A new broadcast-Key Management Scheme for Distributed Wireless Sensor Networks

YingZhi Zeng[1], Yan Xia[1,2], JinShu Su[1], BaoKang Zhao[1], WanRong Yu[1]

[1]School of Computer
National University of Defense Technology
ChangSha Hunan, China
zyz1234@gmail.com

[2]School of Computer and Communication
Hu'nan University
ChangSha Hunan, China
xiayan@hnu.cn

***Abstract*—The management of broadcast-key is one of the most important security problems in the distributed wireless sensor networks. Firstly, the broadcast-key should be calculated and used by each node. Secondly, the broadcast-key should also be updated according to the variation of security condition. In this paper, we propose a new broadcast-key management scheme which has many advantages over the famous μTESLA protocol. The analysis in this paper demonstrates its feasibility, efficiency and security for broadcast-key establishment and renewing in distributed Wireless Sensor Networks.**

Keywords- broadcast-key; key management; wireless sensor networks; network security

I. INTRODUCTION

In a distributed wireless sensor networks (DWSN), the Base Station (BS) may need to broadcast new commands or messages to all the nodes [1]. Due to the unattended deployment and working environment, those broadcast messages have to be encrypted by a kind of key equipped by each node. We call this key as the broadcast-key or the global-key. If the monitor environment is very large, some nodes have to act as relay nodes to re-broadcast the messages sent by the BS. The broadcast message can be propagated to all the nodes hop by hop. However, the wireless connectivity, the absence of physical protection, the close interaction between sensor nodes and their physical environment, and the unattended deployment of sensor nodes make them highly vulnerable to node capture as well as a wide range of network-level attacks. So broadcast-key may be exposed to the enemy by compromised nodes and the enemy can use it to broadcast forged commands or messages to the whole network. Moreover, the constrained energy, memory, and computational capabilities of the employed sensor nodes limit the adoption of security solutions designed for traditional networks.

How to secure the broadcast messages after deployment of sensor nodes? At first, each node is pre-loaded a same broadcast-key before deployment. So the BS can broadcast messages encrypted by this key and each node can get the correct message from BS or relay nodes after decryption. It is obviously that the initial broadcast-key would be exposed to enemy with some nodes being compromised. The next step of authenticating broadcast messages is updating or rekeying the broadcast-key. Since we cannot stop the sensor nodes from being compromised, the secure of the broadcast-key also can not be guaranteed all the time. The best we can do is to use broadcast-key as encryption key to provide confidentiality and authentication for broadcast messages. The former can stop the enemy without broadcast-key from eavesdropping. The object of the latter is to guarantee that each sensor node can authenticate the validity of broadcast messages it just received.

The main contribution of our work is focused on the management mechanism of broadcast-key in DWSN. The propose scheme includes initial broadcast-key pre-shared, broadcast-key establishment and broadcast-key rekeying.

This paper is organized as following: Section Ⅱ describes the related works. Section Ⅲ describes the new scheme in detail. Section Ⅳ deals with the detailed analysis and comparisons. Section Ⅵ concludes the paper with future research directions.

II. RELATED WORKS

Adrian Perrig proposed the first famous broadcast-key management scheme μTESLA in [2], which is based on the authenticated streaming broadcast TESLA protocol [3]. The μTESLA scheme adopts a one-way hash function h ( ) and uses the hash preimages as broadcast-keys in a message authentication code (MAC) algorithm.

Initially, sensor nodes are preloaded with $K_0 = h^n(x)$, where x is the secret held only by the sink. Then, $K_1 = h^{n-1}(x)$ is used to generate MACs for all the broadcast messages sent within time interval I_1. During time interval $I_m (m \geqslant 2)$, the sink broadcasts broadcast-key K_1, and sensor nodes verify $h(K_1) = K_0$. The authenticity of messages received during time interval I_1 are then verified using K_1. This delayed disclosure technique of broadcast-key is used for the entire hash chain and thus demands loosely synchronized clocks between BS and sensor nodes. μTESLA is enhanced in [4] to overcome the length limit of the hash chain.

It is generally held that μTESLA-like schemes have the following shortcomings even in the single-user scenario:

(1) All the sensor nodes have to buffer all the broadcast messages received within at least one time interval, if the

disclosure interval I_m with $m \geqslant 2$, the sensor nodes has to buffer m-1 time intervals;

(2) The sensor nodes are subject to Denial of Service (DoS) attacks, where broadcast messages could be forged by the enemy due to the propagation delay of the disclosed broadcast-keys. Since wireless transmission is very expensive in DWSN and sensor nodes are extremely energy constrained, the DoS attacks can cause devastating damage to the whole networks. Such as compromised nodes A and B in Figure 1.

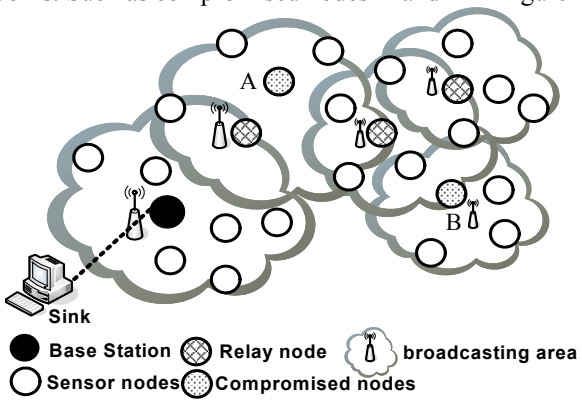

Figure 1. broadcast Example in DWSN

III. A NEW BROADCAST-KEY MANAGEMENT SCHEME

In this section, we will introduce our new scheme for broadcast-key in DWSN.

Compared with the propagation delay of the disclosed broadcast-keys in μTESLA, we propose a new publication scheme for broadcast-keys, which we call it as BPS (Broadcast-key Pre-published Scheme). In BPS, broadcast-keys are published before the related encrypted broadcast messages. The basic framework of BPS is described in Figure 2. Each broadcast key is pre-published by the broadcaster before the encrypted packet.

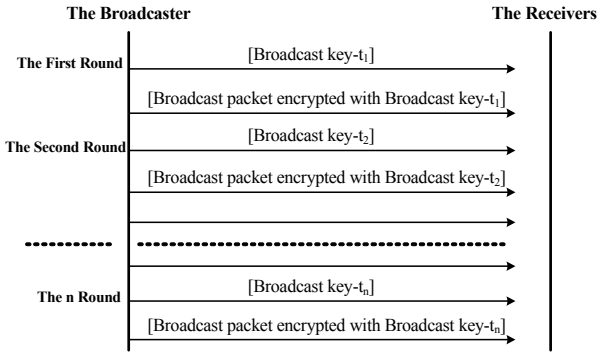

Figure 2. framework of BPS

We can conclude the benefit of BPS as followed:

(1)Less storage. Using BPS, the receiver nodes only need to save the broadcast-key before the related encrypted packets is broadcasted. While in μTESLA, the encrypted packets have to be saved.

(2)High efficiency. In BPS, after receiving the encrypted packet, each receiver node can use the pre-published broadcast-key to decrypt the message immediately.

(3)Anti some kinds of DoS attacks. BPS does not need to receive and save forged encrypted packets.

From the perspective of key management, the shortcoming of BPS is obviously. Since the broadcast-key is pre-published to all nodes before messages, the enemy can acquire the broadcast-key by compromising one node. So the enemy can forge broadcast messages and propagate it into the network.

According to the leakage of broadcast-key by compromised nodes, we must set up the update mechanism for broadcast-key. The update mechanism should guarantee the update process is timely and support the backward security. The latter means that enemy can not calculate the new broadcast-key even some old broadcast-keys are compromised. At the same time, we also should design new scheme that can support Anti-replay.

After analyzing the cons and pros of BPS, we design a new broadcast-key management scheme named GKH (Global Key Hash Chain. In DWSN, the broadcast-key is shared by all the nodes, so is also called global key). GKH use two one-way key hash chains to guarantee the security of broadcast-keys and broadcast messages. Due to the feature of one-way hash function as its inverse function does not exist or has very large computational complexity, even if the enemy master the algorithm and the pre-published broadcast-key, it still can not calculate the next round broadcast-key haven't been published.

A. Two one-way broadcast-key hashing lists

GKH build two one-way key hash chains, K_n and W_{2n}.

K_n: At first, BS generates Kn, the base value of K-list, then use the hash function $Hash_{GL}$ to compute the next hash value along the K-list. The formula is:

$$K_i = Hash_{GL}(K_{i+1}) \qquad (1)$$

$Hash_{GL}$ is pre-loaded on each node before deployment, $0 \leqslant i \leqslant (n-1)$.

Given K_i, we can compute the pre-published broadcast-keys K_j, $0 \leqslant j < i$. But we can not inverse compute the future broadcast-keys K_j, $i < j \leqslant n$.

The W-list has the same construction with K-list, as in Figure 3.

In K-list, each value K corresponds to a broadcast message. Initial value of K is K_n and is pre-loaded on each node before deployment. The broadcast-key is pre-published from K_1 to K_n corresponding to the first n rounds of broadcast messages.

Figure 3. two one-way broadcast-key hash chains of GKH

W_{2n}: We set each round broadcast message as two broadcast packets. The first is the pre-published broadcast-key, and the second is the encrypted message. So the W-list is set up as $W_0 \leftarrow W_1 \leftarrow W_2 \ldots\ldots \leftarrow W_{2n}$. In W-list, each value W corresponding to a broadcast packet. Initial value of W is W_{2n} and is pre-loaded on each node before deployment. The broadcast-key is pre-published from W_1 to W_{2n} corresponding to the 2n packets of first n rounds broadcast messages.

TABLE I. W&K-LIST BROADCAST-KEY LIST

broadcast round	K-list	W-list
1	K_1	W_1 ; W_2
2	K_2	W_3 ; W_4
……	……	……
n-1	K_{n-1}	$W_{2(n-1)-1}$; $W_{2(n-1)}$
n	K_n	W_{2n-1} ; W_{2n}

B. format of broadcast messages

After the construction of two one-way broadcast-key hashing lists, next important step is to set up the format of broadcast packets according to broadcast message.

The parameter and symbol are defined as follows:

i: the round of broadcast;

$Hash_{GL}(M)$:the result of hash function on M;

||: the conjunction of two messages;

M_i: the i round broadcast message;

W,K: global key(broadcast-key);

$\{M_i\}_{K_x}$: encrypt M_i with broadcast-key K_x;

Each round broadcast message is divided into two packets, the formats of the i round broadcast message are :

Packet one: $i \| W_i \| \{K_i\}_{W_{i-1}} \| Hash_{GL}(W_{i+1} \| \{M_i\}_{K_i})$

Packet two: $i \| W_{i+1} \| \{M_i\}_{K_i})$

As described in above formats, broadcast-key Wi and Ki are published in packet one. The encrypted message is carried in packet two. The basic broadcast-key publish strategy is according to BPS.

There is a fixed time interval between the broadcast time of packet one and packet two. Two broadcast packets should be broadcasted retain a certain time interval, so as to ensure that before the BS start broadcast the packet two of this round, the first broadcast packet has been successful reached all the nodes within the network.

C. the broadcast packet process of GKH

In GKH, a sender may be BS or relay nodes. A receiver represents a node who just received a broadcast packet.

The steps of processing the broadcast packet one are listed as follows.

a) Checking the broadcast round i:

If the round number is less than the latest number on one node, this packet can be discarded. If the latest round number is just i-1, then i would replace (i-1) to be the latest round number.

b) Checking the W-list:

According to table 1, each receiver can check W_i through hash computing by using the pre-loaded value W_0 and the latest W-value W_{i-1}. If the result is right, the latest W-value would be replaced by W_i.

c) Acquiring and checking K_i:

After passing check a) and b), each receiver can use W_{i-1} to decrypt encrypted data item. The result would become the latest broadcast-key K_i, instead of K_{i-1}.

If the node receives the packet one of the first round, the initial K_0 is already loaded on each node before deployment.

d) Saving unchecked hash value.

The process of the second packet for the i-th round broadcast would be start after the broadcast-keys (W_i and K_i) have been published for a fixed time interval.

The steps of processing the broadcast packet two are listed as follows.

a) Checking the broadcast round i:

If the round number is less than the latest number on one node, this packet can be discarded.

b) Checking the W-list:

According to table 1, each receiver can check W_i through hash computing by using the latest W-value W_{i-1}. If the result is right, the latest W-value would be replaced by W_i.

c) Checking the integrity of encrypted message:

The receiver can use $Hash_{GL}$ to compute the hash value of latter two data items in packet two and compare the result with unchecked hash value in packet one. If the comparison results are the same, the integrity check is passed. Otherwise the packet two should be discarded.

d) Acquiring the broadcast message of round i:

The receiver use Ki to decrypt the encrypted message in packet two, Ki is the broadcast-key which has been acquired in process of packet one.

D. check mechanism of re-broadcast

According to the situation that some nodes act as relay nodes in the propagation of broadcast messages, some new security mechanism should be added into GKH. Each receiver should first check the identity of relay node before beginning to process the broadcast packets. As a new check value, $ID_{re\text{-}broadcast}$ is added into the header of broadcast packets. If the receiver finds that $ID_{re\text{-}broadcast}$ isn't in its neighbor table, this packet should be discarded.

We settle up some rules to stop the potential Broadcast storm. For two packets in every round broadcast, each relay node only re-broadcast the packets for one time.

E. BMSP check mechanism for GKH

By using GKH, we can prevent some kinds of DoS attack packets. But the enemy may forge broadcast packets of GKH through compromised nodes.

Our propose is based on cryptographic puzzles MSP[5],which is proposed by Prof. PENG NING, to reduce the possibility that an attacker may exploit an observed weak authenticator to forge broadcast packets.

Traditional cryptographic puzzles require interactions between a client and a server [6, 7]. However, broadcast in DWSN, which involves one sender and a large number of receivers, does not permit such interactions. Moreover, we have to prevent an attacker from pre-computing puzzle solutions.

The enhanced GKH with BMSP have the same packets as GKH. Each round broadcast message is divided into two packets, the formats of the i-th round broadcast message are :

Packet one: $i \,||\, W_i \,||\, \{K_i\}_{W_{i-1}} \,||\, Hash_{GL}(W_{i+1} \,||\, \{M_i\}_{K_i}) \,||\, Pi$

Packet two: $i \,||\, W_{i+1} \,||\, \{M_i\}_{K_i}$

Compare with initial GKH, BMSP add a MSP check value into packet one. There is no change to other data item.

BMSP parameter: Pi is the Check parameters of BMSP scheme. Before packet one being broadcasted by BS, at first BS should confirm that all the data item as well as Pi. After applying the hash function $Hash_{GL}$ to the whole packet one, BS can choose suitable Pi value to meet that first L-bit bits are all 0. As described in Figure 4.

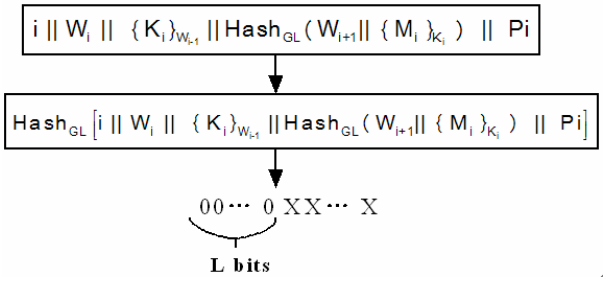

Figure 4. BMSP mechanism

The value of L is settled by BS and would be announced to all nodes before deployment. According to L value, BS would try several times in hash computing to find suitable Pi value in each round of broadcast. Of course, the value of Pi is also included in the object of hash function. BS has sufficient computing resource to applying hash function.

From the prospective of security, the publisher of the broadcasting message must be aware of the numerical values of the hash functions and decide the value of Pi in the possible options (with the calculation attempts might up to 2^L). The calculation cost of hashing functions might be affordable to energetic Base station, but it would be a significant burden for a normal relay node in wireless sensor network. So those relay nodes would keep the values of Pi published by the Base station in the key management during the process of re-broadcasting.

It is obvious that the values of Pi would be distinct in each hashing calculation as the numerical values of the parameters changes, as well as the values of the results of hashing calculation except the leading L bits would be fixed to be zero.

The only change to the receiver nodes is that MSP value validation is added to the process of security check for the first broadcasting package in the ith round.

MSP validation: calculate hashing values for the content of the whole data packet. Check if the leading zero digits equal to L, and abandon the whole packet if not correct.

The advantage of BMSP scheme is that, when the enemy intercepts the first broadcast packet of the ith round via some comprised node, it can only publish the first modified packet after multiple hashing calculations to decide the value of Pi in the possible sets due to the new-adapted MSP validation. The calculation amounts, up to 2L attempts would efficiently delay the time for the enemy to get the appropriated parameters and broadcast the modified packets, thus gains the valuable time for the correct and trustable broadcast packets distribution over the whole network.

Even the enemy has grasped two packets through compromised nodes, and has intercepted the plain text of the broadcast message of the current round, it still needs to finish modification and encryption of the plain text before it start hashing computing and choosing the value of Pi. Thus makes the publishing time of the forged packet is behind the publishing time of correct and trustable broadcast message packet.

The process flowchart of the GKH broadcast packet based on BMSP scheme is show in Fig. 5 and Fig. 6.

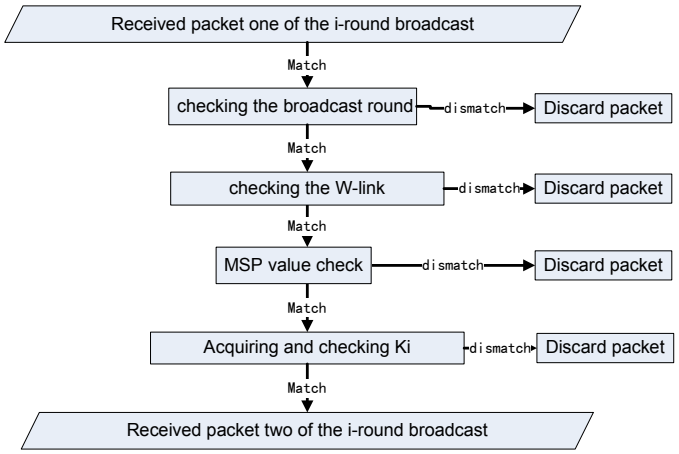

Figure 5. The enhanced process of GKH for packet one

As stated above, the result of hashing function calculation for each MSP validation is theoretically distinct for different parameter of MSP. By setting MSP validation the forge packet or modified data by enemy could be prevented efficiently.

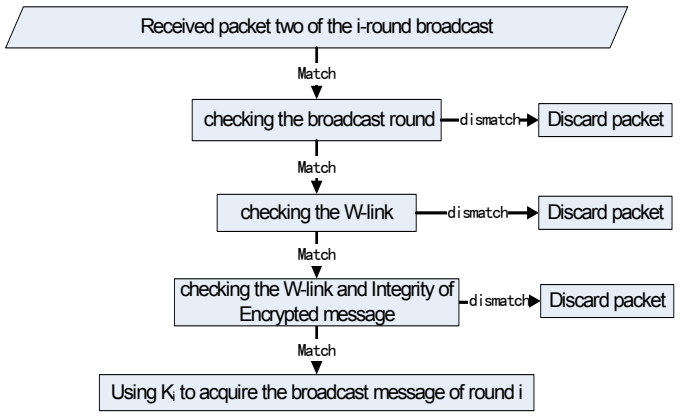

Figure 6. The enhanced process of GKH for packet two

IV. ANALYSIS AND COMPARISONS

This section we will compare GKH and μTESLA in details. As the core of security broadcast in DWSN, the management of broadcast-key is very important in providing authentication and confidentiality for broadcast messages.

a) architecture

The difference of GKH and μTESLA in architecture is the publish mechanism of broadcast-key. The delayed disclosure technique of broadcast-key is used in μTESLA. On the contrary, GKH publish broadcast-key at first, then start broadcasting the encrypted message.

TABLE II. GKH VS μ TESLA IN BROADCAST-KEY

	Key Type	**Key Identify**	**Function**
GKH	two one-way hash key-lists	W; K	W:encrypt broadcast-key K; K:encrypt broadcast message
μTESLA	a one-way hash key-list	K	K:encrypt broadcast message

We can see form Table 2 that, differs from the single and one-way hashing linked list for keys adopted by μ TESLA, GKH uses two one-way hashing linked lists for keys; among them the W-list is the broadcast data packet and provides authentication and security protect for other auxiliary information, and the K-list provides security protect for the broadcast message if each round. The two-listed GKH emphases more on the collaborated validation among the data packets in the timeline, on the global key publishing ahead of the time than the μ TESLA protocol does. The auxiliary authentication design of the broadcast packets based on two-lists in packet of each round improves further security of the encrypted broadcast messages.

b) format of broadcast packet

Comparison of data formats of broadcast packet: The data formats of broadcast packet are different between GKH scheme and that of μ TESLA scheme. GKH uses the data format that provides two broadcast packet corresponding to the message in each round. The encrypted global key is published in the first broadcast packet and the encrypted data is published in the second broadcast message. GKH has much more complicated broadcast data structure than that of μ TESLA; it not only has special bits/rounds for the messages, the data formats of two packets corresponding to the message in each round are different to each other. Each message has corresponding settings in global key, and special settings for hashing values and validation for the whole packet. These values protect and validate for each other.

c) publish time

The publish times of the scheme GKH and μ TESLA are different from each other in spite that they both separate the broadcast of the global broadcast-key and encrypted messages. The comparison is shown in Table 3.

TABLE III. PUBLISH TIME OF BROADCAST-KEY AND ENCRYPTED MESSAGES

	Publish time of broadcast-key	**Publish time of encrypted messages**
GKH	broadcast-key K is carried in packet one of each round broadcast; broadcast-key W is carried in each packet of each round broadcast;	Encrypted messages are broadcasted in packet two of each round broadcast.
μTESLA	Compared with the publish time of encrypted message, the broadcast-key K is delayed broadcasted for several time intervals.	Compared with the publish time of broadcast-key, the encrypted message is broadcasted in advanced for several time intervals.

d) efficient in processing encrypted messages

In GKH, after receiving packet two of a round broadcast, receiver will start the authentication and validations. As long as the check is passed, receiver can decrypt the encrypted message. While in μ TESLA, encrypted messages must be saved till the broadcast-key is published.

e) security

The primary security condition of broadcast security in DWSN is to make sure that the enemy cannot forge correct broadcast packets. Although cannot stop forging broadcast packets, our broadcast management scheme can guarantee that any packet passed the check in GKH must had been broadcasted by BS or trusted relay nodes.

Able to correctly identify and deal with the forged broadcast packets, is a measure of whether a broadcast-key management mechanism has the capability to resist DoS attacks. μ TESLA is vulnerable to DoS attacks. This is a fatal

threat to DWSN because of the limited and depletable battery power on each sensor node.

GKH use the broadcast-key W-list to provide the authentication for broadcast packets. Broadcast-key K in-advance announcement based on one-way cryptographic hashing function is also proposed to prevent attacks on security from the captured keys.

As an authentication approach to GKH for broadcast-keys, BMSP can prevent possible forge messages made by enemy when broadcast packet one has been intercepted in DWSN. Introducing the value of Pi forces multiple hashing calculation and numerical considerations to falsify the intercepted broadcast packet one, thus wins time in broadcasting trusted packet one to cover the whole network. So it is very difficult for the enemy to forge packet one to come into force in a limited time.

Based on intercepted packet one in a round of broadcast, the enemy can start to forge packet two. For those nodes who have successfully received packet one, the hash value of packet two is included in packet one. So the related forged encrypted messages cannot pass the hash validation and cannot get a chance to be propagated in the network.

f) Comprehensive overhead

Unlike μ TESLA, GKH does not need huge storage for encrypted messages. So the storage overhead is less than μ-TESLA. But GKH have to spent more communication and computation in authentication and validations of the checking process for two broadcast packets in each round broadcast than μ TESLA. For the sake of anti DoS attack, the overhead of communication and computation is needed. Our future work is just to reduce the overhead and keep the security level at the same time.

V. CONCLUSION

In this paper we propose a new broadcast-key management scheme for DWSN. A scheme of broadcast-key in-advance announcement is presented in GKH, which differs from classical global keys management protocol; then a scheme of broadcast-key in-advance announcement based on one-way cryptographic hashing function is proposed to prevent attacks on security from the captured keys in compromised nodes. It guarantees that, even when the algorithm and old broadcast-keys are leaked, information in the very next broadcast-key to be announced is still out of induction, which assures the backward security of broadcast-keys for the network.

This paper also appends an authentication approach to GKH for broadcast-keys based on MSP, to prevent possible DoS attacks from enemy when some data packet has been intercepted in DWSN. This method forces multiple hashing calculation and numerical considerations to falsify the first intercepted data packet, thus wins time in broadcasting trusted message packets to cover the whole network. Authentication and validations are added via the additional checking process for middle nodes to effectively avoid possible attacks with fake data packet, to help implement security checking and secrets protection of the broadcasted messages, to guarantee secure, reliable and timely broadcasting of messages among every trusted node in the whole network.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation for the Doctoral Program of Higher Education of China under grant No.20049998027, and the National Science Foundation of China under grant No. 90104001 and No. 90604006.

REFERENCES

[1] Akyildiz I F, Su W, Sankarasubramaniam Y, et al. Wireless sensor networks: a survey. IEEE Communications Magazine, 2002, 40 (8): 102-114.

[2] Perrig A. SPINS : Security protocols for sensor network. Procceedings of the 11th ACM Annual International Conference on Mobile Computing and Networks (Mobicom 2001), Rome, Italy, ACM Press, 2001:189-199.

[3] A. Perrig, R. Canetti, J. Tygar and D. Song, Ef_cient authentication and signing of multicast streams over lossy channels, in: IEEE Symposium on Security and Privacy (2000).

[4] D. Liu and P. Ning, "Multi-level mTESLA: Broadcast authentication for distributed sensor networks," ACMTransactions in Embedded Computing Systems (TECS), vol. 3, no. 4, 2004.

[5] Peng Ning, An Liu, Wenliang Du, Mitigating DoS Attacks against Broadcast Authentication in Wireless Sensor Networks, ACM Transactions on Sensor Networks (TOSN), Vol. 4, No. 1, January 2008.

[6] Ari Juels and John Brainard Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks. In S. Kent, editor, Proceedings of NDSS '99 (Networks and Distributed Security Systems), pages 151-165, 1999.

[7] WANG, X. AND REITER, M. 2004. Mitigating bandwidth-exhaustion attacks using congestion puzzles. In Proceedings ofthe 11th ACM Conference on Computer and Communications Security (CCS '04).257–267.