Comput Syst Sci & Eng (2008) 3: 225–232
© 2008 CRL Publishing Ltd

International Journal of
Computer Systems
Science & Engineering

Improving the security of a secure anonymous routing protocol with authenticated key exchange for ad hoc networks*

Chun-Ta Li[1], Min-Shiang Hwang[2] and Yen-Ping Chu[3]

[1] *Department of Management Information Systems, National Chung Hsing University, 250 Kuo Kuang Road, Taichung, Taiwan 402, R.O.C.*
Email: mshwang@nchu.edu.tw, Fax: 886-4-22857173
[2] *Department of Computer Science and Engineering, National Chung Hsing University, 250 Kuo Kuang Road, Taichung, Taiwan 402, R.O.C.*
[3] *Department of Computer Science and Information Engineering, Tunghai University, 181 Taichung Harbor Road, Taichung, Taiwan 407, R.O.C.*

In recent years, several anonymous routing protocols for wireless ad hoc networks have been proposed. In 2007, Lu, Cao, Wang, and Sun proposed a secure routing protocol with anonymity property and key exchange for ad hoc networks. However, in this paper, it is demonstrated that the so-called secure, anonymous key exchange protocol introduced by Lu et al. is vulnerable to a passive attack and, as a result, the anonymity property of the protocol may be compromised. A simple improvement is suggested to eliminate the vulnerability.

Keywords: Ad hoc network; Anonymity; Key exchange; Passive attack; Security.

1. INTRODUCTION

Owing to the widespread infrastructure of the wireless networks, users can easily communicate with others from all over the world by using their wireless devices. Mobile ad hoc networks (MANET) have been one of the most important applications of wireless communication and have attracted much attention in recent years. An ad hoc network is a self-organized system that does not rely on a predeployed infrastructure and consists of available mobile nodes, so every available node acts as both a host and a router on the network. This feature exists because of the open nature and of the network and the lack of base stations. Available nodes communicate among each other using wireless radio technology and generally, communication is established and maintained via a number of intermediate nodes as long as they are not within the radio communication range of each other.

MANET has been widely used in practical applications, such as battlefields, medical or science areas and even in our home life. In such privacy-vital environment, in order to protect the entire network from malicious attacks [6, 15, 17], some security mechanisms are required for providing data security including secure routing [5, 14, 19], user authentication [2, 4, 8], key exchange [9, 11, 18] and anonymity issues etc. Whenever end-to-end ad hoc nodes attempt to transmit secret information, they want to maintain the necessary privacy without being tracked down for whoever they are, wherever they are and whatever they are doing. It is considered as one of the important security requirements that should be paid more attention. Recently, some researches [1, 7, 12, 16] integrate anonymous channels into rout-

*This work was supported in part by Taiwan Information Security Center (TWISC), National Science Council under the grants NSC 95-2218-E-001-001, NSC95-2218-E-011-015, and NSC96-3114-P-001-002-Y.

ing protocol for ad hoc communications so that intruders cannot confirm which one is source node and which one is destination node in the network.

In the case of anonymous routing, for example, Onion Routing [16] is a communication protocol that the property of anonymity is integrated into routing algorithms, and hence it allows communications between source and destination node to proceed anonymously. This is particularly crucial in environments in which privacy is of vital importance. To establish a secure and anonymous path between source and destination node, in an ad hoc network, the anonymous routing protocol is to allow some intermediate nodes to participate in the path construction algorithm without breaking the anonymity of the communication nodes. Recently, Lu et al. [12] proposed a secure anonymous routing protocol with authenticated key exchange (SARPAKE) for ad hoc networks. In their protocol, they combine the techniques of designated verifier signing [3], the employment of a symmetric encryption/decryption function and a public-key cryptosystem. This protocol not only provides an anonymous link from the source node to the destination node but also integrates the deniable authenticated key exchange into the reactive routing algorithm. However, we found that Lu et al.'s SARPAKE protocol is vulnerable to passive attacks and the anonymity property of their protocol may be compromised. Thus, in this paper, an improved protocol is proposed to solve this problem.

The rest of this paper is organized as follows. Section 2 briefly reviews Lu et al.'s SARPAKE protocol and shows our attack in Section 3. Section 4 proposes an improved protocol and provides a security analysis of the improved protocol followed by conclusion in Section 5.

2. REVIEW OF SARPAKE PROTOCOL

There are three phases in the SARPAKE protocol, namely: path discovery, path reverse, and data transfer. Some notations and assumptions used in the SARPAKE protocol are defined in Tables 1 and 2, respectively. In the following subsections, we briefly review Lu et al.'s SARPAKE protocol.

Table 1 Notations used in SARPAKE protocol

p and g	a large prime number p and a primitive root g in $GF(p)$.
N_i	a mobile node, where N_0 is source node, $N_i (1 \leq i \leq n-1)$is the intermediate nodes, and N_n is the destination node.
sk_i	node N_i's private key, where $sk_i = x_i \in GF(p)$.
pk_i	node N_i's public key, where $pk_i = Y_i = g^{x_i} \bmod p$.
tag#	a unique tag number for a route.
$(x \| y)$	concatenation of message x and y.
K_i^j	shred key between N_i and N_j, where $K_i^j = Y_j^{x_i} = g^{x_i x_j} = Y_i^{x_j}$.
hop_{max}	maximum number of hops that N_0 wishes.
$H(\cdot)$	a public one-way hashing function, i.e., SHA-1 [13].
T_i	a timestamp generated by node N_i.
$\triangle$	a notation for a field in a local route table, which records when the value of this filed is temporarily unknown.
$E_{SK}[\cdot]$	the symmetric encryption function with shared session key SK, i.e., AES [13].
$D_{SK}[\cdot]$	the symmetric decryption function with shared session key SK,
$E_{pk_i}(\cdot)$	the asymmetric encryption function with N_i's public key pk.
$D_{sk_i}(\cdot)$	the asymmetric decryption function with N_i' private key sk.

Table 2 Assumptions used in SARPAKE protocol

A-1	The links between connected mobile nodes are bidirectional.
A-2	Each node is capable of executing $E_{pk_i}(\cdot)$, $D_{sk_i}(\cdot)$, $E_{SK}(\cdot)$ and $H(\cdot)$ algorithms.
A-3	Each node maintains a local route table (LRT) for a specific route and the format of LRT is shown as follow. (tag#, Ancestor, Successor, Lifetime)
A-4	Some intermediate nodes along the route may try to break the anonymity property. Thus, SARPAKE scheme assumes that all the intermediates nodes involved in a specific route are not in collusion.

2.1 Path Discovery Phase

In this phase, the source node N_0 establishs a route to the destination node N_n through a number of intermediate ad hoc nodes. The source node N_0 first generates a unique tag# for this route and then performs the following operations:

1. Choose a random number x to compute $X = g^x$ and $h_0 = H(X \| K_0^n)$, where $K_0^n = Y_n^{x_0} = g^{x_0 x_n}$.
2. Compute $C_0 = E_{pk_n}(M_0)$, where M_0=(tag#$\|N_0\|N_n\|X\|h_0$).
3. Send the path discovery packet *packet* to its neighboring nodes, where *packet*=(tag#$\|hop\|C_0$) and $hop = hop_{max}$.
4. Maintain its local route table LRT$_0$=(tag#,0,$\triangle$,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 it will be 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 operations:

1. Check if $(((hop--) \geq 0)$, if it holds, continue; otherwise, stop.
2. Check 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 does, drop it and stop; otherwise, continue.

3. Check if the node is the destined 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.

   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 the neighboring nodes, where hop has been updated.

   b. If the node is the destined receiver, then it verifies that $h_0 \overset{?}{=} H(X||K_n^0)$ 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.

2.2 Path Reverse Phase

In this phase, the destination node would respond to the source node and performs the following operations:

1. Choose a random number y to compute $Y = g^y$ and $h_n = H(Y||K_n^0)$.

2. Compute the shared session key $SK = (X * Y_0)^{y+x_n} = g^{(x+x_0)(y+x_n)}$.

3. Compute $C_n = E_{pk_0}(M_n)$ and $C_{n-1} = E_{pk_{n-1}}(\text{tag\#})$, where $M_n = (tag\#\ ||N_0\ ||N_n\ ||Y\ ||h_n)$.

4. Send $(C_n||C_{n-1})$ to N_{n-1}.

Upon receiving $(C_n||C_{n-1})$ from N_n, N_{n-1} uses its private key sk_{n-1} to recover $tag\# = D_{sk_{n-1}}(C_{n-1})$ and maintains its local route table $LRT_{n-1} = (tag\#,\ N_{n-2},\ N_n,\ T_{n-1})$. 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, …, N_{n-2} along the route would perform the same operation 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 operations:

1. Compute $D_{sk_0}(C_0)$ to recover tag#.

2. Check tag# from its route table LRT$_0$. If it is found, continue; otherwise, stop.

3. Update its route table LRT$_0$=(tag#,0,N_1,T_0).

4. Compute $D_{sk_0}(C_n)$ to recover (tag#$||N_0||N_n||Y||h_n$).

5. Verify $h_n \overset{?}{=} H(Y||K_n^0)$ is true or not. If it does, continue; otherwise, stop.

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

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.

2.3 Data Transfer Phase

In this phase, the source node N_0 sends a confidential message M to the destination node N_n. Firstly, N_0 computes $C = E_{SK}[M]$ and $h = H(C)$. 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, …, N_{n-1} along the route would perform the same operation as the node N_1. In the end, 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 $h \overset{?}{=} H(C)$. If it holds, N_n uses the shared session key SK to recover the message M from C; otherwise, stop.

3. THE WEAKNESS OF SARPAKE PROTOCOL

Lu et al. claimed that the anonymity of the route from source to the destination is guaranteed by using the public key encryption algorithm. However, we found that the anonymity property of the SARPAKE protocol can not be protected from the passive attack in both path reverse phase and data transfer phase. Using such a passive attack the attacker can un-intrusively monitor on the communication channels between ad hoc nodes in the ad hoc network and discovers some valuable information about the messages being transmitted over the communication channel. Knowing this, suppose that an attacker, Eve, could collect all the communication messages transmitted between ad hoc nodes in the ad hoc network. In the following, we show how Eve may then break the anonymity property of Lu et al.'s SARPAKE protocol.

First of all, the message C_n transmitted from destination node to source node is unchanging in the path reverse phase. Thus, Eve could easily trace down the route from destination to source by comparing C_n with all of the collected messages in the ad hoc network. For example, suppose there is a message transmitted between two neighbor nodes containing C_n. This means that these two nodes are involved in a special route. So, Eve can discover the complete route from destination to source as long as the message transmitted between two neighbor nodes contains C_n. Once Eve finds out the complete route from destination to source, Eve can further verify the relation of a route between two neighbor nodes. For example, Eve could get tag# in path discovery phase and transmitted message C_i $(0 \leq i \leq n-1)$ from the channel between N_i and N_{i-1} in path reverse phase. In order to confirm that there is a route between N_i and N_{i-1}, Eve can compute $E_{pk_{i-1}}(\text{tag\#})$ and compare it with C_i. If it holds, it means that Eve discovers the relation of a route between N_i and N_{i-1}. In this way, Eve can verify other complete route information between involved nodes from the destination node to the source node. For computational complexity, it takes n asymmetric encryption operations for Eve to confirm the complete routing path and the Lu et al.'s SARPAKE protocol cannot protect the anonymity of the involved nodes, namely, the source node and destination node.

Similarly, the anonymity property in data transfer phase is not achieved as well as path reverse phase. Due to the message C

transmitted from source node to destination node is unchanging in the data transfer phase. Thus, Eve can discover the complete route from source to destination as long as the message transmitted between two neighbor nodes contains C. Eve can then compute $E_{pk_{i+1}}(\text{tag\#}||h)$ $(0 \leq i \leq n-1)$ and compare it with the collected message CH transmitted between N_i and N_{i+1}. If it holds, Eve confirms that there is a route between these two neighbor nodes. As a result, the involved nodes in a complete route from source to destination would be available to discovery and the anonymity of the data transfer phase would be compromised.

4. THE PROPOSED PROTOCOL

An improved anonymous routing protocol based on the Lu et al.'s SARPAKE protocol is proposed and the detail of the proposed protocol is described in Section 4.1. In addition, we present the security analysis of the improved protocol and evaluate the performance of the improved protocol against Lu et al.'s protocol that are stated in Section 4.2 and 4.3, respectively.

4.1 The Improved Protocol

The notations and assumptions of the improved protocol are the same as those in Lu et al.'s protocol. Also, we extended additional two fields for the local route table and the introduced nonce-key $nk_{i,j}$ in our improved protocol are provided, where $nk_{i,j}$ is generated by node N_i and shared it with node N_j. Further, $nk_{i,j}$ denotes a 128 bit nonce and simultaneously represents a shared session key between N_i and N_j for symmetric encryption/decryption. The format of each entry in the route table is shown as follows.
(tag#, Ancestor, Successor, Lifetime, $nk_{i,j}$ for ancestor, $nk_{i,j}$ from successor) where the preceding four fields including tag#, ancestor, successor and lifetime are the same as those in SARPAKE Protocol. The fifth field records a nonce-key which is generated by the node itself and would be shared with its ancestor node and the sixth field records a nonce-key which is generated by the node's successor node and will shared with the node itself. The details of the improved protocol are described as follows.

4.1.1 Path Discovery Phase

This phase is almost the same as that in Lu et al.'s protocol. The only difference is that the original message M_0 generated by source node N_0 is changed as follows:M_0=(tag#$||N_0||N_n||X||h_0||nk_{0,n}$), where $nk_{0,n}$ is generated by N_0 and it is shared with N_n. In addition, the ancestor field of LRT$_0$ corresponding to the source node N_0 will record the destination node N_n and the fifth field of LRT$_0$ records a nonce-key $nk_{0,n}$. Moreover, the successor field of LRT$_n$ of destination node N_n holds the record of the source node N_0 and the sixth field of LRT$_n$ records a nonce-key $nk_{0,n}$. Figure 1 shows the path discovery phase conducted from source to destination.

4.1.2 Path Reverse Phase

In this phase, the destination node N_n first generates a nonce-key $nk_{n,n-1}$ and then records it in the fifth field of LRT$_n$ and then performs the following operations:

1. Make the message $M_n = (tag\#\ ||N_0\ ||N_n\ ||Y\ ||h_n\ ||nk_{0,n} + 1)$ and use N_{n-1}'s public key pk_{n-1} to encrypt tag# and $nk_{n,n-1}$ as $C_{n-1} = E_{pk_{n-1}}(\text{tag\#}||nk_{n,n-1})$.
2. Moreover, N_n makes $C_n = E_{nk_{n,n-1}}[E_{pk_0}(M_n)]$ and sends $(C_n||C_{n-1})$ to N_{n-1}.

When the node N_{n-1} receives $(C_n||C_{n-1})$ from N_n, it performs the following operations:

1. Use its private key to recover tag# and nonce-key $nk_{n,n-1}$ by computing $D_{sk_{n-1}}(C_{n-1})$ and records $nk_{n,n-1}$ in the sixth field of LRT$_{n-1}$.
2. Use nonce-key $nk_{n,n-1}$ to recover $E_{pk_0}(M_n)$ by computing $D_{nk_{n,n-1}}[C_n]$.
3. Generate a nonce-key $nk_{n-1,n-2}$ and records it in the fifth field of LRT$_{n-1}$.
4. Finally, N_{n-1} uses N_{n-2}'s public key to compute $C_{n-2} = E_{pk_{n-2}}$ $(tag\#\ ||nk_{n-1,n-2})$ and makes $C_n = E_{nk_{n-1,n-2}}[E_{pk_0}(M_n)]$. After that, N_{n-1} sends $(C_n||C_{n-2})$ to its ancestor node N_{n-2}.

Other nodes N_1, N_2, ..., N_{n-2} along the route would perform the same operation as the node N_{n-1}. In the end, when the source node N_0 receives $(C_n = E_{nk_{1,0}}[E_{pk_0}(M_n)]||C_0 = E_{pk_0}(\text{tag\#}||nk_{1,0})$ from N_1, it performs the following operations:

1. Use its private key to recover tag# and nonce-key $nk_{1,0}$ by computing $D_{sk_0}(C_0)$ and records $nk_{1,0}$ in the sixth field of LRT$_0$.
2. Use nonce-key $nk_{1,0}$ to recover $E_{pk_0}(M_n)$ by computing $D_{nk_{1,0}}[C_n]$ and then uses its private key to recover M_n by computing $D_{sk_0}(E_{pk_0}(M_n))$.
3. Verify $nk_{0,n} + 1$ for freshness checking. If it does not hold, stop; otherwise, N_0 computes the shared session key $SK = (Y * Y_n)^{x+x_0} = g^{(x+x_0)(y+x_n)}$ and the route from the source to the destination is established. Figure 2 shows the path reverse phase conducted from destination to source.

4.1.3 Data Transfer Phase

In this phase, the source node N_0 would send a confidential M to the destination node N_n. See Figure 3. Firstly, N_0 computes $C = E_{SK}[M||nk_{0,n} + 2]$ and $h = H(C)$. Then, N_0 sends $(E_{nk_{1,0}}[C]||CH)$ to its successor node N_1, where $CH = E_{pk_1}(\text{tag\#}||h)$. Upon receiving $(E_{nk_{1,0}}[C]||CH)$ 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 $D_{nk_{1,0}}[E_{nk_{1,0}}[C]]$ to recover C and verifies whether or not $H(C)$ equals h. If it does not hold, stop; otherwise, N_1 computes $CH = E_{pk_2}(\text{tag\#}||h)$ and forwards $(E_{nk_{2,1}}[C]||CH)$ to its successor node N_2. Other nodes N_2, N_3, ..., N_{n-1} along the route would perform the same operation as the node N_1. In the end, when the destination node N_n

Path discovery phase		
N_0		N_1
$X = g^x, h_0 = H(X, K_0^n)$		
$M_0 =(\text{tag\#}\|\|N_0\|\|N_n\|\|X\|\|h_0\|\|nk_{0,n})$		
$C_0 = E_{pk_n}(M_0)$		
$packet=(\text{tag\#}\|\|hop\|\|C_0)$		
$LRT_0 \to(\text{tag\#}, N_n, \triangle, T_0, nk_{0,n}, \triangle)$	$\xrightarrow{packet}$	
N_i		$N_{i+1}\ (1 \le i \le n-1)$
$LRT_i \to(\text{tag\#}, N_{i-1}, \triangle, T_i, \triangle, \triangle)$	$\xrightarrow{packet}$	
N_{n-1}		N_n
$LRT_{n-1} \to(\text{tag\#}, N_{n-2}, \triangle, T_{n-1}, \triangle, \triangle)$	$\xrightarrow{packet}$	$M_0 = D_{sk_n}(C_0), h_0 \overset{?}{=} H(X, K_n^0)$
		$LRT_n \to(\text{tag\#}, N_{n-1}, N_0, T_n, \triangle, nk_{0,n})$

Figure 1 Path discovery phase

Path reverse phase		
N_{n-1}		N_n
$(\text{tag\#}\|\|nk_{n,n-1})=D_{sk_{n-1}}(C_{n-1})$		$Y = g^y, h_n = H(Y, K_n^0)$
$E_{pk_0}(M_n) = D_{nk_{n,n-1}}[C_n]$		$session\ key\ SK = (X * Y_0)^{y+x_n}$
$C_n = E_{nk_{n-1,n-2}}[E_{pk_0}(M_n)]$		$M_n =(\text{tag\#}\|\|N_0\|\|N_n\|\|Y\|\|h_n\|\|nk_{0,n}+1)$
$C_{n-2} = E_{pk_{n-2}}(\text{tag\#}\|\|nk_{n-1,n-2})$		$C_n = E_{nk_{n,n-1}}[E_{pk_0}(M_n)]$
$LRT_{n-1} \to(\text{tag\#}, N_{n-2}, N_n, T_{n-1},$		$C_{n-1} = E_{pk_{n-1}}(\text{tag\#}\|\|nk_{n,n-1})$
$nk_{n-1,n-2}, nk_{n,n-1})$		$LRT_n \to(\text{tag\#}, N_{n-1}, N_0, T_n, nk_{n,n-1},$
	$\xleftarrow{(C_n\|\|C_{n-1})}$	$nk_{0,n}+1)$
N_{i-1}		$N_i(1 \le i \le n-1)$
		$C_n = E_{nk_{i,i-1}}[E_{pk_0}(M_n)]$
		$C_{i-1} = E_{pk_{i-1}}(\text{tag\#}\|\|nk_{i,i-1})$
		$LRT_i \to(\text{tag\#}, N_{i-1}, N_{i+1}, T_i, nk_{i,i-1},$
	$\xleftarrow{(C_n\|\|C_{i-1})}$	$nk_{i+1,i})$
N_0		N_1
$(\text{tag\#}\|\|nk_{1,0})=D_{sk_0}(C_0)$		$C_n = E_{nk_{1,0}}[E_{pk_0}(M_n)]$
$E_{pk_0}(M_n) = D_{nk_{1,0}}[C_n]$		$C_0 = E_{pk_0}(\text{tag\#}\|\|nk_{1,0})$
$M_n = D_{sk_0}(E_{pk_0}(M_n)), h_n \overset{?}{=} H(Y, K_0^n)$		$LRT_i \to(\text{tag\#}, N_0, N_2, T_1, nk_{1,0}, nk_{2,1})$
$session\ key\ SK = (Y * Y_n)^{x+x_0}$		
$LRT_0 \to(\text{tag\#}, N_n, N_1, T_0,$	$(C_n\|\|C_0)$	
$nk_{0,n}+1, nk_{1,0})$	$\longleftarrow$	

Figure 2 Path reverse phase

receives $(E_{nk_{n,n-1}}[C]||CH)$ from N_{n-1}, it computes $D_{sk_n}(CH)$ to recover tag# and h and verifies $h \overset{?}{=} H(D_{nk_{n,n-1}}[E_{nk_{n,n-1}}[C]])$. If it does not hold, stop; otherwise, N_n uses the shared session key SK to recover the message M and $nk_{0,n} + 2$ from C. Furthermore, in our protocol, N_n can further verify $nk_{0,n} + 2$ for freshness checking as well as nonce.

4.2 Security Analysis of the Improved Protocol

In the following, we describe how our improved protocol maintains the anonymity of the route from the source to destination in both the path reverse phase and data transfer phases.

1. During the path reverse phase, as pointed out by our attack in Section 3, an attacker Eve traces C_n with all of the collected messages in the ad hoc network and thus the anonymity property of their routing protocol may be compromised. However, as suggested by our improvement in Section 4.1, Eve is unable to trace down the route from destination to source by comparing C_n in our improved routing protocol. Due to the use of nonce-key $nk_{n_i,n_{i-1}}$, the message C_n transmitted between any two neighbor nodes N_i and N_{i-1} in a specific route is constantly changing, where $1 \leq i \leq n$. Thus, Eve cannot discover the complete route from destination to source by comparing C_n. Additionally, as a result of the use of nonce-key $nk_{n_i,n_{i-1}}$, Eve has no way to derive the relation of a route between two neighbors nodes N_i and N_{i-1} by computing $E_{pk_i}(\text{tag\#}||nk_{n_i,n_{i-1}})$ and comparing it with the collected message C_i unless Eve knows the nonce-key $nk_{n_i,n_{i-1}}$. Finally, the anonymity of the route from destination to source can be achieved in this phase.

2. Similarly, during the data transfer phase, before sending the message C to N_{i+1}, N_i uses the shared nonce-key nk_{n_{i+1},n_i} for encryption and the message $E_{nk_{i+1,i}}[C]$ is constantly changing between any two involved nodes N_i and N_{i+1} in a established route, where $0 \leq i \leq n-1$. As a result, Eve has no way to discover the complete route from source to destination by comparing C. In addition, Eve cannot decrypt the message C from $E_{nk_{i+1,i}}[C]$ to compute $h = H(C)$. In essence, Eve cannot compute the $CH = E_{pk_i}(tag\#||h)$ to derive the route from the source to destination and the anonymity can be protected in our improved protocol.

4.3 Performance Evaluation of the Improved Protocol

In this subsection, we evaluate the performance of the improved protocol and compare it with original SARPAKE protocol in terms of efficiency in Table 3. For memory requirement, our improved protocol introduces a nonce-key mechanism that must be generated and stored in the local route table, before communications can begin. Therefore, the necessity to add two fields and store ancestor/successor nonce keys in each node of the improved protocol. In addition, to compare the computational cost of the improved protocol against SARPAKE protocol, we measured the cryptographic operations needed to secure the transmission packets for path reverse phase and data transfer phase. To evaluate performance, we define several computational parameters as follows:

Table 3 Comparisons of the efficiency

	Our improved protocol	SARPAKE protocol [12]
Path Discovery Phase	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}	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: 2 T_{asym}+1 T_{hash}+1 T_{exp}+1 T_{sym} Intermediate node: 2 T_{asym}+2 T_{sym} Destination node: 2 T_{exp}+1 T_{hash}+2 T_{asym}+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: 1 T_{asym}+1 T_{hash}+2 T_{sym} Intermediate node: 2 T_{asym}+2 T_{sym} Destination node: 1 T_{asym}+1 T_{hash}+2 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}

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

The computational cost of path discovery phase taken by our improved protocol and SARPAKE protocol is identical. During path reverse phase of our protocol and compare it to that of SARPAKE protocol, although the source and destination node perform one additional symmetric operation for each C_n, and all the intermediate nodes perform two additional symmetric operations for each C_n per session, the weakness of SARPAKE protocol would be prevented by our improvement. Similarly, during the data transfer phase of our protocol, the source and destination node perform one additional symmetric operation for $E_{nk_{1,0}}[C]$ and $D_{nk_{n,n-1}}[E_{nk_{n,n-1}}[C]]$, and other intermediate nodes N_1, N_2, …, N_{n-1} along the route perform two additional symmetric operations for each $D_{nk_{i,i-1}}[E_{nk_{i,i-1}}[C]]$ and $E_{nk_{i+1,i}}[C]$ per session, where $1 \leq i \leq n-1$. The computational cost that we found [10] for one symmetric operation was about 10 one-way hashing operations and so, it is clear that the overhead of 10-20 one-way

Data transfer phase		
N_0		N_1
$C = E_{SK}[M\|\|nk_{0,n}+2],\ h = H(C)$		$\text{tag\#}\|\|h = D_{sk_1}(CH)$
$CH = E_{pk_1}(\text{tag\#}\|\|h)$		$C = D_{nk_{1,0}}[E_{nk_{1,0}}[C]],\ H(C) \overset{?}{=} h$
	$\xrightarrow{(E_{nk_{1,0}}[C]\|\|CH)}$	$CH = E_{pk_2}(\text{tag\#}\|\|h)$
N_i		$N_{i+1}\ (1 \le i \le n-1)$
$CH = E_{pk_{i+1}}(\text{tag\#}\|\|h)$		$\text{tag\#}\|\|h = D_{sk_{i+1}}(CH)$
		$C = D_{nk_{i+1,i}}[E_{nk_{i+1,i}}[C]],\ H(C) \overset{?}{=} h$
	$\xrightarrow{(E_{nk_{i+1,i}}[C]\|\|CH)}$	$CH = E_{pk_{i+2}}(\text{tag\#}\|\|h)$
N_{n-1}		N_n
$CH = E_{pk_n}(\text{tag\#}\|\|h)$		$\text{tag\#}\|\|h = D_{sk_n}(CH)$
		$C = D_{nk_{n,n-1}}[E_{nk_{n,n-1}}[C]]$
		$H(C) \overset{?}{=} h$
	$\xrightarrow{(E_{nk_{n,n-1}}[C]\|\|CH)}$	$M\|\|nk_{0,n}+2 = D_{SK}[C]$

Figure 3 Data transfer phase

hashing operations for each node is negligible, especially in view of the level of security the improved protocol offers.

5. CONCLUSIONS

In this paper, we point out that anonymity property of Lu et al.'s SARPAKE protocol is vulnerable to a passive attack. As a result, an attacker can easily find out the complete route from source node to destination node in both path reverse phase and data transfer phase. More importantly, a new improved proposal for a protocol that can resist the passive attack and still maintain the anonymity between communicating nodes in a MANET is demonstrated.

REFERENCES

1. 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, June 2005.
2. Zhenchuan Chai, Zhenfu Cao, and Rongxing Lu. Threshold password authentication against guessing attacks in ad hoc networks. *Ad Hoc Networks*, 2006.
3. X. Huang, W. Susilo, Y. Mu, and F. Zhang. Short (identity-based) strong designated verifier signature schemes. In *Second International Conference Information Security Practice and Experience*, pages 214–225, Lecture Notes in Computer Science, 2006.
4. Min-Shiang Hwang and Chi-Yu Liu. Authenticated encryption schemes: Current status and key issues. *International Journal of Network Security*, 1(2):61–73, Sep. 2005.
5. 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.
6. Nikos Komninos, Dimitris Vergados, and Christos Douligeris. Detecting unauthorized and compromised nodes in mobile ad hoc networks. *Ad Hoc Networks*, 5(3):289–298, April 2007.
7. J. Kong and X. Hong. Anodr: Anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In *Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 291–302, 2003.
8. 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.
9. Jung-San Lee and Chin-Chen Chang. Secure communications for cluster-based ad hoc networks using node identities. *Journal of Network and Computer Applications*, 2006.
10. Chun-Ta Li, Min-Shiang Hwang, and Yen-Ping Chu. A secure and efficient communication scheme with authenticated key establishment and privacy preserving for vehicular ad hoc networks. *Computer Communications*, 2007.
11. Zhenjiang Li and J.J. Garcia-Luna-Aceves. Non-interactive key establishment in mobile ad hoc networks. *Ad Hoc Networks*, 2006.
12. 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.
13. W. Mao. *Modern Cryptography: Theory and Practice.* Prentice Hall PTR, 2003.

14. C. Perkins, E. Belding-Royer, and S. Das. *Ad Hoc on-Demand Distance Vector (AODV) Routing*. IETF RFC 3561, 2003.
15. Asad Amir Pirzada and Chris McDonald. Detecting and evading wormholes in mobile ad-hoc wireless networks. *International Journal of Network Security*, 3(2):191–202, Sep. 2006.
16. M. Reed, P. Syverson, and D. Goldschlag. Proxies for anonymous routing. In *Proceedings of the 12th Annual Computer Security Applications Conference*, pages 95–104, 1995.
17. 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.
18. 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.
19. Feng Zhu, Matt Mutka, and Lionel Ni. Facilitating secure ad hoc service discovery in public environments. *The Journal of Systems and Software*, 76(1):45–54, April 2005.