01 Jan 2006

A Hierarchical Secure Routing Protocol Against Black Hole Attacks in Sensor Networks

Jian Yin

Sanjay Kumar Madria
Missouri University of Science and Technology, madrias@mst.edu

Recommended Citation

J. Yin and S. K. Madria, "A Hierarchical Secure Routing Protocol Against Black Hole Attacks in Sensor Networks," *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2006. The definitive version is available at https://doi.org/10.1109/SUTC.2006.1636203

A Hierarchical Secure Routing Protocol against Black Hole Attacks in Sensor Networks*

Jian Yin and Sanjay Kumar Madria
Department of Computer Science, University of Missouri-Rolla, MO 65401, USA
{jian, madrias}@umr.edu

Abstract

A black hole attack is a severe attack that can be easily employed against routing in sensor networks. In a black hole attack, a malicious node spuriously announces a short route to the sink node (the destination) to attract additional traffic to the malicious node and then drops them. In this paper, we propose a hierarchical secure routing protocol for detecting and defending against black hole attacks. The proposed protocol uses only symmetric key cryptography to discover a safe route against black hole attacks. The comparison of the proposed protocol with two other existing approaches proves that the proposed scheme is faster in detecting black hole attacks with much lower message overhead.

Keywords Black hole attack, Sensor networks

1. Introduction

Many sensor network applications, such as border security application [1], run in untrustworthy environments, which require secure communication [2-4] against different types of attacks. However, traditional security protocols [5] are designed for resource rich machines with large computation, which are not applicable to sensor networks due to resource limitation. Secure routing in sensor networks presents challenges due to low computing power, small memory, limited bandwidth, and especially very limited energy [6]. A black hole attack [5] is a severe attack that can be easily employed against routing in sensor networks. In a black hole attack, a malicious node spuriously announces a short route to the sink node in order to attract additional traffic to the malicious node and then drops them. The black hole attack forms a serious threat as data packets are dropped by the black hole node.

In this paper, a hierarchical secure routing protocol (HSRBH) for detecting and defending against black hole attacks is presented. It uses only symmetric key cryptography to discover a safe route against black hole attacks. In the HSRBH protocol, the sensor network is divided into different groups with one group leader in each group through the localized self-organization process. Group members are organized in a tree structure with a group leader as the root. An intra-group shared key among the neighbors of the group leader and an inter-group shared key between the two neighboring group leaders are established. Most of the black hole attacks are detected only locally. To detect the black hole attacks which are caused by the cooperation between the group leader and its neighbor, the randomized data acknowledgement scheme is proposed. The proposed HSRBH protocol is robust against black hole attacks. Comparing with the two existing schemes AODVBH [5] and CoBH [7], the HSRBH scheme detects the black holes faster while establishing the route with much lower communication overhead.

The rest of the paper is organized as follows. Section 2 gives a motivating example. Section 3 presents the system model. Section 4 describes the HSRBH protocol. Section 5 gives the theoretical analysis. Section 6 provides the performance evaluations. Section 7 gives related work, and section 8 concludes the paper.

2. Motivating Example

In the border security application, sensor nodes are deployed along the border area, which monitor the border and send the information only if they detect new events such as people crossing the border. Heidemann et al. [8] mentioned that ad hoc routing such as AODV and DSR can be used in sensor networks. However, the table driven scheme [9] is energy consuming since the route update must be done frequently. That's the reason our HSRBH scheme uses on-demand routing scheme.

2.1. Problem Statement

In a black hole attack, a malicious node spuriously announces a short route to the sink node in order to attract additional traffic to it and then drops them. It sends route replies immediately after receiving the

*This research is partially supported by Intelligent Research Center, UMR, and NSF (EIA-0323630).

route request even though it does not have any route. If the route reply from the malicious node reaches the source node earlier, the source node establishes the route to the sink node through the malicious node. Normally the route reply from the malicious node can have more chances to be accepted by the source node. The black hole attack forms a serious threat as data packets are dropped by the black hole node.

3. System Model

This section provides the system model based on the system description, localized self-organization and key establishment.

3.1. System Description

Two types of sensor nodes are deployed in the sensor network [10]: level-0 and level-1 sensor nodes. The level-1 nodes are manually deployed [11]. The distribution of level-1 nodes is roughly uniform. The number of level-1 nodes is 10-20% of the total number of sensor nodes. We assume that all the level-0 and level-1 nodes are preloaded with a global initial key K_0 in memory. Besides, each node holds its individual secret key shared with the sink node. Each node has a unique identity (*ID*) and a preloaded pseudo-random function *f* [4]. We assume that the sink node is secure and trusted, whereas all other sensor nodes can be compromised. However, we assume there is a lower bound on the time interval T_{min} (about several seconds [4]) that is necessary for an adversary to compromise a node. Table 1 provides the notation description.

Table 1. Notation description

Notation	Description
A, B	Principles, such as sensor nodes
$M_1 \| M_2$	The concatenation of messages M_1 and M_2
K_{AB}	The secret shared key shared between *A* and *B*
K_A	The secret key held by *A*
MAC(*K,M*)	The message authentication code of message *M* using a symmetric key *K* [12]
$E_K(M)$	Encryption of message M with key *K* [13]
$f_K(u)$	A pseudo-random function with inputs of symmetric key K and the identity of node u
N_A	A nonce generated by node *A*, which is a random number
ID_A	The identity of node *A*

3.2. Localized Self-Organization

The level-1 nodes as the group leaders start the localized self-organization by sending a hello message $<ID_G|N_G|MAC(K_0,ID_G|N_G)>$, where ID_G is the group leader's *ID*, and N_G is a nonce. Each node (say node A) only accepts the first verified hello message through *MAC*. The receiver A sets the sender as its parent. It then sends a reply message $<ID_A|ID_G|N_G|MAC(K_0,ID_A|ID_G|N_G)>$ to the sender. It also broadcasts the updated hello message $<ID_A|ID_G|N_A|MAC(K_0, ID_A|ID_G|N_A)>$. The group leader accepts the verified reply message through *MAC* and sets the node A as its child. This process is recursively continued. When a sensor node receives a hello message, it must decide whether it stops broadcasting it. If it has received a hello message from other group leaders earlier, it stops broadcasting. After a certain time, it reports its all group leaders about all other group leader's *ID*. Then the localized self-organization process ends. After the localized self-organization process, group members are organized in a tree structure with the group leader as the root. The information stored in each group leader includes (a) one-hop neighbors' *ID*, and (b) its neighbor group leader's *ID*. The information stored in other sensor nodes includes (a) parent node's *ID*, (b) child node's *ID*, and (c) group leader's *ID*.

3.3. Key Establishment

This section provides inter-group shared key establishment between two neighboring group leaders and intra-group shared key establishment among the one-hop neighbors of the group leader.

3.3.1. Inter-group Shared Key Establishment. First, the group leader derives its own secret key $f_{K_0}(ID)$ [4]. Second, it derives its neighboring group leaders' secret key $f_{K_0}(ID)$, where *ID* is its neighboring group leaders' *ID*. Third, it establishes the inter-group shared key with its neighboring group leaders. Let node A_0 and B_0 be group leaders. The shared key K_{AB} establishment is as follows. If $ID_{A_0} \geq ID_{B_0}$, node A_0 computes the pair-wise shared key $K_{AB} = f_{K_{B_0}}(ID_{A_0})$. Node B_0 computes K_{AB} as $K_{AB} = f_{K_{B_0}}(ID_{A_0})$ independently. Fourth, after a specified time (much less than T_{min}), all sensor nodes other than the sink node remove K_0, *f* and all secret keys of other nodes.

3.3.2. Intra-group Shared Key Establishment. The group leader sends the message $\{ID|ID_R|MAC(K_0, ID|ID_R)\}$ to its one-hop neighbors, where ID_R is a random *ID*. The neighbors then verify the authentication of the message through MAC. Then the neighbors establish the intra-group shared key $f_{K_0}(ID_R)$. Finally, the group leader A_0 removes the initial key K_0, the random function *f*, and random identity ID_R. Therefore, only the neighbors of the group leader share the intra-group shared key.

4. Secure Routing Protocol against Black Hole Attacks

This section provides the HSRBH protocol. We first give an overview (Section 4.1) and then provide the detailed protocol (Section 4.2).

4.1. An Overview

After the localized self-organization and key establishment process, the sensor network is divided into groups. Each group has a group leader. An intra-group shared key among neighbors of the group leader and an inter-group shared key between the two neighboring group leaders are established.

In the HSRBH protocol, the source node initiates the route discovery process through sending a route request (*RREQ*) to the sink node. When the sink node or an intermediate group leader having a fresh enough route receives the *RREQ*, it generates a route reply (*RREP*) and sends it to the source node. Each *RREP* includes the message authentication code (MAC) which is calculated using the inter-group shared key. For each *RREP* packet, two verification steps are executed as follows. First step, the neighbor of the group leader who generates the *RREP* packet must make the verification through sending a further verification message to the next hop of the group leader on the route to the sink node. The further verification message includes a MAC, which is calculated using intra-group shared key. The next hop receiving the verification message sends the verification result to the sender. If the verification succeeds, the node forwards the *RREP* packet to its previous node. Otherwise, if the verification fails or the node did not get the verification result within a certain time, the node drops the *RREP* packet. Second step, the neighboring group leader receiving the *RREP* packet must verify the authentication of the *RREP* packet through MAC since only the two neighboring group leaders share their inter-group shared key. If the verification succeeds, the neighboring group leader forwards the *RREP* packet to its previous node. Otherwise, it must drop it.

The nodes between two neighboring group leaders send the *RREP* packet immediately after receiving the *RREQ* packet even though they do not have any route to make the black hole attack. These attacks can be locally detected using the two verification steps.

The most challenging black hole attack is made by the collusion between the group leader and other nodes. The black hole node's goal is to drop the packets. We use the randomized data acknowledgement mechanism to detect this attack. In this mechanism, the source node randomly sends the control message to the sink node to send the acknowledgement. The acknowledgement includes the MAC using the shared key between the source node and the sink node. If the source node can receive the acknowledgement and the verification succeeds, the route is secure against the black hole. Otherwise, the route is considered to have a black hole node inside. Then the source node adds the node generating the *RREP* to its black list. The source node will not accept any *RREP* from the nodes in the black list.

After the secure route discovery, each node on the route has built the appropriate routing table, which includes the next hop on the route to the sink node.

4.2. Protocol Description

This section gives a detailed description of HSRBH including a route request process, a route reply process, a data acknowledgement process, and a route maintenance process.

4.2.1. Route Request. Source node (S) initiates the route discovery by broadcasting *RREQ* as follows:

$S \rightarrow *: ID_S|ID_{sink}|ID_{RREQ}|Seq|N_S|$
$\text{MAC}(K_S, ID_S|ID_{sink}|ID_{RREQ}|Seq|N_S)$

where ID_{RREQ} is a random number and *Seq* is the sequence number. When the intermediate node (node I) receives it, six cases will be considered:

Case 1: If the *RREQ* is not from its group leader and it is not the parent of the sending node, it drops it.

Case 2: If the *RREQ* is not from its group leader but it is the parent of the sending node, and if it has already received the same packet, it drops it. If it has not receive it, it updates its routing table to add ID_{send}, ID_S, ID_{sink} and ID_{RREQ}. Here ID_{send} is the ID of the sending node. Then it sends the *RREQ* as follows:

$I \rightarrow *: ID_I|ID_S|ID_{sink}|ID_{RREQ}|Seq|N_S|$
$\text{MAC}(K_S, ID_S|ID_{sink}|ID_{RREQ}|Seq|N_S)$

Case 3: If the *RREQ* is originally from its group leader and it is not the child of the sending node or it has received the RREQ before, it drops it.

Case 4: If the *RREQ* is originally from its group leader and it is the child of the sending node, and if it has not received it before, it updates its routing table with ID_{send}, ID_S, ID_{sink} and ID_{RREQ}. Then it sends the *RREQ*:

$I \rightarrow *: ID_I|ID_G|ID_S|ID_{sink}|ID_{RREQ}|Seq|N_S|$
$\text{MAC}(K_S, ID_S|ID_{sink}|ID_{RREQ}|Seq|N_S)$

Case 5: If it is the group leader which receives the *RREQ* and if it has received it before, it drops it.

Case 6: If it is the group leader which receives the *RREQ* and if it has not received it before, it checks its routing table to determine whether it has a fresh enough route. If it has a fresh route, it generates a *RREP* to the source node *S*. If it has no fresh route, it updates its routing table with ID_{send}, ID_S, ID_{sink} and ID_{RREQ}. Then it sends the updated *RREQ* as follows:

$G \rightarrow *: ID_G|ID_s|ID_{sink}|ID_{RREQ}|Seq|N_S|$
$\text{MAC}(K_S, ID_S|ID_{sink}|ID_{RREQ}|Seq|N_S)$

When the sink node receives the *RREQ*, it derives the source node *S* secret key $K_S = f_{K_0}(ID_S)$. It only accepts the *RREQ* with the valid *MAC*, which first reaches it. If *Seq* is bigger than the same route stored in its routing table, it updates the routing table to store ID_S, ID_{RREQ}, *Seq*, and the previous group leader's *ID*. Finally, it sends the *RREP* to the source node.

4.2.2. Route Reply. As shown in Figure 1, if the sink node generates a *RREP*, it sends the *RREP* as follows:

$Sink \rightarrow C$: $ID_P|ID_C|ID_S|ID_{sink}|ID_{RREQ}|Seq|N|$
MAC($K_{<C, sink>}$, $ID_S|$ $ID_{sink}|$ $ID_{RREQ}|$ $Seq|N|ID_C)|$
MAC(K_S, $ID_S|$ $ID_{sink}|$ $ID_{RREQ}|Seq|N$)

where ID_P is the *ID* of the previous node, ID_C is the *ID* of the previous group leader and $K_{<C, sink>}$ is the inter-group shared key between C and the sink node.

If the intermediate group leader generates a *RREP* packet, it sends the *RREP* packet as follows:

$C \rightarrow B$: $ID_P|ID_B|ID_C|$ ID_{C_2} $| ID_S|ID_{sink}|ID_{RREQ}|Seq|N|$
MAC(K_{BC}, $ID_S|ID_{sink}|ID_{RREQ}|Seq|N|$ ID_{C_2} $|ID_C|ID_B)|$

where ID_P is the *ID* of the previous node, ID_B is the *ID* of the previous group leader B, ID_C is the *ID* of the sending group leader C, ID_{C_2} is the ID of the next node C_2.

When the previous node of the group leader receives the *RREP* generated by the group leader, it sends a verification message to the next hop of the group leader as follows:

$C_1 \rightarrow C_2$: ID_{C_1} $| ID_{C_2}$ $| ID_S|ID_{sink}|$ $Seq|$
MAC($K_{C_1C_2}$, $ID_S|$ $ID_{sink}|$ $Seq|$ ID_{C_1} $| ID_{C_2}$)

where ID_{C_1} is the ID of the previous node, ID_{C_2} is the ID of the next hop, and $K_{C_1C_2}$ is the intra-group key among the neighbors of the group leader C.

The next hop of the group leader sends the verification result to the previous node of the group leader as follows:

$C_2 \rightarrow C_1$: ID_{C_1} $| ID_{C_2}$ $|ID_S|ID_{sink}|Seq|R_{verify}|$MAC($K_{C_1C_2}$,
$ID_S|$ $ID_{sink}|$ $Seq|$ ID_{C_1} $| ID_{C_2}$ $|R_{verify})$

Where R_{verify} is the result of verification. If C_2 has fresh enough route to sink, R_{verify} is *YES*. Otherwise, it is *NO*.

If the verification result is *NO*, the next hop of the group leader has no fresh enough route and node C_1 drops the *RREP*. If the verification result is *YES*, the next hop of the group leader has a fresh route. Then the *RREP* is sent to the previous node. The intermediate node receiving it updates its routing table.

When the group leader receives the *RREP*, it verifies the first *MAC* to make sure it is from the next group leader. If the verification fails, it drops it. If the *ID* of the previous group leader embedded in the *RREP* is not the *ID* of the current group leader, it drops it. Otherwise, it updates its routing table. Then, if the *RREP* is originally from the sink node, the group leader sends the *RREP* as follows (in Figure 1):

$B \rightarrow A$: ID_{B_1} $|ID_A|ID_B|$ ID_{B_2} $|ID_S|ID_{sink}|ID_{RREQ}|Seq|N|$
MAC($K_{AB}|ID_S|ID_{sink}|ID_{RREQ}|Seq|N|$ ID_{B_2} $|ID_B|ID_A)|$
MAC(K_S,$ID_S|ID_{sink}|ID_{RREQ}|Seq|N$)

where ID_{B_1} is the ID of the previous node B_1, A and B are the neighboring group leaders, and ID_{B_2} is the ID of the next node B_2.

If the *RREP* is originally from the intermediate group leader C, it sends the *RREP* as follows:

$B \rightarrow A$: ID_{B_1} $|ID_A|ID_B|$ ID_{B_2} $|ID_C|ID_S|ID_{sink}|ID_{RREQ}|Seq|N|$
MAC(K_{AB},$ID_C|ID_S|ID_{sink}|ID_{RREQ}|Seq|N|$ ID_{B_2} $|ID_B|ID_A)|$

where ID_{B_1} is the ID of the previous node B_1, A and B are the neighboring group leaders, ID_{B_2} is the ID of the next node B_2, and ID_C is the ID of the node who generates the *RREP*.

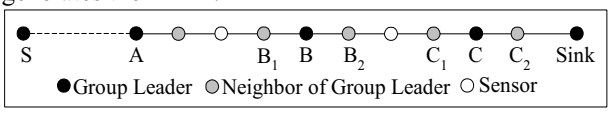

Figure 1. Route discovery

When the source node receives the *RREP*, it verifies the first *MAC*. If the *RREP* has not been tampered with, the source node inserts the *ID* of the next hop on the route and the ID of the node who generates the *RREP* packet to its routing table. If the source node cannot receive the *RREP* within a specified period, it triggers another route discovery.

4.2.3. Data Acknowledgement. The data acknowledgement mechanism is used to detect the black hole attack due to the collusion between the group leader and other nodes. It is designed based on the observation that the black hole node will drop the data packets during the data dissemination process. The data acknowledgement mechanism includes two phases: control message forwarding phase, data acknowledgement phase.

Control Message Forwarding Source node sends the data packets to the sink node at a message rate R_d. Here the message rate is defined as the number of the messages per minute. It also sends control message to the sink node at a message rate R_c. R_c can be changed within a small range randomly and is designed much smaller than R_d to decrease message overhead. The control message includes the information to ask the sink node to send the acknowledgement, which is carefully crafted. For example, the control message can be of the same size as of data packet. It is encrypted. The control message is sent as follows:

$S \rightarrow Sink$: $ID_N|ID_C|ID_S|ID_{sink}|N_S|$ $E_{K_S}(M)|$
MAC(K_S,$ID_S|ID_{sink}|N_S|$ $E_{K_S}(M)$)

where, ID_N is the *ID* of the next hop on the route, and node C is the group leader generating the *RREP*.

Data Acknowledgement The sink node sends the acknowledgement as follows:

$Sink \rightarrow S$: $ID_P|ID_S|ID_{sink}|N_S|N_{Sink}|$
MAC(K_S,$ID_S|ID_{sink}|N_S|N_{Sink}$)

Here, ID_P is the *ID* of the previous hop in the route.

If the source node receives the acknowledgement and passes the verification through *MAC*, the route is secure against black hole attack. Only the sink node can create the correct MAC using K_S since only the source node and the sink node holds the source node's secret key K_S. If the verification fails, the source node will suspect that the route may not be secure against black hole attacks. To further confirm this judgment, the source node sends further control messages more frequently at a message rate R_c' ($R_c < R_c' < R_d$). If it can not get the correct acknowledgement within a threshold time, the source node can be sure that the route is not secure against black hole attacks. It then adds the group leader which generates the *RREP* into black hole list. If it can successfully receive the data acknowledgement within a threshold time, the route is secure against black hole attacks.

Note that messages could be lost either because of the black hole attack or because of the collision. But we assume in this paper that collision could be handled [14]. For example, the sending node can resend the data packets when it finds that the data packet is lost because of the collision. But the black hole node will not send the data packets again after it drops them since its objective is to drop the packets to disrupt the sensor network.

4.2.4. Route Maintenance. If a sensor node wants to send data to the sink node and there is no route information in its routing table, it initiates the route discovery. If the source node gets the error message after it sends data or routing packet, or the threshold time for the data acknowledgement is expired, or run out of the specified waiting time from the last route discovery, it triggers another route discovery. It never accepts *RREP* generated by the group leader in its black hole list.

5. Theoretical Analysis

This section gives security analysis against different types of black hole attacks, and cost analysis.

5.1. Security Analysis

In our sensor network architecture, four types of black hole attacks could occur: (1) A single node other than the group leader makes the black hole attack; (2) Several nodes between two neighboring group leaders collude to make the black hole attack; (3) A single group leader makes the black hole attack; (4) The group leader colludes with its neighbors to make the black hole attack.

5.1.1. Single Node other than the Group Leader Makes the Black Hole Attack. In Figure 2, node S is the source node, nodes A and B are the neighboring group leaders. When the malicious node M_1 receives the *RREQ*, it immediately sends the *RREP* to node A even it does not have fresh route to the sink. In our secure routing protocol, group leader A receives the *RREP* and verifies whether the *RREP* is from its neighboring group leader. Only group leader B or A can generate the correct *RREP* with *MAC* calculated using K_{AB} since only group leader A and its neighboring group leader B shares an inter-group shared key K_{AB}. If M_1 generates the *RREP* after it receives the *RREQ*, the group leader A can detect it immediately through the verification of *MAC*.

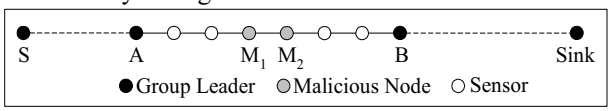

Figure 2. Black hole attack

5.1.2. Multiple Nodes between Two Neighboring Group Leaders Collude to Make a Black Hole Attack. In [5], the author wants to detect whether *RREP* is from the black hole node by sending further *RREQ* to its next hop node M_2. But if the next hop node M_2 is also a malicious node, called a colluding black hole attack, the [5] does not provide any solution to this problem. Our secure routing protocol can detect this kind of colluding black hole attack where several malicious nodes collude to act as black holes between two neighboring group leaders. This is because the group leader A can verify *RREP* through MAC using the shared key K_{AB}, which is shared between two group leaders A and B.

5.1.3. Single Group Leader Makes the Black Hole Attack. The HSRBH protocol can detect this kind of attack using local verification. When one of the group leader's neighbors receives the *RREP*, it verifies the authenticity of the route from the next hop of the group leader in the path to the sink. If the group leader is part of the black hole attack, the verification result will be "*NO*". This verification result can not be tampered by the group leader since it does not have the intra-shared key of its neighbors. The spoofed *RREP* will not be forwarded to the next group leader by the node that had received the *RREP* earlier.

5.1.4. Group Leader Colludes with its Neighbors to Make a Black Hole Attack. The probability of this

kind of black hole attack is low since the attacker needs special effort to conquer multiple nodes' key. However, our proposed secure routing protocol can also detect this kind of attack through data acknowledgement process. The source node randomly sends control messages and waits for the data acknowledgement from the sink node. The data acknowledgement is signed through MAC using the source node's secret key, which is executed only by the source node or the sink node since only they hold the source node's secret key.

5.2. Cost Analysis

This section gives the computational complexity and storage overhead analysis. Define N as the total number of sensor nodes, N_G as the number of groups, N_{nb} as the average number of neighboring groups for each group leader, and M as the number of nodes to make black hole attacks.

5.2.1. Computational Complexity. The worst case is that none of group leaders have the fresh enough route to the destination. The source node sends the *RREQ* once to start the secure route discovery process, thus the computational cost is *O(1)*. Each intermediate sensor node only accepts the first *RREQ*, then broadcasts it, thus the total cost is *O(N-2)*. The cost that the sink node generates *RREP* is *O(1)*. Since the number of sensor nodes in one group is N/N_G ($<<N$) and the number of hops between two neighboring group leaders is much smaller than N/N_G, we can assume the number of hops between two neighboring group leaders is constant. We assume the total number of hops along the route between the source node and the sink node is N_H, then the cost that RREP is forwarded to the source node is $O(N_H)$, which is *O(1)* since in general $N_H << N$ if the sensor nodes are deployed in a rectangle region (length × breadth), where length is not far away from breadth. The total cost is *O(N)*. The computational cost is not relevant to the number of black holes since most of the black hole attacks are detected only locally. However, the computational cost of AODVBH [5] is $O(M \cdot N)$. The computational comlexity of CoBH [7] is $O(M \cdot N)$. The computational cost for both of AODVBH [5] and CoBH [7] increases when the number of black hole attacks increase.

5.2.2. Storage Overhead. In CoBH [7], each node maintains an additional Data Routing Information (DRI) table. Each node stores two additional bits in DRI for each other node. The total additional storage for maintaining DRI for each node is *O(2N)* bits, that is *O(N/4)* bytes. If there are 1000 sensor nodes, each node must maintain additional at least 256 bytes for DRI storage. In the HSRBH, all the nodes maintain a shared key with the sink node. Each group leaders also need to maintain additional inter-group shared key with its neighboring group leaders. All the one-hop neighbors of the group leader also need to maintain additional intra-group shared key. We assume the key size is S_k bytes. The sensor node other than the group leader need the additional key storage $O(S_k)$. The group leader needs the additional key storage including shared key with the sink node and shared key with its neighboring group leader, which is $O((1+N_{nb})S_k)$. The one-hop neighbors of the group leader needs the additional key storage including shared key with the sink node and shared key with other one-hop neighbors of the group leader, which is $O(2S_k)$. If the key size is 8 bytes [2], the node other than the group leader needs additional 8 bytes storage for key information. Assume that the average number of neighboring group leaders for each group leader is 6 then each group leader needs to maintain additional 56 bytes for key storage. Each one-hop neighbor of the group leader needs to maintain additional 16 bytes for key storage.

In the sensor network, the storage is only 512 bytes RAM, and 512 bytes EEPROM [2]. The HSRBH only uses a small part of memory and thus it is suitable to be used in the sensor network, whereas CoBH [7] needs huge additional storage, which is not suitable for the sensor network. AODVBH [5] does not need any additional information, Therefore the storage overhead is low. However, it only considers single black hole, which is obviously not enough to build a secure route against different kinds of black hole attacks including cooperative black hole attacks.

6. Performance Evaluations

The metrics such as route discovery overhead and black hole detection time are measured from the simulation results. We evaluate the proposed HSRBH by comparing it with AODVBH [5] and CoBH [7].

6.1. Performance Comparison: Non Cooperative Black Holes

The message overhead to build a secure route against multiple non cooperative black hole attacks is shown in Figure 3. The message overhead in HSRBH includes the message overhead in the initialization phase and in the route discovery. The message overhead in AODVBH and CoBH includes route requests and route replies. The message overhead for HSRBH is almost same when the number of black holes increases, which is consistent with the computation cost analysis result of O(N), where N is the number of sensor nodes. But it is much lower than the message overhead of the AODVBH scheme and

CoBH scheme. This is because in HSRBH the black hole node can be detected only locally. But, for both AODVBH and CoBH, if the source node gets route reply, it will send an additional route request to the next hop of the black hole node to verify the route reply. When the number of black holes increases, more additional route discoveries are triggered, which is consistent with the computation cost analysis result of AODVBH and CoBH $O(M \cdot N)$, where M is the number of black hole nodes, and N is the total number of sensor nodes.

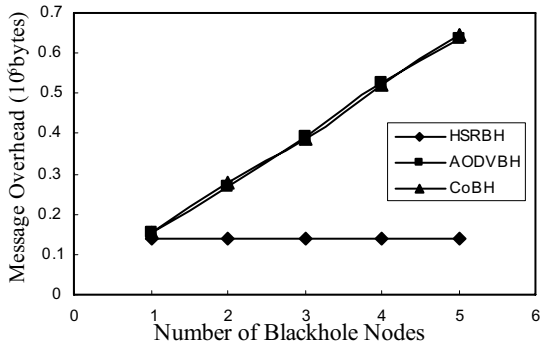

Figure 3. Route discoveries overhead

The average time to detect the black holes is shown in Figure 4. HSRBH is faster to detect malicious nodes acting as black holes than AODVBH and CoBH. This is because in HSRBH the detection of black hole is executed only locally. But in AODVBH or CoBH the source node needs to send another route request to the next hop of the intermediate node to verify the authenticity of the route. This verification process greatly delays the detection of the black hole node.

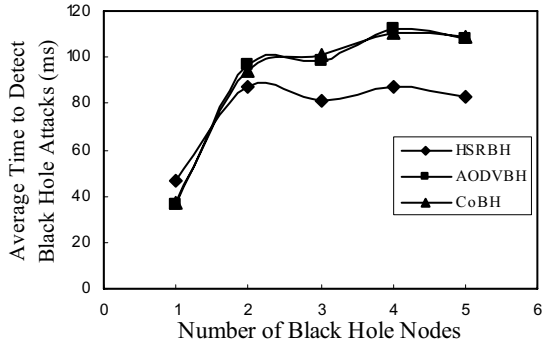

Figure 4. Average time to detect black hole

6.2. Performance Comparison: Cooperative Black Holes

The message overhead for HSRBH protocol is much lower than the message overhead for CoBH scheme as shown in Figure 5. This is because in HSRBH the black hole nodes can be detected locally whenever they are single nodes acting as black holes or a set of cooperative black holes except the group leaders. But, for CoBH, the source node always sends an additional route request to the next hop of the black hole node until the next hop of the black hole is a reliable node [7]. When the number of cooperative black holes increases, more additional route discoveries are triggered. Therefore, the message overhead in route discovery in CoBH is much more than in HSRBH.

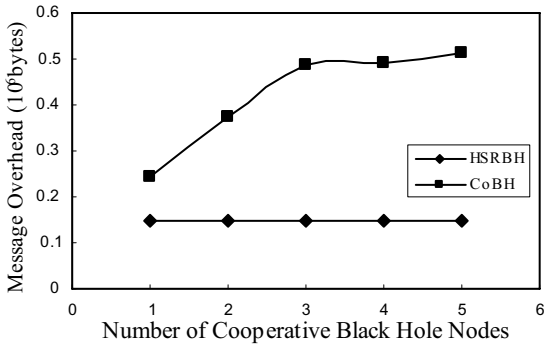

Figure 5. Messages overhead during route discovery

As shown in Figure 6, HSRBH is faster to detect malicious nodes making cooperative black hole attacks than CoBH. This is because in HSRBH the detection of cooperative black holes is executed only. But in CoBH if several nodes cooperatively make a black hole attack, the source node sends another route request to the next hop of the next hop of black hole node until it finds a normal node with a route to the sink. Therefore, the time to detect cooperative black holes increases with the increase in the number of cooperative black holes. This verification process in CoBH greatly delays the detection of black hole.

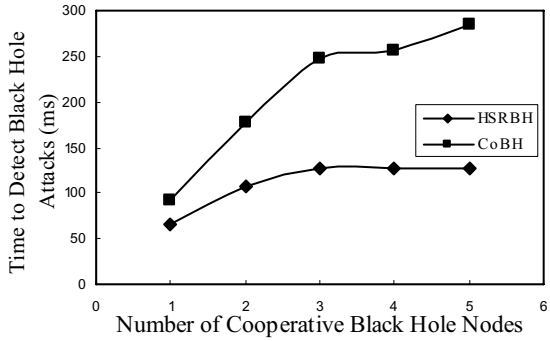

Figure 6. Time to detect cooperative black holes

7. Related Work

Perrig et al. [2] presented two security protocols, SNEP and μTESLA. SNEP provides data confidentiality, two-party data authentication, and data freshness, while μTESLA provides authenticated broadcast for severely resource-constrained

environments using a time-released key chain. Zhu et al. [4] proposed localized encryption and authentication protocol (LEAP) and established four types of keys for each sensor node. Oliveira et al. [10] proposed a solution for securing heterogeneous hierarchical sensor network with an arbitrary number of levels. Deng et al. [5] proposed a solution for the black hole problem in AODV routing protocol. They allowed the intermediate node to send a reply message if it had a fresh enough route to the destination. But the intermediate node could be a malicious node and could send route reply even if it had no fresh enough route to the destination to make a black hole attack. They proposed a solution that the source node would send another route request to the next hop of the intermediate node to verify the authenticity of the route from the intermediate node to the destination node. If the route exists, the intermediate node is trusted; otherwise, the reply message from the intermediate node is discarded. Ramaswamy et al. [7] addressed the problem of coordinated attack by multiple black holes acting in group in mobile ad hoc networks (MANETs). They proposed a technique using the Data Routing Information (DRI) table to identify multiple black holes cooperating with each other and discover a safe route avoiding cooperative black hole attack.

8. Conclusions

In this paper, we proposed a hierarchical secure routing protocol to detect and find a secure path against black hole attacks in sensor networks. The protocol uses only symmetric key cryptography to discover a safe route against black hole attacks. Most of black hole attacks except the group leader colludes with other nodes to make black hole attack, are detected only locally. Therefore, it is much faster in detecting the black hole attacks, and the message overhead is very low. The proposed protocol also provides the scheme to detect the black hole attack caused by the group leader colluding with other nodes. The simulation results show that the proposed protocol has a much better performance for secure route discovery against black hole attacks in comparison with the secure AODV against black hole presented in [5] and the protocol given in [7] against cooperative black hole attacks.

9. References

[1] Sasikanth Avancha, Jeffrey Undercoffer, Anupam Joshi, John Pinkston. Secure Sensor Networks for Perimeter Protection. Computer Networks: The International Journal of Computer and Telecommunications Networking, Vol43 Issue 4 November 2003.

[2] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J.D. Tygar. SPINS: Security Protocols For Sensor Networks. In *Proceedings of Mobicom*, 2001.

[3] C. Karlof and D. Wagner, Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures. *First IEEE International Workshop on Sensor Network Protocols and Application*, May 2003.

[4] S. Zhu, S. Setia, and S. Jajodia. LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. In *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS '03)*, Washington D.C., October, 2003.

[5] Hongmei Deng, Wei Li, and Dharma P. Agrawal. Routing Security in Wireless Ad Hoc Networks. *IEEE Communications Magazine*, vol. 40, no. 10, October 2002.

[6] M. Tubaishat, J. Yin, B. Panja, and S. Madria. A Secure Hierarchical Model for Sensor Network. *ACM SIGMOD Record*, Vol. 33, No. 1, March, 2004.

[7] Sanjay Ramaswamy, Huirong Fu, Manohar Sreekantaradhya, John Dixon, and Kendall Nygard. Prevention of Coopoerative Black Hole Attack in Wireless Ad Hoc Networks. *In proceedings of the 2003 International Conference on Wireless Networks (ICWN'03)*, Las Vegas, Nevada, USA, June 2003.

[8] John Heidemann, Fabio Silva, Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, and Deepak Ganesan. Building Efficient Wireless Sensor Networks with Low-Level Naming. In *Proceedings of the Symposium on Operating Systems Principles*, pp. 146-159. Chateau Lake Louise, Banff, Alberta, Canada, ACM. October, 2001.

[9] Anthony D. Wood and John A. Stankovic. Denial of Service in Sensor Networks. *IEEE Computer*, 35(10):54-62, 2002.

[10] Leonardo B. Oliveira, Hao Chi Wong, and Antonio A. F. Loureiro. LHA-SP: Secure Protocols for Hierarchical Wireless Sensor Networks. *9th IFIP/IEEE International Symposium on Integrated Network Management (IM'05)*, May 2005, Nice, France, (pages 31-44).

[11] Vijay Kumar, Daniela Rus, Sanjiv Singh. Robot and Sensor Networks for First Responders. *IEEE Pervasive Computing*, vol.03, no.4, pp.24-33, October-December, 2004.

[12] Krawczyk, H., Bellare, M., and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. *RFC 2104*, February 1997.

[13] R. L. Rivest. The RC5 Encryption Algorithm. *Proc. 1st Workshop on Fast Software Encryption*, pages 86–96, 1995

[14] R. Gandhi, S. Parthasarathy, and A. Mishra. Minimizing Broadcast Latency and Redundancy in Ad Hoc Networks. MobiHoc'03, Annapolis, Maryland, USA, June 2003.