Message and Its Origin Authentication Protocol for Data Aggregation in Sensor Networks*

HongKi Lee[1], DaeHun Nyang[2,**], and JooSeok Song[1]

[1] Department of Computer Science, Yonsei University, Seoul, Korea
{lhk, jssong}@emerald.yonsei.ac.kr
[2] Graduate School of Information Technology and Telecommunications, Inha University, Incheon, Korea
nyang@inha.ac.kr

Abstract. In distributed sensor networks, the researches for authentication in sensor network have been focused on broadcast authentication. In this paper, we propose a message and its origin authentication protocol for data aggregation in sensor networks, based on one way hash chain and Merkle tree authentication with pre-deployment knowledge. Proposed protocol provides not only for downstream messages but also for upstream messages among neighbors, and it solves the secret value update issue with multiple Merkle trees and unbalanced energy consumption among sensor nodes with graceful handover of aggregator. In treating compromised node problem, our protocol provides an equivalent security level of pair-wise key sharing scheme, while much less memory requirements compared to pair-wise key sharing scheme.

Keywords: sensor networks, aggregation, authentication, Merkle tree, hash chain.

1 Introduction

Recently, distributed sensor networks have been paid lots of attentions because of its valuable applications, such as monitoring of disaster site, observation of valuable creature's habitation, and surveillance of critical spot in battlefields where human can not approach or stay for observation all the time.

Distributed sensor networks typically consist of a large number of resource-constrained sensor nodes and one or a few powerful control nodes called as base station (BS). The computing ability and the communication range of SN have restrictions because of resource-constrained characteristics of SN. The limitation

* This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment), and supported by grant No. R01-2006-000-10957-0(2006) from the Basic Research Program of the Korea Science & Engineering Foundation.

** Corresponding author.

X. Zhou et al. (Eds.): EUC Workshops 2006, LNCS 4097, pp. 281–290, 2006.

of computation ability urges us not to adapt the conventional security techniques, but to develop new mechanisms adaptable to sensor networks.

The traffic of sensor network is classified into downstream and upstream in terms of its direction. The former is usually called as control message directed from BS to SNs while the latter usually contains sensed data directed from SNs to BS, and the former is usually transmitted by broadcast fashion while the latter depends on routing protocol. To guarantee the reliability of message in sensor network, authentication should be applied to all traffics including message origin.

In this paper, we propose an authentication scheme for message and its origin without disclosing of node's secret key for authentication by utilizing hash chain and Merkle hash tree (MT) authentication [4]. We assume that we can get the pre-deployment knowledge [7], and construct a group with potential neighbor nodes which are the nodes of higher probability to communicate with each other. After that, we build MT with the group member nodes as leaf nodes, and choose a header node as aggregator in which the tree information is stored. After deployment, message and its origin authentication at the aggregator node (AN) can be easily achieved using pre-distributed authentication values. To protect the replay attack with reusing the authentication value, AN maintains the counter values of each nodes and updates for every communications.

The contributions of this work are:

- Provide a real-time message and its origin authentication scheme for upstream and downstream traffics.
- Reduce the computation overhead for authentication, and thus extend the lifetime of sensor networks by utilizing symmetric cryptography.
- Provide graceful degradation by minimizing the impact of compromised node (CN), which is limited to CN itself.
- Provide almost equivalent security level of pair-wise key sharing scheme with much less memory.
- Provide graceful aggregator handover scheme without leakage of security.

The remainder of this paper is organized as follows. Section 2 explains the motivation of this work with some overviews of related works. Then, we present proposed protocol in section 3. In section 4, analysis of security and performance evaluation of our protocol are presented. Section 5 finally concludes this paper.

2 Motivation

Considering that the main purpose of sensor networks is gathering information or detecting events through SNs and reporting to user through BSs, we can easily guess that the volume of upstream traffic is much larger than that of downstream traffic. Moreover, upstream traffic which contains various data sensed by lots of SNs at different time is more diverse than downstream traffic which usually contains control messages broadcasted by BS at a stroke.

In sensor networks, one of the most significant security threats is CN which is captured and installed malicious codes by adversaries. Since it cannot be

prevented completely because of unattended characteristic of sensor networks, it is required to guarantee the resilience against the CN and to minimize its effect. Authentication of message and its origin before accepting is one of alternative solutions to minimize affection of CNs.

For authentication of downstream traffic, μTESLA and its variants have been proposed in [1,2,3]. The μTESLA accomplishes the authenticated broadcast by loosely time synchronization and delayed disclosure of symmetric key fashion with hash chain. The essential problem in scaling up μTESLA is how to distribute and authenticate the parameters of its instances. The multilevel μTESLA uses higher-level μTESLA instances to authenticate the parameters of lower-level ones. The μTESLA and the multilevel μTESLA have the same problem of authentication delay, which connotes the possibility of denial-of-service attacks to disrupt the distribution of its parameters [3].

For authentication of upstream traffic, any remarkable works has not been proposed yet. It commonly relies on conventional cryptographic techniques such as unicast with the shared pair-wise secret between correspondent nodes. Sharing pair-wise keys between all possible pairs is not commonly considered as practical solution because of huge memory requirement. Furthermore, in data gathering application, since authenticity of data is more important while exposure of contents is not a significant issue, shared key is not always required. Considering large volume and diverse messages from lots of SNs, a simple and efficient authentication scheme is surely required.

3 Message and Its Origin Authentication Protocol

In this section, we describe the message and its origin authentication protocol for data centric sensor network in detail. We use the following notations to describe the security protocol and cryptographic operations in this paper.

ID_i	identifier of sensor node i
SN_i	sensor node with ID_i
AN, A	aggregator node of a group
m	the number of member nodes in a group
ctr	current counter value of hash chain
CTR	maximum counter value, i.e., the length of hash chain
N	the number of MT, i.e., the number of fraction of hash chain
L	the lifetime of sensor node, i.e., $L = N \times CTR$
$h(x)$	one-way hash function with input value x
$h^n(x)$	compute n times of $h(x)$
$\|\|$	concatenation

3.1 Assumptions

In our proposed protocol, we assume to construct a security architecture that some deployment knowledge is available as *priori* and sensor network is static.

In many cases, if we have some deployment knowledge, it can be very useful to construct security architecture such as key management and clustering. In fact,

certain deployment knowledge may be available as a *priori* in many practical scenarios depends on deployment method [7]. We assume that some deployment knowledge to approximate the topology of sensor network is available.

We also assume that the sensor network is static, which means that the group membership of SNs will not be changed after deployment. Since sensor networks tends to use redundancy by a large number of cheap SNs to cover wide area, each SN will be responsible for only small area. Therefore, the group membership of SN can be kept even if it has mobility.

3.2 Pre-deployment Phase

Before deployment, grouping based on deployment knowledge and constructing initial security architecture for authentication are performed.

At first, based on deployment knowledge, potential neighbor nodes are combined into a group. A secret key, S_i, for authentication is allocated to each SN. S_i is the seed of hash chain for authentication and should be kept securely to any other nodes including legitimated nodes. After then, authentication key, K_i for each SN is computed by CTR times of iterative hash computation with ID_i and S_i, i.e., $K_i = h^{CTR}(ID_i||S_i)$, and stores it to the SN's memory.

Next, a MT for authentication of K_i is constructed with K_is of member nodes. The root hash value of the MT, K_R, and the authentication evidence of K_i, CK_i, are stored in each member node.

Figure 1 shows an example of the MT with $m = 8$ and $CTR = 100$. $K_R = K_{1,8}$, where $K_1 = h(K_1^*)$, $K_{1,2} = h(K_1||K_2)$, $K_{1,4} = h(K_{1,2}||K_{3,4})$, and $K_{1,8} = h(K_{1,4}||K_{5,8})$. In this figure, AN has this tree architecture and K_R, while node ID_5 will have K_5 as own authentication key, and $CK_5 = (K_{1,4}, K_6, K_{7,8})$ (circled ones in Figure 1) as authentication evidence of K_5. To authenticate K_5, compute $K'_R = h(K_{1,4}||h(h(K_5||K_6)||K_{7,8}))$, and check whether or not $K_R = K'_R$.

Within a group, one member node is selected as AN of the group. It maintains the lists of all the member nodes ID and their current *ctr*s in the table. And its role is aggregating the sensed data received from member nodes after checking authenticity of sensed data and its origin, and broadcasting control messages.

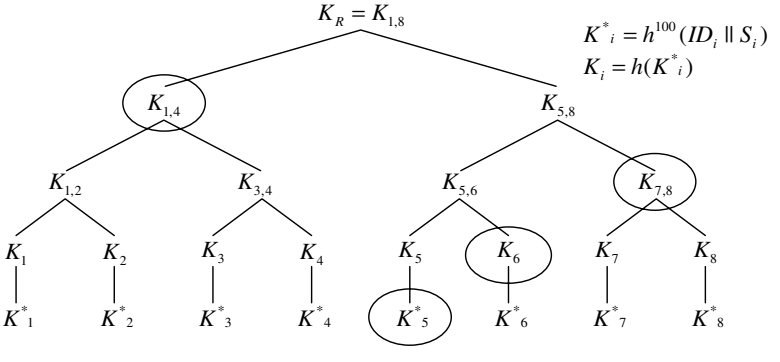

Fig. 1. An example of Merkle hash tree authentication

3.3 Message and Its Origin Authentication Phase

After deployment, SNs confirm its membership of group and perform their basic operations. Message and its origin authentication will be carried out by a part of the basic operations.

In proposed protocol, the message format is as follow:

ID_S, ID_D, Flag, K_S, ctr_S, CK_S, M, MAC(M, ctr_S)
where S is source, D is destination,
and Flag is combination of I, D, B, H, U

We classify the messages into 5 types by the purpose of message, and the each message type is assigned by a Flag character: I for initialization, D for reporting sensed data, B for broadcasting control messages, H for aggregator handover procedure, and U for secret value update procedure.

Initialization

After deployment, every SN including AN broadcasts its ID and initial hash value K_i with CK_i for initialization. AN checks the list of member nodes from these messages by verifying K_i with CK_i. If initial authentication is successful, AN saves the ctr_is of each SN for next authentication. AN also broadcasts its ID and K_A with CK_A, and then, SNs authenticate AN by checking weather the K_R computed with AN's message is the same with its own K_R or not. After the initialization ends, newly appeared nodes will be treated as CN.

$SN_i \rightarrow$ broadcast ID_i, $*$, (B, I), K_i, ctr_i, CK_i, M, MAC(M, ctr_i)
where $M =$ "Initialization of ID_i".

$AN \rightarrow$ broadcast ID_A, $*$, (B, I), K_A, ctr_A, CK_A, M, MAC(M, ctr_A)
where $M =$ "ID of Aggregator Node is ID_A".

Upstream Authentication. For data aggregation, SNs send sensed data with authentication information which includes the node's ID, its authentication value $h^{ctr_i}(ID_i||S_i)$ (briefly denoted as h_i^{ctr}), latest counter ctr_i, and CK_i with MAC(M, ctr_i) to AN. After transmitting, SN changes its counter value for next reporting.

$SN_i \rightarrow AN$ ID_i, ID_A, D, h_i^{ctr}, ctr_i, CK_i, M, MAC(M, ctr_i)
where $M =$ Sensed data

SN_i $ctr_i = ctr_i - 1$

AN authenticates the received data and its origin by checking MAC and freshness of ctr_i, and rebuild K_i' by $h^{CTR-ctr}(h_i^{ctr})$ computation. And with this K_i', yield K_R' and compare it with initial K_R. If all tests are successful, AN accepts the message and updates the SN's counter in the table. Otherwise, AN considers that the data is transmitted from unauthorized node and discards it.

AN Check MAC, and the freshness of ctr_i
Compute $K_i' = h^{CTR-ctr_i}(h_i^{ctr})$, and K_R' with K_i' and CK_i
Compare K_R with K_R'
IF all tests are successful, Accept data and $ctr_i = ctr_i - 1$
OR Discard it

Downstream Authentication. Downstream authentication can be achieved by the same way of the upstream authentication. AN broadcasts control message with authentication information and setting B Flag.

$$AN \rightarrow SNs \qquad ID_A,\ *,\ B,\ h_A^{ctr},\ ctr_A,\ CK_A,\ M,\ \mathrm{MAC}(M, ctr_A)$$
$$\text{where } M = \text{Control message of } AN.$$

When a SN receives the broadcasted message, it checks the ID_A with AN's ID already known and verify the authenticity of the message. Following procedure of downstream authentication is all the same as upstream authentication.

3.4 Handover of Aggregator Node Phase

Since it has much possibility that AN's lifetime could be shorter than other SNs to perform the AN's role, it is required that the role of AN is transferred to another depending on the remained energy status of AN.

If AN has to be changed into another, old AN (OAN or OA) chooses an appropriate SN as new AN (NAN or NA) and transmits the information for AN to NAN by means of upstream message with setting H Flag. Transferred message includes the list of member nodes and their current counters. After the AN's role has been transferred to NAN, OAN broadcasts an announcement of the ID of NAN.

$$OAN \rightarrow NAN \qquad ID_{OA}, ID_{NA}, H, h_{OA}^{ctr}, ctr_{OA}, CK_{OA},\ \mathrm{MAC}(M, ctr_{OA})$$
$$\text{where } M = (ID_1, ctr_1), (ID_2, ctr_2), \cdots, (ID_m, ctr_m)$$
$$OAN \rightarrow \text{broadcast } ID_{OA}, *, (B, H), h_{OA}^{ctr}, ctr_{OA}, CK_{OA}, M,\ \mathrm{MAC}(M, ctr_{OA})$$
$$\text{where } M = ID_{NA},\ ctr_{NA},\ Timestamp$$

3.5 Secret Value Update

In our proposal, we utilize hash chain technique to protect replay attack which utilize the previous hash value achieved by overhearing. However, one of the critical issues for hash chain is secret value update since it has an original limitation of hash chain length.

A naïve approach for this problem is to make use of pair-wise keys between every pair of SNs. It avoids wholesale sensor network compromise upon node capture since selective key revocation is possible. However, this solution requires pre-distribution and storage of $m-1$ keys in each SN, and $\frac{m(m-1)}{2}$ per sensor network, which renders it impractical for sensor networks of more than 10,000 nodes, for both intrinsic and technological reasons [8]. We will analyze this point by comparing with our proposal in section 4.

Another approach is to use long enough hash chain to last until the end of sensor network's lifetime. However, long hash chain makes the lifetime of sensor network shorten by consuming node energy to perform lots of hash function for every authentication. The average number of hash computation per one authentication is Therefore, one long hash chain is not a good solution.

We propose a secret value update mechanisms: pre-computing multiple Merkle trees from one long hash chain. During pre-deployment phase, each SN extracts

N of $\frac{L}{N}$ length hash chains from L length of a long hash chain derived from one initial value, and construct N of MTs with these fractions. This method is very efficient because the next initial value of MT could be authenticated by old hash value. It requires more memory to SNs naturally. However, it can solve secret value update problem. Moreover, it can reduce the number of hash computation for every authentication compared to one long hash chain mechanism.

3.6 Treatment of Node Compromising

In general, the detection of compromised or captured sensor is considered difficult, but feasible at least in certain scenarios such as in battlefields [3]. In this paper, we do not consider the process or mechanism to detect CN, but assume such results are given.

When CN is detected, AN can simply block the CN by setting its *ctr* to "0". Then it node can not be authenticated, and therefore, the messages from that node are treated as invalid data in aggregation.

4 Analysis of the Proposed Protocol

4.1 Security Analysis

Proposed protocol is secure against the adversary which can overhear every communication messages and can compromise small number of SNs.

In proposed protocol, S_i is the only one secret which should be kept securely by each node, because it is only used for producing K_i with hashed form and should not transmitted or exposed to anyone else after deployment. AN maintains only a table of member nodes and their latest counter additionally. Since there is no shared secret, transmitted information in our protocol are not encrypted data but just public data which can be obtained through overhearing. Therefore, even if adversary capture and compromise SN including AN, it can not get any other secrets without its S_i.

We adopted group mechanism and BS does not intervene in any procedures. Thus, all the security problems are limited in each group and the possibility of attack to communication between SN and BS are removed. Furthermore, because an adversary can get only each node's S_i through node compromising, small number of CN can not effect the correct operation of entire sensor networks.

This protocol provides complete authentication not only for data aggregation but also for broadcasting. And it also provides secret update and handover of AN schemes. Therefore, any other assistant schemes are not required for message and its origin authentication in sensor networks.

4.2 Overhead Analysis: Computation and Memory

In proposed protocol, the number of hash computation is decided by the length of hash chain and the height of MT. For one authentication, the average number of hash computation is $CTR/2$ times to get a new hash chain value in SN and

to get K_i in AN, and $\log_2 m$ times of additional computations to verify that K_i in MT in AN. If m is fixed, the hash chain length is the determinant factor, and by controlling this we can reduce the number of hash computation.

To shorten the hash chain, we divide long hash chain into multiple fragments and adopte multiple MTs with the fragments. If we divide a length L of long hash chain into N fragments of length $\frac{L}{N}$ hash chains, the average number of hash computation in each node becomes simply $\frac{L}{2N}$. Table 1 shows the average number of hash computation in each SN and AN.

Each SN has to maintain the authentication information such as its own ID, secret value for authentication, hash counter, authentication key, the authentication evidence for MT authentication, root hash value of MT, and ID and counter of AN for authentication of control messages. AN additionally maintains the authentication information for the member nodes, which is the table of member node ID and its counter value set.

SN_i	$ID_i,\ S_i,\ ctr_i,\ K_i,\ CK_i,\ K_R,\ ID_A,\ ctr_A$
AN	$ID_A,\ S_A,\ ctr_A,\ K_A,\ CK_A,\ K_R$ + table of (ID_i, ctr_i) set

If we use N of MTs, SN has to maintain N sets of K_i and CK_i. If we use multiple MT scheme with fraction of long hash chain, and thus, the next authentication key K_i' can be verified by previous K_i, the memory for K_i can be saved. However, AN does not need to maintain additional information for multiple MTs, because only current *ctr* of each node is required for authentication.

When we use multiple MTs, we can reduce the memory to maintain *ctr*. The counter means the length of hash chain, thus, $\lceil \log_2 L \rceil$ bits are required for *ctr*. If we divide long hash chain into N of multiple fragments, the length of counter goes down by $\lceil \log_2 \frac{L}{N} \rceil$ of each node.

On the other hand, in pair-wise key scheme, to protect the replay attack, it requires also countermeasures such as pseudo-random number generator (PRNG) or challenge and response (C/R) protocol. It has the computational burden for naïve pairwise scheme, whereas our scheme is strong against replay attack. And the pair-wise key scheme requires $m - 1$ of memory space for key pairs.

Table 1 briefly shows the required computation for one authentication and the required memory for maintaining authentication information in each SN compared with those of the pair-wise key scheme.

We assume the environments sensor network is as below:

L (Lifetime of SN)	10,000
m (Number of member node)	1,024
Platform of SN	Atmega 128
Hash algorithm	SHA1
Time to perform SHA1 for 64 byte plaintext	7,700 μseconds [9]

Based on these parameters, SN can communicate 10,000 times with other SNs. If SN reports sensed data per 1 hour, it is reasonable considering more than 1 year of SN's lifetime. Table 2 shows the average number of hash computation and its

Table 1. Required resources for one authentication and memory

	Avg. # of hash computation	# of information
SN	$\frac{L}{2N}$	$(2+\log_2 m) \times N$
AN	$\frac{L}{2N} + \log_2 m$	$(2+\log_2 m) \times N + \log_2 \frac{L}{N} \times m$
Pair-wise key scheme	PRNG or C/R protocol	$m-1$

computational time, and the number of authentication values to be maintained by each SN for various N. It also shows the memory gain compared with the pair-wise key scheme. When we divide the 10,000 length of hash chain into 50 fragments, it takes about 0.48 seconds to generate one authentication value while the gains of memory are 58% in SN. By [10], the energy consumption is smaller than RSA signing operation considering RSA signing consumes 304 mJ per 1 byte whereas SHA1 consumes 5.9 μJ per 1 byte. Table 2 also shows the energy consumption of 1 authentication for 64 byte message.

Figure 2 shows the relationship of m and the required memory by changing N. We can see that our scheme has more advantages by increasing m, and we can choose various N by the application and environments such as the mission, available memory, required lifetime, etc.

Table 2. Comparison of hash computation and memory ($L = 10,000$, $m = 1,024$)

N	Avg. # of hashing	Comp. time	# of hash key	Memory gain	Energy consume
10	500	3.85 sec	120	12 %	188.8 mJ
20	250	1.92 sec	240	23 %	94.4 mJ
50	100	0.77 sec	600	58 %	37.8 mJ
80	62.5	0.48 sec	960	93 %	23.6 mJ
100	50	0.39 sec	1200	117 %	18.9 mJ

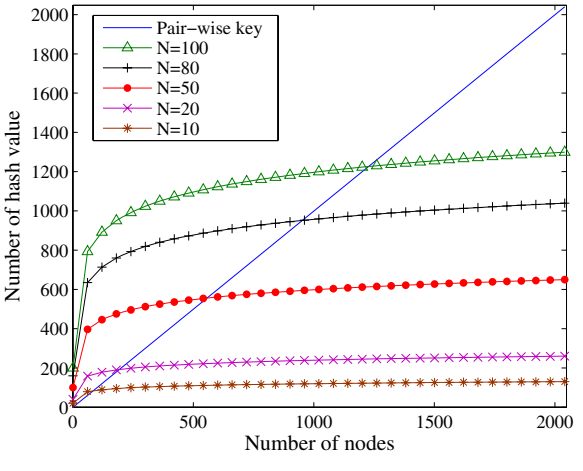

Fig. 2. The number of required hash key for the number of fraction

5 Conclusion

In this paper, we proposed a message and its origin authentication protocol based on pre-deployment knowledge in static sensor networks.

The proposed protocol provides an authentication scheme for traffic of downstream and upstream at real time by utilizing hash chain and MT. With partitioning of long hash chain and constructing multiple MTs, we solved secret update problem, and mitigated energy consumption. Moreover our protocol achieved the graceful degradation by limiting the affection of CN to CN itself.

By analyzing with example, we showed the relationship between the computation overhead and required memory compared with pair-wise key scheme. In the example, our scheme can control the computational energy and required memory depends on applications.

Proposed protocol fits for group based data centric sensor network with lots of member nodes. And there are only upstream channels and thus, BS can not control SNs, our protocol works well. Since utilizing group mechanism, our scheme is well matched for cluster based sensor network routing protocols such as LEACH [5], and PEGASIS [6].

References

1. A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J.D. Tygar, "SPINS: Security Protocols for Sensor Networks," Proceedings of the MOBICOM, 2001.
2. D. Liu and P. Ning, "Multilevel μTESLA: Broadcast Authentication for Distributed Sensor Networks," ACM Transactions on Embedded Computing Systems, Vol. 3, No. 4, Pages 800-836, 2004.
3. D. Liu, P. Ning, S. Zhu, and S. Jajodia, "Practical Broadcast Authentication in Sensor Networks," Proceedings of the MobiQuitous, 2005.
4. R. Merkle, "Protocols for public key cryptosystems," Proceedings of the IEEE Symposium on Research in Security and Privacy, 1980.
5. W. Heinzelman, A. Chandrakasan, and H. Balakrshnan, "Energy-efficient communication protocol for wireless sensor network," Proceedings of the HICSS, 2000.
6. S. Lindsey, and C. Raghavendra, "PEGASIS: Power-Efficient gathering in sensor information systems," Proceedings of the IEEE Aerospace Conference, Vol. 3, Pages 1125-1130, 2002.
7. W. Du, J. Deng, Y.S. Han, S. Chen, and P.K. Varshney, "A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge," Proceedings of the IEEE INFOCOM, 2004.
8. L. Eschenauer and V. Gligor, "A Key Management Scheme for Distributed Sensor Networks," Proceedings of the ACM CCS 2002.
9. P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, and M. Sichitiu, "Analyzing and modeling encryption overhead for sensor network nodes," Proceedings of the ACM WSNA 2003.
10. A.S. Wander , N. Gura, H. Eberle, V. Gupta, and S.C. Shantz, "Enery Analysis of Public-Key Cryptography for Wireless Sensor Networks," Proceedings of the IEEE PerCom, March 2005.