes that submit incorrect payment. However, due to the nature of the statistical methods, the colluding nodes may manage to steal credits, and some honest nodes may be falsely accused of cheating which is called false accusations. Moreover, some cheating nodes may not be identified which is called missed detections, and it may take long time to identify the cheating nodes.

In [15], a payment scheme has been proposed for hybrid ad-hoc networks, but involving the base stations in every communication session may lead to suboptimal routes when the source and destination nodes reside in the same cell. In addition, corrupted messages are relayed to the base stations before they are dropped because the intermediate nodes cannot verify the authenticity and the integrity of the messages. In [16], each node has to contact the AC in each communication session to get coins to buy packets from the previous node in the route. However, the interactive involvement of the AC in each session is inefficient, causes long delay, and creates a bottleneck.

ESIP [17] proposes a communication protocol that can be used for a payment scheme. ESIP transfers messages from the source to the destination nodes with limited number of public key cryptography operations by integrating public key cryptography, identity-based cryptography, and hash function. Public key cryptography and hash function are used to ensure message integrity and payment nonrepudiation to secure the payment. Identity-based cryptography is used to efficiently compute a shared symmetric key between the source node and each node in the route. Using these keys, the source node computes and sends a keyed hash value for each intermediate node to verify the message integrity. Comparing to PIS, ESIP requires fewer public key cryptography operations but with larger receipts' size. Unlike ESIP that aims to transfer messages efficiently from the source to the destination nodes, RACE aims to reduce the overhead of submitting the payment data to the AC and processing them. Although the communication protocol proposed in ESIP can be used with RACE, we use a simple protocol due to space limitation and to focus on our contributions.

| | RACE | Receipt-based schemes [11-13, 17] | CDS |
|---|---|---|---|
| Communication overhead | LOW | LARGE | LOW |
| Payment processing overhead | Fair reports: light overhead Cheating reports: Cryptographic operations are applied | Cryptographic operations are systematically applied | Lightweight statistical operations |
| Payment Clearance delay | Much shorter than CDS in case of cheating | The shortest delay | Very long delay in case of cheating |
| Storage area | More than receipt based schemes | More than CDS and less than RACE | Smallest storage area |
| Security | • No false accusations and missed detections. • Strong protection against colluders • Cheaters are identified after the first cheating action | • No false accusations and missed detections. • Strong protection against colluders • Cheaters are identified after the first cheating action | • False accusations and missed detections. • Vulnerable to collusion attacks • Long time to identify the cheaters |

*Table 1 Comparison between RACE and the Existing Payment Schemes*

A mechanism is proposed in [18] to thwart packet dropping attacks. Payment is used to thwart the rational packet-dropping attacks, and a reputation system is used to identify and evict the irrational packet dropping attackers once their packet-dropping rates exceed a threshold. In [19], Zhu et al. propose a payment scheme, called SMART, for delay tolerant wireless networks (DTNs). SMART uses layered coins and can secure the payment against a wide range of attacks such as Credit-Forgery, Nodular-Tontine, and Submission-Refusal. Lu et al. [20] propose a payment scheme for DTNs which focuses on the fairness issue. The intermediate nodes earn credits for forwarding the delivered messages and gain reputation for forwarding the undelivered messages which gives them preference in forwarding future messages. However, the payment schemes designed for DTNs may not be efficiently applicable to MWNs because DTNs lack fully connected end-to-end routes and tolerate long packet delivery delay. Table 1 summarizes the main features of RACE and the existing payment schemes. RACE is more secure than CDS because it does not suffer from false accusations, missed detections, and delay in identifying attackers, and it can thwart collusion attacks. Moreover, RACE requires much less communication and processing overhead comparing to receipt-based schemes [11], [12], [13], [17], yet with more and acceptable storage area and payment clearance delay.

## 3. System Design

### 3.1 Network Model

For military and disaster recovery applications, the network can be considered ephemeral because it is used for a specific purpose and short duration. In this paper, we adopt the network model used in [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17] that targets the civilian applications of MWNs, where the network has long life and the nodes have long-term relations with the network. As illustrated in Fig. 1, the considered MWN has an offline TP and mobile nodes. The TP contains the AC and the certificate authority (CA). The AC maintains the nodes' credit accounts and the CA renews and revokes the nodes' certificates. Each node A has to register with the trusted party to receive a symmetric key $K_A$, private/public key pair, and certificate. The symmetric key is used to submit the payment reports and the private/public keys are required to act as source or destination node. Once the AC receives the payment reports of a session and verifies them, it clears the payment if the reports are fair; else, it requests the Evidences to identify the cheating nodes. The CA evicts the cheating nodes by denying renewing their certificates.

RACE can be used with any source routing protocol, such as DSR [22], which establishes end-to-end routes before transmitting data. Source nodes' packets may be relayed several hops by intermediate nodes to their destinations. The nodes can contact the TP at least once during a period of few days. In this connection, the nodes submit the payment reports and the Evidences (if requested), and receive renewed certificates to be able to continue using the network. The nodes also can purchase credits with real money to enable the nodes that cannot earn sufficient credits, such as those at the network border, to communicate, and also to avoid credit decline because the total charges may be more than the rewards when routes are broken. This connection can occur via the base stations of cellular networks, Wi-Fi hotspots, or wired networks such as Internet.

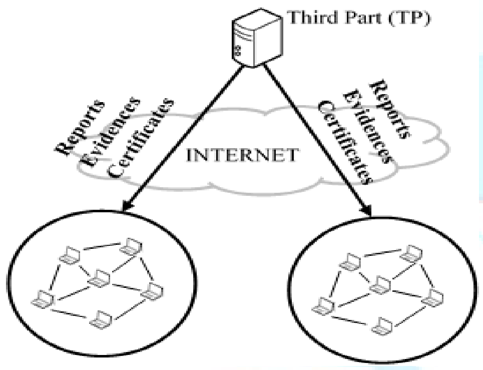

*Figure 1 The architecture of the considered network*

For the payment model, source nodes are charged for every transmitted message even if it does not reach the destination nodes, but the intermediate nodes are rewarded only for the delivered messages. Some schemes, such as [23], consider that the reward of relaying a packet is proportional to the incurred energy in relaying the packet. This rewarding policy can be integrated with RACE, but similar to [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], we use fixed rewarding rate, e.g., **A** credits per unit-sized packet to simplify our description and focus on our contributions.

### *3.1 Adversary Model*

The mobile nodes are probable attackers but the TP is fully secure. The mobile nodes are autonomous and self-interested and thus motivated to misbehave. The TP is run by an operator that is motivated to ensure the network proper operation. As discussed in [24], it is impossible to realize secure payment between two entities without a trusted third party. The attackers have full control on their nodes and can change their operation and infer the cryptographic data. The attackers can work individually or collude with each other under the control of one attacker to launch sophisticated attacks. These strong assumptions are necessary due to implementing payment in the network.

| Symbol | Description |
|---|---|
| X, Y | X is concatenated to Y |
| F | A flag bit indicating whether the last received packet by a node is for acknowledgment (ACK) or data. |
| $h^{(i)}$ | The hash value number i in hash chain created by the destination node. |
| H(P) | The hash value resulted from hashing P |
| $H_K(P)$ | The keyed hash value resulted from hashing P using the key K |
| $ID_A$ | The identity of an intermediate node A |
| $ID_S$ and $ID_D$ | The identities of the source node (S) and the destination node (D), respectively. |
| $K_A$ | The shared key between node A and the TP |
| $M_X$ | The message send in the $X^{th}$ data packet |
| n | The number of nodes in a route |
| $P_c(n)$ | The average payment clearance delay for a route with n nodes. |
| R | The concatenation of the identities of the nodes in a route e.g., $R=ID_S, ID_A, \ldots ID_D$ |
| $Sig_S()$, $Sig_D()$ | Signatures of Source and Destination nodes |
| $T_{cert}$ | Life time of a certificate |
| $T_S$ | Time Stamp of a route establishment |

*Table 2 Description of the Symbols*

Similar to [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], the attackers are rational in the sense that they misbehave only when they can achieve more benefits than behaving honestly. Particularly, the attackers aim to steal credits, pay less, and communicate for free. Table 2 gives the description of the used symbols in this paper.

## 4. Proposed System

The as shown in Fig. 2, RACE has four main phases. In Communication phase, the nodes are involved in communication sessions and Evidences and payment reports are composed and temporarily stored. The nodes accumulate the payment reports and submit them in batch to the TP. For the Classifier phase, the TP classifies the reports into fair and cheating. For the Identifying Cheaters phase, the TP requests the Evidences from the nodes that are involved in cheating reports to identify the cheating nodes. The cheating nodes are evicted and the payment reports are corrected. Finally, in Credit-Account Update phase, the AC clears the payment reports.

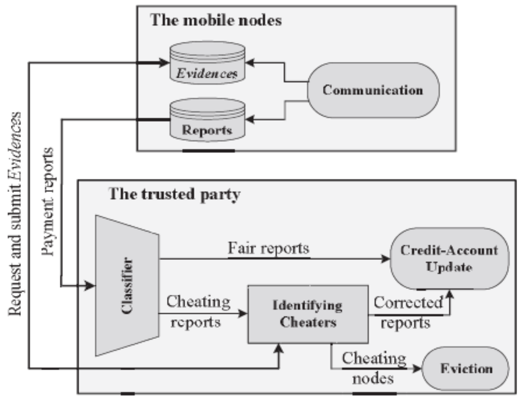

*Figure 2 The architecture of RACE*

## 4.1 Communication

The *Communication* phase has four processes: route establishment, data transmission, Evidence composition, and payment report composition/submission.

***Route establishment:*** In order to establish an end-to-end route, the source node broadcasts the *Route Request (RREQ)* packet containing the identities of the source ($ID_S$) and the destination ($ID_d$) nodes, time stamp ($T_s$), and Time-To-Live (TTL). TTL is the maximum number of intermediate nodes. After a node receives the *RREQ* packet, it appends its identity and broadcasts the packet if the number of intermediate nodes is fewer than TTL. The destination node composes the *Route Reply (RREP)* packet for the nodes broadcasted the first received *RREQ* packet, and sends the packet back to the source node. The destination node creates a hash chain by iteratively hashing a random value ($h^{(K)}$) K times to produce the hash chain root ($h^{(0)}$), where $h^{(i-1)}) = H(h^{(i)})$ and $1 \leq i \leq K$. The optimal value of K depends on many factors such as the number of messages the source node needs to send, and the average number of messages sent through a route before it is broken, i.e., due to node mobility. Estimating a good value for K can save the destination node's resources because once a route is broken, the unused hash values in the hash chain should not be used for another route to secure the payment. The nodes can estimate the value of K and periodically tune it.

The *RREP* packet contains the identities of the nodes in the route (e.g., R = $ID_S$, $ID_a$; $ID_b$; $ID_d$ in the route shown in Fig.3), $h^{(0)}$, and the destination node's certificate and signature ($Sig_D(R; Ts; h^{(0)})$). This signature authenticates the hash chain and links it to the route. The intermediate nodes verify the destination node's signature, relay the *RREP* packet, and store the signature and $h^{(0)}$ for composing the *Evidence.*

***Data transmission:*** The source node sends data packets to the destination node through the established route and the destination node replies with ACK packets. For the Xth data packet, the source node appends the message $M_X$ and its signature to R, X, $T_s$, and the hash value of the message ($H(M_X)$) and sends the packet to the first node in the route. The security tokens of the Xth data and ACK packets are illustrated in Fig. 3. The source node's signature is an undeniable proof for transmitting X messages and ensures the message's authenticity and integrity. Signing the hash of the message instead of the message can reduce the *Evidence* size because the smaller-size $H(M_X)$ is attached to the *Evidence* instead of MX. Before relaying the packet, each intermediate node verifies the signature to ensure the message's authenticity and integrity, and verifies R and *X* to secure the payment. Each node stores only the last signature for composing the *Evidence,* which is enough to prove transmitting X messages, e.g., after receiving the Xth data packet, the nodes should store $Sig_S(R, X, Ts, H(M_X))$ and remove $Sig_S(R, X\text{-}1, Ts, H(M_X\text{-}1))$, and so on. The data transmission process ends when the source node transmits its last message, or if the route is broken, e.g., due to node mobility or channel impairment. Algorithm 1 gives the pseudo code of the processes of data transmission and composition of *Evidence* and report.

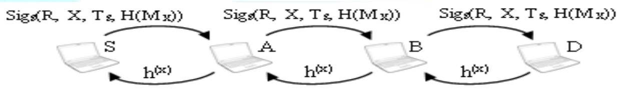
$Sig_S(R, X, T_s, H(M_X))$
$Sig_S(R, X, T_s, H(M_X))$
$Sig_S(R, X, T_s, H(M_X))$
S
A
B
D
$h^{(x)}$
$h^{(x)}$
$h^{(x)}$

*Figure 3 The security tokens of the Xth data and ACK packets*

After receiving the Xth data packet, Fig. 3 shows that the destination node sends back an ACK packet containing the pre-image of the last sent hash value (or $h^{(X)}$) to acknowledge receiving the message $M_X$, where $h^{(1)}$ is released in the first ACK and $h^{(2)}$ in the second and so on. Each intermediate node verifies the hash value by making sure that $h^{(X-1)}$ is obtained from hashing $h^{(X)}$. The nodes store only the last released hash value for composing the *Evidence.* The possession of $h^{(X)}$ by a node is a proof of delivering *X* messages, but the possession of $Sig_S(R, X, Ts, H(M_X))$ is a proof of delivering *X-1* messages and receiving one. The number of delivered messages can be computed from the number of hashing operations to map $h^{(X)}$ to $h^{(0)}$, and the number of transmitted messages (X) is signed by the source node. An intermediate node cannot drop the Xth data packet and claim delivering it because the hash function is one way, i.e., it is computationally infeasible to compute $h^{(X)}$ from $h^{(X-1)}$. Hash chains have been used for many purposes due to their low energy and computational overhead, and non-repudiation and one-way properties. In RACE, hash chains are used to reduce the number of public key cryptography operations, i.e., instead of generating a signature per ACK packet to secure the payment, one signature is generated by the destination node per K ACK packets.

If a node in the route does not receive a data or ACK packet within a time interval, the session is considered stale. The node A can estimate this interval as product of {$n_a$ , (cryptographic_delay + transmission_delay)}, where $n_A$ is the number of nodes between A and the source node for data packets, and the number of nodes between A and the destination node for ACK packets. The cryptographic_delay is the maximum computation time required by a node to perform the cryptographic operations, and the transmission_delay includes any other delays, such as the propagation, queuing, and channel contention delays.

```
Algorithm: 1
1:// $n_i$ is the source, intermediate, or destination node that is running the algorithm
2: if($n_i$ is the source node) then
3:        $P_X \leftarrow [R, X, T_S, MX, Sig_S(R, X, T_S, H(MX))]$;
4:        Send($P_x$); //send x to the first node in the route
5: else
6: if ((R, X, $T_S$ are correct) and Verify($Sig_S(R, X, TS, H(M_X))$)==TRUE) then
7: if($n_i$ is an intermediate node) then
8: Relay the packet;
9: Store $Sig_S(R, X, T_S, H(M_X))$;
10: end if
11: if($n_i$ is the destination node) then
12: Send(h(X));
13: end if
14: else
15: Drop the packet;
16: Send error packet to the source node;
17: end if
18: end if
19: if ($P_X$ is last packet) then
20: Evidence = {R, X, $T_S$, H(MX), h(0), h(x),
              $H(Sig_S(R,X,T_S, H(M_X)), Sig_D(R,T_S,h(0)))$};
21: Report = { R, $T_S$, F, X};
22: Store Report and Evidence;
23: end if
```

***Evidence* composition:** *Evidence* is defined as information that is used to establish proof about the occurrence of an event or action, the time of occurrence, the parties involved in the event, and the outcome of the event. The purpose of an *Evidence* is to resolve a dispute about the amount of the payment resulted from data transmission. Fig. 4 gives the general format of an *Evidence.* The figure shows that an *Evidence* contains two main parts called DATA and PROOF. The DATA part describes the payment, i.e., who pays whom and how much, and contains the necessary data to regenerate the nodes' signatures. From Fig. 4, the DATA contains the identities of the nodes in the route (R), the number of received messages (X), the session establishment time stamp, the root of the destination node's hash chain $h^{(0)}$, the hash value of the last message ($H(M_X)$), and the last received hash value ($h^{(v)}$) where $v = x\text{-}1$.

The PROOF is an undeniable security token that can prove the correctness of the DATA and protect against payment manipulation, falsification, and repudiation. Instead of attaching the signature to compose PROOF, it is composed by hashing the destination node's signature and the last signature received from the source node, to reduce the *Evidence* size.

*Evidences* have the following main features:

1. *Evidences* are unmodifiable: If X messages are delivered, the intermediate nodes can compose *Evidences* for fewer than X messages, but not for more. This is because the intermediate nodes have $Sig_S(R; i; T_s; H(M_i))$ and $h^{(i)}$ for $i = \{1; 2;... ;X\}$, which are sufficient for composing *Evidences* for fewer than X messages. However, the intermediate nodes cannot compose *Evidences* for more than X because it is computationally infeasible to compute $Sig_S(R;i;Ts;H(Mi))$ or $h^{(i)}$ for i > X.
2. If the source and destination nodes collude, they can create *Evidences* for any number of messages because they can compute the necessary security tokens.
3. *Evidences* are unforgeable: If the source and destination nodes collude, they can create *Evidence* for sessions that did not happen, but the intermediate nodes cannot, because constructing the source and destination nodes' signatures is infeasible.
4. *Evidences* are undeniable: This is necessary to enable the TP to verify them to secure the payment. A source node cannot deny initiating a session or the amount of payment because it signs the number of transmitted messages and the signature is included in the *Evidence.*
5. An honest intermediate node can always compose

valid *Evidence* even if the route is broken or the other nodes in the route collude to manipulate the payment. This is because it can verify the *Evidences* to avoid being fooled by the attackers.

Reducing the storage area of the *Evidences* is important because they should be stored until the AC clears the payment. Onion hashing technique can be used to aggregate *Evidences.* The underlying idea is that instead of storing one PROOF per session, one compact PROOF can be computed to prove the credibility of the payment of a group of sessions. The compact *Evidence* contains the concatenation of the DATAs of the individual *Evidences* and one compact PROOF that is computed by onion hashing the PROOFs of the individual *Evidences.* Let PROOF(i) refer to the PROOF of the *Evidence* number i, the compact PROOF is computed as follows:

```
H(…..,
        H(H(PROOF(1), PROOF(2)), PROOF(3)),
                        . . . , PROOF(n) )
```

*Figure 4 The general format of an Evidence*

PROOF(1) and PROOF(2) are concatenated and hashed, and then PROOF(3) is added to the compact PROOF by adding one hashing layer and so on. The compact PROOF has the same size of the PROOF of individual Evidence, but it can prove the credibility of the payment of multiple sessions. The onion hashing technique enables the nodes to aggregate a recent Evidence with the old compact Evidence, i.e., Evidences are always stored in an aggregated form to reduce their storage area. The technique is called onion hashing because each aggregation operation requires adding one hashing layer.

However, the Evidence aggregation process is irreversible because the hash function is unidirectional, i.e., the compact Evidence cannot be decomposed to individual Evidences. Thus, if the TP requests an Evidence that is aggregated in the compact Evidence, the node has to submit the compact Evidence and the TP has to verify all the PROOFs of the sessions of the compact Evidence, instead of verifying only the PROOF of the requested Evidence. Therefore, aggregating more Evidences can further reduce their storage area, but with more communication and processing overhead if an Evidence is requested. This is acceptable because Evidences are requested only in case of cheating and RACE requests the Evidences from few nodes instead of all the nodes in the cheating reports. The aggregation level can be flexible and dependent on the available memory space, e.g., a storage-constrained node can aggregate all Evidences in only one compact Evidence.

| Session identifier | F | X |
|---|---|---|
| $ID_A$, $ID_W$, $ID_C$, $ID_Z$, $T_{S1}$ | 0 | 12 |
| $ID_C$, $ID_W$, $ID_Y$, $ID_Z$, $ID_A$, $T_{S2}$ | 1 | 17 |
| $ID_W$, $ID_Y$, $ID_A$, $ID_Z$, $T_{S3}$ | 0 | 15 |

*Table 3 Numerical Example Reports submitted by Node A*

***Payment report composition/submission:*** A payment report contains the session identifier, a flag bit (F), and the number of messages (X). The session identifier is the concatenation of the identities of the nodes in the session and the time stamp. The flag bit is zero if the last received packet is data and one if it is ACK. Table 3 gives numerical examples for the payment reports of node A. For the first report, A is the source node and claims sending 12 messages, but it did not receive the ACK of the last message because F is zero. For the second report, A is the destination node and claims receiving 17 messages. For the third report, A is an intermediate node and claims receiving 15 messages, but it did not receive the ACK of the last message. The submission of reports and Evidences are illustrated in Algorithm 2 and Fig. 5.

```
Algorithm: 2
1: ni → TP: Submit(Reports[ti-1, ti));
2: TP→ni: Evidences_Request(Ses_IDS[ti-2,ti-1));
3: ni → TP: Submit(Req_Evs[ti-2, ti-1));
4: TP: Identify_Cheaters();
5: TP: Clear the payment of the reports;
6: if(ni is honest) then
7: TP→ ni: A renewed certificate;
8: end if
```

As shown in Fig. 5, node A sends a *Report Submission Packet (RSP)* to the TP at time $t_i$ to submit the reports of the sessions held since the last contact at $t_i$_1. The packet contains the reports of the sessions held in $[t_{i-1},t_i)$ (Reports$[t_{i-1},t_i)$), time stamp, and a keyed hash value ($H_{ka}Q$) to ensure the packet's integrity and authenticity, where $K_a$ is the long-term symmetric key shared between node A and the TP. Thus, the TP can make sure that the packet has not been manipulated and the reports are indeed sent by the intended node, which is important to secure the payment and hold the nodes accountable for any misbehavior. If the TP requests *Evidences* from node A, it sends an *Evidences Request Packet (EREQ)* containing the identifiers of the reports that their *Evidences* are requested (Ses_ID$_s[t_{i-2}\ t_{i-1})$). Node A replies with *Evidences Reply*

*Packet (EREP)* containing the requested *Evidences* (Req_Ev$_S$[$t_{i-2}$, $t_{i-1}$]). If node A is honest, the TP sends a *Renewed Certificate Packet (RCP)* containing a renewed certificate for node A with the same identity and public/ private keys but with updated lifetime. Therefore, only the efficient hashing operations are used to submit the reports and *Evidences* securely to the TP.

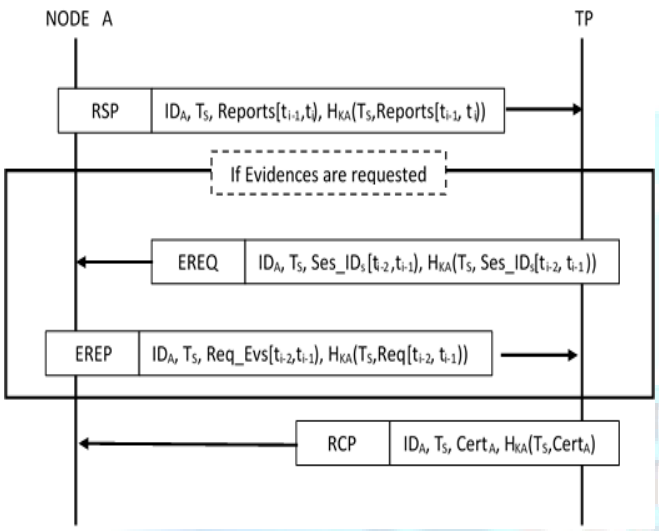
NODE A
TP
RSP
$ID_A$, $T_S$, Reports[$t_{i-1}$,$t_i$], $H_{KA}$($T_S$,Reports[$t_{i-1}$, $t_i$])
If Evidences are requested
EREQ
$ID_A$, $T_S$, Ses_ID$_S$[$t_{i-2}$,$t_{i-1}$], $H_{KA}$($T_S$, Ses_ID$_S$[$t_{i-2}$, $t_{i-1}$])
EREP
$ID_A$, $T_S$, Req_Evs[$t_{i-2}$,$t_{i-1}$], $H_{KA}$($T_S$,Req[$t_{i-2}$, $t_{i-1}$])
RCP
$ID_A$, $T_S$, Cert$_A$, $H_{KA}$($T_S$,Cert$_A$)

*Figure 5 The submission of reports and Evidences.*

## 4.2 Classifier

After receiving a session's payment reports, the AC verifies them by investigating the consistency of the reports, and classifies them into fair or cheating. For fair reports, the nodes submit correct payment reports, but for cheating reports, at least one node does not *submit the reports* or *submits incorrect reports*, e.g., to steal credits or pay less. Fair reports can be for complete or broken sessions. For a complete session, all the nodes in the session report the same number of messages and F of one.

| Case No | | S | A | B | C | D |
|---|---|---|---|---|---|---|
| 1 | X | 11 | 11 | 11 | 11 | 11 |
| | F | 1 | 1 | 1 | 1 | 1 |
| 2 | X | 11 | 11 | 11 | 11 | 11 |
| | F | 0 | 0 | 1 | 1 | 1 |
| 3 | X | 8 | 8 | 7 | 7 | 7 |
| | F | 0 | 0 | 1 | 1 | 1 |
| 4 | X | 1 | 1 | 1 | -- | -- |
| | F | 0 | 0 | 0 | -- | -- |

*Table 4 Numerical Examples for Fair Reports*

If a session is broken during relaying the Xth **DATA** packet, the reports of the nodes from S to the last node that received the packet report X and F of zero, but the other nodes report *X*-1 and F of one.

If a session is broken during relaying the Xth **ACK** packet, the nodes in the session report *X* messages, and the nodes from D to the last node that received the ACK report F of one, but the other nodes report F of zero. The reports are classified as cheating if they do not achieve one of the aforesaid rules.

Table 4 gives numerical examples for fair reports. Case 1 is reports for complete session and Cases 2 to 4 are reports for broken sessions. For Case 1, all the nodes report the same number of messages and F of one. For Case 2, the session was broken during relaying the ACK packet number 11 and B is the last node that received the packet. For Case 3, the session was broken during relaying the data packet number 8 and node A is the last node that received the packet. For Case 4, the session was broken during relaying the first data packet, and node B is the last node that received the packet, and therefore nodes C and D did not submit the payment report of the session.

## 4.3 Identifying Cheaters

As shown in Fig. 2, in the Identifying Cheaters' phase, the TP processes the cheating reports to identify the cheating nodes and correct the financial data. Our objective of securing the payment is preventing the attackers (singular of collusive) from stealing credits or paying less, i.e., the attackers should not benefit from their misbehaviors. We should also guarantee that each node will earn the correct payment even if the other nodes in the route collude to steal credits. The AC requests the Evidence only from the node that submits report with more payment instead of all the nodes in the route because it should have the necessary and undeniable proofs (signatures and hash chain elements) for identifying the cheating node(s). In this way, the AC can precisely identify the cheating nodes with requesting few Evidences. Numerical examples will be given in Section 5 to clarify how cheating nodes can be identified without false accusations.

To verify an Evidence, the TP composes the PROOF by generating the nodes' signatures and hashing them. The Evidence is valid if the computed PROOF is similar to the Evidence's PROOF.

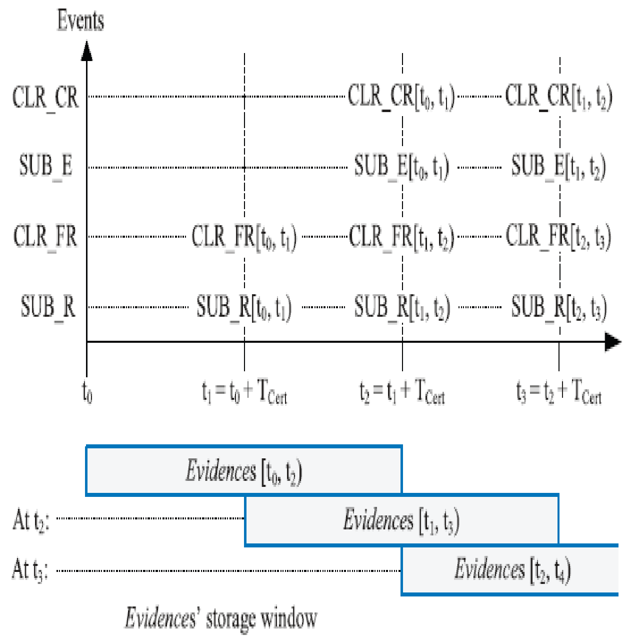

*Figure 6 The worst case timing of the reports submission and clearance*

## 4.4 Credit-Account Update

As shown in Fig. 2, the *Credit-Account Update* phase receives fair and corrected payment reports to update the nodes' credit accounts.

In receipt-based payment schemes, a receipt can be cleared once it is submitted because it carries undeniable security proof, but the AC in RACE has to wait until receiving the reports of all nodes in a route to verify the payment. The maximum payment clearance delay (or the worst case timing) occurs for the sessions that are held shortly after at least one node contacts the AC and the node submits the report after the certificate lifetime ($T_{Cert}$), i.e., at least one report is submitted after $T_{Cert}$ of the session occurrence. It is worth to note that the maximum time duration for a node's two consecutive contacts with the TP is $T_{Cert}$ to renew its certificate to be able to use the network.

Fig. 6 shows the worst case timing of the submission and clearance of the reports with considering that the reports are submitted every $T_{Cert}$, where SUB_R, SUB_E, CLR_FR, and CLR_CR are the events of submitting reports, submitting *Evidences,* clearing fair reports, and clearing cheating reports respectively. At $t_1$, the nodes submit the payment reports of the sessions held in $[t_0, t_1)$ and the fair reports of these sessions are cleared. Thus, the maximum payment clearance delay of fair reports is $T_{Cert}$ for the sessions held shortly after $t_0$, but the average payment clearance delay is $T_{Cert}/2$ for the sessions held in $[t_0,t_1)$ assuming that the sessions are held according to uniform random distribution. At $t_2$, the TP requests the *Evidences* of the cheating reports of the sessions held in $[t_0,t_1)$. Thus, the maximum payment clearance delay for cheating reports is 2 **x** $T_{Cert}$ for the sessions held shortly after $t_0$, but the average payment clearance delay is 1.5 $T_{Cert}$ for the cheating reports of the sessions held in $[t_0,t_1)$. The figure also shows that the maximum time for storing an *Evidence* is 2 **x** $T_{Cert}$, e.g., for the reports of sessions held shortly after t0. At $t_2$, the nodes delete the *Evidences* of the sessions held in $[t_0,t_1)$ because the AC must have cleared their reports.

However, the nodes submit the reports at different times because the connection to the TP may not be available on a regular basis, and thus the duration between each two submissions may not be the same and may be less than or equal to $T_{Cert}$. Hence, the maximum payment clearance delay may be less than $T_{Cert}$. $T_i$ is a continuous random variable that denotes the time duration between two submissions for a node, where Ti ϵ [0, $T_{Cert}$]. The submission durations of the nodes are independent and identically distributed (i.i.d.) random variables. In order to estimate the average and maximum payment clearance delay, we consider two models.

For model I, each node contacts the TP when it accumulates a large number of reports or when the remaining time of its certificate's lifetime is short, to reduce the communication overhead. We model this behavior with truncated exponential distribution given in (1), where the probability that a node contacts the TP is high as $T_i$ approaches $T_{Cert}$.

For model II, a node submits the reports once it has a connection to the TP and the connections are uniformly distributed over the time interval [0, $T_{Cert}$].

## 5. Analysis

Our security objective is preventing an attacker or even a group of colluding attackers from achieving gains such as *stealing credits* or *paying less*. The signatures of the source node can ensure the messages integrity and authenticity and secure the payment. Signatures and hash chains have nonrepudiation property because it is computationally infeasible to compute a node's signature without knowing the private key used in generating the signature and to compute $h^{(i)}$ from $h^{(i-1)}$. This nonrepudiation property is used to secure the payment by enabling the nodes to compose valid *Evidences* and enabling the TP to verify the

*Evidences* to identify the cheating nodes. In order to evaluate the *Identifying Cheaters'* phase, numerical examples for cheating reports are given in Table 5.

In Cases 1 and 2, the reports of the intermediate and destination nodes are consistent but the source node claims sending fewer number of messages. The source node can compose valid *Evidence* if it cheats because it has the tokens $Sig_s(R,6,T_s,H(M_6))$ and $h^{(1)}$ and thus it is ineffective to request the *Evidence* from the source node. The TP can request the *Evidence* from an intermediate node or the destination node. The source node is a cheater if the *Evidence* is correct because the *Evidence* cannot be composed without the source node's signature for 10 messages, and the intermediate and destination nodes cannot compute this signature.

For Case 2, it is obvious that the route was broken at node B during relaying the data packet number ten. For Case 3, the source and destination nodes reports are consistent but the intermediate nodes claim relaying more messages. If an intermediate node submits valid *Evidence,* the source and destination nodes are cheaters because the *Evidence* should contain the source node's signature for 12 messages and $h^{(11)}$ or $h^{(12)}$. It is ineffective to request the Evidence from the source or the destination node because they can collude to generate valid Evidence.

In Case 4, the reports of the intermediate and destination nodes are consistent but the source node claims sending more messages. This case may be rare because the rational attackers will attempt to steal credits or pay less. The TP can clear the payment according to the nodes' reports without requesting Evidences to achieve our security strategy and discourage submitting incorrect reports because the source node pays more if it lies and the other nodes lose credits if they lie. However, the TP can identify the cheating nodes by requesting the Evidence from the source node because it should contain $h^{(12)}$ or $h^{(11)}$.

If the Evidence is correct, the destination node is evicted but the intermediate nodes should not be evicted because eight messages may be indeed relayed in the session and the source and destination nodes collude to falsely accuse the intermediate nodes.

Case 5 is similar to Case 4 but the source and destination nodes report the same number of messages. The payment is cleared according to the nodes' reports to punish the nodes that submit incorrect reports without stealing credits.

| Case No | | S | A | B | C | D |
|---|---|---|---|---|---|---|
| 1 | X | 6 | 10 | 10 | 10 | 10 |
| | F | 1/0 | 1 | 1 | 1 | 1 |
| 2 | X | 6 | 10 | 10 | 9 | 9 |
| | F | 1/0 | 0 | 0 | 1 | 1 |
| 3 | X | 5 | 12 | 12 | 12 | 5 |
| | F | 1 | 1/0 | 1/0 | 1/0 | 1 |
| 4 | X | 12 | 8 | 8 | 8 | 8 |
| | F | 1/0 | 1 | 1 | 1 | 1 |
| 5 | X | 9 | 4 | 4 | 4 | 9 |
| | F | 1/0 | 1 | 1 | 1 | 1/0 |
| 6 | X | 14 | 14 | 22 | 14 | 14 |
| | F | 1 | 1 | 1/0 | 1 | 1 |
| 7 | X | 7 | 7 | 7 | 7 | 6 |
| | F | 0 | 0 | 1 | 0 | 1 |
| 8 | X | 7 | 7 | 7 | 7 | 7 |
| | F | 0 | 0 | 1 | 0 | 1 |
| 9 | X | -- | 4 | -- | -- | -- |
| | F | -- | 1/0 | -- | -- | -- |
| 10 | X | 6 | -- | -- | -- | 6 |
| | F | 1/0 | -- | -- | -- | 1/0 |

*Table 5 Numerical Examples for Cheating Reports*

In Cases 6 and 7, node B can prove the credibility of its reports and earn the deserved payment even if the other nodes in the session collude. For Case 7, node B claims delivering seven messages but the other nodes claim receiving seven messages and delivering only six messages. If node B is honest, its Evidence should have $h^{(7)}$. If the Evidences of node B are valid, the source and destination nodes are cheaters in Case 6, but only the destination node is a cheater in Case 7. For Case 8, as long as the destination node acknowledges receiving the message number seven, the intermediate nodes are rewarded for seven messages. In Cases 9 and 10, "–" means that the node does not submit the payment report of the session. For Case 9, if node A submits valid *Evidence,* the source and destination nodes are cheaters because they established a session but did not submit the payment reports. The nodes B and C are not rewarded to discourage un-submitting the payment reports. For Case 10, the source node is charged but the intermediate nodes are not rewarded without requesting *Evidences* in order to punish the nodes that do not submit payment reports.

The attackers may launch *Multiple-Redemptions and One-Redemption* attacks to deceive the AC. For *Multiple-Redemptions* attack, the attackers submit the same payment reports multiple times to be rewarded several times for the same sessions. The AC can thwart the attack and identify the attackers because it can ensure whether the payment of a session has been cleared before using the session unique identifier that includes the identities of the nodes in the session and time stamp. For *One-Redemption* attack, the attackers attempt to make the payment reports of different sessions have the same identifier to pay once because the AC clears the reports having the same identifier once. This attack is not possible in RACE because a session's identifier changes once the route or the time changes, i.e., the reports are different even if the same nodes participate in different sessions at different times.

*Evidence Forgery and Manipulation* attacks target the *Evidence* composition process. For *Evidence-Forgery* attack, the attackers attempt to forge *Evidences* for sessions that did not happen to steal credits, and for *Evidence-Manipulation* attack, the attackers attempt to manipulate valid *Evidences* to increase their rewards. In RACE, *Evidences* are undeniable, unforgeable, and unmodifiable. The source node cannot deny initiating a session and the amount of payment because its signature is included in the *Evidence.* The attackers cannot forge *Evidences* or manipulate them with using secure hash function and public-key cryptosystem because it is impossible to compute $h^i$ from $h^{i-1}$ or compute the nodes' signatures without knowing the private keys. Moreover, it is also impossible to modify the source nodes' signatures, compute the private keys from the public ones, and compute the hash value of the signatures without computing the signatures. The TP can identify the attackers that forge *Evidences* because the *Evidences'* verifications fail.

The attackers may launch *Impersonation, Packet-Replay, and Free-Riding* attacks to target route establishment, report submission, and data transmission processes. For *Impersonation* attack, the attackers impersonate legitimate nodes to communicate freely or steal credits. This attack is impossible because the nodes use their private keys in signing the packets and use their secret symmetric keys in submitting the payment reports. In *Packet-Replay* attack, the attackers record valid packets and replay them in different place and/or time to establish sessions under the name of others to communicate freely. In RACE, stale packets cannot be used to establish sessions because time stamps are used to verify the freshness of the packets. For *Free-Riding* attack, two colluding intermediate nodes in a legitimate session attempt to communicate freely by manipulating the packets to add their data. This is impossible in RACE because the integrity of the packets can be verified at each node, and thus the first intermediate node after the attacker can detect any addition or modification to the packets and thwart the attack by dropping them. The integrity of the data packets can be ensured by verifying the source nodes' signatures, and the integrity of the ACK packets can be ensured by verifying the hash chain elements.

The attackers may attempt to manipulate their reports to launch *Reduced-Payment* and *False-Accusation* attacks. For *Reduced-Payment* attack, some intermediate nodes may collude with the source node to submit reports with less payment to charge the source node less. For example, if ***p*** intermediate nodes launch this attack successfully in a session with n nodes, the colluders can save $((n-2-p)*(\mathbf{X}-\omega) \mathbf{x}\ \lambda)$ credits, where X and ω are the correct and the submitted number of messages, respectively, and thus the source node can compensate the colluding intermediate nodes. In RACE, even if a group of nodes colludes to reduce the rewards of an honest node, the honest node can compose valid *Evidence* and earn the correct payment, such as Cases 6 and 7 in Table 5. For *False-Accusation* attack, the attackers manipulate their reports to insert nonexistent nodes in a session to let the TP accuse them of not reporting the session. In RACE, if the victim nodes are intermediate, the payment is cleared without punishing or rewarding them as discussed in Cases 9 and 10 in Table 5, but if the victim nodes are source or destination, the attackers are evicted because they cannot submit correct *Evidences.*

The charging and rewarding policy can counteract rational cheating actions. If the nodes are charged only for the successfully delivered messages, the destination nodes may collude with the source nodes to not send ACK packets so as not to pay. To prevent this, the source nodes are charged for undelivered messages. If the intermediate nodes are rewarded for the relayed messages that do not reach the destination, the colluding intermediate nodes can increase their rewards with consuming low resources by relaying only the smaller size signatures but not the messages to compose valid *Evidences* and claim relaying the messages. To prevent this, the intermediate nodes are rewarded only for delivered messages.

## 6. Conclusion

In this paper, we have proposed RACE, a report-based payment scheme for MWNs. The nodes submit lightweight payment reports containing the alleged charges and rewards (without proofs), and temporarily store undeniable security tokens called Evidences. The fair reports can be cleared with almost no cryptographic operations or processing overhead, and Evidences are submitted and processed only in case of cheating reports in order to identify the cheating nodes. Our analytical and simulation results demonstrate that RACE can significantly reduce the communication and processing overhead comparing to the existing receipt-based payment schemes with acceptable payment clearance delay and Evidences' storage area, which is necessary for the effective implementation of the scheme. Moreover, RACE can secure the payment, and identify the cheating nodes precisely and rapidly without false accusations or missed detections.
In RACE, the AC can process the payment reports to know the number of relayed/dropped messages by each node. In our future work, we will develop a trust system based on processing the payment reports to maintain a trust value for each node. The nodes that relay messages more successfully will have higher trust values, such as the low-mobility and the large-hardware-resources nodes. Based on these trust values, we will propose a trust-based routing protocol to route messages through the highly trusted nodes (which performed packet relay more successfully in the past) to minimize the probability of dropping the messages, and thus improve the network performance in terms of throughput and packet delivery ratio. However, the trust system should be secure against singular and collusive attacks, and the routing protocol should make smart decisions regarding node selection with low overhead.

## References

[1] G. Shen, J. Liu, D. Wang, J. Wang, and S. Jin, "Multi-Hop Relay for Next-Generation Wireless Access Networks," Bell Labs Technical J., vol. 13, no. 4, pp. 175-193, 2009.
[2] C. Chou, D. Wei, C. Kuo, and K. Naik, "An Efficient Anonymous Communication Protocol for Peer-to-Peer Applications Over Mobile Ad-Hoc Networks," IEEE J. Selected Areas in Comm., vol. 25, no. 1, pp. 192-203, Jan. 2007.
[3] H. Gharavi, "Multichannel Mobile Ad Hoc Links for Multimedia Communications," Proc. IEEE, vol. 96, no. 1, pp. 77-96, Jan. 2008.
[4] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," Proc. MobiCom '00, pp. 255-265, Aug. 2000.
[5] G. Marias, P. Georgiadis, D. Flitzanis, and K. Mandalas, "Cooperation Enforcement Schemes for MANETs: A Survey," Wiley's J. Wireless Comm. and Mobile Computing, vol. 6, no. 3, pp. 319-332, 2006.
[6] Y. Zhang and Y. Fang, "A Secure Authentication and Billing Architecture for Wireless Mesh Networks," ACM Wireless Networks, vol. 13, no. 5, pp. 663-678, Oct. 2007.
[7] L. Buttyan and J. Hubaux, "Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks," Mobile Networks and Applications, vol. 8, no. 5, pp. 579-592, Oct. 2004.
[8] Y. Zhang, W. Lou, and Y. Fang, "A Secure Incentive Protocol for Mobile Ad Hoc Networks," ACM Wireless Networks, vol. 13, no. 5, pp. 569-582, Oct. 2007.
[9] A. Weyland, "Cooperation and Accounting in Multi-Hop Cellular Networks," PhD thesis, Univ. of Bern, Nov. 2005.
[10] A. Weyland, T. Staub, and T. Braun, "Comparison of Motivation- Based Cooperation Mechanisms for Hybrid Wireless Networks," J. Computer Comm., vol. 29, pp. 2661-2670, 2006.
[11] S. Zhong, J. Chen, and R. Yang, "Sprite: A Simple, Cheat-Proof, Credit Based System for Mobile Ad-Hoc Networks," Proc. IEEE INFOCOM '03, vol. 3, pp. 1987-1997, Mar./Apr. 2003.
[12] M. Mahmoud and X. Shen, "FESCIM: Fair, Efficient, and Secure Cooperation Incentive Mechanism for Hybrid Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 11, no. 5, pp. 753-766, May 2012.
[13] M. Mahmoud and X. Shen, "PIS: A Practical Incentive System for Multi-Hop Wireless Networks," IEEE Trans. Vehicular Technology,
vol. 59, no. 8, pp. 4012-4025, Oct. 2010.
[14] M. Mahmoud and X. Shen, "Stimulating Cooperation in Multihop Wireless Networks Using Cheating Detection System," Proc. IEEE INFOCOM '10, Mar. 2010.
[15] N. Salem, L. Buttyan, J. Hubaux, and M. Jakobsson, "Node Cooperation in Hybrid Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 5, no. 4, pp. 365-376, Apr. 2006.
[16] J. Pan, L. Cai, X. Shen, and J. Mark, "Identity-Based Secure Collaboration in Wireless Ad Hoc Networks," Computer Networks, vol. 51, no. 3, pp. 853-865, 2007.
[17] M. Mahmoud and X. Shen, "ESIP: Secure Incentive Protocol with Limited Use of Public-Key Cryptography for Multi-Hop Wireless Networks," IEEE Trans. Mobile Computing, vol. 10, no. 7, pp. 997- 1010, July 2011.
[18] M. Mahmoud and X. Shen, "An Integrated Stimulation and Punishment Mechanism for Thwarting Packet Drop in Multihop Wireless Networks," IEEE Trans. Vehicular Technology, vol. 60, no. 8, pp. 3947-3962, Oct. 2011.
[19] H. Zhu, X. Lin, R. Lu, Y. Fan, and X. Shen, "SMART: A Secure Multilayer Credit Based Incentive Scheme for Delay-Tolerant Networks," IEEE Trans. Vehicular Technology, vol. 58, no. 8, pp. 4628-4639, Oct. 2009.
[20] R. Lu, X. Lin, H. Zhu, X. Shen, and B.R. Preiss, "Pi: A Practical Incentive Protocol for Delay Tolerant Networks,"

IEEE Trans. Wireless Comm., vol. 9, no. 4, pp. 1483-1493, Apr. 2010.

[21] B. Wehbi, A. Laouiti, and A. Cavalli, "Efficient Time Synchronization Mechanism for Wireless Multi Hop Networks," Proc. IEEE Personal, Indoor and Mobile Radio Comm. (PIMRC), 2008. [22] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," Mobile Computing, Chapter 5, pp. 153-181, Kluwer Academic Publishers, 1996.

[23] L. Anderegg and S. Eidenbenz, "Ad Hoc-VCG: A Trustful and Cost-Efficient Routing Protocol for Mobile Ad Hoc Networks with Selfish Agents," Proc. ACM MobiCom, Sept. 2003.

[24] H. Pagnia and F. Gartner, "On the Impossibility of Fair Exchange Without a Trusted Third Party," Technical Report TUD-BS-1999- 02, Darmstadt Univ. of Technology, Mar. 1999.

[25] K. Sanzgiri, D. LaFlamme, B. Dahill, B. Levine, C. Shields, and E. Belding-Royer, "Authenticated Routing for Ad Hoc Networks," IEEE Selected Areas in Comm., vol. 23, no. 3, pp. 598- 610, Mar. 2005.

[26] Y. Hu, A. Perrig, and D. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," Proc. ACM MobiCom, Sept. 2002.

[27] B. Wu, J. Chen, J. Wu, and M. Cardei, "A Survey of Attacks and Countermeasures in Mobile Ad Hoc Networks," Wireless Network
Security, Springer Network Theory and Applications, vol. 17, pp 103- 135, 2007.

[28] S. Even, O. Goldreich, and S. Micali, "On-Line/off-line Digital Signatures," Crypto '89: Proc. Advances in Cryptology, pp. 263-277, 1990.

[29] O. Nibouche, M. Nibouche, A. Bouridane, and A. Belatreche, "Fast Architectures for FPGA-Based Implementation of RSA Encryption Algorithm," Proc. IEEE Field-Programmable Technology Conf., Dec. 2004.