INTERNATIONAL JOURNAL OF RECENT TRENDS IN ENGINEERING & RESEARCH
Impact Factor 4.101 ISSN -2455-1457
RECENT ADVANCES IN ELECTRONICS & COMMUNICATION ENGINEERING (RAECE-2K17)

Efficient Cryptography Technique Of Authenticated Join Protocols In Wireless Sensor Networks

S.IRFAN
Lecturer, Kombolcha institute of technology
Wollo university,ETHIOPIA

Abstract: **Wireless sensor networks are a challenging field of research when it comes to security issues. Using low cost sensor nodes with limited resources makes it difficult for cryptographic algorithms to function without impacting energy consumption and latency. In this paper, we focus on key management issues in multi-hop wireless sensor networks. These networks are easy to attack due to the open nature of the wireless medium. Intruders could try to penetrate the network, capture nodes or take control over particular nodes. In this context, it is important to revoke and renew keys that might be learned by malicious nodes. We propose several secure protocols for key revocation and key renewal based on symmetric encryption and elliptic curve cryptography. All protocols are secure, but have different security levels. Each proposed protocol is formally proven and analyzed using Scyther, an automatic verification tool for cryptographic protocols. For efficiency comparison sake, we implemented all protocols on real testbeds using TelosB motes and discussed their performances.**

Keywords: **renewing; revocation; authentication; wireless sensor network; security; multihop; verification; formal proof**

1. Introduction

Nowadays, the Internet of Things (IoT) is a reality: more and more devices are used to monitor our environment and to interconnect such embedded objects. IoT relies on wireless sensor networks (WSNs) for ensuring connectivity between nodes on the lower level of the network architecture. In such a context, some sensitive applications often require cryptographic mechanisms in order to achieve security. For instance, most of the military applications of WSNs require a high level of security [1]. Thus, it is important to design secure communication mechanisms between nodes of the network. These mechanisms can be achieved thanks to modern cryptographic primitives. Once we have established a secure communication channel in the network, several situations might occur; a node can run out of battery, even get destroyed, or just leave the network, or a new node can join the network. In addition, an intruder could capture a node and get all of its secret data (including secret cryptographic keys). An attacker could also try to join the network and be part of the authenticated nodes of the network. There exist several intrusion detection systems (IDS) in the literature [2,3] in order to detect such malicious behavior. In general, an IDS either searches for signs of malicious activity in the network, or monitors the internal behavior of one node. Several works use signature-based or anomaly-based detection techniques. Using the results of IDS, the next step is to revoke identified malicious nodes and to renew the cryptographic keys used by the nodes of the network.

Contributions

In this paper, we propose key renewal and key revocation protocols. Based on the cryptographic mechanisms used in [11], we propose centralized protocols for revoking and renewing keys when an IDS has detected an abnormal behavior. These mechanisms can also be used for periodical key renewal. More precisely, we provide a key revocation protocol (KR), a protocol for renewing symmetric keys between any node in the network and the sink (RSK), four protocols to renew asymmetric keys of nodes ($RAKnk_a$, $RAKnk_b$, $RAKdh_a$ and $RAKdh_b$), a protocol to renew the network key (RNK) and two multihop shared key establishment protocols (MSK_a and MSK_b). When several protocols with the same goal are given, they use different cryptographic primitives.

In this paper, we not only propose original protocols that solve the secure join/revocation/renewal challenge, but also prove them to be secure, and we perform an experimental evaluation on real testbeds. This paper is an extension of the short paper presented in [13]. It contains additional optimized protocols, and we present additional in-depth details. Adding new nodes to the network was studied and evaluated in [11,14], and the scalability evaluation of these protocols was studied in [15].

Related Work

The main issue for secure communications in WSNs is how to set up secret keys between nodes, which is known as the key agreement. This task is a challenge due to resource constraints and the size of networks.

Many proposed methods are based on random key pre-distribution [16–18]. The main idea is that any two sensor nodes have a probability for choosing the same key from the pre-distributed key pool. The weakness of these schemes is when sensor nodes are captured by an attacker. All encryption keys in the captured nodes will be revealed. In [19], the authors proposed an approach to resolve this issue. The proposed approach is to hash the keys in the key pool with a one-way hash function. Once a key is chosen by a sensor node, it will be hashed. If another sensor node chooses the same key afterwards, instead of using the same key as the previous sensor node, it will derive a new different key by hashing the key value. This technique can reduce the amount of information revealed when a sensor node is captured for about 30% to 50% according to the authors of [19].

Table 1 summarizes the comparison between the related work schemes and our proposition. The comparison is based on the type of cryptographic algorithms used, the type of cryptographic technique and the evaluation and verification methods. We do not include the complexity of the protocols listed in the table, since these analyses are not always given by the authors of the protocols, and often, it is not feasible to perform a fair analysis of the protocol complexity only based on the presentation given in the papers.

As the table shows, and according to our knowledge, none of the existing revocation and key renewal protocols were verified using an automatic formal verification tool. In addition, most of the results in the state of the art are obtained through simulations or complexity estimation when evaluating the cost of the

cryptographic scheme. Most of the existing schemes do not specify the cryptographic algorithms of encryption/decryption, leaving the choice of these algorithms to the application. We provide automatic formal verification and real testbed implementation to further support our proposal. We used standard algorithms and used both symmetric and asymmetric cryptography.

Table 1. Comparison of related work schemes and our proposition.

Proposed Schemes	Standard Algorithms	Cryptographic Technique	Simulation	Implementation	Verification
Chan *et al.* 2003 [22]	not specified	symmetric	none	none	none
Chan *et al.* 2005 [21]	not specified	symmetric	none	none	none
Chattopadhyay *et al.* 2012 [23]	not specified	symmetric	none	none	none
Chuang *et al.* 2010 [26]	yes	asymmetric	none	none	none
Dini *et al.* 2006[25]	not specified	symmetric	none	none	none
Jiang *et al.* 2008 [24]	not specified	symmetric	none	none	none
Jolly *et al.* 2003 [31]	not specified	symmetric	yes	none	none
Purohit *et al.* 2011 [28]	not specified	symmetric	none	none	none
Wang *et al.* 2006 [29]	none	symmetric	none	none	none
Wang *et al.* 2010 [30]	not specified	symmetric	yes	none	none
Wang *et al.* 2007 [27]	not specified	symmetric	yes	none	none
All of our schemes	yes	symmetric/asymmetric	none	TelosB	automatic

1. Authenticated Join Protocols

Before recalling the two authenticated join protocols given in [11], we introduce some cryptographic primitives and notations used in the rest of the paper

Cryptographic Primitives and Notations

We use public key elliptic curve cryptography (ECC), using parameters secp160r1 given by the Standards for Efficient Cryptography Group [32]. Our implementation of ECC on TelosB is based on the optimized TinyECC library [33]. More precisely, we use the elliptic curve integrated encryption scheme (ECIES 160 bits), the public key encryption system proposed by Victor Shoup in 2001 [34].For all symmetric encryptions, we use an optimized implementation of AES [35] with a key of 128 bits proposed by [36]. In what follows, we also use the following notations to describe exchanged messages in our protocols:

- I: a new node that initiates the protocol,
- R: a neighbor of node I,
- S: the sink of the network (also called the base station),
- n_A: a nonce generated by node A,
- $\{x\}_k$: the encryption of message x with the symmetric or asymmetric key k,
- $pk(A)$: the public key of node A,
- $sk(A)$: the secret (private) key of node A,
- $K(I, S)$ or $K(S, I)$: the symmetric session key between I and S,
- NK: the symmetric network key between all nodes of the network,
- $K_{DH}(N, S)$ or $K_{DH}(S, N)$: the shared symmetric key between N and S using the Diffie–Hellman key exchange without the interaction described below.

Before deployment, each node N knows the public key $pk(S)$ of the sink and also its Based on ECC, we have that $pk(N) = sk(N) \times G$, where G is a public generator point of the elliptic curve. From $pk(N)$ and G, it is difficult to find $sk(N)$; this problem is called the elliptic curve discrete logarithm problem (ECDLP) [37,38].Using this material, each node N can compute a shared key with the sink S using a variation of the Diffie–Hellman key exchange without interaction, denoted $K_{DH}(N, S) = K_{DH}(S, N)$.

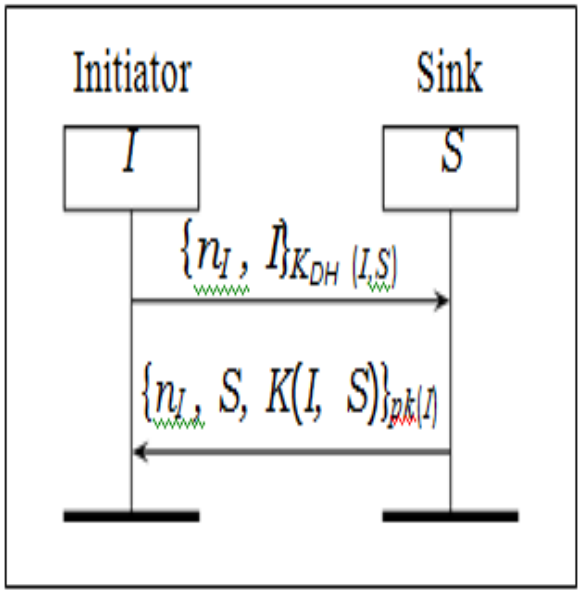
- The sink knows its own secret key $sk(S)$ and the public key $pk(N)$ of any node N. The sink computes $K_{DH}(N, S) = sk(S) \times pk(N)$.
- Node N multiplies its secret key $sk(N)$ by the public key of the sink $pk(S)$ to get $K_{DH}(N, S)$.

Both computations give the same shared key, since: $K_{DH}(N, S) = sk(N) \times pk(S) = sk(N) \times (sk(S) \times G) = (sk(N) \times G) \times sk(S) = pk(N) \times sk(S)$. In our protocols, we intensively use this mechanism.

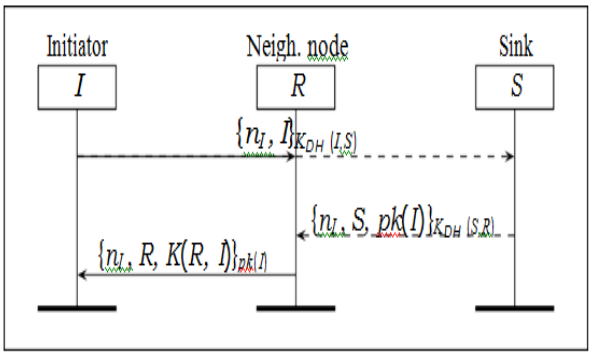
Join Protocols (DJS and IJS)

In Figure 1, we present two key establishment protocols given in [11]. The first protocol, called direct join to the sink (DJS), allows a node to join directly through the sink and is described in Figure 1a. In DJS, a new node I sends a direct request to S in order to establish a session key with it. Node I begins the join process by computing the symmetric key $K_{DH}(I, S)$ with the sink S. Then, node I generates a nonce n_I and adds its identity then encrypts it with $K_{DH}(I, S)$ and sends it to S. Upon reception, S computes $K_{DH}(I, S)$ to decrypt the request. Then, S verifies the identity of I and generates a new session key $K(I, S)$. The own pair of public and private keys, denoted $pk(N)$ and $sk(N)$, respectively.
join response contains n_I, the identity of S and the new symmetric session key $K(I, S)$. The response is encrypted using $pk(I)$ and is sent to I. Only I is able to decrypt the response with its secret key $sk(I)$. We note that n_I helps I to authenticate S.

The second protocol, called indirect join to the sink (IJS), allows a new node I to join the network through a neighbor node R that is already authenticated in the network. Node I sends an indirect request to S in order to establish a session key with R. Node R forwards without any modification the request to S through intermediate nodes that are trusted to route the request towards S. Only nodes I and S are able to decrypt the messages encrypted with $K_{DH}(I, S)$, and only R and S are able to decrypt the messages encrypted with $K_{DH}(S, R)$.



(a)



(b)

Figure 1. Join protocols. (a) DJS: direct join to the sink. Node I directly joins the network by communicating directly with the sink S. (b) IJS: indirect join to the sink. Intermediate nodes between R and S forward messages without any encryption or decryption.

2. Key and Protocol Dependency

3. It is important to note that cryptographic keys are interdependent. In order for a certain protocol to be executed, nodes should have the appropriate keys. In this section, we recapitulate the dependency between the keys for each of our protocols.

Figure 8 presents the dependency between keys for each of our protocolspublic/private keys of nodes are used to compute the shared symmetric keys using Diffie–Hellman without interaction described in Section 2.1.

Protocol KR is done based on information given by an IDS in order to revoke the keys in possession of compromised nodes. In executing KR, the sink informs the nodes in the neighborhood of the compromised node to delete all shared keys. Therefore, protocols RAK and MSK can be executed regularly or after the revocation process in order to maintain the security level of the application in the case of RAK or to create more secure links between nodes in the case of MSK. Protocol RNK must be renewed in a short interval or after a revocation process.

All of these keys, besides the network key NK, are predistributed to nodes before the deployment. Therefore, all protocols in Figure 8 plus KR can be executed at any time after deployment, except protocol RSK. Indeed, protocols DJS and IJS end with establishing a session key between the initiator and its neighbor. Therefore, protocol RSK must be executed after DJS and IJS in order to renew the session key.

Protocol name	Delivering
DJS	$pk(I) \xrightarrow{S:I} K(I,S)$
IJS	$K_{DH}(N,S) \xrightarrow{S:N} pk(I) \xrightarrow{N:I} K(I,N)$
RSK	$K(I,R)$ & $pk(R) \xrightarrow{I:R} K'(I,R)$
RAKnk$_a$	$NK \xrightarrow{S:I} pk'(S)$ $K_{DH}(S,I) \xrightarrow{S:I} pk'(I)\|sk'(I)$
RAKnk$_b$	$NK \xrightarrow{S:I} pk'(S)$ $K_{DH}(S,I) \xrightarrow{S:I} pk'(I)\|sk'(I)\|K'_{DH}(S,I)$
RAKdh$_a$	$K_{DH}(S,I) \xrightarrow{S:I} pk'(S)$ $K_{DH}(S,I) \xrightarrow{S:I} pk'(I)\|sk'(I)$
RAKdh$_b$	$K_{DH}(S,I) \xrightarrow{S:I} pk'(S)$ $K_{DH}(S,I) \xrightarrow{S:I} pk'(I)\|sk'(I)\|K'_{DH}(S,I)$
MSK$_a$	$K_{DH}(I,S) \xrightarrow{S:I} pk(R)$ $K_{DH}(R,S) \xrightarrow{S:R} pk(I)$
MSK$_b$	$K_{DH}(I,S) \xrightarrow{S:I} K_{DH}(I,R)$ $K_{DH}(R,S) \xrightarrow{S:R} K_{DH}(I,R)$

Key dependency for our protocols.

Results and Discussion

In Table, we provide the execution time for all of our protocols (please note that the execution time for the cryptographic algorithms can be found in [46]). We also present the results without the execution time of the sink, since in many applications, the base station is a special node with extra resources. All results are the averages of 100 experiments of each protocol. We also provide the standard deviations for execution time

including the time of $\mathcal{S}$. We notice that these values are small compared to the execution time of the protocol.

These variations are normal according to the motes used, physical parameters like 2.4 GHz interference, battery level, humidity, temperature, *etc*. Moreover, the cause of the high standard deviation of join protocols and RSK$_b$ presented in Table 2 is essentially related to therandom number (that is, of course, different at each generation) that is used in multiplication operations in the asymmetric encryption. In fact, when the random number is small, the multiplication operations take much less time than what they take for a large random number. To confirm this claim, we have tested with a given constant random number coded on 20 bytes. We obtained small standard deviation values, 5.94 ms for the DJS protocol and 5.71 ms for the IJS protocol.

4. Conclusions and Perspectives

We have proposed several protocols to revoke a set of nodes, to renew symmetric and asymmetric keys and to establish a shared key between two authenticated nodes of the network. Our solutions use the ECC, which allows nodes to easily construct shared keys without interaction. Moreover, all of our protocols have been automatically verified as secure using Scyther. This ensures the security of our solutions. We also have implemented and tested all of our protocols on TelosB nodes in order to evaluate their execution time relative to one another. These results show that according to the load of the network and to the topology, one protocol might be more efficient than another. Then, according to the context (size of the network, size of the battery, type of mote, energy consumption for communication, computation resources of the motes), one solution might be better than another one. All of these parameters should be taken into account before choosing one real solution

References

1. Hussain, M.A.; Khan, P.; Sup, K.K. WSN research activities for military
2. application. In Proceedings of the 11th International Conference on Advanced Communication Technology, Phoenix Park, Korea, 15–18 February 2009; Volume 1, pp. 271–274.
3. Debar, H.; Dacier, M.; Wespi, A.Towards a taxonomy of intrusion-detection systems. *Comput. Netw.* 1999, *31*, 805–822.
4. Sun, B.; Osborne, L.; Xiao, Y.; Guizani, S. Intrusion detection techniques in mobile ad hoc and wireless sensor networks. *IEEE Wirel. Commun.* 2007, *14*, 56–63.
5. Rivest, R.L.; Shamir, A.; Adleman, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *ACM Commun.* 1978, *21*, 120–126.

   El Gamal, T. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Trans. Inf. Theory* 1985, *31*, 469–472.
6. Eisenbarth, T.; Kumar, S.; Uhsadel, L.; Paar, C.; Poschmann, A. A Survey of Lightweight-Cryptography Implementations. *IEEE Des. Test Comput.* 2007, *24*, 522–533 .
7. Cazorla, M.; Marquet, K.; Minier, M. Survey and Benchmark of Lightweight Block Ciphers for Wireless Sensor Networks. In Proceedings of the 10th International Conference on Security and Cryptography (SECRYPT 2013), Reykjavík, Iceland, 29–31 July 2013; pp. 543–548.
8. Jeong, K.; Lee, C.; Lim, J. Improved differential fault analysis on lightweight block cipher LBlock for wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.* 2013, *2013*, doi:10.1186/1687-1499-2013-151.
9. Couroussé, D.; Robisson, B.; Lanet, J.;

   Barry, T.; Noura, H.; Jaillon, P.; Lalevée, P.COGITO:

10. Code Polymorphism to Secure Devices. In Proceedings of the 11th International Conference on Security and Cryptography (SECRYPT 2014), Vienna, Austria, 28–30 August 2014; pp. 451–456.
11. Zhang, J.; Varadharajan, V. Wireless sensor network key management survey and taxonomy. *J. Netw. Comput. Appl.* 2010, *33*, 63–75.
12. Mansour, I.; Chalhoub, G.; Misson, M. Security architecture for multi-hop wireless sensor networks. In *Security for Multihop Wireless Networks*; CRC Press: Boca Raton, FL, USA, 2014; pp. 157–178.
13. Cremers, C. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. In Proceedings of the 20th International Conference on Computer Aided Verification, Princeton, NJ, USA, 7–14 July 2008; pp. 414–418.
14. Mansour, I.; Chalhoub, G.; Lafourcade, P.; Delobel, F. Secure Key Renewal and Revocation for Wireless Sensor Networks. In Proceedings of the 39th IEEE Conference on Local Computer Networks (LCN), Edmonton, AB, Cananda, 8–11 September 2014; pp. 382–385 .
15. Mansour, I.; Chalhoub, G.; Lafourcade, P. Evaluation of Secure Multi-Hop Node Authentication and Key Establishment Mechanisms for Wireless Sensor Networks. *J. Sens. Actuator Netw.* 2014, *3*, 224–244.
16. Mansour, I.; Chalhoub, G.; Lafourcade, P. Secure Multihop Key Establishment Protocols for Wireless Sensor Networks. In Proceedings of International Conference on Cryptography and Security Systems, Lublin, Poland, 22–24 September 2014; pp. 166–177.
17. Mehta, M.; Huang, D.; Harn, L. RINK-RKP: A scheme for key predistribution and shared-key discovery in sensor networks. In Proceedings of the 24th IEEE International on Performance, Computing, and Communications Conference, Phoenix, AZ, USA, 7–9 April 2005; pp. 193–197.
18. Park, J.; Kim, Z.; Kim, K. State-based key management scheme for wireless sensor networks. In Proceedings of IEEE International Conference on Mobile Adhoc and Sensor Systems, Washington, DC, USA, 7 November 2005.
19. Park, J.; Kim, Z.; Kim, K. Random key assignment for secure wireless sensor networks. In Proceedings of the 1st ACM workshop on Security of Ad Hoc and Sensor Networks, Washington, DC, USA, 27–30 October 2003; pp. 62–71.
20. Cheng, Y.; Malik, M.; Xie, B.; Agrawal, D. Enhanced Approach for Random Key Pre-Distribution in Wireless Sensor Networks. In Proceedings of International Conference on Communication, Networking and Information Technology, Amman, Jordan, 6–8 December 2007.
21. Cheng, Y.; Agrawal, D. An Improved Key Distribution Mechanism for Large-Scale Hierarchical Wireless Networks Key Distribution. *AD HOC Netw. J.* 2007, *5*, 35–48.
22. Chan, H.; Gligor, V.; Perrig, A.; Muralidharan, G. On the distribution and revocation of cryptographic keys in sensor networks. *IEEE Trans. Dependable Secur.*

