Secure OLSR

Fan Hong
College of Computer, Huazhong University of Science & Technology
Wuhan 430074, China

Liang Hong
College of Computer, Huazhong University of Science & Technology
Wuhan 430074, China

Cai Fu
College of Computer, Huazhong University of Science & Technology
Wuhan 430074, China

kaiserking@sohu.com

Abstract

Mobile Ad hoc Networks (MANET) is a new networking paradigm for wireless hosts. Because of infrastructureless, self-organization, dynamic topology and openness of wireless links, the routing security problem in MANET is more seriously than in wired networks. Optimized Link State Routing (OLSR)[1] is proposed by IETF's MANET Group at 2003. In OLSR, neighbor detection is not invulnerable when two bad nodes perform wormhole attack. Further more, OLSR's security cannot simply rely on IPSec, because OLSR's packets are often broadcasted and IPsec provides end-to-end security. In this paper, we propose a solution to secure OLSR, which apply the wormhole detective mechanism and authentication to strengthen the neighbor relationship establishment, and use hash-chain and digital signature to protect the routing packets.

1. Introduction

An ad hoc network is a collection of mobile computers (or nodes) that cooperate to forward packets for each other to extend the limited transmission range of each node's wireless network interface. An ad hoc network is often defined as an "infrastructureless" network, meaning a network without the usual routing infrastructure like fixed routers and routing backbones. A routing protocol in such a network finds routes between nodes, allowing a packet to be forwarded through other network nodes towards its destination.

Due to "infrastructureless", dynamic topology, and openness of wireless links, ad hoc network routing protocols face more security problems than traditional network routing protocols.

Several routing protocols for ad hoc networks have been developed, particularly in the MANET working group of the Internet Engineering Task Force (IETF). Surveys of routing protocols for ad hoc wireless networks are presented in [2] and [3]. In this paper, we consider the security of the OLSR protocol. Section 2 takes a look at related work. Section 3 describes OLSR's security flaws and analyzes the security requirements of OLSR. Section 4 presents our plan to protect OLSR's neighbor detection process and the routing information. Section 5 analyzes our plan. Section 6 summarizes and shows the next step of this work.

2. Related Work

The collaborative, self-organizing environment of the Mobile Ad Hoc Networking technology opens the network to numerous security attacks that can actively disrupt the routing protocol and disable communication. Attacks on ad hoc network routing protocols generally fall into one of two categories: 1) Routing-disruption attacks. The attacker attempts to cause legitimate data packets to be routed in dysfunctional ways. 2) Resource-consumption attacks. The attacker injects packets into the network in an attempt to consume valuable network resources such as bandwidth or to consume node resources such as memory (storage) or computation power.

Recently, a number of protocols have been proposed to secure wireless ad hoc routing. Papadimitratos and Haas proposed the Secure Routing Protocol [4], which we can use with DSR or the Interzone Routing Protocol in the Zone Routing Protocol (ZRP). They designed SRP as an extension header that is attached to ROUTE REQUEST and ROUTE REPLY packets. SRP doesn't attempt to secure ROUTE ERROR packets but instead delegates the route-maintenance function to the Secure Route Maintenance portion of the Secure Message Transmission protocol. SRP requires that, for every route discovery, source and destination must have a security association between them. Furthermore, the paper does not even mention route error messages. Therefore, they are not protected, and any malicious node can just forge error messages with other nodes as source. Ariadne [5] is a secure on-demand routing protocol based on DSR and TESLA, which withstands node compromise and relies on highly efficient symmetric cryptography and requires clock synchronization. ARAN [6] is based on AODV and proposed by Dahill In ARAN, each node has a certificate signed by a trusted authority. Every node that forwards a route discovery or a route reply message must also sign it, which is very computing power consuming and causes the size of the routing messages to increase at each hop. Manel Guerrero Zapata and N. Asokan propose Secure AODV

Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05)
1550-445X/05 $20.00 © 2005 IEEE

(SAODV)[7], another protocol designed to secure AODV. The idea behind SAODV is to use a signature to authenticate most fields of a route request (RREQ) and route reply (RREP) and to use hash chains to authenticate the hop count. SAODV designs signature extensions to AODV. Network nodes authenticate AODV routing packets with an SAODV signature extension, which prevents certain impersonation attacks.

3. OLSR

3.1 Overview

Optimized Link State Routing Protocol (OLSR) was proposed in [1]. OLSR aims at large and dense MANETs.

It is based on a Multipoint Relaying (MPR) flooding technique to reduce the number of topology broadcast packets. The procedure of OLSR is:

(1) Every node broadcasts HELLO messages that contain one-hop neighbor information periodically. The TTL of HELLO message is 1, so its neighbors do not forward them. With the aid of HELLO messages, every node obtains local topology information. Please notice here, the neighbor not forwarding HELLO message isn't forced, that means if someone forwards the message the sender wouldn't know it.

(2) A node (also called selector) chooses a subset of its neighbors to act as multi-point relaying nodes for it based on the local topology information, which are specified in the periodic HELLO messages later. MPR nodes have two roles: 1) When the selector sends or forwards a broadcast packet, only its MPR nodes among all its neighbors forward the packet; 2) The MPR nodes periodically broadcast its selector list throughout the MANET (again, by means of MPR flooding). Thus every node in the network knows by which MPR node the node could be reached.

Note that there is no security guarantee to ensure the selected MPR node is not a bad man.

(3) With global topology information stored and updated at every node, a shortest path from one node to every other node could be computed with Dijkstra's algorithm, which goes along a series of MPR node.

3.2 Security flaws of OLSR

Since OLSR has no security mechanisms, malicious nodes can perform many attacks by these ways as follows:

1. There is no security mechanism for a good node to distinguish an attacker from his neighbors, once an attacker becomes his MPR node, then the attacker can create a blackhole which drops all packets from or to the selector, or just drops the packets selectively, or temper the packet contents and then relay it. All of those behaves can cause the good node cannot work normally;
2. An attacker can generate lots of false Topology Control Message to broadcast. Because there is no source authentication, other nodes will accept it and update the global topology information.
3. OLSR don't protect the routing packets in networks, so an attacker can easily modify them and won't be detected.
4. OLSR use HELLO packets for neighbor detection, so if an attacker tunnels to B all HELLO packets transmitted by A and tunnels to A all HELLO packets transmitted by B, then A and B will believe that they are neighbors, which would cause the routing protocol to fail to find routes when they aren't actually neighbors. This attack is called wormhole attack [8].

Here we describe the 4th attack in detail. As shown in Figure 1, A and B are two good nodes, but they cannot hear each other. M is a bad nodes, he can hear A and B.

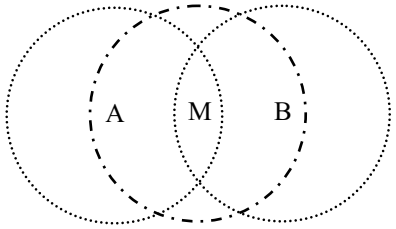

Figure 1. Three nodes' location sketch map

The attack is processed by M as follows:

1. A → *: HELLO_MESSAGE, A, neighbor(A);
   Here neighbor(A) denotes A's neighbors.
2. M → * **:** HELLO_MESSAGE, A, neighbor(A);
3. B hears the A's HELLO_MESSAGE, so he thinks A is his new neighbor, then he inserts A to his neighbor table.
4. B → * : HELLO_MESSAGE, B, neighbor(B);
5. M → * : HELLO_MESSAGE, B, neighbor(B);
6. A hears the B's HELLO_MESSAGE, so he thinks B is his new neighbor, then he inserts B to his neighbor table.

So M makes that A and B establishing the neighbor relationship. This attack can be detected by Watchdog [9], which maintaining a buffer of recently sent packets and comparing each overheard packet with the packet in the buffer to see if there is a match. If so, the node can determine that there is a bad node relaying the HELLO_MESSAGE.

But this method cannot take affect when two bad nodes collude to create wormhole. The attack figure is show in Figure 2.

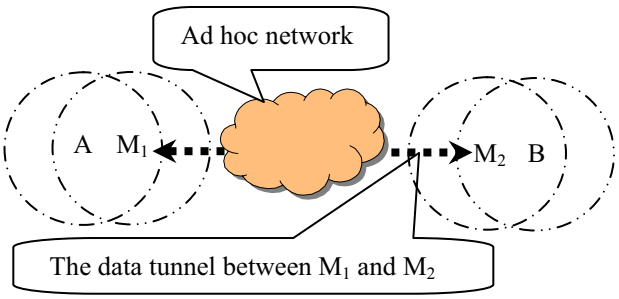

Figure 2. Two bad nodes perform wormhole attack

Wormhole attack:

1. $A \rightarrow *$: HELLO_MESSAGE, A, neighbor(A);
2. $M_1 \rightarrow M_2$: $\{\text{HELLO\_MESSAGE, A, neighbor(A)}\}_k$;
   Here *k* is the shared key between M_1 and M_2.
3. $M_2 \rightarrow *$: HELLO_MESSAGE, A, neighbor(A)
4. B hears the A's HELLO_MESSAGE, he will insert A to his neighbor table.
5. $B \rightarrow *$: HELLO_MESSAGE, B, neighbor(B)
6. $M_2 \rightarrow M_1$: $\{\text{HELLO\_MESSAGE, B, neighbor(B)}\}_k$
7. $M_1 \rightarrow *$: HELLO_MESSAGE, B, neighbor(B)
8. A hears the B's HELLO_MESSAGE; he will insert B to his neighbor table.

In this way, watchdog cannot detect this attack, because the HELLO packet is encrypted by M_1 and sent to M_2. When M_2 receives the packet, he decrypts the packet and broadcasts. This attack will cause the routing protocol to fail to find routes when A and B aren't actually neighbors.

3.3 Security Requirement in OLSR

1. We should provide a mechanism to detect the wormhole attack during node's performing neighbor detection;
2. We should import identity authentication before two nodes begin to establish the neighbor relationship. This prevents the bad node from becoming one good node's MPR node;
3. We need to be able to verify that the routing packet's sender is the one it claims to be;
4. In addition, we need to be able to verify that the arrived routing packet has not been altered during transfers.

4. Securing OLSR

4.1 Overview

Here we propose our plan — SOLSR which will secure OLSR from two sides. First, we strengthen the neighbor relationship establishment. Only the node who is in my radio coverage and passes the identity authentication can become my neighbor. Second, we secure the routing packet to protect the packet's integrity and offer the source authentication. At the same time because OLSR's each routing packet owns a unique serial number, if we can prevent the number from altering, then we will block the replay attack.

Next we will describe the plan in detail.

4.2 Assumptions

The physical layer of a wireless network is often vulnerable to denial of service attacks such as jamming. Mechanisms such as spread spectrum have been extensively studied as means of providing resistance to physical jamming. Medium Access Control protocols are also often vulnerable to attack. But in this paper, we disregard attacks on these two layers.

We assume that network links are bidirectional, if node A is able to transmit to some node B, then B is able to transmit to A.

We assume that each legitimate node is equipped with a public/private key pair, named K and K^{-1}. There is a key certification that can be provided by a coalition of N nodes and the use of threshold cryptography, or a distributed instantiation of a *CA*. Here we should notice that only legitimate nodes could get key certifications. This paper won't discuss the legitimate node's Byzantine behavior.

The content of the key certificate are the identity of the node i, the node's public key, and the validity period of the certificate, all these are encrypted by the CA's private key, i.e.

$$Cert_i =< ID_i, K_i, VTime_i >_{K_{CA}^{-1}}$$

We assume that the network may drop, corrupt, reorder, or duplicate packets in transmission.

4.3 Secure Neighbor Detection

From the analysis, we find that the wormhole attack on neighbor detection is a threat to the routing protocols in wireless networks.

In a wormhole attack, an attacker receives the packets at one point in the network, "tunnels" them to another point in the network, and then replays them into the network from that point. The tunneled distances are concealed with the normal wireless transmission range of a single hop. This is due to the "neighbor" inaccurately defined, once the node receives some node's HELLO packet, he will think the sender is his neighbor, even the packet is relayed from far away.

Now we define the neighbor as follows:

Only the node that is within the range of a single hop far from me and then passes the identity authentication can be my neighbor.

The wormhole's distance must be longer than a single hop distance, so the time of a packet traveling from the wormhole must be longer than from a single hop. If we can get the packet's actual travel time *t*, then we can calculate the actual travel distance $L = t \times c$, where c is the propagation speed of our wireless signal (i.e., the speed of light in air, which is very close to the speed of light in a vacuum). At the same time, the node's wireless LAN adapter's radio coverage R is known to us, for example, 3Com® 11 Mbps Wireless LAN PCI Adapter link coverage is 100m. Thus if L>R, there may exist a wormhole, if L<=R, there not. From this way we can detect whether there exists a wormhole or not.

In our protocols — SOLSR, when a node *B* receives some node *A*'s HELLO packets who is a new face, he will perform the follow steps:

- *B* sends a probe packet to *A,* and at the same time starts a time counter.
- When *A* receives the probe packet, he will send a reply immediately and starts a time counter.
- Once *B* receives *A's* reply, he stops the time counter, and send a reply to A immediately. After *B* gets the time interval $\triangle t_b$, the distance *S* between *A* and *B* is $(\triangle t_b/2) \times c$, if $S>R$, *B* will not insert *A* into the neighbor table, or else will take the identity authentication process with *A*.
- When *A* receives *B*'s reply, he stops the time counter, writes down the time interval $\triangle t_a$, and calculates the distance $S=(\triangle t_a/2) \times c$, if $S>R$, *A* won't insert *B* into the neighbor table, or else if $S<=R$, *A* will takes the identity authentication process with *B*

When passing the neighbor detection, the next is the identity authentication.

- *A* generates a packet which contains a big random R_a, the key certificate, its identifier, and the packet's hash values, then sends it to B:

  $A \rightarrow B: A, B, Cert_A, R_A, sign(H(A, B, Cert_A, R_A))$;

  Here H() is a hash function, *sign*() is denoted as digital signature operation.
- After *B* receives the packet, at first he verifies the *A*'s key certificate, then he gets *A*'s public key from the certificate to verify the packet's digital signature. When all those done, *B* sends a packet which contains a big random number R_b, its key certificate, its node identifier and the packet's hash values.

  $B \rightarrow A: B, A, Cert_B, R_B, sign(H(B, A, Cert_B, R_A, R_B))$

- After *A* receives *B*'s packet and verifies *B*'s certificate and the packet's signature, **A** answers *B* a packet and believes that **B** is a neighbor.

  $A \rightarrow B: A, B, sign(H(A, B, R_A, R_B))$

- When *B* receives *A*'s reply, he will know that *A* is his neighbor.

After passing these two steps, nodes *A* and *B* can establish the neighbor relationship. The frame of the secure neighbor detection is showed in Figure 3.

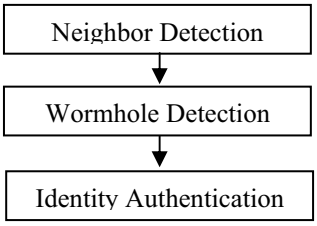

Figure 3. Framework of the secure neighbor detection

4.4 Secure Routing Packets

OLSR communicates using a unified packet format for all data related to the protocols. The basic packet in OLSR is as follows (omitting IP and UDP headers):

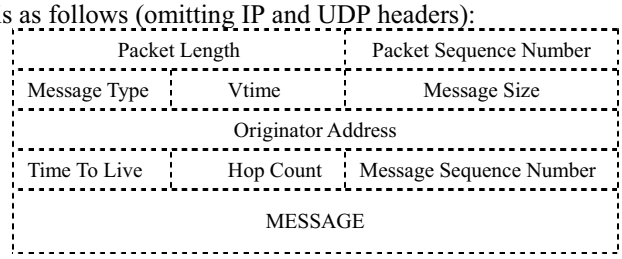

Figure 4. OLSR packet format

We can divide the fields of the packet into two kinds: ones is the unchanged during transmission, such as Packet Length, Packet Sequence Number, Vtime, Originator Address, Message Type etc., another is changeable, such as Time To Live and Hop Count. We can use digital signature to ensure the non-mutable fields unaltered before received. As for the last kind, we can use hash chains to secure Hop Count and TTL. The security extensions are as follows.

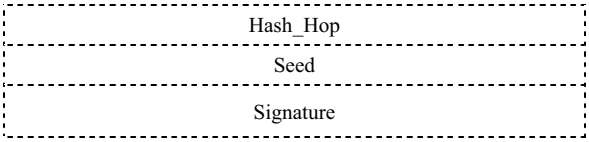

Figure 5. OLSR packet's security extensions format

When a node generates a new packet, it performs the following steps:

- Generates a random number *Seed*.
- $Hash\_Hop = H^{Time\ To\ Live}(Seed)$.

- *Signature* = *Sign* (The unchanged contents).

Every time a node receives an OLSR packet, it performs the following operations:

- First, performs $H^{Time\ To\ Live-Hop\ Count}(Seed)$, if the value equals the *Hash_Hop* in the packet, then verifies the packet's signature, if correct, does next step.
- *Time to Live* = *Time to Live* –1;
  Hop Count = *Hop Count* +1;
  Seed = *H* (*Seed*)

Thus SOLSR provide a mechanism, which include source authentication and integrity protection, for the receiver to verify the received packet validation.

5. Analysis

SOLSR's security is based on the assumption "only the legitimate node can get the key certificate from authority". So the malicious node can't get the key certificate, then he can't generate the validate signature which means he can't generate false Topology Message or alter other's routing packets undetectably. And at the same time he also can't pass the identity authentication, which means that he has no chance to be one good node's MPR node. Also SOSLR apply the wormhole detection to prevent bad nodes performing wormhole attack.

There are two ways for nodes to get its certificates. One is from the certificate authority [10]. We can define one or more certificate authorities (CAs) to take charge of signing the legitimate node's certificate. Another is from transitive trust and PGP trust graphs [11]. In this way, each node signs certificates for other nodes. A node can search the network for a chain of certificates leading from the node initiating the query and ending at the node trying to authenticate a message. Of course, such schemes require transitive trust.

Next we present a formal analysis of the identity authentication course and verify that the goals are achieved. The analysis follows the methodology of BAN logic [12]. We follow the notation and inference rules in [13].

Three exchanged messages in the identity authentication course can be formalized as follows. *A* and *B* are two principals, K_s^{-1} is the *CA's* private key.

1. $A \to B: \{\xrightarrow{k_a} A\}_{k_s^{-1}}, R_b, \{H(\xrightarrow{k_a} A, R_a)\}_{k_a^{-1}}$
2. $B \to A: \{\xrightarrow{k_b} B\}_{k_s^{-1}}, R_b, \{H(\xrightarrow{k_b} B, R_a, R_b)\}_{k_b^{-1}}$
3. $A \to B: \{R_a, R_b\}_{k_a^{-1}}$

The purpose of the identity authentication is that after three messages exchanged A will believe the message 2's signature is come from B and B believes the signatures of message 1 and 3 are come from A. In a word, the aims are $A \models B \ni k_b^{-1}$ and $B \models A \ni k_a^{-1}$.

To analyze the protocol, we first give the following assumptions:

(i) $A \models \xrightarrow{k_a} A$, $A \models \xrightarrow{k_s} S$, $A \ni k_s$,

$B \models \xrightarrow{k_b} B$, $B \models \xrightarrow{k_s} S$, $B \ni k_s$

$A \models S \Rightarrow \xrightarrow{k_b} B$, $B \models S \Rightarrow \xrightarrow{k_a} A$

$A \models \#(R_a)$, $B \models \#(R_b)$

$A \models \#(\{\xrightarrow{k_b} B\}_{k_s^{-1}})$, $A \models \phi(\{\xrightarrow{k_b} B\}_{k_s^{-1}})$

$B \models \#(\{\xrightarrow{k_a} A\}_{k_s^{-1}})$, $B \models \phi(\{\xrightarrow{k_a} A\}_{k_s^{-1}})$

The first group of six is about the node's public key and private key and the *CA*'s public key. The next group of two indicates that *A* and *B* believe the authority of *S*. And the next group of two indicates *A* and *B* can generate the fresh random number. The final group of four indicates *A* and *B* believes the certificate can be recognized and is generated freshly by *CA*.

(ii) Now from recognizability rules, we can obtain:

$$\frac{B \models \phi(\{\xrightarrow{k_a} A\}_{k_s^{-1}}),\ B \ni k_s}{B \models \phi(\xrightarrow{k_a} A)}$$

B receives Message 1, and then B can get:

$$\frac{B < \{\xrightarrow{k_a} A\}_{k_s^{-1}},\ B \ni k_s,\ B \models \phi(\xrightarrow{k_a} A)}{B \models S|: \{\xrightarrow{k_a} A\}}$$

Use the freshness rules:

$$\frac{B \models \#(\{\xrightarrow{k_a} A\}_{k_s^{-1}}),\ B \ni k_s}{B \models \#(\xrightarrow{k_a} A)}$$

From the two previous results, we get:

$$\frac{B \models S|: \{\xrightarrow{k_a} A\},\ B \models \#(\xrightarrow{k_a} A)}{B \models S| \models \xrightarrow{k_a} A}$$

Now using the jurisdiction rules, we get:

$$\frac{B \models S \Rightarrow \xrightarrow{k_a} A,\ B \models S| \models \xrightarrow{k_a} A}{B \models \xrightarrow{k_a} A}$$, which means *B* believes K_a is public key of *A*.

(iii) When *A* receives the message 2, similarly *A* can get the following:

$$\frac{A \models \phi(\{\xrightarrow{k_b} B\}_{k_s^{-1}}),\ A \ni k_s}{A \models \phi(\xrightarrow{k_b} B)};$$

$$\frac{A < \{\xrightarrow{k_b} B\}_{k_s^{-1}},\ A \ni k_s,\ A \models \phi(\xrightarrow{k_b} B)}{A \models S|: \{\xrightarrow{k_b} B\}};$$

$$\frac{A \models \#(\{\xrightarrow{k_b} B\}_{k_s^{-1}}),\ A \ni k_s}{A \models \#(\xrightarrow{k_b} B)};$$

$$\frac{A \models S \models \{\xrightarrow{k_b} B\} \quad , \quad A \models \#(\xrightarrow{k_b} B)}{A \models S \models \xrightarrow{k_a} B}$$

$$\frac{A \models S \Rightarrow \xrightarrow{k_b} B \quad , \quad A \models S \models \xrightarrow{k_b} B}{A \models \xrightarrow{k_b} B}$$

Now we get that A believes K_b is B's public key. At the same time, using freshness rules, A gets:

$$\frac{A \models \#(R_a)}{A \models \#(\xrightarrow{k_b} B, R_a, R_b)},$$

$$\frac{A \models \#(\xrightarrow{k_b} B, R_a, R_b)}{A \models \#(H(\xrightarrow{k_b} B, R_a, R_b))}$$

From above result and using message interpretation rules 5, we get:

$$\frac{A \ni \{H(\xrightarrow{k_b} B, R_a, R_b)\}_{k_b^{-1}}, A \ni k_b, A \models \xrightarrow{k_b} B, A \models \#(H(\xrightarrow{k_b} B, R_a, R_b)), \ A \models \#(H(\xrightarrow{k_b} B, R_a, R_b))}{A \models B \ni k_b^{-1}}$$

Now, A believes B has k_b^{-1};

(iv) From message 3 and (ii), B get:

$$\frac{B \ni \{H(R_a, R_b)\}_{k_a^{-1}}, B \ni k_a, B \models \xrightarrow{k_a} A, B \models \#(H(R_a, R_b)), B \models \#(H(R_a, R_b))}{B \models A \ni k_a^{-1}}$$

Now B also believes that A has k_a^{-1}.

From above analysis, we find that after three messages exchanged identity authentication achieves the goal.

6. Conclusion and Future work

In this paper we proposed a solution to secure OLSR for mobile ad hoc networks. SOLSR secure neighbor discovery and the use of wormhole detective mechanism defend against the wormhole attack. Applying identity authentication sets up the reliable communication between two nodes. The solution also provides source authentication and integrity verification mechanism for the receiver to verify the received packets validation.

As the next step of our research, digital signatures have been used for each routing packets in our plan, so we want to provide a reasonable digital signature process for nodes based on recently research. Moreover we want to present a detailed performance evaluation of SOLSR for various network instances and node processing capabilities.

References

[1] T. Clausen, Ed., P. Jacquet, Ed., "Optimized Link State Routing Protocol (OLSR)", *IETF INTERNET DRAFT*, RFC 3626, 2003.

[2] S. Ramanathan and M. Streenstrup, "A survey of routing techniques for mobile communications networks", *Mobile networks and Applications*, 1(2): 89-104, 1996.

[3] E.M.. Royer and C.-K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks", *IEEE Personal Communications*， pp 46-55, Apr,1999.

[4] P. Papadimitratos and Z.J. Haas, "Secure Routing for Mobile Ad Hoc Networks," *Proc. SCS Communication Networks and Distributed Systems Modeling and Simulation Conf.* (CNDS 2002), Jan. 2002

[5] Y.-C. Hu, A. Perrig, and D.B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks", *Proc. 8th Ann. Int'l Conf. Mobile Computing and Networking* (MobiCom 2002), ACM Press, 2002, pp.12–23.

[6] K. Sanzgiri et al., "A Secure Routing Protocol for Ad Hoc Networks", *Proc. 10th IEEE Int'l Conf. Network Protocols* (ICNP '02), IEEE Press, 2002, pp. 78–87.

[7] M. Guerrero Zapata and N. Asokan, "Securing Ad Hoc Routing Protocols", *Proc. ACM Workshop on Wireless Security* (WiSe), ACM Press, 2002, pp. 1–10.

[8] Y.-C. Hu, A. Perrig, and D.B. Johnson, "Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks", *Proc. 22nd Ann. Joint Conf. IEEE Computer and Communications Societies* (INFOCOM 2003), IEEE Press, 2003, pp. 1976–1986.

[9] S. Marti et al., "Mitigating Routing Misbehaviour in Mobile Ad Hoc Networks", *Proc. 6th Ann. Int'l Conf. Mobile Computing and Networking* (MobiCom 2000), ACM Press, 2000, pp. 255–265.

[10] L. Zhou and Z.J. Haas, "Securing Ad Hoc Networks", *IEEE Network Magazin*e, IEEE Press, vol. 13, no. 6, 1999, pp. 24–30.

[11] S. Capkun, L. Buttyan and J.-P. Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks", *IEEE Transactions On Mobile Computer*, IEEE Press, vol.2, no.1, 2003, pp. 52–63.

[12] M. Burrows, M. Abadi, and R. Needham, "A Logic of Authentication", *ACM Transactions on Computer System*, vol.8, no. 1, February 1990, pp. 18–36.

[13] L. Gong, R. Needham, and Yahalom, "Reasoning about Belief in Cryptographic Protocols", *Proceeding of the 1990 IEEE Symposium on Research in Security and Privacy,* IEEE Computer Society Press, 1990, pp. 234–248.