An Elliptic Curve based Hierarchical Cluster Key Management in Wireless Sensor Network

Srikanta Kumar Sahoo and Manmanth Narayan Sahoo

National Institute of Technology Rourkela, Odisha, India
srikantsahoo1784@gmail.com,sahoom@nitrkl.ac.in

Abstract. In wireless sensor networks(WSN), because of the absence of physical protection and unattended deployment, the wireless connections are prone to different type of attacks. Hence, security is a measure concern in WSN. Moreover, the limited energy, memory and computation capability of sensor nodes, lead to difficulty in implementing security mechanisms effectively. In this paper we proposed an elliptic curve based hierarchical cluster key management scheme, which is very much secure, have better time complexity and consumes reasonable amount of energy. The proposed work uses digital signature scheme and encryption-decryption mechanisms using elliptic curve cryptography(ECC). As it is using ECC, the same level of security can be achieved with smaller key size compared to other mechanisms. The result shows that the proposed work is faster than Hamed and Khamys work, and it also guards against different type of attacks. Energy consumption, number of messages exchanged and key storage are three other aspects addressed in this work.

Keywords: WSN, ECC, ECDLP, ECDSA, ECDH, ECDH, RCH, CH, ECHCKM.

1 Introduction

Wireless sensor networks (WSNs) comprise a large number of spatially distributed small autonomous devices (called sensor nodes) cooperatively monitoring environmental conditions and sending the collected data to a command center using wireless channels [1]. Because of the size and cost of sensor nodes there is constraints on energy, memory, computation speed and bandwidth. Most of the applications of WSN needs secure communication. Because of the absence of the physical protection and the unattended deployment wireless communication and sensor nodes are prone to different type of attacks such as: impersonation, masquerading, spoofing and interception etc. Hence, security mechanisms in WSN is an important concern. Different security mechanisms in WSN are described in [2] and [4].

For implementing key management in WSN, it is important to select appropriate cryptographic methods. Cryptographic methods should meet the constraints of sensor nodes in WSNs. These cryptographic methods could be evaluated by code size, data size, processing time, and power consumption. Security

mechanisms can be implemented by using public key cryptography or symmetric key cryptography.

Most important public key algorithms include Rabin's Scheme, RSA, and Elliptic Curve Cryptography (ECC). In RSA to implement security operations thousands of multiplication instructions are performed, which is time consuming. Brown et al. found that encryption and decryption operations in RSA usually takes on the order of tens of seconds [12]. Recent studies have shown that it is feasible to apply public key cryptography to sensor networks by selecting proper algorithms and associated parameters. Most of the literatures gives emphasis on RSA and ECC algorithms. Researchers are more attracted towards ECC, because it provides same level of security with much smaller key size. For example, RSA with 1024 bit key provides an accepted level of security whereas ECC with 160 bit key provides same level of security. The RSA private key operation limits its use in sensor nodes. ECC has no such problem because both the public key and private key operation use the same point multiplication operations.

Rest of this paper is structured as follows: Section 2 includes related works. Section 3 presents proposed Elliptic Curve based Hierarchical Cluster Key Management scheme (ECHCKM). Section 4 depicts security analysis and section 5 includes performance analysis. Conclusions are finally drawn in Section 6.

2 Related works

Naureen et al. in [10] proposed performance and security assessment of a PKC based key management scheme for hierarchical sensor networks. The author claims the architecture is secure one but there is possibility of snooping, modification, replay and masquerading attack. Jabeenbegum et al. in [8] proposed a cluster based cost effective contributory key agreement protocol for secure group communication. In practical applications suppose an adversary node present in the group. Group controller gets public key of adversary node also. Group Controller generates group key as GK=n_{GK} * (P1+P2+P3+55P_n+Padv) and distributes to all. Thus adversary gets the key. Dahshan et al. in [9] proposed a distributed key management protocol in MANET using elliptic curve cryptography. In this scheme if an ordinary node wants to join the network, it has to contact a threshold number of servers. The node joining time may be bit more.

Khamy et al. in [7] proposed a new low complexity key exchange and encryption protocols for wireless sensor networks clusters based on elliptic curve cryptography. Here base node (BN) generates cluster symmetric key then sign it with own private key and asks for signed public key to cluster head (CH). CH generates signature, sends its public key and signature to BN. BN verifies the signature and gets the public key. BN generates Elliptic Curve Diffie Hellman(ECDH) key by multiplying public key of CH and its own private key. BN encrypts cluster symmetric key and signature with ECDH key and sends to CH. CH generates ECDH key by multiplying public key of BN and its own private key. CH decrypts cluster symmetric key and signature by using ECDH key. CH now finally verifies the signature. If signature is verified successfully,

then cluster symmetric key for the CH is accepted as shared key between BN and particular CH. Khamy used the same procedure to generate key between CH and cluster node (CN). The problem in Khamy's scheme is that, it uses two signature generation and verification (for public key signature and cluster symmetric key signature), two ECDH key generation and three encryption decryption (for cluster symmetric key and two signatures generated) hence it takes much time for key generation. Again each node maintains 4 keys (private key, public key, ECDH key and shared key).

Now a days, people need faster performance in every aspect of life. Organizations even work on improving computation speed of different products. In such a scenario message transfer time between nodes in wireless sensor network a big concern. Key generation time, key storage, number of computations have a great impact on message transfer. Moreover while concentrating on faster performance we compromise on security aspects. In the above discussed methods we found that, authors concentrate on security and leave behind key generation time. Problems on the above scheme motivate us to work on faster and secure key generation scheme, with minimized storage space and number of computation.

3 Proposed ECHCKM scheme

3.1 Network Architecture

In this work we consider a hierarchical sensor network setup, where nodes are classified into three categories, root cluster head (RCH), cluster head (CH) and sensor node (SN). RCH is highly powerful node with no constraint on energy and memory, which manages the CHs to work properly. CHs have sufficient energy and memory so that they can handle all cluster operations. RCH and CHs are nearly equal powerful. SNs are low end nodes with limited energy, memory, and computation capability. SNs are only capable of sensing the environmental activities and forward to RCH through CH. So CHs have additional responsibility for data aggregation and data processing. RCH is responsible for cluster key generation. CHs are responsible for key generation for SNs in that cluster and inter cluster data communication.

3.2 Assumptions

RCH and CHs are more powerful nodes in the network and cannot be compromised. We have not considered physical layer and media access control layer so that sharing the physical connection to the network is not possible. Before network key establishment the network divided into different cluster by using appropriate clustering algorithm. After cluster formation if an adversary node tries to pretend itself as a sensor node then CH is able to identify the adversary node and discard its requests. Every CH knows the identity (ID) and location of SNs present in that cluster.

The key generation process is divided into four phases: Initialization, cluster group key generation between sensor node and cluster head, root cluster group

key generation between cluster head and root cluster head and finally global group key generation. Figure 1 shows basic ECHCKM scheme in a cluster.

3.3 Initialization

All nodes in the network should have private-public key pair. Every node chooses a random number d_i between 0 and *n-1* as their private key and keeps it secret. Every node then finds their public key Q_i by multiplying generator point with private key. The key pair will be (d_i, Q_i=d_i * P), where P is generator point. To work with ECC for key management, all nodes in the network should agree upon elliptic curve parameters. Agreement upon an elliptic curve parameter is important because, if every node in the network choose their own elliptic curve then key synchronization cannot be done. Every node will work on different points on different elliptic curve and key generation is impossible. Hence it is required to work with a single elliptic curve and points on that curve. We consider an elliptic curve $y^2=x^3+ax+b$ defined over prime field. The ECC parameters in prime case are *(q,a,b,P,n,h)*. Where q is prime number defining field of elliptic curve, a and b are coefficients of elliptic curve, P is the generator point on elliptic curve, n is a non negative number defines order of P and h denotes cofactor preferable h=1.

3.4 Cluster group key generation (between CH and SNs):

The cluster group key for CH and SNs is generated as per the following steps:

1. CH generates a cluster group key GK_C:
   (a) CH chooses a random number s.
   (b) CH finds cluster group key $GK_C = s$ * P.
   For every sensor nodes in the cluster steps 2 to 10 are repeated.
2. CH chooses a random number x. And generates the message M=xP. Where M will be a point on elliptic curve defined.
3. CH encrypts M with the public key Q_i of sensor node SN_i:
   (a) CH chooses a random number r.
   (b) CH generates two points on elliptic curve: C1 = r * P and C2=r * Q_i +M.
4. CH generates ECDSA Signature of the message sig(M) by using its private key.
5. CH sends the encrypted points C1,C2 and sig(M) to SN_i.
6. The SN_i decrypts C1 and C2 using it's private key.
   (a) SN_i gets M by calculating: M=C2 - d_i * C1.
7. Sensor node SN_i verifies the signature. If signature is verified successfully SN_i sends an acknowledgment (ACK) to CH informing that signature is verified and the message is received. If signature verification fails, then SN_i sends request (REQ) to CH for another message and its signature. As soon as CH receives a REQ it generates another message, encrypts it, generate the signature and send to SN. The process continues until an ACK is received.

8. Now both CH and SN_i knows M. The shared key (K_{shi}) between CH and SN_i will be the X coordinate of M. simply speaking the shared key is M.
9. CH finds the multiplicative inverse of K_{shi}, multiply it with cluster group key GK_C to find K_{temp} and sends it to SN_i.

$$K_{temp} = GK_C * K_{shi}^{-1} \tag{1}$$

10. SN_i multiplies K_{shi} with K_{temp} to get GK_C ($GK_C = K_{temp} * K_{shi}$).
11. After generation of cluster group key SN_i deletes shared key K_{shi}.

Every SN will get the same cluster group key because, initial GK_C value is fixed for all and

$$K_{shi} * K_{shi}^{-1} = 1(modq) \tag{2}$$

3.5 Root Cluster group key generation (between RCH and CHs):

The above method and steps followed to generate the group key for different CHs with RCH. At the end of this phase every CHs and RCH have the same root cluster group key. Fig 1 shows the steps for generating the cluster group key between CH and SNs.

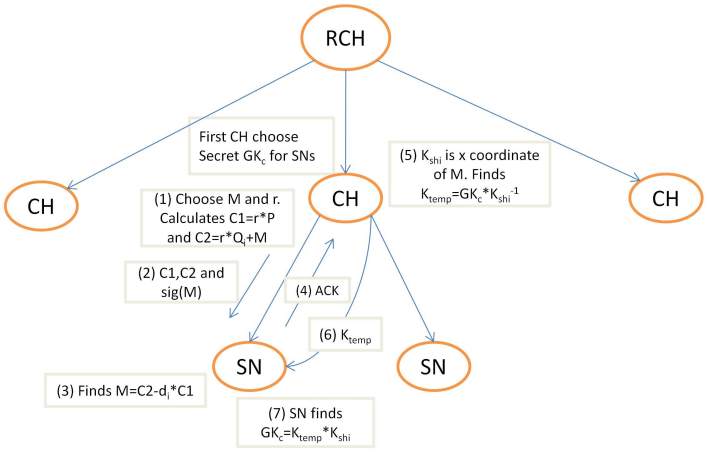

Fig. 1. ECHCKM Scheme

3.6 Global Group key generation:

The global group key for all network nodes generated in the following steps:

1. RCH generates a global group key GGK.
2. RCH encrypts GGK with root cluster group key (GK_{RC}) and sends to each CH.
3. CH decrypts it to get GGK.
4. CH encrypts GGK with cluster group key (GK_C) and sends to each SN.
5. CH decrypts it to get GGK.

3.7 Node addition and deletion

In our scheme node addition and deletion can be done easily without any complexity. No rekeying is required here, because in the key generation process every SN gets the group key with separate communication with CH. Each node is having the location and ID of near by CHs. When a SN wants to join another cluster, by clustering algorithm it gets into it and gets public key of CH of that cluster. Now it can separately establish key with CH and finally gets GK. When a SN leaves CH it first informs CH, delete the group key and then leaves.

4 Security Analysis

In the proposed scheme for generating the intermediate shared key between CH and SN or RCH and CH the encrypted message and its signature are sent. The group key is also encrypted and sent to the descendant node. Hence, even though attacker intercepts the data in between, she cannot get the secret message. Suppose attacker intercepts data during transmission. She will get encrypted message, not the concrete message. To modify concrete message she has to decrypt the message. Breaking an ECC encryption is too difficult, she has to solve ECDLP problem. Hence, our scheme is secure against snooping and modification. We have considered CHs are highly secure nodes with tamper resistance hardware, which cannot be compromised. So attacker cannot masquerade as CH. Here most important transmissions are from CH. So replay attack will not help attacker much. She may replay the REQ to get another message, again that message sent to particular node with its ID and encrypted with its public key. Attacker needs private key of SN to break it which is not available with her. The SN cannot repudiate because CH has an ACK value that indicates message received.

In ciphertext-only attack, eve has access to some cipher texts. If eve knows the algorithm she can analyze different ciphertexts to get plaintext. Knowing signature generation algorithm and encryption algorithm is not sufficient for eve to get secret message, for this she has to solve ECDLP problem. For decryption she has to do point divisions, which is practically very difficult. By knowing signature he can not get message as it is signed by private key of CH. In other case by knowing K_{temp} she cannot get GK_C for the same reason. In known-plaintext attack, eve has access to some plaintext/ciphertext pairs. In our method plaintext is never revealed. Let it be the case, then attacker has to solve

$$K_{shi}^{-1} = K_{temp}/GK_C \tag{3}$$

ECPoint/ECpoint never defined in ECC. On the other case, by knowing M, C1 and C2 attacker can not know r. For this she has to solve two equations C1=r*P and C2-M=r*Q_i, which is difficult. For Chosen - plaintext and chosen - ciphertext attack attacks attacker needs access to sender/receiver computers. We have not considered physical layer security. Again attacker cannot have access CHs computer, as it is loaded with tamper resistance hardware.

5 Performance Analysis

In this section we have compared Group Key generation time, total number of messages exchanged, total key storage and energy consumption with Khamy's scheme.

5.1 Simulation platform

In our key generation scheme we consider elliptic curve cryptography among all other public key cryptography techniques in order to get better performance. We used java programming for ECC implementation. To provide Java Cryptographic Extension (JCE) we used open code library Bouncy Castle provider 1.47 [25]. Bouncy Castle provides packages, interfaces and classes to handle different cryptographic algorithms. We have considered named elliptic curve P-224 defined in the package org.bouncycastle.jce.ECNamedCurve.

5.2 Time for Key Generation

Here we have analyzed group key generation time for ECHCKM and Khamy's scheme. We have executed both key generation algorithm on a system with Pentium-4 processor, 3.20GHz processor speed and 2GB RAM. The result graph shows that with increase in number of nodes in the network the difference between key generation time for both schemes increases. We have calculated summation of time taken for key generation, for varying number of nodes from 50-250 with an interval 50 and found that our scheme is near about 30 percent faster than Khamy's scheme. Fig 2 shows the key generation time in the complete network and in a cluster.

5.3 Key storage

- The RCH maintains one private key, one public key, m shared key, m signatures, a root cluster group key and a global group key. All total RCH stores ($2m$+4) keys.
- Each CH maintains one private key, one public key, two group keys (one each for RCH and CH), (1+n) shared keys (one for RCH and one each for n number of SNs in the cluster), (1+n) signatures (one for RCH and one each for n number of SNs in the cluster) and one global group key. All total each CH stores ($2n$+7) keys.

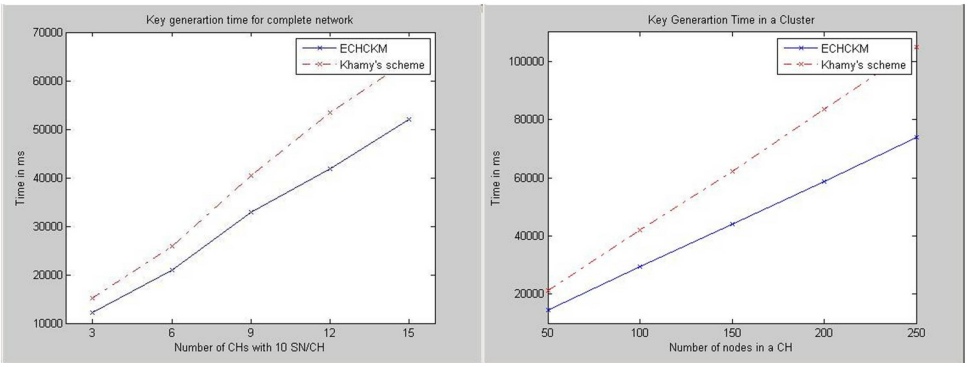

Fig. 2. Key generation time

– Each SN maintains one private key, one public key, one signature, one shared key, one cluster group key and one global group key. All total each SN maintains 6 keys.

So total number of keys stored in the network is *8mn+9m+4*. Whereas in Khamy's scheme total number of keys stored is *10mn+8m+4*.

5.4 Number of Operations

– RCH does one private/public key pair generation, $3m$ encryptions (encryption of message M, GK_{RC} and GGK) and m number of signature generation (for each CH).
– Each CH does one private/public key pair generation, n signature generations (for each SN in the cluster), one signature verification, $3n$ encryptions (encryption of message M, GK_C and GGK) and three number of decryption.
– Each SN does one private/public key pair generation, one signature verification and three number of decryption.

So total number of operations for group key generation in the network is *9mn+9m+1*. Whereas in Khamy's scheme total number of operations is *13mn+16m+3*.

5.5 Number of Messages Exchanged

For cluster group key generation four messages exchanged. (1) CH sends signature of message. (2) CH sends encrypted message. (3) SN replays with a REQ for another message and signature or with an ACK indicating that signature is verified. (4) CH sends encrypted GK_C.

For root cluster group key generation also four messages exchanged. (1) RCH sends signature of message. (2) RCH sends encrypted message. (3) CH replays with a REQ for another message and signature or with an ACK indicating that signature is verified. (4) RCH sends encrypted GK.

For global group key generation two messages exchanged. (1) RCH broadcasts encrypted GGK to CHs. (2) CH broadcasts encrypted GGK to SNs.

Hence total number of messages exchanged is $4mn+4m+2$. Whereas in Khamy's scheme total number of messages exchanged is $5mn+5m+4$.

5.6 Energy Consumption

In the proposed work we assumed that the radio dissipates E_{elec}=50 nJ/bit to run the transmitter or receiver circuitry and E_{amp}=100pJ/bit/m^2 for the transmit amplifier to achieve an acceptable E_b/Node [15]. So the energy required to transmit k bit of data over a distance d is $E_t = E_{elec}\text{k} + E_{amp}\text{k}d^2$ and energy required to receive k bit of data is $E_r = E_{elec}\text{k}$. Where d is distance between source and sink. Hence in the direct communication with base station the energy consumption is E=E_t+E_r. Here we have considered that in the worst case the sensor nodes are at a distance of $100m$ from cluster head, which is the transmission range of CH. Fig 3 shows energy consumptions for both the schemes.

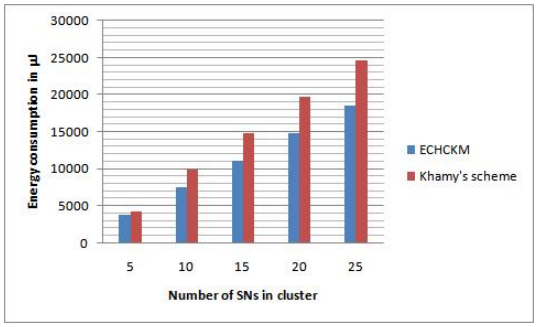

Fig. 3. Energy consumption in a cluster

6 CONCLUSION

The key generation time in wireless sensor network has been a major research area in recent years, because industry needs computationally faster and secure products. In the above study we proposed ECHCKM scheme which is a secure and resilient hierarchical cluster key establishment technique with shorter keys. The experimental result shows that our scheme is much faster than Khamy's scheme. Our scheme also have better performance in terms of key storage, number of operations performed and number of messages exchanged during key establishment.

References

1. Junqi Zhang , Vijay Varadharajan; "Wireless sensor network key management survey and taxonomy"; Journal of Network and Computer Applications, vol. 33, pp. 63-75, 2010.
2. X Chen, K Makki, K Yen and N Pissinou; "Sensor Network Security: A Survey"; IEEE communication survey and tutorials, vol. 11, pp. 52-73, 2009..
3. D Johnson, A Menezes and S Vanstone; "The Elliptic Curve Digital Signature Algorithm (ECDSA)", Certicom Corporation, 2001.
4. Yong Wang, Garhan Attebury, Byrav Ramamurthy; "A Survey of Security Issues In Wireless Sensor Networks", IEEE Communications Surveys and Tutorials, volume 8, pp. 2-23, 2nd quarter 2006.
5. K Lauter; "The advantages of elliptic curve cryptography for wireless security", IEEE Wireless Communications , vlo 11, pp. 62-67, February 2004
6. Nils Gura, Arun Patel, Arvinderpal Wander,Hans Eberle, Sheueling Chang Shantz; "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs",Springer, vol. 3156, pp.925-943, 2004.
7. A Hamed and S EL-Khamy; "New Low Complexity Key Exchange and Encryption protocols for Wireless Sensor Networks Clusters based on Elliptic Curve Cryptography", 26tl1 NATIONAL RADIO SCIENCE CONFERENCE, NRSC, pp.1-13, 2009.
8. S Jabeenbegum, T Purusothaman; "A Cluster Based Cost Effective Contributory Key Agreement Protocol for Secure Group Communication", Second International conference on Computing, Communication and Networking, IEEE, pp.1-12, 2010.
9. Hisham Dahshan and James Irvine; "An Elliptic Curve Distributed Key Management for Mobile Ad Hoc Networks", IEEE Communication, pp-1-5, 2010.
10. A Naureen, K Kim, F Ahmed; "Performance and Security Assessment of a PKC Based Key Management Scheme for Hierarchical Sensor Networks", IEEE Communication, pp.163-167, 2008.
11. Darrel Hankerson, Alfred Menezes, Scott Vanstone; "Guide to Elliptic Curve Cryptography"; Springer-Verlag New York, Inc; ISBN 0-387-95273-X; 2004.
12. D. W. Carman, P. S. Kruus, and B. J. Matt, "Constraints and Approaches for Distributed Sensor Network Security", NAI Labs, Tech. Report 00-010, 2000.
13. M. Brown, "PGP in Constrained Wireless Devices", Proc. 9th USENIX Security Symp., Aug. 2000.
14. S. Wander, N Gura, H Eberle, V Gupta, C Shantz; "Energy Analysis of Public-Key Cryptography for Wireless Sensor Networks", PerCom '05: Proc. 3rd IEEE Int'l. Conf. Pervasive Computing and Commun., pp. 324-328, Mar. 2005
15. R Heinzelman, A Chandrakasan, H Balakrishnan; "Energy Efficent Communication Protocol for Wireless Microsensor Network" , Proceedings of the 33rd Hawaii International Conference on System Sciences(HICSS'00),Maui, HI, Jan, 2000.
16. A. Simplicio, M. Barreto, B. Margi; "A survey on key management mechanisms for distributed Wireless Sensor Networks", Computer Networks, vol-54, ELSEVIER, 2010.
17. David Hook; "Beginning Cryptography with Java", Wrox Press, ISBN:0764596330, 2005.