A Lightweight Anonymous Routing Protocol without Public Key En/Decryptions for Wireless Ad Hoc Networks

Chun-Ta Li [†] Min-Shiang Hwang[‡]

Department of Information Management[†]
Tainan University of Technology
529 Jhong Jheng Road, Yongkang, 710 Tainan, Taiwan, R.O.C.
E-mail: th0040@mail.tut.edu.tw

Department of Management Information Systems[‡]
National Chung Hsing University
250 Kuo Kuang Road, Taichung, Taiwan 402, R.O.C.
E-mail: mshwang@nchu.edu.tw

November 1, 2010

[‡]Responsible for correspondence: Prof. Min-Shiang Hwang.

A Lightweight Anonymous Routing Protocol Without Public Key En/Decryptions for Wireless Ad Hoc Networks

Abstract

More attention should be paid to anonymous routing protocols in secure wireless ad hoc networks. However, as far as we know, only a few papers on secure routing protocols have addressed both issues of anonymity and efficiency. Most recent protocols adopted Public Key Infrastructure (PKI) solutions to ensure the anonymity and security of route constructing mechanisms. Since PKI solution requires a huge and expensive infrastructure with complex computations and the resource constraints of small ad hoc devices; a Two-layer Authentication Protocol with Anonymous Routing (TAPAR) is proposed in this paper. TAPAR does not adopt public key computations to provide secure and anonymous communications between source and destination nodes over wireless ad hoc networks. Moreover, TAPAR accomplishes mutual authentication, session key agreement, and forward secrecy among communicating nodes; along with integration of non-PKI techniques into the routing protocol allowing the source node to anonymously interact with the destination node through a number of intermediate nodes. Without adopting PKI en/decryptions, our proposed TAPAR can be efficiently implemented on small ad hoc devices while at least reducing the computational overhead of participating nodes in TAPAR by 21.75%. Our protocol is certainly favorable when compared with other related protocols.

Keywords: Anonymous routing; Mutual authentication; PKI solution; Information security; Wireless ad hoc networks.

1 Introduction

Due to significant advances in wireless and mobile communication techniques and the broad development of potential applications, wireless ad hoc networks have attracted great attention in recent years. Many experiments involving various scenarios regarding wireless ad hoc networks have been conducted. Examples include: Vehicular Ad hoc Networks (VANETs) [2, 25], Wireless Sensor Networks (WSNs) [26, 37], and Mobile Ad hoc Networks (MANETs) [27] etc. Generally, ad hoc networks can be quickly deployed to designated areas while consists of available nodes that two of the nodes can directly communicate with each other through wireless medium, as long as they are within the wireless radio communication range of each other without any fixed infrastructure. Therefore, participating nodes must cooperate with each other and serve as both router and host simultaneously [1, 32, 47].

Routing protocols in wireless ad hoc networks, such as AODV [15, 46], DSR [12], ZRP [13], and IERP [8] are generally classified into two categories: reactive and proactive protocols. In reactive protocols (also called source-initiated on-demand driven protocols), routing information does not need to be periodically maintained and related routing information is initiated whenever data packets need to be sent. On the other hand, in proactive protocols (also called table-driven protocols), participating nodes are required to periodically maintain, update, and store routing information in their own routing tables, even if no data packets need to be sent. In general, most recent ad hoc routing protocols rely on implicit trust-your-neighbor relations to route data packets among participating nodes, but this untrustworthy model may incur security attacks that could damage a wireless ad hoc network. Therefore, a method of securing route packets in ad hoc networks must be developed. In order to achieve secure routing, various secure routing protocols have been proposed, such as SRP [38], SEAD [10], SMT [39], TLR [43], and DIMH [48], etc.

Unlike traditionally wired networks that are protected by several lines of defense such as basic infrastructure, centralized administration, and firewalls, security threats on such wireless environments could come from any direction and target all participating nodes [11, 16]. Therefore, wireless ad hoc networks are susceptible to hackers ranging from passive eavesdropping to active tampering, interfering, and spamming due to the absence of fixed infrastructure and centralized administration. A number of security threats to wireless ad hoc networks have been addressed [4, 5, 14, 19, 26, 33, 36, 40, 41, 42, 44, 45] and can be generally classified into two types: passive attacks and active attacks. In passive attacks (such as *eavesdropping* attacks), a hacker can un-intrusively monitor on the communication channel between two communicating nodes to collect and discover valuable information without disturbing the communication [7, 18, 19, 20, 21, 22, 23, 24, 28, 29, 30, 31]. On the other hand, active attacks (such as *Sybil* attacks, *replay* attacks, *guessing* attacks, and *modification* attacks) can be further classified into two categories: external attacks and internal attacks. In external attacks, a hacker does not belong to an ad hoc network and he/she can first eavesdrop on messages sent or received by normal participating nodes for the eventual purpose of malicious tempering, interfering, guessing, or spamming, then finally injects false messages to disrupt the network functionalities. Internal attacks (such as *node replication* attacks and *node compromised* attacks) are usually caused by compromised members who belong to the ad hoc network in question, hence internal attacks are more difficult to safeguard against than external attacks.

Additionally, the main challenge facing wireless ad hoc networks is about user privacy. Whenever source nodes attempt to send confidential information to a designated node, they should maintain user privacy and avoid disclosing information about the identity, whereabouts, or behavior of the user. This process is particularly crucial in environments in which privacy is of vital

importance. It is important for the mechanisms to maintain confidentiality and privacy in ad hoc routing protocols to resist various security intrusions while providing anonymous interactions between two communication nodes. To date, only a small number of papers focused on the subject of wireless ad hoc network have addressed the secure routing protocols and anonymity issues [3, 9, 34, 35, 49]. In fact, most protocols have adopted PKI solutions to ensure the anonymity and security of route constructing mechanisms. Due to hardware constraints, an ad hoc node is usually built as a small device which has limited power and computational ability. Motivated by the above-mentioned situations, in this paper, a lightweight and secure anonymous routing protocol for wireless ad hoc networks without public key en/decryptions is proposed. Besides preventing the above-mentioned security attacks, the proposed protocol features several notable properties as follows:

Anonymous routing: The proposed protocol integrates the property of anonymity into a reactive routing algorithm, hence it allows anonymous interactions between source and destination node.

Session key agreement: Two communicating nodes both partially contribute to security so both of them can agree on a session key based on their own contributions. Thus, a session key can be used to secure future communications between them.

Mutual authentication: The proposed protocol can achieve a mutual authentication between the two communicating nodes; and not only can the destination node authenticate the legal source node, but the source node can also authenticate the legal destination node.

Forward secrecy: Even if an attacker has the ability to compromise a node to obtain the partial contributions to session keys from previous sessions, she/he would still be unable to derive the correct session keys from them

and this is so-called forward secrecy.

Group secrecy: All trustworthy members in an ad hoc network must authenticate all the packets they received before forwarding the data. Hence, the proposed protocol can prevent security threats from external attacks. A simple solution for achieving this goal is to employ a network-wide group key shared by all participating members.

Efficiency: All involved nodes in a specific route are not required to perform heavy operations to authenticate received route packets from their neighbors, and the proposed protocol only requires a few inexpensive operations for subsequent traffic authentications.

In summary, our proposed protocol has two main advantages compared with other related works. One advantage is that our protocol allows communications between source and destination node to proceed anonymously, and the other advantage is that we presented the alternative solution to public key solutions in a simpler way with respect to energy and bandwidth savings while reducing the complexity of the computational operations. To the best of our knowledge, none of the recently proposed secure anonymous routing protocols include non-PKI solutions for authenticating route packets. As a result, we attempted to keep our protocol free of PKI operations to demonstrate that it is important to provide a lightweight anonymous routing protocol for wireless ad hoc networks.

The rest of this paper is organized as follows: In Section 2, we will review some of the previous and related anonymous routing protocols for ad hoc networks. In Section 3 we will present our two-layer authentication protocol with anonymous routing (TAPAR). In Section 4, security and performance analyzes are presented; followed by our conclusion in Section 5.

2 Overview of Related Anonymous Routing Protocols

If only two communication nodes exchange confidential information through insecure networks and nodes, even though a network might be otherwise secure, outsiders can discover which node is the source and which node is the destination. Many researches have been invested [3, 35] in the anonymity and authentication issues in wireless ad hoc communications.

One simple solution that provides anonymity is to adopt public key cryptography into the routing procedure. Recently, two anonymous routing protocols have been presented that utilize public key solutions for wireless ad hoc networks. In 2005, Boukerche et al. [3] proposed an anonymous routing protocol for wireless ad hoc networks, basing the security of their protocol on Onion En/Decryption Routing. In addition, a similar idea was also presented for the design of anonymous routing of ad hoc networks, based on designated verifier signatures (DVS) and public key cryptography, in 2007 by Lu et al. [35]. They proposed a Secure Anonymous Routing Protocol with Authenticated Key Exchange (SARPAKE) to provide not only anonymity between two communication nodes but also to establish session keys over ad hoc networks. In the following subsection, a brief review of the two protocols is provided.

2.1 A Review of Boukerche et al.'s Protocol

Boukerche et al.'s protocol consists of three phases: path discovery phase, path reverse phase, and data transfer phase. The main task of the first phase is to elucidate and establish a path from a source node N_0 to a destination node N_n while preventing other nodes from discovering the identity of either source and destination nodes. In the second phase, the destination node triggers the path reverse phase. Finally, when the source node receives an acknowledgement from a path reverse message, the node could then use the shared session keys

of the intermediate nodes to encrypt the confidential data until data reaches the destination node. We will briefly describe the phases of their protocol as follows:

- Path Discovery Phase: At the outset, the source node N_0 generates a temporary one-time pubic key TPK and its corresponding private key TSK and then performs the following steps:

  Step 1: Broadcasts the messages $M_0 = \{TYPE, TPK,\ E_{PK_n}(N_n, K_0),\ E_{K_0}(N_0, PK_0,\ TPK,\ TSK,\ SN_{N_0},\ E_{SK_0}(M_0))\}$ to its neighbors, where $TYPE$ is the message type, E_{PK_n} signifies public key encryption with N_n's public key, K_0 is a symmetric key generated by N_0, SN_{N_0} is a random number generated for the route, and $E_{SK_0}(M)$ signifies the signature of message M which is generated and signed by N_0's private key SK_0.

  Step 2: When an intermediate node receives the packet, it attempts to decrypt $E_{PK_n}(N_n, K_0)$ with its private key. If this node is not the targeted destination node, the decrypted result is meaningless and the node forwards the message $M_i = \{TYPE,\ TPK,\ E_{PK_n}(N_n, K_0), E_{K_0}(N_0, PK_0, TPK, TSK, SN_{N_0}, E_{SK_0}(M_0)), E_{TPK}[N_i,\ K_i,\ SN_{N_i},\ E_{SK_i}(M_i)]\}$ to its neighbors within its wireless transmission range, where $i = 1, 2, \cdots, n-1$ and $E_{TPK}[\cdot]$ signifies symmetric encryption with key TPK.

  Step 3: If the node is the targeted destination node, it decrypts (N_n, K_0) with its private key and receives the common key K_0. Then, N_n decrypts $(N_0, PK_0, TPK, TSK, SN_{N_0}, E_{SK_0}(M_0))$ with common key K_0 and receives the temporary private key TSK. Finally, N_n can use TSK to retrieve session keys K_i for all intermediate nodes along the path and it sends the path reverse message to the source node through the intended intermediate nodes along the reverse path.

- Path Reverse Phase: In this phase, N_n computes the message $M_0' = (E_{K_0}[SN_{N_1}, K_1, SN_{N_2}, K_2, \cdots, K_{n-1}, SN_{N_n}], SN_{N_0})$ and sends

  $$(E_{K_{n-1}}[E_{K_{n-2}}[\cdots(E_{K_2}[(E_{K_1}[M_0'], SN_{N_1})], SN_{N_2}), \cdots], SN_{N_{n-2}}], SN_{N_{n-1}})$$

  to its ancestor node N_{n-1}. Then, each intermediate node N_{n-i} accords $SN_{N_{n-i}}$ to retrieve the key K_{n-i} to decrypt its own encryption layer, and forwards this message to the ancestor node along the reverse path until it reaches the source node, where $i = 1, 2, \cdots, n-1$. Finally, when the source node receives the path reverse message, N_0 decrypts $(SN_{N_1}, K_1, SN_{N_2}, K_2, \cdots, K_{n-1}, SN_{N_n})$ with key K_0 and knows the complete route and all shared session keys from itself to the destination node.

- Data Transfer Phase: When N_0 wants to send a confidential message to N_n, it uses the shared session keys of the intermediate nodes to generate a layer of encryptions as in the Onion Routing protocol and sends the message to its successor node. Then, each intermediate node decrypts one encryption layer and forwards the message to its successor node until the message reaches the destination node.

2.2 A Review of Lu et al.'s Protocol

Lu et al.'s protocol also consists of three phases: path discovery phase, path reverse phase, and data transfer phase. The main task of the first phase is to ensure that the source node establishes an anonymous route from itself to the destination node through a number of intermediate nodes. The second phase mainly concerns with confirming the completeness of the reverse path from the destination to the source and establishes a common session key between end-to-end communication nodes. Finally, the third phase primarily ensures that the source node sends the confidential data to the destination by using a session key. Lu et al.'s SARPAKE protocol can briefly be summarized as follows.

- Path Discovery Phase: In this phase, the source node N_0 is initiated to establish a route to the destination node N_n through a number of intermediate ad hoc nodes which are passed from N_1 to N_{n-1}. First, N_0 generates a unique tag# for this route, and then it performs the following steps:

  Step 1: Chooses a random number x to compute $X = g^x \bmod p$ and $h_0 = H(X||K_0^n)$, where g is the primitive element in $GF(p)$, $H(.)$ is a one-way hash function, $Y_n = g^{x_n}(\mathrm{mod}p)$ is the destination's public key, x_n is its private key, and $K_0^n = Y_n^{x_0} = g^{x_0 x_n}$.

  Step 2: Computes $C_0 = E_{pk_n}(M_0)$, where M_0=(tag#$||N_0||N_n||X||h_0$).

  Step 3: Sends the path discovery packet to its neighboring nodes, where $packet$=(tag#$||hop||C_0$) and hop signifies the number of hops that a packet can send.

  Step 4: Maintains its local route table $LRT_0 = (tag\#,\ 0,\ ?,\ T_0)$. The second field records the ancestor node and N_0 is the source of this route, this field records 0. The third field records the successor node, and is added later during the path reverse phase. The fourth field T_0 is the time for the route generated by N_0.

  When a node receives the packet, it performs the following steps:

  Step 5: Checks if $(((hop--) \geq 0)$, if it holds, continue; otherwise, stop.

  Step 6: Checks if the packet has already been received from other nodes within its wireless transmission range using the unique tag# as the unique identifier for this route. If it holds, drop it and stop; otherwise, continue.

  Step 7: Checks if the node itself is the designated receiver. (Try to decrypt C_0 with the private key of the node and compare the tag# and the N_n to the node's ID. There are two possible outcomes:

a. If the node is NOT the intended receiver, then it maintains its local route table LRT$_i$=(tag#,N_{i-1},?,T_i) and forwards the new packet (tag#$||hop||C_0$) to neighboring nodes, where hop has been updated.

b. If the node is the destined receiver, then it verifies whether $h_0 \stackrel{?}{=} H(X||K_n^0)$ is true or not, where $K_n^0 = y_0^{x_n} = g^{x_0 x_n} = y_n^{x_0} = K_0^n$. If it does not hold, drop it and stop; otherwise, N_n stores the entry (tag#,N_{n-1},1,T_n) to its local route table LRT$_n$ and the path discovery phase ends.

- Path Reverse Phase: In this phase, the destination node would respond to the source node and performs the following steps:

Step 1: Chooses a random number y to compute $Y = g^y$ and $h_n = H(Y||K_n^0)$. Then, the shared session key can be generated by computing $SK = (X * Y_0)^{y+x_n} = g^{(x+x_0)(y+x_n)}$.

Step 2: Computes $C_n = E_{pk_0}(M_n)$ and $C_{n-1} = E_{pk_{n-1}}(\text{tag\#})$ and sends $(C_n||C_{n-1})$ to N_{n-1}, where $M_n = (tag\# \ ||N_0 \ ||N_n \ ||Y \ ||h_n)$ and $E_{pk_i}(.)$ signifies asymmetric encryption with node i's public key.

Upon receiving $(C_n||C_{n-1})$ from N_n, N_{n-1} uses its private key sk_{n-1} to recover $tag\#$ by computing $D_{sk_{n-1}}(C_{n-1})$ and maintains its local route table $LRT_{n-1} = (tag\#,\ N_{n-2},\ N_n,\ T_{n-1})$, where $D_{sk_i}(.)$ signifies the asymmetric decryption with node i's private key. Then, N_{n-1} forwards $(C_n||C_{n-2})$ to its ancestor node N_{n-1}, where $C_{n-2} = E_{pk_{n-2}}(\text{tag\#})$. Other nodes N_1, N_2, $\cdots$, N_{n-2} along the route perform the same operations as the node N_{n-1}. Finally, when the source node N_0 receives $(C_n||C_0 = E_{pk_0}(\text{tag\#}))$ from N_1, it performs the following steps:

Step 3: Computes $D_{sk_0}(C_0)$ to recover $tag\#$ and check whether tag# is in its route table LRT_0 or not. If it is found, continue; otherwise,

stop.

Step 4: Updates its route table $LRT_0 = (tag\#, 0, N_1, T_0)$ and computes $D_{sk_0}(C_n)$ to recover $(\text{tag}\#||N_0||N_n||Y||h_n)$.

Step 5: Verifies whether $h_n \stackrel{?}{=} H(Y||K_n^0)$ is true or not. If it holds, continue; otherwise, stop.

Step 6: Computes the shared session key $SK = (Y * Y_n)^{x+x_0} = g^{(x+x_0)(y+x_n)}$.

Step 7: Finally, the route from the source node to the destination node is established and the shared session key between N_0 and N_n is also established.

- Data Transfer Phase: In this phase, the source node N_0 sends a confidential message M to the destination node N_n securely. First, N_0 computes $C = E_{SK}[M]$ and $h = H(C)$, where $E_{SK}[.]$ signifies symmetric encryption with session key SK. Then, N_0 sends $(C||CH)$ to its successor node N_1, where $CH = E_{pk_1}(\text{tag}\#||h)$. Upon receiving $(C||CH)$ from N_0, N_1 computes $D_{sk_1}(CH)$ to recover tag# and h. If tag# is found in LRT_1, N_1 computes $CH = E_{pk_2}(\text{tag}\#||h)$ and forwards $(C||CH)$ to its successor node N_2. Other nodes N_2, N_3, $\cdots$, N_{n-1} along the route perform the same operation as the node N_1. Finally, when the destination node N_n receives $(C||CH = E_{pk_n}(\text{tag}\#||h))$ from N_{n-1}, it computes $D_{sk_n}(CH)$ to recover tag# and h and verifies whether $h \stackrel{?}{=} H(C)$ holds or not. If it holds, N_n uses the shared session key SK to recover the message M from C; otherwise, stop.

3 The Proposed Protocol

In our proposed protocol, a sample of system architecture for anonymous routing over a wireless ad hoc network is given in Figure 1. In general, a routing protocol consists of three types of participants, namely: source node N_0, des-

tination node N_n, and intermediate nodes N_i, where $i = 1$ *to* 5 in this sample. The source node initiates a path discovery phase from itself to the destination node by sending a $PathDiscovery$ message through a number of intermediate nodes (depicted in real line). Then, the destination node responds with a $PathReverse$ message in the reverse direction until the message reaches the source node, verifying the complete route (depicted by dotted line). In addition, all messages transmitted between nodes should be verified and protected in the protocol, and thus even if an attacker eavesdrops on the communications between nodes or injects false messages into networks, the protocol still provides an adequate level of security. Furthermore, the proposed lightweight protocol also took into account the resource constraints of mobile ad hoc nodes by minimizing computational overhead without resorting to PKI solutions. We have largely succeeded in ensuring our protocol of maintaining low computational overhead, as seen in a detailed description of our proposed TAPAR protocol.

3.1 Notations and Assumptions

Before presenting the details of our TAPAR protocol, we must outline some basic notations and assumptions, as seen in Tables 1 and 2, respectively.

3.2 A Two-Layer Authentication Protocol with Anonymous Routing (TAPAR)

Our two-layer authentication protocol with anonymous routing (TAPAR) allows source node to anonymously interact with the destination node. It contains three participants, namely: the source node N_0, the destination node N_n, and the intermediate nodes N_i, where $i = 1, 2, \cdots, n-1$. In addition, it consists of four phases, namely: a preprocessing phase, a path discovery phase, a path reverse phase, and a data transfer phase. Motivated by the types of security threats mentioned in Section 1, we have proposed a two-layer

Table 1: Notations

$(\mathbf{G}, g, p)$	An element g of large prime order p in a finite cyclic group $\mathbf{G}$.
ID_i	The identity of node i.
N_0	The source node.
N_n	The destination node.
N_i	The intermediate nodes, where $i = 1, 2, \cdots, n-1$.
(pk_i, sk_i)	A public key and private key of node i, where $pk_i = g^{x_i} = Y_i \in \mathbf{G}, sk_i = x_i \in GF(p)$.
tag#	An unique tag number for a route.
SK_i^j	A static shared key between node N_i and node N_j, where $SK_i^j = Y_j^{x_i} = g^{x_i x_j} = Y_i^{x_j} = SK_j^i$.
GSK	The group secret key shared among all nodes in the network.
CSK_i^j	The common session key established by node i and j.
T_i	A timestamp, which node i attaches.
$H(.)$	A public and collision-free one-way hash function.
MAC	The message authentication code and is defined by $MAC = H(k; m)$, where m denotes the message under the protection key of k.
$\oplus$	Exclusive OR operation.
hop	The number of hops that a message can transmit.
$a\|\|b$	Concatenation of message a and b.
$E_K[.]$	The symmetric encryption function with key K.
$D_K[.]$	The symmetric decryption function with key K.

Table 2: Assumptions

A-1	All nodes have the same transmission range and the links between connected nodes are bidirectional wireless links. Each node has a unique, non-zero, and integer-valued ID and its corresponding public key is clear to every node that participates in the networks. On the contrary, a private key of corresponding node is undisclosed to any others.
A-2	Node i maintains its local route table (LRT_i) for a special route and the format of LRT_i is shown as follows. $(tag\#, Ancesotr, Ancestor\_SN, Succesor, Sucessor\_SN, Lifetime)$ where the 1st field of LRT_i means the unique tag number for a route, the 2nd field records its ancestor node, the 3rd field records a serial number which is generated by the node itself and would be shared with its ancestor node, the 4th field records its successor node, the 5th field records a serial number which is generate by its successor node and would be shared with the node itself, and the 6th field means a lifetime which controls how long a route is valid in LRT_i. If LT_i hits 0, the entry will be removed from LRT_i.
A-3	Each node is capable of executing $E_{SK}[.]$, $D_{SK}[.]$, and $H(.)$ algorithms.
A-4	The proposed protocol provides perfect forward secrecy means that if the group secret key and two communication nodes's private key are simultaneously revealed to an attacker, it is still unable to the attacker derive the session keys of previous sessions.
A-5	An attacker may un-intrusively eavesdrop, alter, or replay messages into the wireless channel between two communication nodes in networks. However, we assumed that the proposed protocol does not provide a mechanism to against a Denial-of-Service (DoS) attack. For this attack, an attacker can simply disrupt, subvert, or destroy a network and this kind of attack is common though to all protocols, which is not our research focus in this paper.
A-6	Involved intermediate nodes along the route may try to break the anonymity property. Thus, the proposed protocol assumes that all the intermediates nodes involved in a specific route are not in collusion.

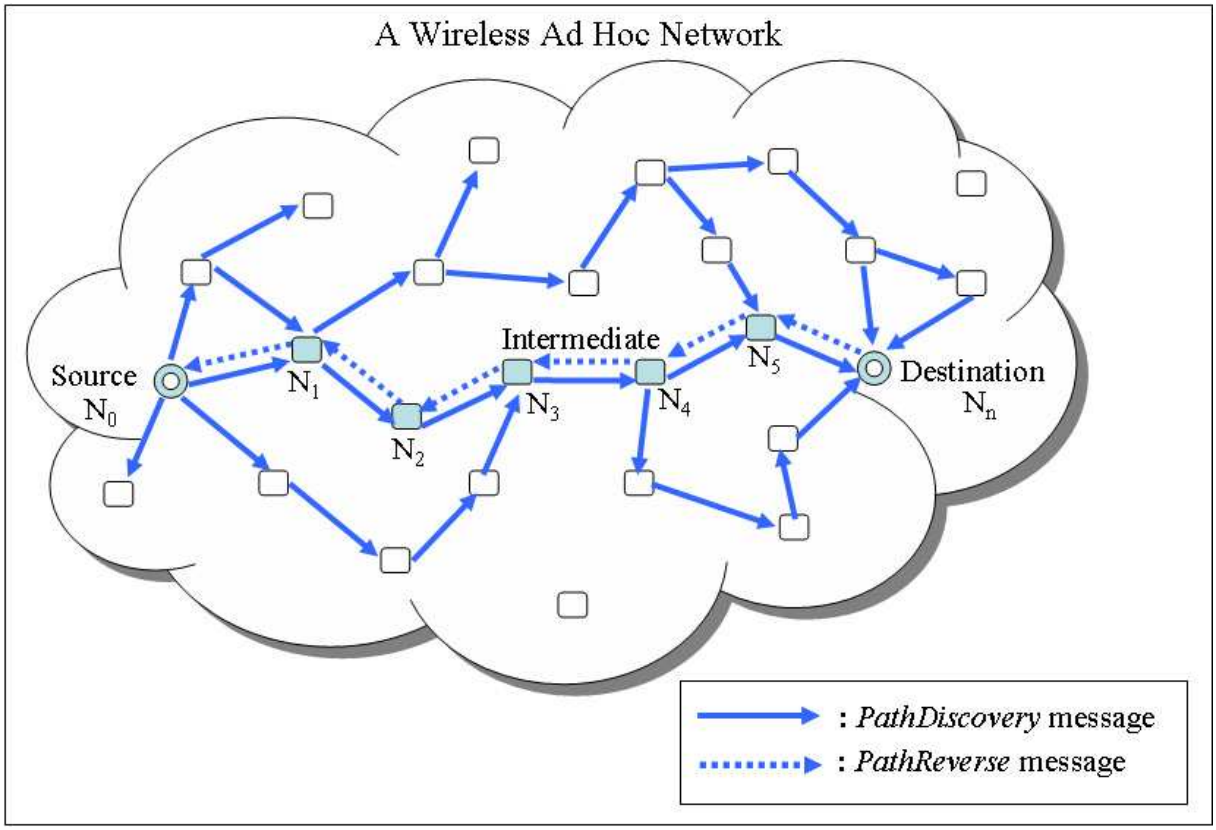

Figure 1: Ad hoc routing protocol architecture

authentication protocol to secure communications over insecure networks in all involved phases except the preprocessing phase. The goal of the first-layer is to confirm whether a message generated by group members is valid and the concept of message authentication code (MAC) would be used for quick verification during this phase. Thus, an invalid outsider would be unable to send a malevolent message into the networks and legal group members can confirm that a received message has come from a valid member, not from malicious outsiders. In addition, the goal of the second-layer is to verify whether a message has come from a specific user in a group and utilizes the concept of static shared key SK_i^j to make sure a received message has come from a specific user. Finally, the proposed protocol allows trustworthy intermediate nodes to participate in the route construction procedure without damaging the anonymity of two communicating nodes. The details of TAPAR are described

Table 3: Hashed key table of node i

Identity	Hashed session key
ID_1	$H(SK_i^1) = H(Y_1^{x_i}) = H(g^{x_i x_1}) = H(Y_i^{x_1}) = H(SK_1^i)$
ID_2	$H(SK_i^2) = H(Y_2^{x_i}) = H(g^{x_i x_2}) = H(Y_i^{x_2}) = H(SK_2^i)$
$\vdots$	$\vdots$
ID_{i-1}	$H(SK_i^{i-1}) = H(Y_{i-1}^{x_i}) = H(g^{x_i x_{i-1}}) = H(Y_i^{x_{i-1}}) = H(SK_{i-1}^i)$
ID_{i+1}	$H(SK_i^{i+1}) = H(Y_{i+1}^{x_i}) = H(g^{x_i x_{i+1}}) = H(Y_i^{x_{i+1}}) = H(SK_{i+1}^i)$
$\vdots$	$\vdots$
$ID_{\mathbf{N}-1}$	$H(SK_i^{\mathbf{N}-1}) = H(Y_{\mathbf{N}-1}^{x_i}) = H(g^{x_i x_{\mathbf{N}-1}}) = H(Y_i^{x_{\mathbf{N}-1}}) = H(SK_{\mathbf{N}-1}^i)$
$ID_{\mathbf{N}}$	$H(SK_i^{\mathbf{N}}) = H(Y_{\mathbf{N}}^{x_i}) = H(g^{x_i x_{\mathbf{N}}}) = H(Y_i^{x_{\mathbf{N}}}) = H(SK_{\mathbf{N}}^i)$

as follows.

3.2.1 Preprocessing Phase

Before an anonymous routing request is sent, each node maintains a hashed key table, which records hashed session keys shared between itself and corresponding members, to authenticate the correctness of a user's identity. Note that each node never shares its hashed key table with anyone else. As a result, our anonymous routing protocol uses a pre-computed hashed key table, which includes both member's ID information and hashed session keys, to authenticate the identity of members so it does not require any public key en/decryptions during the subsequent path discovery and path reverse phases. Assume that the number of members in an ad hoc network is $\mathbf{N}$, where $1 \leq i \leq \mathbf{N}$. Table 3 shows the format of each entry in the hashed key table.

3.2.2 Path Discovery Phase

In Figure 2, the path discovery phase of an anonymously routed two-layer authentication is provided. When a node wants to broadcast a request to neighbors or forward a message to its next node in an ad hoc network, it can be accepted only if it has the correct group key GSK and its transmission time T_i must be within a tolerance period of current time. At the beginning, when

a source node N_0 wants to securely and privately discover an available route path from itself to the destination node N_n, it first generates a unique $tag\#$ and initiates a path discovery procedure within the network to find a feasible route by performing the following operations.

Path discovery phase		
N_0		N_1
$A = g^a$, $M_{PD} = \{tag\#\|\|hop\|\|M_0\}$ $M_0 = [tag\#\|\|N_0\|\|N_n\|\|A\|\|SN_0\|\|T_{N_0}]$ $\oplus H(SK_{N_0}^{N_n})$ $packet = E_{GSK}[MAC_0\|\|T_{N_0}\|\|M_{PD}]$ $LRT_0 = (tag\#, N_n, SN_0, ?, ?, LT_0)$	$\xrightarrow{packet}$	
N_i		N_{i+1} $(1 \leq i \leq n-1)$
$packet = E_{GSK}[MAC_i\|\|T_{N_i}\|\|M_{PD}]$ $LRT_i = (tag\#, N_{i-1}, ?, ?, ?, LT_i)$	$\xrightarrow{packet}$	
N_{n-1}		N_n
$packet = E_{GSK}[MAC_{n-1}\|\|T_{N_{n-1}}\|\|M_{PD}]$ $LRT_{n-1} = (tag\#, N_{n-2}, ?, ?, ?, LT_{n-1})$	$\xrightarrow{packet}$	$(tag\#\|\|N_0\|\|N_n\|\|A\|\|AN_0\|\|T_{N_0}) = M_0 \oplus H(SK_{N_n}^{N_0})$ $LRT_n = (tag\#, N_{n-1}, SN_n, N_0, SN_0, LT_n)$

Figure 2: Path discovery phase.

1. Computes $A = g^a (\mathrm{mod} p)$ and sets a suitable number of hop depending on the actual frequency of usage, where a is a random number.

2. Makes a path discovery message M_{PD} as $M_{PD} = \{tag\#||hop||M_0\}$, where $M_0 = H(SK_{N_0}^{N_n}) \oplus [tag\#||N_0||N_n||A||SN_0||T_{N_0}]$, SN_0 is a ancestor-serial number which is stored in the third field of LRT_0, and T_{N_0} is a timestamp of current time.

3. Broadcasts a packet $E_{GSK}[MAC_0||T_{N_0}||M_{PD}]$ to neighbors within wireless transmission range, where $MAC_0 = H(GSK; T_{N_0})$.

4. Stores the entry $(tag\#, N_n, SN_0, ?, ?, LT_0)$ in its local route table LRT_0.

Since N_0 itself is the source of this route and no ancestor node exists, the second field records the identity of N_n. Similarly, the third field records a serial number which will be shared with N_n. The fourth and fifth fields are temporarily unknown, and these two fields will be added later during the path reverse phase. The sixth field records the life-time of an entry in the route. The entry will be deleted if the timer hits 0.

When a node N_i receives the packet, it first reveals $(MAC_0||T_{N_0}||M_{PD})$ by computing $D_{GSK}[E_{GSK}[MAC_0||T_{N_0}||M_{PD}]]$. Then it checks the validity of MAC_0 for fist-layer authentication and whether T_{N_0} is within a reasonable time delay range or not. If the above conditions are satisfied, N_i checks whether the packet has already been received from other nodes by comparing the $tag\#$. If the packet has been received, N_i drops it; otherwise, N_i checks if $((hop--) \geq 0)$. If this condition does not hold, drop it; otherwise, check if the node itself is the intended destination node by computing $H(SK_{N_i}^{N_0}) \oplus M_0$ with all the hashed session keys of its hashed key table, where $1 \leq i \leq \mathbf{N}$. According to Algorithm 1, one of two conditions is met by decrypted results, shown as follows: **Condition (i)** is satisfied only when the decrypted results are meaningful; that is, $(tag\#||N_0||N_n)$ are correct and the node is the intended destination node. Otherwise, under **Condition (ii)**, the node is not the intended destination node.

Algorithm 1. Confirmation of destination node for receiving M_{PD} packet

```
1: function Confirm_M_PD( ){
2:   while (M_PD message is received by N_i){
3:     Retrive hashed session keys from N_i's hashed key table;
4:       for (j = 1; j <= N; j++) do
5:         C = M_0 ⊕ H(SK_{N_i}^{N_j});          // For 2nd-layer authentication //
6:         if (C <> [tag#||N_0||N_n||A||SN_0||T_{N_0}]) {
7:         Discard this message; continue;
8:         }
9:         if (C == [tag#||N_0||N_n||A||SN_0||T_{N_0}]) {
10:          Confirmation of node itself is destination node;
11:        return Condition(i);
12:        }
13:        return Condition(ii);
14:      }
15:    }
```

Condition(i): If the node is the intended destination node N_n, it generates a successor-serial number SN_n and stores the following entry ($tag\#$, N_{n-1}, SN_n, N_0, SN_0, LT_n) in its local route table LRT_n. Since N_n itself is the destination of this route and has no successor node, the fourth field records the identity of N_0. Similarly, the fifth field records SN_0 which is generated by N_0. The second field records its ancestor node N_{n-1} and the third field records a serial number SN_n which is generated by itself and will be shared with its ancestor node N_{n-1}. The sixth field records the life-time LT_n of an entry in the route. Upon entry of all necessary records, the path discovery phase ends.

Condition(ii): Under this condition, the process is repeated until the entire hashed session keys of the hashed key table have been tried and no meaningful results remain. Thus, the node N_i ($N_1 \leq N_i \leq N_{n-1}$) confirms that it is not the destination node and forwards $E_{GSK}[MAC_i \;||\; T_{N_i} \;||\; M_{PD} = (tag\# \;||\; hop -- \;||\; M_0)]$ to neighbors within its wireless transmission range, where $MAC_i = H(GSK; T_{N_i})$. In addition, N_i stores the entry ($tag\#$, N_{i-1}, ?, ?, ?, LT_i) in its local route table LRT_i.

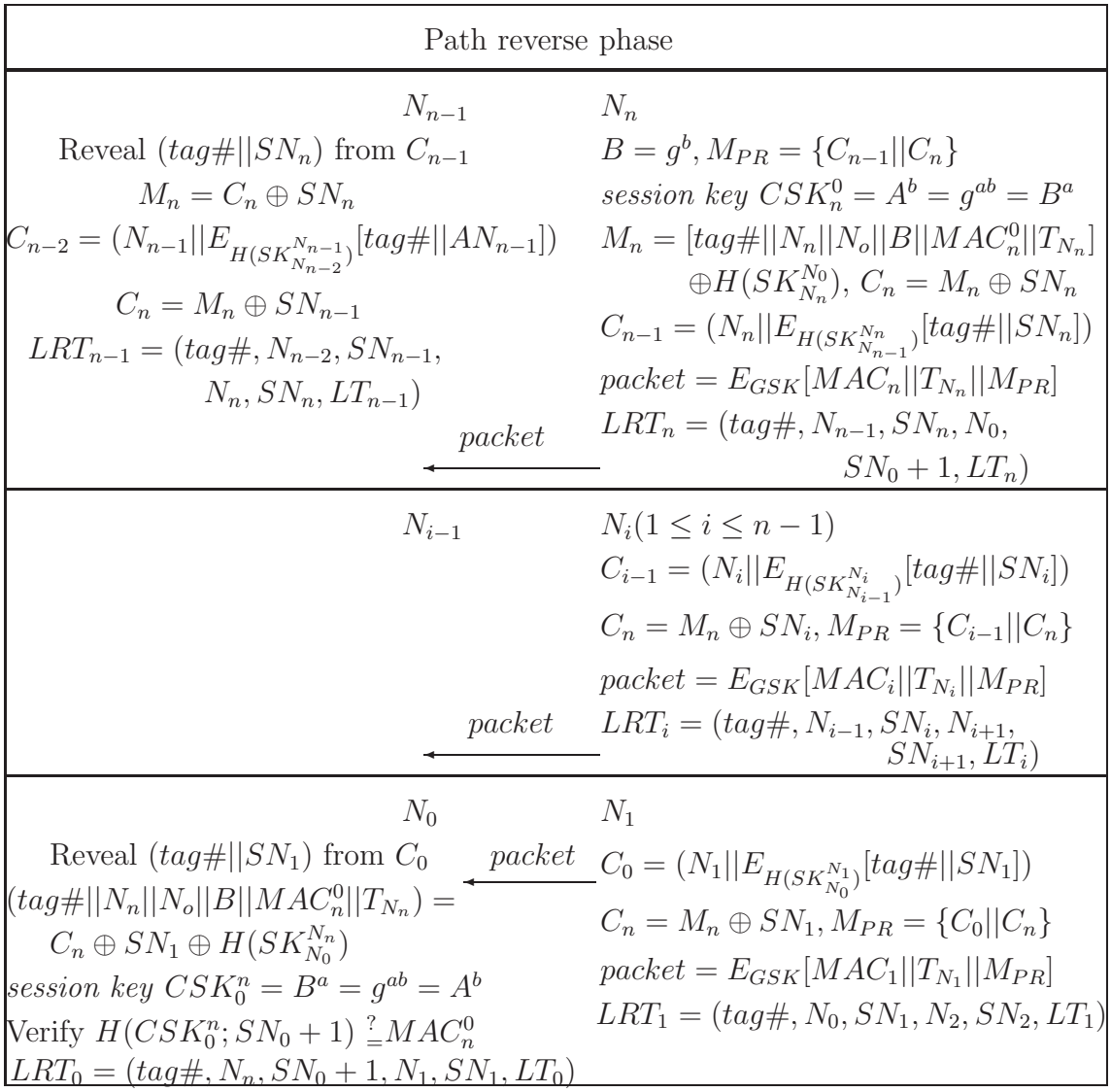

Figure 3: Path reverse phase.

3.2.3 Path Reverse Phase

Figure 3 illustrates the path reverse phase of our proposed TAPAR protocol. To initiate a path reverse session, the destination node N_n first initiates a path reverse procedure in the reverse direction to confirm a complete route from itself to the source node by performing the following operations.

1. Selects a random number b and computes $B = g^b (\mathrm{mod} p)$. N_n can then compute the common session key $CSK_n^0 = A^b = g^{ab} = B^a (\mathrm{mod} p)$, which is only shared between N_0 and N_n. It then uses the CSK_n^0 to

create the message authentication code $MAC_n^0 = H(CSK_n^0; SN_0 + 1)$, which includes the serial number SN_0+1.

2. Makes a path reverse message M_{PR} as $M_{PR} = \{C_{n-1}||C_n\}$, where $C_{n-1} = (N_n||E_{H(SK_{N_{n-1}}^{N_n})}[tag\#||SN_n])$, $C_n = M_n \oplus SN_n$, SN_n is generated by N_n and is shared with its ancestor node N_{n-1}; $H(SK_{N_{n-1}}^{N_n} = Y_{n-1}^{x_n} = g^{x_{n-1}x_n} = Y_n^{x_{n-1}})$ is a hashed static shared key between node N_n and node N_{n-1}, and $M_n = [tag\#||N_n||N_0||B||MAC_n^0||T_{N_n}] \oplus H(SK_{N_n}^{N_0})$.

3. Unicasts a packet $E_{GSK}[MAC_n||T_{N_n}||M_{PR}]$ to its ancestor node N_{n-1}, where $MAC_n = H(GSK; T_{N_n})$.

4. Maintains its local route table LRT_n as $LRT_n = (tag\#,\ N_{n-1},\ SN_n,\ N_0,\ SN_0 + 1,\ LT_n)$.

Upon receiving the path reverse packet from N_n, N_{n-1} would then deal with this message according to following steps.

1. Reveals $(MAC_n||T_{N_n}||M_{PR})$ by computing $D_{GSK}[E_{GSK}[MAC_n||T_{N_n}||M_{PR}]]$ and verifies the validity of MAC_n for first-layer authentication. If the above condition is met, continue; otherwise, stop.

2. Uses the static shared key $H(SK_{N_{n-1}}^{N_n})$ to reveal $(tag\#||SN_n)$ and checks whether $tag\#$ exists in LRT_{n-1} or not. If it is found, continue; otherwise, stop.

3. Uses SN_n to reveal M_n by computing $C_n \oplus SN_n$ and creates a new M_{PR} packet as $M_{PR} = \{C_{n-2}||C_n\}$, where $C_{n-2} = (N_{n-1}\ ||\ E_{H(SK_{N_{n-1}}^{N_{n-2}})}\ [tag\#\ ||\ SN_{n-1}])$, $C_n = M_n \oplus SN_{n-1}$, and SN_{n-1} is a serial number which is generated by itself and shared with N_{n-2}.

4. According to the second field of LRT_{n-1}, N_{n-1} unicasts the packet E_{GSK} $[MAC_{n-1}\ ||\ T_{N_{n-1}}\ ||\ M_{PR}]$ to its ancestor node N_{n-2}.

5. Maintains its local route table LRT_{n-1} as $LRT_{n-1} = (tag\#,\ N_{n-2},\ SN_{n-1},\ N_n,\ SN_n,\ LT_{n-1})$.

Algorithm 2. Forward path reverse packet between intermediate nodes

```
1: function Forward_Path_Reverse_Packet( ){
2:   while (path reverse packet is received by N_i){
3:     for (i = n − 1; i <= 1; i − −) do
4:       Decrypt E_GSK[MAC_{i+1}||T_{N_{i+1}}||M_PR];
5:       if (MAC_{i+1} <> H(GSK; T_{N_{i+1}})) {      // For 1st-layer authentication //
6:       Discard this message and stop;
7:       }
8:          Retrieve H(SK_{N_i}^{N_{i+1}}) to reveal (tag#||SN_{i+1});
9:          if (tag# is not found in LRT_i or LT_i hits 0) {
10:         Discard this message and stop;
11:         }
12:           Compute D = C_n ⊕ SN_{i+1};
13:           Generate SN_i;
14:           Make C_{i−1} = (N_i||E_{H(SK_{N_i}^{N_{i−1}})}[tag#||SN_i]);
15:           Make C_n = D ⊕ SN_i;
16:           Unicast E_GSK[MAC_i||T_{N_i}||M_PR = (C_{i−1}||C_n)] to its ancestor node N_{i−1};
17:     }
18:   }
```

Other intermediate nodes N_i along the route perform the same steps as the node N_{n-1}, where $i = 1, 2, 3, \cdots, n-3, n-2$ (See Algorithm 2). As a result, the last intermediate node N_1 should unicast a packet $E_{GSK}[MAC_1||T_{N_1}||M_{PR}]$ to its ancestor node N_0 and maintain its local route table LRT_1 as $LRT_1 = (tag\#, N_0, SN_1, N_2, SN_2, LT_1)$, where $M_{PR} = \{C_0||C_n\}$, $C_0 = (N_1\ ||\ E_{H(SK_{N_0}^{N_1})}[tag\#\ ||\ SN_1])$, and $C_n = M_n \oplus SN_1$.

Finally, when the source node N_0 receives the packet $E_{GSK}[MAC_1||T_{N_1}||M_{PR}]$ from N_1, N_0 performs the following operations.

1. Reveals $(MAC_1||T_{N_1}||M_{PR})$ by computing $D_{GSK}[E_{GSK}[MAC_1||T_{N_1}||M_{PR}]]$ and verifiesy the validity of MAC_1 by first-layer authentication. If the above condition is met, continue; otherwise, stop.

2. Uses the static shared key $H(SK_{N_0}^{N_1})$ to reveal $(tag\#||SN_1)$ from C_0 and checks whether $tag\#$ exists in LRT_0 or not. If it is found, continue;

otherwise, stop.

3. Uses SN_1 to reveal M_n by computing $C_n \oplus SN_1$ and use $H(SK_{N_0}^{N_n})$ to reveal $(tag\#||N_n||N_0||B||MAC_n^0||T_{N_n})$ by computing $M_n \oplus H(SK_{N_0}^{N_n})$.

4. Computes the common session key CSK_0^n by computing $CSK_0^n = B^a = g^{ab} = A^b(\bmod p)$ and verifies the validity of CSK_0^n by comparing $H(CSK_0^n; SN_0+1) \stackrel{?}{=} MAC_n^0$ for the second-layer authentication. If the above condition holds, not only has the route from N_0 to N_n been confirmed, but the common session key CSK_0^n shared between N_0 and N_n would have been established by them as well; which is used to secure communications during the following data transfer phase.

5. Maintains its local route table LRT_0 as $LRT_0 = (tag\#, N_n, SN_0+1, N_1, SN_1, LT_0)$. After this step path reverse phase ends.

3.2.4 Data Transfer Phase

Figure 4 describes the data transfer phase of the TAPAR protocol. The source node N_0 securely and privately sends a confidential data M to the destination node N_n. N_0 first encrypts M as $M' = E_{CSK_0^n}[M||SN_0+2]$ and computes $F = H(M')$, where $E_{CSK_0^n}$ is a common session key shared between N_0 and N_n. Then, N_0 initiates a data transfer procedure within the network for the purpose of sending the secret packet along the route till it reaches N_n by performing the following operations.

1. Creates a data transfer message M_{DT} as $M_{DT} = \{I||G\}$, where $I = M' \oplus SN_1 + 1$ and $G = (N_0||E_{H(SK_{N_0}^{N_1})}[tag\#||F])$

2. Unicasts a packet $E_{GSK}[MAC_0||T_{N_0}||M_{DT}]$ to its successor node N_1, where $MAC_0 = H(GSK; T_{N_0})$.

3. Maintains its local route table LRT_0 as $LRT_0 = (tag\#, N_n, SN_0+2, N_1, SN_1+1, LT_0)$.

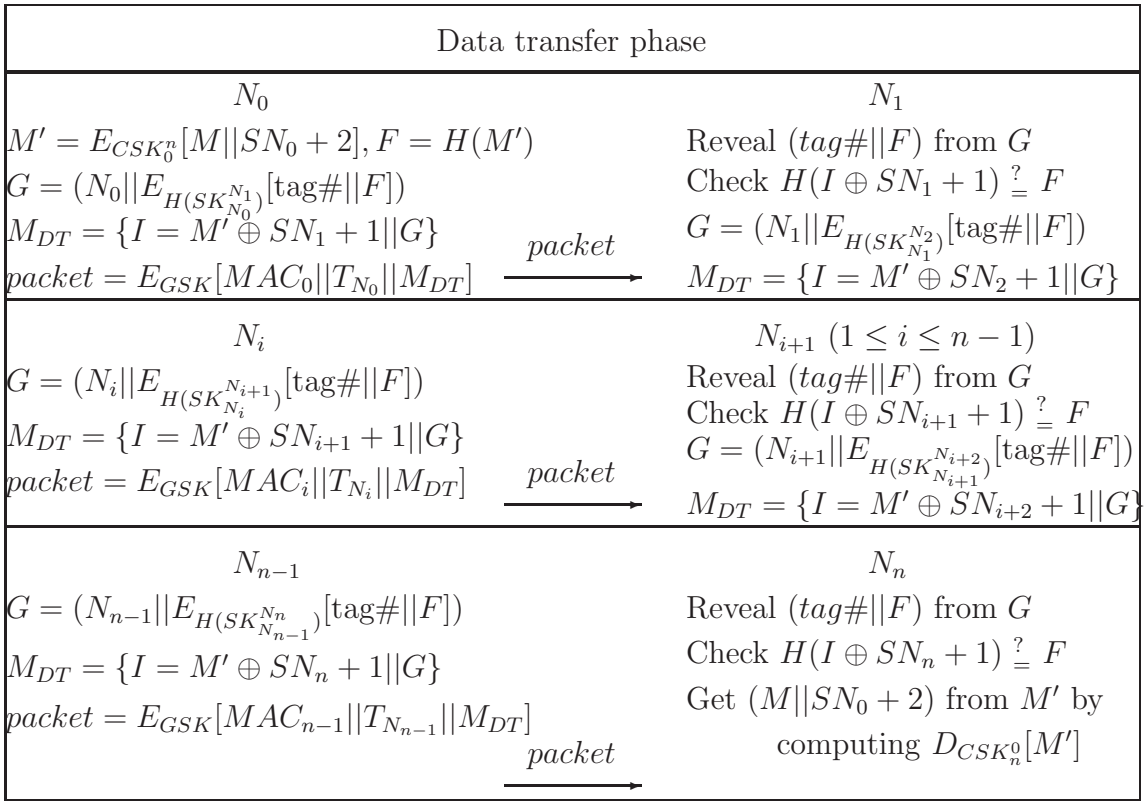

Figure 4: Data transfer phase.

Algorithm 3. Forward data transfer packet between intermediate nodes

```
1: function Forward_Data_Transfer_Packet( ){
2:   while (data transfer packet is received by N_i){
3:     for (i = 1; i <= n − 1; i++) do
4:       Decrypt E_GSK[MAC_{i−1}||T_{N_{i−1}}||M_DT];
5:       if (MAC_{i−1} <> H(GSK; T_{N_{i−1}})) {    // For 1st-layer authentication //
6:       Discard this message and stop;
7:       }
8:          Retrieve H(SK_{N_i}^{N_{i−1}}) to reveal (tag#||F);
9:          if (tag# is not found in LRT_i or LT_i hits 0) {
10:         Discard this message and stop;
11:         }
12:            Compute I ⊕ SN_i + 1;
13:            if (H(I ⊕ SN_i + 1) <> F) {
14:            Discard this message and stop;
15:          }
16:          Make G = (N_i||E_{H(SK_{N_i}^{N_{i+1}})}[tag#||F]);
17:          Make M_DT = {I = M' ⊕ SN_{i+1} + 1||G};
18:          Unicast E_GSK[MAC_i||T_{N_i}||M_DT] to its successor node N_{i+1};
19:    }
20:  }
```

Upon receiving the data transfer message, each intermediate node N_i deals with this message according to Algorithm 3, where $i = 1, 2, \cdots, n-1$. As a result, the last intermediate node N_1 should unicast a data transfer packet $E_{GSK}[MAC_{n-1}||T_{N_{n-1}}||M_{DT}]$ to its successor node N_n and maintain its local route table LRT_{n-1} as $LRT_{n-1} = (tag\#, N_{n-2}, SN_{n-1}+1, N_n, SN_n+1, LT_{n-1})$ upon completion of the route, where $M_{DT} = \{I||G\}$, $I = M' \oplus SN_n + 1$, and $G = (N_{n-1}||E_{H(SK^{N_n}_{N_{n-1}})}[tag\#||F])$.

When the destination node N_n receives the packet $E_{GSK}[MAC_{n-1}||T_{N_{n-1}} || M_{DT}]$ from N_{n-1}, it performs the following operations.

1. Reveals $(MAC_{n-1}||T_{N_{n-1}}||M_{DT})$ by computing $D_{GSK}[E_{GSK}[MAC_{n-1} || T_{N_{n-1}} || M_{DT}]]$ and verifies the validity of MAC_{n-1} for first-layer authentication. If the above condition holds, continue; otherwise, stop.

2. Uses the static shared key $H(SK^{N_{n-1}}_{N_n})$ to reveal $(tag\#||F)$ from G and checks whether $tag\#$ exists in LRT_n or not. If it is found, continue; otherwise, stop.

3. Uses $SN_n + 1$ to reveal M' by computing $I \oplus SN_n + 1$ and verifies $H(M') \stackrel{?}{=} F$. If it holds, continue; otherwise, stop.

4. Uses the common session key CSK^0_n to recover message $(M||SN_0 + 2)$ from M' by computing $D_{CSK^0_n}[E_{CSK^n_0}[M']]$.

5. Maintains its local route table LRT_n as $LRT_n = (tag\#, N_{n-1}, SN_n + 1, N_0, SN_0 + 2, LT_n)$. After this step the data transfer phase ends.

4 Analysis and Discussion

In this section, we shall analyze the requirements and the security of our proposed TAPAR. Furthermore, in terms of computational complexity, we have evaluated and compared the performance of TAPAR with the other related

protocols mentioned in Section 2. The details of the above-mentioned analyzes are briefly described in the following subsections.

4.1 Security Analysis

In this section, we will provide proof of the correctness of TAPAR and evaluate its security with respect to its robustness in the presence of the attacks presented in Section 1. As mentioned earlier, our proposed TAPAR does not consider DoS attacks and all intermediate nodes involved in a specific route are not in collusion, which is common throughout all ad hoc anonymous routing protocols, and TAPAR could satisfy the following theorems.

Theorem 4.1. *TAPAR is secured against passive and active attacks while maintaining the data integrity of the nodes that have participated in a specific route over ad hoc networks; including the source node, intermediate nodes, and the destination node.*

Proof. During path discovery phase, the source node sends a path discovery message through a number of group members until it reaches the destination node, and the message is encrypted with the group secret key that is only shared among valid members. Thus, beginning with the source node, invalid outsiders are unable to participate in routing procedures to further decrypt the message, also the security of the secret key is ensured based on MAC and one-way hashing cryptography. Hence, data confidentiality and integrity are well-protected by first-layer authentication in our proposed protocol, preventing passive attacks such as eavesdropping. Moreover, in active attacks, an attacker could intercept the authorized credential message and replay it. However, such attacks can be prevented by timestamps and MAC, both of which are used in our protocol to guarantee the freshness of transmission messages. Furthermore, static shared key $SK_{N_0}^{N_n}$, that is, the security mechanism of second-layer authentication, rend it impossible for valid group members

to be impersonated as some source node N_0 to run the anonymous routing procedure with some destination node N_n as well.

During the path reverse phase, the destination node sends a path reverse message along the reverse order path of a route through a number of intermediate nodes until it reaches the source node; also the message is encrypted with the group secret key as well. Finally, every session between two nodes is well-protected by the static shared key $SK_{N_i}^{N_{i-1}}$, thus no one has the ability to acquire or derive their communication messages, where N_i is N_{i-1}'s successor node. As a result, many passive and active attacks can be prevented by TAPAR. □

Theorem 4.2. *TAPAR ensures anonymous interactions between the source N_0 and the destination N_n through a number of intermediate nodes, and neither intermediate nodes nor outsiders can ascribe any session to a particular source or destination during any session.*

Proof. During path discovery phase, if an attacker Eve intercepts N_0's path discovery message $E_{GSK}[MAC_0||T_{N_0}||M_{PD}]$ and was able to derive $M_{PD} = \{tag\# \;||\; hop \;||\; M_0\}$, she still could not know which node is the source and which node is the destination because she lacks the static shared key $SK_{N_0}^{N_n}$.

Similarly, during the path reverse phase, if an attacker Eve intercepts N_n's path reverse message $E_{GSK}[MAC_n||T_{N_n}||M_{PR}]$ and was able to obtain $M_{PR} = \{C_{n-1}||C_n\}$, she would still be unable to discover which node is the destination and which node is the source. Not only because she lacks static shared key $SK_{N_0}^{N_n}$, but also because she lacks the key $H(SK_{N_{n-1}}^{N_n})$, where N_{n-1} is N_n's ancestor node in a specific route. As a result, the source node can anonymously interact with the destination node without disclosing either's personal information. Thus absolute privacy is achieved for wireless ad hoc networks in our proposed TAPAR. □

Theorem 4.3. *TAPAR is able to provide mutual authentication of fresh ses-*

sion key agreement between two communication nodes in wireless ad hoc networks.

Proof. Let A and B be two communication entities, namely: the source and the destination. Let $A \overset{K_{ab}}{\longleftrightarrow} B$ denotes the fresh session key K_{ab} shared between A and B, and SK_a^b is a static shared key between A and B. Hence, the mutual authentication of K_{ab} is achieved between A and B if there exists a session key K_{ab}, and A believes $A \overset{K_{ab}}{\longleftrightarrow} B$ and B believes $A \overset{K_{ab}}{\longleftrightarrow} B$. As a result, we stated that a strong mutual authentication of K_{ab} should satisfy the following equations:

$$A \ \ believes \ \ B \ \ believes \ \ A \overset{K_{ab}}{\longleftrightarrow} B. \tag{1}$$

$$B \ \ believes \ \ A \ \ believes \ \ A \overset{K_{ab}}{\longleftrightarrow} B. \tag{2}$$

In **Condition(i)** of the path discovery phase, when B receives the message M_0, he decrypts $(tag\#||A||B||g^a||SN_A||T_{N_A})$ by using the shared secret key SK_a^b, where g^a is A's contribution to K_{ab} and SN_A is a challenge. Then B can generate its own contribution g^b and computes the session key $K_{ab} = g^{ab}$ and believe $A \overset{K_{ab}}{\longleftrightarrow} B$. Also, B's contribution g^b and a message authentication code $MAC_b^a = H(K_{ab}; SN_A + 1)$ are sent to A in the path reverse phase, where $SN_A + 1$ is the appropriate response.

Upon receiving g^b and MAC_b^a from B, A computes the session key $K_{ab} = g^{ab}$ and verifies the message MAC_b^a to confirm whether this message contains $SN_A + 1$. If it does, A believes $A \overset{K_{ab}}{\longleftrightarrow} B$. Since a is chosen by A, A believes B believes $A \overset{K_{ab}}{\longleftrightarrow} B$.

During the data transfer phase, after B receives $M' = E_{K_{ab}}[M||SN_A + 2]$, B will use K_{ab} to decrypt M by computing $D_{K_{ab}}[M']$, and checks if the result contains a fresh response $SN_A + 2$. If it does, B believes $A \overset{K_{ab}}{\longleftrightarrow} B$. Since g^b is generated by B, B believes A believes $A \overset{K_{ab}}{\longleftrightarrow} B$. Finally, Equations (1) and (2) are satisfied and collectively accomplish the good property of mutual authentication of the dynamic session key. □

Theorem 4.4. *Based on the well-known Diffie-Hellman problem, TAPAR also meets the security requirement for forward secrecy; even if the group secret key GSK, static shared key $H(SK_{N_0}^{N_n})$, N_0's contribution g^a, and N_n's contribution g^b are all known by an attacker.*

Proof. We assumed that the attacker Eve intercepts information including GSK and $H(SK_{N_0}^{N_n})$ and knew the relationship between N_0 and N_n. Then Eve could decrypt M_{PD} to obtain N_0's contribution g^a and could also decrypt M_{PR} to obtain N_n's contribution g^b. However, Eve could not derive the common session key $CSK_0^n = g^{ab}$ since such task is as difficult as solving the Diffie-Hellman problem [6]. Thus our proposed TAPAR provides perfect forward secrecy. □

4.2 Functionality Analysis

As illustrated in Table 4, the functionality requirements of our protocol are better than those of other anonymous routing protocols. In particular, the proposed TAPAR have accomplished several crucial goals, such as session key agreement with forward secrecy, mutual authentication of session key by second-layer authentication, group secrecy by first-layer authentication, and lightweight computations as a result of using non-PKI operations.

4.3 Performance Analysis

In this subsection, the computational overhead of the proposed TAPAR protocol is analyzed. Moreover, we compared our TAPAR protocol with the previous protocols mentioned in Section 2. The results of the comparison of efficiencies between TAPAR, Boukerche et al.'s protocol [3], and Lu et al.'s protocol [35] are shown in Table 5. To evaluate performances, we have defined several computational parameters as follows:

- T_{exp} denotes the time required by the modular exponentiation.

Table 4: Functionality comparisons between our protocol and two related protocols

	Ours	Boukerche et al.'s [3]	Lu et al.'s [35]
Anonymous routing	Yes	Yes	Yes
Session key agreement	Yes	No	Yes
Mutual authentication of session key	Yes	No	No
Forward secrecy	Yes	No	Yes
Group secrecy	Yes	No	No
Computation cost	Low	High	Medium
Based problem	MAC, Hashing operations, Symm. operations, Discrete logarithms	PKI operations, Symm. operations	PKI operations, Symm. operations, Hashing operations, Discrete logarithms

- T_{hash} denotes the time required by the hashing operation.
- T_{sym} denotes the time required by the symmetric encryption/decryption operation.
- T_{asym} denotes the time required by the asymmetric encryption/decryption operation.
- T_{xor} denotes the time required by the XOR operation.

For instance, as introduced in [17], a symmetric encryption/decryption is at least 100 times faster than an asymmetric encryption/decryption in software, an exponential operation is approximately equal to 60 symmetric encryptions/decryptions, and a hashing operation is at least 10 times faster than a symmetric encryption/decryption. Therefore, we present the following equation:

$$1\ T_{asym} \approx 100\ T_{sym}, 1\ T_{exp} \approx 60\ T_{sym}, and\ 1\ T_{sym} \approx 10\ T_{hash}. \quad (3)$$

Moreover, it requires 0.0005 second to perform a one-way hashing operation and 0.0087 second to perform a symmetric encryption/decryption. According

Table 5: Efficiency comparisons between our protocol and two related protocols

	Ours	Boukerche et al.'s [3]	Lu et al.'s [35]
Path Discovery Phase	Source node: 1 T_{exp}+1 T_{hash}+ 1 T_{xor}+1 T_{sym} Intermediate node: 2 T_{hash}+2 T_{sym}+ (**N**-1) T_{xor} Destination node: 1 T_{sym}+1 T_{hash}+ ($\leq$ **N**-1) T_{xor}	Source node: 2 T_{asym}+1 T_{sym} Intermediate node: 2 T_{asym}+1 T_{sym} Destination node: (n+2) T_{asym}+n T_{sym}	Source node: 2 T_{exp}+1 T_{hash}+ 1 T_{asym} Intermediate node: 1 T_{asym} Destination node: 1 T_{exp}+1 T_{hash}+ 1 T_{asym}
Path Reverse Phase	Source node: 1 T_{exp}+2 T_{hash}+ 2 T_{xor}+2 T_{sym} Intermediate node: 2 T_{hash}+4 T_{sym}+ 2 T_{xor} Destination node: 2 T_{exp}+2 T_{hash}+ 2 T_{xor}+2 T_{sym}	Source node: 1 T_{sym} Intermediate node: 1 T_{sym} Destination node: n+1 T_{sym}	Source node: 2 T_{asym}+1 T_{hash}+ 1 T_{exp} Intermediate node: 2 T_{asym} Destination node: 2 T_{exp}+1 T_{hash}+ 2 T_{asym}
Data Transfer Phase	Source node: 3 T_{sym}+2 T_{hash}+ 1 T_{xor} Intermediate node: 3 T_{sym}+2 T_{hash}+ 2 T_{xor} Destination node: 3 T_{sym}+2 T_{hash}+ 1 T_{xor}	Source node: n+1 T_{sym} Intermediate node: 1 T_{sym} Destination node: 1 T_{sym}	Source node: 1 T_{asym}+1 T_{hash}+ 1 T_{sym} Intermediate node: 2 T_{asym} Destination node: 1 T_{asym}+1 T_{hash}+ 1 T_{sym}
Computation Costs (n=1)	$\approx$ 3616 T_{hash}	$\approx$ 7110 T_{hash}	$\approx$ 16626 T_{hash}
Computation Time (n=1)	1.808 seconds	3.555 seconds	8.313 seconds

N: The number of group members in a ad hoc network.
n: The number of intermediate nodes in a specific route.

to Equation (3), in comparison with [35], nearly 4802 T_{hash} are required by the path discovery phase of Lu et al.'s protocol, while our protocol requires about 1634 T_{hash}; and the computational overhead of the path discovery phase of our protocol can be reduced by 34.02%. Note that we assumed there is only one intermediate node involved ($n = 1$) in a specific route and the computational costs of the Exclusive OR operations are ignored, since these kinds of operations have much lighter costs than hashing operations. Additionally, the path reverse phase of [35] requires nearly 7802 T_{hash}, while ours only requires 1886 T_{hash}; and the computational overhead of the path reverse phase of our protocol can be reduced by 24.17%. During data transfer phase, [35] requires nearly 4022 T_{hash}, while ours only requires 96 T_{hash}; and the computational overhead during this phase of our protocol can be reduced by 2.39%.

As described in Table 5, the total computational time of ours and Lu et al.'s protocol are 1.808 seconds and 8.313 seconds, respectively. As a result, the computational costs of our scheme can be reduced by 21.75% in comparison with Lu et al.'s protocol. Therefore, the proposed TAPAR protocol is highly efficient in the sense of computational overhead.

On the other hand, for the memory consumption, the proposed TAPAR protocol must store **N** pieces of hashed session keys. For example, if SHA-256 algorithm is used and **N** = 100, then each node will need 100 x 256 = 25,600 bits = 3,200 bytes of memory to store the hashed key table. As a result, the proposed protocol requires much less bandwidth and memory than traditional PKI-solution protocols; and that makes TAPAR protocol quite applicable to wireless ad hoc networks.

5 Conclusions

In this paper, a two-layer authentication protocol with anonymous routing (TAPAR) for wireless ad hoc networks is proposed. We have attempted to

introduce a novel solution without resorting to PKI operations to achieve anonymity between two communication entities over insecure networks. Moreover, by comparison with other related protocols, the proposed TAPAR protocol not only compares favorably (e.g., low computational costs, session key establishment with forward secrecy, mutual authentication, group secrecy etc.), but also provides the advantage of anonymous routing. Hence, a source node can privately establish a route and securely send confidential data from itself to a destination node with complete confidentiality and anonymity (e.g., private communication and transmission integrity). Additionally, TAPAR also takes into account the resource constraints of the mobile ad hoc devices by minimizing the computational overhead with non-PKI operations. Thus, in comparison with Boukerche et al.'s, and Lu et al.'s protocols, the overheads of involved participants in our protocol are quite low in terms of computational complexity. Specifically, the computational costs of our TAPAR protocol are as much as 21.75% lower than those of Lu et al.'s protocol. As a result, TAPAR is suitable for various ad hoc networks and privacy-vital applications in ubiquitous computing environments since it provides security, privacy, and efficiency.

Acknowledgements

This work was supported in part by National Chung Hsing University (NCHU), Aiming for the Top University and Elite Research Center Development Plan under the grant CC98118. We also thank anonymous referees for their valuable suggestions and comments to improve this paper.

References

[1] Doina Bein and S. Q. Zheng. Energy efficient all-to-all broadcast in all-wireless networks. *Information Sciences*, 180(10):1781–1792, 2010.

[2] Mohamed Salah Bouassida, "Authentication vs. Privacy within Vehicular Ad Hoc Networks," *International Journal of Network Security*, vol. 12, no. 3, pp. 259-272, 2011.

[3] Azzedine Boukerche, Khalil El-Khatib, Li Xu, and Larry Korba. An efficient secure distributed anonymous routing protocol for mobile and wireless ad hoc networks. *Computer Communications*, 28(10):1193–1203, 2005.

[4] Chih-Yung Chang, Chao-Tsun Chang, Tzung-Shi Chen, and Hsu-Ruey Chang. Hierarchical management protocol for constructing a QoS communication path in wireless ad hoc networks. *Information Sciences*, 177(13):2621–2641, 2007.

[5] Hsing-Chung Chen, Shiuh-Jeng Wang, and Jyh-Horng Wen. Packet construction for secure conference call request in ad hoc network systems. *Information Sciences*, 177(24):5598–5610, 2007.

[6] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, Nov. 1976.

[7] M. Razvi Doomun and K. M. Sunjiv Soyjaudah, "LOTKIP: Low Overhead TKIP Optimization for Ad Hoc Wireless Networks," *International Journal of Network Security*, vol. 10, no. 3, pp. 225-237, 2010.

[8] Z. J. Haas, M. Perlman, and P. Samar. *The Interzone Routing Protocol (IERP) for Ad Hoc Networks*. IETF MANET Working Group, 2001.

[9] W. Hu, K. Xue, P. Hong, and C. Wu, "ATCS: A Novel Anonymous and Traceable Communication Scheme for Vehicular Ad Hoc Networks," *International Journal of Network Security*, vol. 12, no. 3, pp. 239-246, 2011.

[10] Y. C. Hu, A. Perrig, and D. B. Johnson. Ariadne: a secure on-demand routing protocol for adhoc networks. In *Proceedings of MOBICOM*, pages 255–265, 2002.

[11] Min-Shiang Hwang, Cheng-Chi Lee, and Ji-Zhe Lee. A new anonymous channel protocol in wireless communications. *International Journal of Electronics and Communications*, 58(3):218–222, 2004.

[12] D. B. Johnson, D. A. Maltz, and Y. C. Hu. *The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)*. IETF Internet-Draft: Work in Progress, 2004.

[13] Y. B. Ko and N. H. Vaidya. Location-aided routing in mobile ad hoc network. In *Proceedings of the MOBICOM*, pages 66–75, 1998.

[14] Nikos Komninos, Dimitris Vergados, and Christos Douligeris. Detecting unauthorized and compromised nodes in mobile ad hoc networks. *Ad Hoc Networks*, 5(3):289–298, 2007.

[15] Satoshi Kurosawa, Hidehisa Nakayama, Nei Kato, Abbas Jamalipour, and Yoshiaki Nemoto. Detecting blackhole attack on AODV-based mobile ad hoc networks by dynamic learning method. *International Journal of Network Security*, 5(3):338–346, 2007.

[16] Cheng-Chi Lee, Min-Shiang Hwang, and I-En Liao. Security enhancement on a new authentication scheme with anonymity for wireless environments. *IEEE Transactions on Industrial Electronics*, 53(5):1683–1687, 2006.

[17] Jung-San Lee and Chin-Chen Chang. Secure communications for cluster-based ad hoc networks using node identities. *Journal of Network and Computer Applications*, 30(4):1377–1396, 2007.

[18] Chun-Ta Li. An efficient and secure communication scheme for trusted computing environments. *Journal of Computers*, 20(3):17–24, 2009.

[19] Chun-Ta Li and Yen-Ping Chu. Cryptanalysis of threshold password authentication against guessing attacks in ad hoc networks. *International Journal of Network Security*, 8(2):166–168, 2009.

[20] Chun-Ta Li and Min-Shiang Hwang. An efficient biometrics-based remote user authentication scheme using smart cards. *Journal of Network and Computer Applications*, 33(1):1–5, 2010.

[21] Chun-Ta Li and Min-Shiang Hwang. An online biometrics-based secret sharing scheme for multiparty cryptosystem using smart cards. *International Journal of Innovative Computing, Information and Control*, 6(5):2181–2188, 2010.

[22] Chun-Ta Li and Min-Shiang Hwang. A batch verifying and detecting the illegal signatures. *International Journal of Innovative Computing, Information and Control*, accepted for publication, 2010.

[23] Chun-Ta Li, Min-Shiang Hwang, and Yen-Ping Chu. Further improvement on a novel privacy preserving authentication and access control

scheme for pervasive computing environments. *Computer Communications*, 31(18):4255–4258, 2008.

[24] Chun-Ta Li, Min-Shiang Hwang, and Yen-Ping Chu. Improving the security of a secure anonymous routing protocol with authenticated key exchange for ad hoc networks. *International Journal of Computer Systems Science and Engineering*, 23(3):227–234, 2008.

[25] Chun-Ta Li, Min-Shiang Hwang, and Yen-Ping Chu. A secure and efficient communication scheme with authenticated key establishment and privacy preservation for vehicular ad hoc networks. *Computer Communications*, 31(12):2803–2814, 2008.

[26] Chun-Ta Li, Min-Shiang Hwang, and Yen-Ping Chu. An efficient sensor-to-sensor authenticated path-key establishment scheme for secure communications in wireless sensor networks. *International Journal of Innovative Computing, Information and Control*, 5(8):2107–2124, 2009.

[27] Chun-Ta Li, Min-Shiang Hwang, and Chi-Yu Liu. An electronic voting protocol with deniable authentication for mobile ad hoc networks. *Computer Communications*, 31(10):2534–2540, 2008.

[28] Chun-Ta Li, Checng-Chi Lee, and Lian-Jun Wang. On the security enhancement of an efficient and secure event signature protocol for P2P MMOGs. In *The 2010 International Conference on Computational Science and Its Applications*, volume 6016 of *Lecture Notes in Computer Science*, pages 599–609. Springer-Verlag, 2010.

[29] Chun-Ta Li, Checng-Chi Lee, and Lian-Jun Wang. A two-factor user authentication scheme providing mutual authentication and key agreement over insecure channels. *Journal of Information Assurance and Security*, 5(2):201–208, 2010.

[30] Chun-Ta Li, C. H. Wei, and Y. H. Chin. A secure event update protocol for peer-to-peer massively multiplayer online games against masquerade attacks. *International Journal of Innovative Computing, Information and Control*, 5(12(A)):4715–4723, 2009.

[31] Chun-Ta Li, C. H. Wei, C. C. Lee, Y. H. Chin, and L. J. Wang. A secure and undeniable billing protocol among charged parties for grid computing environments. *International Journal of Innovative Computing, Information and Control*, accepted for publication, 2010.

[32] Hongwei Li and Atam P. Dhawan, "MOSAR: A Secured On-demand Routing Protocol for Mobile Multilevel Ad Hoc Networks," *International Journal of Network Security*, vol. 10, no. 2, pp. 121-134, 2010.

[33] Zhenjiang Li and J.J. Garcia-Luna-Aceves. Non-interactive key establishment in mobile ad hoc networks. *Ad Hoc Networks*, 5(7):1194–1203, September 2007.

[34] Xiaodong Lin, Rongxing Lu, Haojin Zhu, Pin-Han Ho, Xuemin Shen, and Zhenfu Cao. ASRPAKE: An anonymous secure routing protocol with authenticated key exchange for wireless ad hoc networks. In *IEEE International Conference on Communications*, pages 1247–1253, 2007.

[35] Rongxing Lu, Zhenfu Cao, Licheng Wang, and Congkai Sun. A secure anonymous routing protocol with authenticated key exchange for ad hoc networks. *Computer Standards & Interfaces*, 29(5):521–527, 2007.

[36] S. S. Manvi and M. S. Kakkasageri. Multicast routing in mobile ad hoc networks by using a multiagent system. *Information Sciences*, 178(6):1611–1628, 2008.

[37] Changsoo Ok, Seokcheon Lee, Prasenjit Mitra, and Soundar Kumara. Distributed routing in wireless sensor networks using energy welfare metric. *Information Sciences*, 180(9):1656–1670, 2010.

[38] P. Papadimitratos and Z. J. Haas. Secure routing for mobile ad hoc networks. In *Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference*, pages 30–38, 2002.

[39] P. Papadimitratos and Z. J. Haas. Secure message transmission in mobile ad hoc networks. *Ad Hoc Networks*, 1(1):193–209, 2003.

[40] M. Ramkumar and N. Memon. An efficient key predistribution scheme for ad hoc network security. *IEEE Journal on Selected Areas in Communications*, 23(3):611–621, 2005.

[41] Wei Ren. Pulsing roq ddos attacking and defense scheme in mobile ad hoc networks. *International Journal of Network Security*, 4(2):227–234, March 2007.

[42] Bartlomiej Sieka and Ajay D. Kshemkalyani. Establishing authenticated channels and secure identifiers in ad-hoc networks. *International Journal of Network Security*, 5(1):51–61, July 2007.

[43] Eyad Taqieddin, S. Jagannathan, and Ann Miller. Optimal energy-delay routing protocol with trust levels for wireless ad hoc networks. *International Journal of Network Security*, 7(2):207–217, 2008.

[44] Yuh-Ren Tsai and Shiuh-Jeng Wang. Two-tier authentication for cluster and individual sets in mobile ad hoc networks. *Computer Networks*, 51(3):883–900, February 2007.

[45] Johann van der Merwe, Dawoud Dawoud, and Stephen Mcdonald. A survey on peer-to-peer key management for mobile ad hoc networks. *ACM Computing Surveys*, 39(1):1–45, 2007.

[46] Nen-Chung Wang, Yung-Fa Huang, and Jhu-Chan Chen. A stable weight-based on-demand routing protocol for mobile ad hoc networks. *Information Sciences*, 177(24):5522–5537, 2007.

[47] Wei Wang, Xiaohong Guan, Beizhan Wang, and Yaping Wang. A novel mobility model based on semi-random circular movement in mobile ad hoc networks. *Information Sciences*, 180(3):399–413, 2010.

[48] ZhengYou Xia and Jian Wang. DIMH: A novel model to detect and isolate malicious hosts for mobile ad hoc network. *Computer Standards & Interfaces*, 28(6):660–669, 2006.

[49] Yanchao Zhang, Wei Liu, and Wenjing Lou. Anonymous communications in mobile ad hoc networks. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1940–1951, 2005.