International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

An Enhanced Secure Data Transfer using ECRSA - DSA with Deffie Hellman Key Exchange in Wireless Sensor Networks

Sindu .B[1], S. Sathya Priya[2]

[1]M. Phil Scholar, Computer Science, Sankar College of Science & Commerce, Saravanampatty, Coimbatore

[2]Professor, Computer Science, Sankar College of Science & Commerce, Saravanampatty, Coimbatore

Abstract: ***Recently, several data aggregation schemes based on privacy homomorphism encryption have been proposed and investigated on wireless sensor networks. These data aggregation schemes provide better security compared with traditional aggregation since cluster heads (aggregator) can directly aggregate the cipher texts without decryption; consequently, transmission overhead is reduced. However, the base station only retrieves the aggregated result, not individual data, which causes two problems. First, the usage of aggregation functions is constrained. Second, the base station cannot confirm data integrity and authenticity via attaching message digests or signatures to each sensing sample. This paper attempted to overcome the above two drawbacks. In this proposed design, the base station can recover all sensing data even these data has been aggregated. This property is called "recoverable." Furthermore, the design has been generalized and adopted on both homogeneous and heterogeneous wireless sensor networks. The proposed work uses the elliptic curve Deffie- Hellman key exchange with RSA for file encryption and decryption and the signature was generated using message digest method namely Digital signature Algorithm. The proposed work was implemented using .Net. Experiment results demonstrate that the transmission overhead is still reduced even if this approach is recoverable on sensing data.***

Keywords: Data aggregation, wireless sensor networks, privacy homomorphism encryption

1. Introduction

A WSN is a network consisting of numerous sensor nodes with sensing, wireless communications and computing capabilities. These sensor nodes are scattered in an unattended environment (i.e. sensing field) to sense the physical world. WSNs can be used for infrastructure security and counterterrorism applications. Critical buildings and facilities such as power plants, airports, and military bases have to be protected from potential invasions. Networks of video, acoustic, and other sensors can be deployed around these facilities [1]. Since only aggregated results reach the base station.

Unfortunately, an adversary has the ability to capture cluster heads. It would cause the compromise of the whole cluster; consequently, several schemes, such as ESPDA [4] and SRDA [5], have been proposed. However, these schemes restrict the data type of aggregation or cause extra transmission overhead. Besides, an adversary can still obtain the sensing data of its cluster members after capturing a cluster head.

To solve above problems completely, two ideas are used in recent research [6], [7], [8]. First, data are encrypted during transmission. Second, cluster heads directly aggregate encrypted data without decryption. A well-known approach named Concealed Data Aggregation (CDA) [6] has been proposed based on these two ideas. CDA provides both end-to-end encryption and in-networking processing in WSN. Since CDA applies privacy homomorphism (PH). Encryption with additive homomorphism, cluster heads are capable of executing addition operations on encrypted numeric data. Later, several PH-based data aggregation schemes [7], [8] have been proposed to achieve higher security levels

2. Related Works

Numerous secure data aggregation schemes have been proposed. These schemes are designed for different security requirements. Recently many data aggregation protocols have been proposed to eliminate the data redundancy in sensor data of the network, hence reducing the communication cost and energy expenditure in data collection. Wagner[11] proposed a mathematical framework for formally evaluating the security of several resilient aggregation techniques. For example, median is a more robust estimator than mean; truncation and trimming can be used to eliminate possible outliers. This work, however, is not really about data aggregation because it assumes the BS has already collected all the raw data. Also, abnormal data are discarded without further reasoning. Hu and Evans [12] propose a secure hop-by-hop data aggregation scheme that works if one node is compromised. They also assume that only leaf nodes in the tree topology sense data whereas the intermediate nodes do not have their own readings. SDAP can tolerate more compromised nodes and allows every node to input its own readings.

Du et al. [13] proposes a mechanism that allows the base station to check the aggregated values submitted by several designated aggregators, based on the endorsements provided by a certain number of witness nodes around the aggregators. Their scheme does not provide per-hop aggregation. Also it is assumed that sensing nodes can be trusted and witness nodes do not collude with the aggregators. However, this condition may not always hold in practice.

Przydatek et al. [9] present SIA, a Secure Information Aggregation scheme for sensor networks where a fraction of sensor nodes may be compromised. In their model, the BS is the only aggregator, which collects the authenticated raw data from all the sensor nodes in the network. The aggregator then computes an aggregation result over the raw data together with a commitment to the data based on a Merkle-hash tree and then sends them to a trustable remote user, who later challenges the aggregator to verify the aggregate. In [10] showed that even if a few compromised nodes contribute false sub-aggregate values, this result in large errors in the aggregate computed at the root of the hierarchy. The authors present modifications to the aggregation algorithms that guard against such attacks, i.e., algorithms for resilient hierarchical data aggregation despite the presence of compromised nodes in the aggregation hierarchy. The performance and costs of our approach via both analysis and simulate on is evaluated. Our results show that our approach is scalable and efficient.

3. Overview of Proposed Architecture with Algorithm

In this chapter, describes about the network models and define the attack model. Then, Mykletun et al.'s and Boneh schemes are reviewed since they are the foundation of the proposed schemes

3.1 Network Model

A WSN is controlled by a base station (BS). A BS has large bandwidth, strong computing capability, sufficient memory, and stable power to support the cryptographic and routing requirements of the whole WSN. Besides the BS, sensors (SNs) are also deployed to sense and gather responsible results for the BS. Typical SNs are small and low cost; hence, SNs are limited on computation, storage, and communication capability. Generally, all SNs in a WSN may be divided into several clusters. Cluster-based WSN has several advantages such as efficient energy management, better scalability of MAC or routing. Each cluster has a cluster head responsible for collecting and aggregating sensing data from SNs within the same cluster. A CH then sends the aggregation results to the BS. In a homogeneous WSN, cluster heads act as normal SNs. On the other hand, cluster heads act as by powerful high-end sensors (H-Sensors), in a heterogeneous WSN which incorporates different types of SNs with different capabilities.

3.2 Attack Model

The attack model is defined based on the ability of adversaries. Here, we consider the following three cases:

1) Without compromising any SN or CH. An adversary can only eavesdrop on packets in the air, so he can modify or inject the forged messages with this public information.
2) Compromising SNs. After compromising a SN, an adversary can obtain secrets such as encryption/ decryption keys. Then, an adversary can obtain sensing data and packets passed through the captured SN or impersonate this compromised sensor to forge malicious data.
3) Compromising CHs. After compromising a CH, an adversary can obtain the secrets and perform the following attacks. First, an adversary can decrypt the ciphertext of sensing data sent by its cluster members. Second, an adversary can generate forged aggregation results.

3.3 Proposed Encryption Scheme

The proposed encryption scheme is a concealed data aggregation scheme based on the elliptic curve RSA (EC-RSA) cryptosystem. It consists of four procedures: key generation (KeyGen), encryption (Enc), aggregation (Agg), and decryption (Dec). The symbol þ and $_3$ denote addition and scalar multiplication on elliptic curve points, respectively. Proposed Signature Scheme Boneh proposed an aggregate signature scheme which merges a set of distinct signatures into one aggregated signature. This scheme consists of five procedures: key generation (KeyGen), signing (Sign), verifying (Verify), aggregation (Agg), and verifying aggregated signature (Agg-Verify). Boneh et al.'s scheme is based on bilinear map The process latest several months during which Rivest proposed approaches, Adleman attacked them and Shamir recalls doing some of each. In cryptography, RSA is an algorithm for public-key cryptography which was given by Rivest, Shamir and Adleman. According to Mathematically The RSA algorithm is based on the mathematical part that is easy to find and multiple two large prime numbers together, but it is extremely difficult to factor their product.There are some important steps are involved in a RSA algorithm to solve a problem as given below:

Step 1: Assume two large prime numbers p & q.
Step 2: Compute: N = p*q Where N is the factor of two large prime number.
Step 3: Select an Encryption key (E) such that it is not a factor of (p-1)*(q-1) i.e. Ø(n)= (p-1)*(q-1) for calculating encryption exponents E, should be 1< E < Ø(n) such that gcd(E, Ø(n)=1 The main purpose of calculating gcd is that E & Ø(n) should be relative prime. Where Ø (n) is the Euler Totient Function & E is the Encryption Key.
Step 4: Select the Decryption key (D), which satisfy the Equation D*E mod (p-1)*(q-1) = 1
Step 5: For Encryption: Cipher Text= (Plain Text)E mod N
CT = (PT) E mod N

4. Cryptography Diffie Hellman Key Exchange (ECDH)

Public key cryptography was first publicly proposed in 1975 by Stanford University researchers Whitfield Diffie and Martin Hellman to provide a secure solution for confidentially exchanging information online. This paper looks at the implementation of the Diffie Hellman algorithm using Elliptic Curve Cryptography.

4.1 Diffie-Hellman Key Agreement

Diffie-Hellman key agreement is not based on encryption and decryption, but instead relies on mathematical functions that enable two parties to generate a shared secret key for

exchanging information confidentially online. Essentially, each party agrees on a public value *g* and a large prime number *p*. Next, one party chooses a secret value *x* and the other party chooses a secret value *y*. Both parties use their secret values to derive public values, *g* x mod *p* and *g* y mod *p*, and they exchange the public values. Each party then uses the other party's public value to calculate the shared secret key that is used by both parties for confidential communications. A third party cannot derive the shared secret key because they do not know either of the secret values, *x* or *y*.

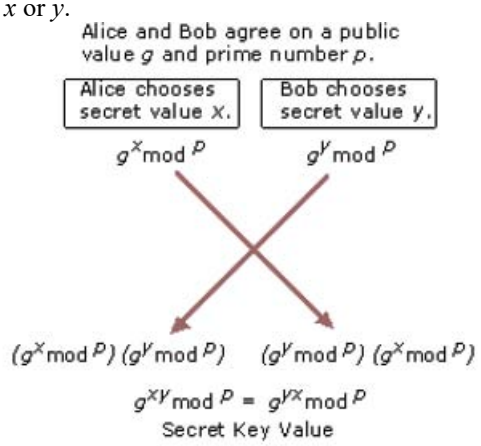

Figure 4.1: Shows the basic Diffie-Hellman Key

agreement process. For example, Alice chooses secret value *x* and sends the public value *g* x mod *p* to Bob. Bob chooses secret value *y* and sends the public value *g* y mod *p* to Alice. Alice uses the value *g* xy mod *p* as her secret key for confidential communications with Bob. Bob uses the value *g* yx mod *p* as his secret key. Because *g* xy mod *p* equals *g* yx mod *p*, Alice and Bob can use their secret keys with a symmetric key algorithm to conduct confidential online communications. The use of the mod function ensures that both parties can calculate the same secret key value, but an eavesdropper cannot. An eavesdropper can intercept the values of g and p but because of the extremely difficult mathematical problem created by the use of a large prime number in mod p, the eavesdropper cannot feasibly calculate either secret value x or secret value y . The secret key is known only to each party and is never visible on the network.

4.2 Elliptic Curve Domain parameters

Apart from the curve parameters a and b, there are other parameters that must be agreed by both parties involved in secured and trusted communication using Elliptic Curve Cryptography. These are domain parameters. The domain parameters for prime fields and binary fields are described below. The generation of domain parameters is out of scope of this paper. Generally the protocols implementing the Elliptic Curve Cryptography specify the domain parameters to be used.

4.3 Domain parameters for EC over field Fp

The domain parameters for Elliptic curve over Fp are p, a, b, G, n and h. p is the prime number defined for finite field Fp. a and b are the parameters defining the curve y2 mod p= x3 + ax + b mod p. G is the generator point (xG, yG), a point on the elliptic curve chosen for cryptographic operations. n is the order of the elliptic curve. The scalar for point multiplication is chosen as a number between 0 and n – 1. h is the cofactor where h = #E(Fp)/n. #E(Fp) is the number of points on an elliptic curve. Hasse Theorem states that: It is possible to define an addition rule to add points on *E*. The addition rule is specified as follows:

1) Rule to add the point at infinity to itself:

$$O+O = O:$$

2) Rule to add the point at infinity to any other point:

$$(x,y)+ O=O+(x,y)$$

4.4 Elliptic Curve Diffie Hellman (ECDH)

ECDH is a key agreement protocol that allows two parties to establish a shared secret key that can be used for private key algorithms. Both parties exchange some public information to each other. Using this public data and their own private data these parties calculates the shared secret. Any third party, who doesn't have access to the private details of each device, will not be able to calculate the shared secret from the available public information.
An overview of ECDH process is defined below.

Elliptic Curve Diffie Hellman Algorithm

- Decide domain parameters.
  Pa=na*P
  Pb=nb*P
  The end Alice computes KA=na*pb
- The end Bob computes
  KB= nb*pa
- Since
  na*pb=na*nb*P=nb*na*P
  =nb*pa .therefore KA=KB

Hence the shared secret key is K.

Verify: While receiving (ĉ,σ) from CH1, BS can recoverand verify each sensing data. Similarly, the BS may receive other ciphertext and signature pairs form other clusters. The BS can recover all sensing data within the whole WSN. After confirming the integrity of all data, the BS can perform any operations if it wants since all individual data are reverted.

5. RSA Encryption Scheme

1. Selection of large prime number (p, q): The main feature of RSA algorithm is the selection of large prime number (p, q) because it is logical that fraction of large number is always typical and any users or force attackers could not be able to find the capable numbers, timely to force attack is shortly non-feasible.
Example:
p = 5, q = 3 N= p*q
= 5*3 = 15
= 1*15 = 15*1 = 3*5 = 5*3

2. Selection of Encryption Key(E):
Selection of large of large prime fraction always create impact during the selection of Encryption key, if the factor is high then the estimation of Encryption is infeasible.
Example:

If p=7, q=17 must not be a factor of (p-1)*(q-1)
i.e. (7-1)*(17-1) = 6*16 = 96
= 2*2*2*2*2*3
So, E can be 5, 7, 11…

3. Selection of Decryption Key (D): Selection of large factors always create an effect on the Decryption key,there may be an inversely relation.
(E*D) mod (p-1)*(q-1) =1
D α 1/ [(E) (p) (q)]
Note: Some important points as given below:

According to Euler's Totient Function :
1. Ø (1) = 0
2. Ø (p) = p – 1 {if p is prime number}
3. Ø (m*n) = Ø (m)* Ø (n) {if m and n is relative prime number}
4. Ø (pe) = pe – p e -1 {if p is a prime number}

There is no need for a user to know the secret parameters p, q and Ø (n). The plain text or message (M) has the form of one or more positive integer M<N. Any user can use his private key to authenticate the communication. RSA cryptosystem provides the facility of digital signature scheme. The message consists of letters, numbers and special characters (i.e. stop, colon, space etc.). Each character is re presented by its own arrangement of Eight bits (o & 1). The most of the hardware & software products and standards that use public key technique for Encryption, Decryption etc. are based on RSA cryptosystem.

5.1 Elliptic Curve Digital Signature Algorithm based signature scheme

Suppose Alice wants to send a signed message to Bob. Initially, the curve parameters $(CURVE, G, n)$ must be agreed upon. In addition to the field and equation of the curve, need G, a base point of prime order on the curve; n is the multiplicative order of the point G.

Alice creates a key pair, consisting of a private key integer d_A, randomly selected in the interval $[1, n-1]$; and a public key curve point $Q_A = d_A * G$. We use $*$ to denote elliptic curve point multiplication by a scalar.

For Alice to sign a message m, she follows these steps:
1) Calculate $e = \text{HASH}(m)$, where HASH is a cryptographic hash function, such as SHA-1.
2) Let z be the L_n leftmost bits of e, where L_n is the bit length of the group order n.
3) Select a random integer k from $[1, n-1]$.
4) Calculate the curve point $(x_1, y_1) = k * G$.
5) Calculate $r = x_1 \pmod n$. If $r = 0$, go back to step 3.
6) Calculate $s = k^{-1}(z + rd_A) \pmod n$. If $s = 0$, go back to step 3.
7) The signature is the pair (r, s).

When computing S, the string Z resulting from HASH(m) shall be converted to an integer. Note that Z can be *greater* than n but not longer.

As the standard notes, it is crucial to select different k for different signatures, otherwise the equation in step 6 can be solved for d_A, the private key: Given two signatures (r, s) and (r, s'), employing the same unknown k for different known messages m and m', an attacker can calculate z and z', and since $s - s' = k^{-1}(z - z')$ (all operations in this paragraph are done modulo n) the attacker can find $k = \frac{z - z'}{s - s'}$. Since $s = k^{-1}(z + rd_A)$, the attacker can now calculate the private key $d_A = \frac{sk - z}{r}$

6. A RCDA Scheme for Homogeneous WSN (RCDA-HOMO)

RCDA-HOMO is composed of four procedures: Setup, Encrypt-Sign, Aggregate, and Verify. The Setup procedure is to prepare and install necessary secrets for the BS and each sensor. When a sensor decides to send sensing data to its CH, it performs Encrypt-Sign and sends the result to the CH. Once the CH receives all results from its members, it activates Aggregate to aggregate what it received, and then sends the final results (aggregated cipher text and signature) to the BS. The last procedure is Verify. The BS first extracts individual sensing data by decrypting the aggregated ciphertext. Afterward, the BS verifies the authenticity and integrity of the decrypted data based on the corresponding aggregated signature.

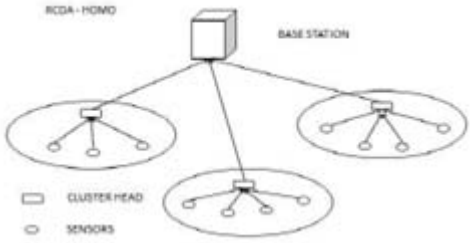

Figure 6.1: An example of Homogeneous WSN

To present RCDA-HOMO in a simple way, we choose Cluster 1 (see Fig. 4.2) as an example. SNω is selected as CH of Cluster 1 which contains the remaindering sensors, {SN1; SNω-1}. The detailed procedures are listed as follows:
Setup: BS generates the following key pairs:
1)(PSNi, RSNi): For each sensor SNi, the BS generates (PSNi; RSNi) by KeyGen procedure (see Boneh et al.'sscheme in Fig. 1) where PSNi=vi and RSNi=xi.
2)(PBS, RBS): These keys are generated by KeyGen procedure (see Mykletun et al.'s scheme in Fig. 1) where PBS= fY ; E; p; G; ng and RBS=ζ. After that, RSNi , PBS, and H are loaded to SNi for all i. Finally, the BS keeps all public keys PSNi and its own RBS in privacy.

Encrypt-Sign
This procedure is triggered while a sensor decides to send its sensing data to the cluster head. Detailed steps are listed as follows:

Receiver transmits her public key to sender and keeps the private key secret. Sender then wishes to send message M to Receiver. He first turns M into an integer by using an agreed-upon reversible protocol known as a padding scheme. He then computes the cipher text corresponding to $c = me(\mathrm{mod} n)$.

Receiver can recover from by using her private key exponent via computing $m = c^d (\mathrm{mod}\ n)$.

Given, she can recover the original message M by reversing the padding scheme

Elliptic Curve Cryptography Arithmetic

Elliptic Curve Cryptography involves mathematics of a different kind than the type used in other cryptographic algorithms. The figure 6.1 shows a hierarchical model of Elliptic Curve Cryptography. Elliptic Curve Cryptography is divided into three kinds of fields. Field over real numbers, field over prime numbers, and a binary Galois field. The main operations in Elliptic Curve Cryptography are Point Multiplication, Point Addition and Point Doubling. These operations can be performed over all kinds of fields, however this implementation deals only with the prime field, which is better suited for software implementation purposes.

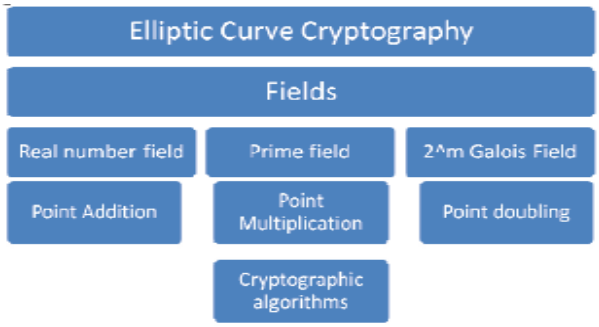

Figure 6.2: Hierarchical Elliptic Curve Cryptography model

7. RCDA for Heterogeneous WSN (RCDA-HETE)

Here, another environment is considered, heterogeneous WSN. A concealed data aggregation scheme for heterogeneous WSN has been proposed; however, this scheme does not provide data integrity and recovery. The RCDA-HETE scheme is first proposed. Later, another scheme named RCDA-HETE is proposed if HSensors are designed to be tamper-resistant.

A. RCDA-HETE Scheme

Actually, RCDA-HOMO can be applied to heterogeneous WSN without modification. The proposed approach native RCDA-HETE. Since H-Sensors are capable of stronger computation ability and stable power supply, they can perform more complex tasks than L-Sensors. Thus, H-sensors can act as cluster heads. Obviously, native RCDAHETE also achieves the Recovery property.

B. RCDA-HETE Scheme

Here, the attempt to fully exploit H-Sensors which have stronger computing capability. Operations on L-Sensors could be switched to H-Sensors. In addition, H-Sensors can be designed to be tamper-resistant, so this may allow H Sensors to store the partial secret information if required. With these considerations, redesign an RCDA scheme named RCDA-HETE.

While the use of tamper-resistant devices may raise the hardware cost; however, in a heterogeneous WSN, majority of sensors are low-end sensors. In our design, computation cost on L-Sensors is switched to H-Sensors, so L-Sensors can be very cheap and simple. In fact, the overall hardware cost is reduced. RCDA-HETE is composed of five procedures: Setup, Intra cluster Encrypt, Inter cluster Encrypt, Aggregate, and Verify. In the Setup procedure, necessary secrets are loaded to each H-Sensor and L-Sensor. Intra cluster Encrypt procedure involves when L-Sensors desire to send their sensing data to the H-Sensor. In the Inter cluster Encrypt procedure, each H-Sensor aggregates the received data and then encrypts and signs the aggregated result.

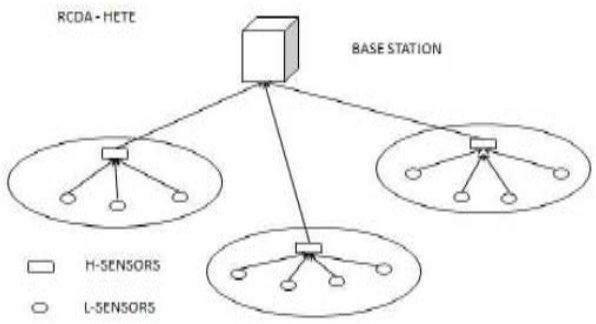

Figure 7.1: RCDA Heterogeneous

An Example of Heterogeneous WS

In addition, if an H-Sensor receives cipher texts and signatures from other H-Sensors on its routing path, it activates the Aggregate procedure. Finally, the Verify procedure ensures the authenticity and integrity of each aggregated result. To explain RCDA-HETE clearly, a heterogeneous WSN is given in Fig.7.1.

C. Proposed Algorithm Diagram

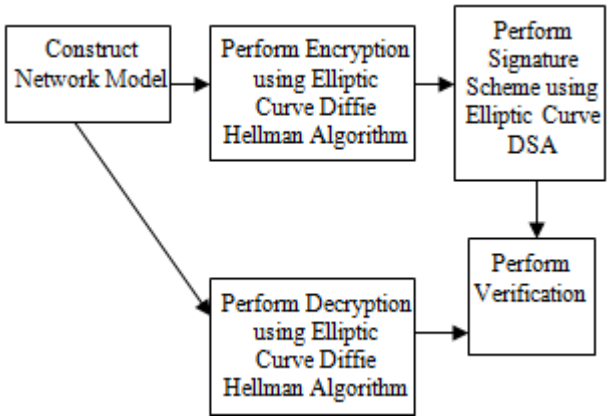

Figure 4.5: Proposed algorithm diagram

The Proposed scheme is composed of four procedures: Setup, Encrypt-Sign, Aggregate, and Verify. The Setup procedure is to prepare and install necessary secrets for the BS and each sensor. When a sensor decides to send sensing

data to its CH, it performs Encrypt-Sign and sends the result to the CH. Once the CH receives all results from its members, it activates Aggregate to aggregate what it received, and then sends the final results (aggregated ciphertext and signature) to the BS. The last procedure is Verify. The BS first extracts individual sensing data by decrypting the aggregated ciphertext. Afterward, the BS verifies the authenticity and integrity of the decrypted data based on the corresponding aggregated signature.

Proposed Algorithm

1. Setup the wireless network

BS generates the following key pairs:

- (PSNi , RSNi): For each sensor SNi, the BS generates (PSNi ; RSNi) by KeyGen procedure using Elliptic Curve Diffie Hellman key exchange where PSNi=vi and RSNi=xi.
- (PBS , RBS): These keys are generated by KeyGen procedure of Elliptic curve digitial Signature where PBS= fY ; E; p; G; ng and RBS=ζ. After that, RSNi , PBS, and H are loaded to SNi for all i. Finally, the BS keeps all public keys PSNi and its own RBS in privacy.

Then, the BS loads PBS to all L-Sensors. On the other hand, each H-Sensor is loaded

Its own key pair (PHi , RHi),PBS and several necessary aggregation functions. Intracluster Encrypt: This procedure ensures the establishment of a secure channel between L-Sensors and their HSensor. L_i^1 encrypts d_i^1 with K_i^1 and sends $E_{K_i^1}(d_i^1)$ to H1. After receiving $E_{k_i^1}(d_i^1)$, H1 decrypts the cipher texts to obtain the plaintext d_i^1

Intercluster Encrypt: After collecting all sensing data from all cluster members, an H-Sensor performs the preferred aggregation function on these data as its result. H1 select d_i^1 as the aggregated result by predefined property, such as maximum or minimum.

8. Experimental Results

To evaluate the performance of the proposed schemes, execution time (or "delay") is the main measurement of performance evaluation. Without loss of generality, this defines processing delay and aggregation delay for deployed sensors. Processing delay indicates the execution time for sensors to produce cipher texts and corresponding signatures before transmission. Aggregation delay is also evaluated by measuring time spent on processing time on aggregating cipher texts and signatures in the proposed schemes. The last delay, decryption delay, is not considered since the base station is considerably powerful as a workstation. Therefore, this delay is negligible and can be ignored. Another criterion is cost evaluation. Cost evaluation involves communication and computation aspects.

This proposed work was implemented using vb.net. The performance of this proposed work RCDA using Elliptic curve Deffie Hellman key exchange with DSA Signature Scheme was compared with two existing approaches Native RCDA and RCDA with Elliptic curve Elgamal Cryptography. The figure 5.1 below shows the configuration of the system requirement

	Configuration
CPU	Intel® Core™2 P7350 @ 2.00GHz
Memory	4.00GB DDR2 SDRAM
Hard-Disk	320GB (5400RPM)
operating system	Windows Vista™ Home Premium

Figure 5.1: Configuration of system requirement

The table 5.1 shows the performance comparison of the proposed method with other existing approaches based on the six different metrics processing delay, processing energy, aggregation delay , aggregation energy, payload size and communication cost.

Table 5.1: Performance Comparison

	RCDA-HOMO	RCDA using Elliptic Curve Elgamal	Proposed RCDA using Elliptic Curve Deffie Hellman Key Exchange
Processing Delay	3702.09	2984.06	2108.08
Processing Energy	12079.9	11106.2	10568.3
Aggregation Delay	73.71	70.68	49.69
Aggregation Energy	64.52	57.81	48.21
Pay Load Size	476	396	256
Communication Cost	338.4	253.6	153.6

Table 5.2: Comparison table between RSA and ECC

RSA and Diffie Hellman Key Size in bits	*Elliptic curve key size in bits*
1024	160
2048	224
3072	256
7680	384
15360	512

Performance comparison of proposed RCDA using EC-DH with existing approaches based on Processing Delay

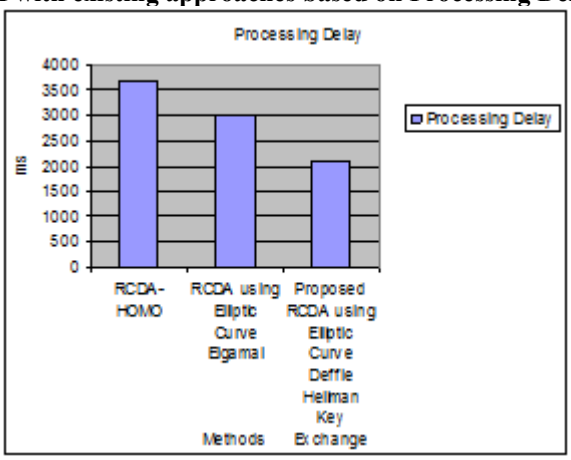

Figure 5.2: Comparison based on processing delay

From the chart it shows the performance measure based on the processing delay and our proposed approach RCDA using EC-DHE took less time while comparing the other methods and the worst time complexity is RCDA-HOMO

Performance comparison of proposed RCDA using EC-DH with existing approaches based on Processing Energy

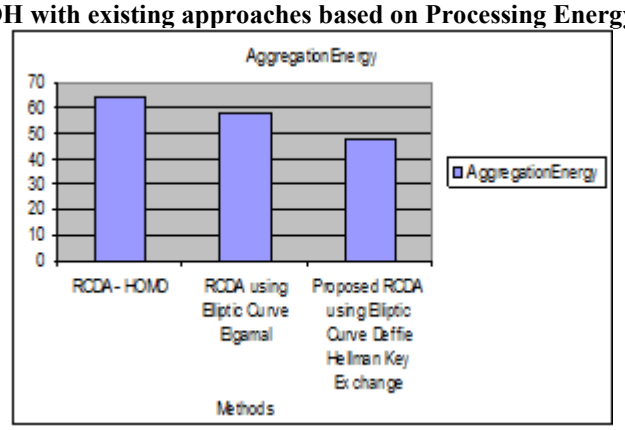

Figure 5.3: Comparison based on processing energy

Performance comparison of proposed RCDA using EC-DH with existing approaches based on Aggregation Delay

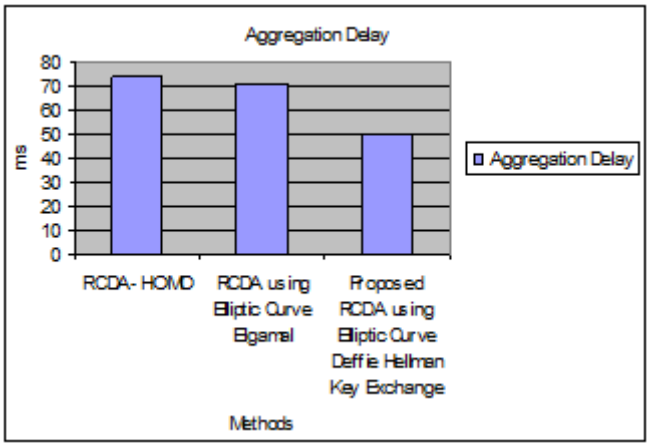

Figure 5.4: Comparison based on aggregation delay

From the chart it shows the performance measure based on the processing energy and our proposed approach RCDA using EC-DHE took less energy while comparing the other methods and the worst time complexity is RCDA-HOMO

Performance comparison of proposed RCDA using EC-DH with existing approaches based on Aggregation Energy

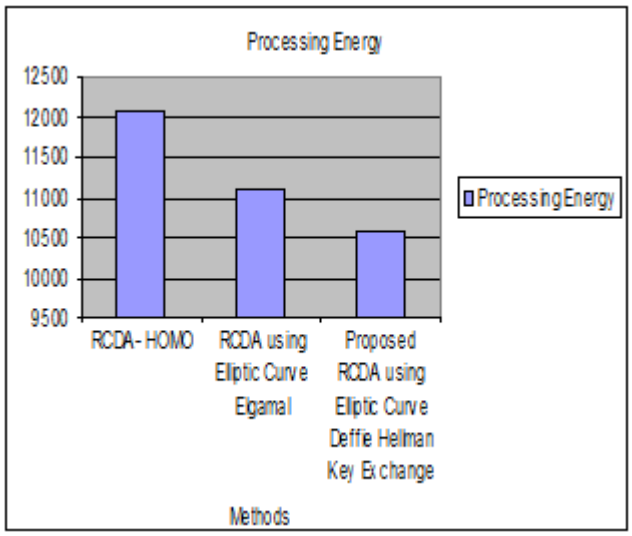

Figure 5.5: Comparison based on aggregation energy

From the chart it shows the performance measure based on the aggregation delay and our proposed approach RCDA using EC-DHE took less time while comparing the other methods and the worst time complexity is RCDA-HOMO.

Performance comparison of proposed RCDA using EC-DH with existing approaches based on Pay Load Size

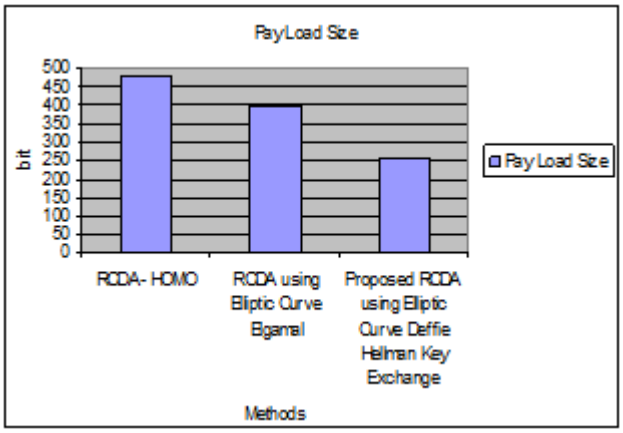

Figure 5.6: Comparison based on pay load size

From the chart it shows the performance measure based on the Payload Size and our proposed approach RCDA using EC-DHE took less load balancing while comparing the other methods and the worst time complexity is RCDA-HOMO

Performance comparison of proposed RCDA using EC-DH with existing approaches based on Communication Cost

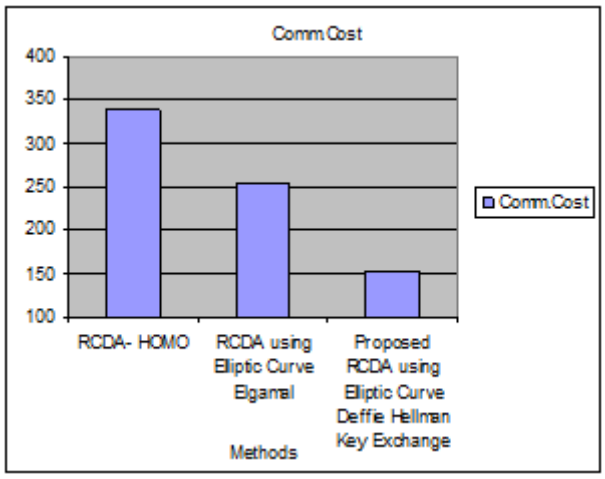

Figure 5.7: Comparison based on communication cost

From the chart it shows the performance measure based on the Aggregation Energy and our proposed approach RCDA using EC-DHE took less communication cost while comparing the other methods and the worst time complexity is RCDA-HOMO

9. Performance and Cost Evaluation

To evaluate the performance of the proposed schemes, execution time (or "delay") is the main measurement of performance evaluation. Without loss of generality, this defines processing delay and aggregation delay for deployed sensors. Processing delay indicates the execution time for sensors to produce cipher texts and corresponding signatures before transmission. Aggregation delay is also evaluated by measuring time spent on processing time on aggregating cipher texts and signatures in the proposed schemes. The last delay, decryption delay, is not considered since the base station is considerably powerful as a workstation. Therefore, this delay is negligible and can be ignored. Another criterion is cost evaluation. Cost evaluation involves communication and computation aspects.

This proposed work was implemented using vb.net. The performance of this proposed work RCDA using Elliptic curve Deffie Hellman key exchange with DSA Signature Scheme was compared with two existing approaches Native RCDA and RCDA with Elliptic curve Elgamal Cryptography.

10. Conclusion

Wireless sensor networks (WSN) have been widely deployed in many applications, e.g., military field surveillance, health care, environment monitor, accident report, etc. A WSN is composed of a large number of sensors which collaborates with each other. Each sensor detects a target within its radio range, performs simple computations, and communicates with other sensors. Generally, sensors are constrained in battery power, communication, and computation capability; therefore, reducing the power consumption is a critical concern for a WSN. This thesis has proposed recoverable concealed data aggregation schemes for homogeneous/heterogeneous WSNs. A special feature is that the base station can securely recover all sensing data rather than aggregated results, but the transmission overhead is still acceptable. Moreover, it integrates the aggregate signature scheme to ensure data authenticity and integrity in the design. Even though signatures bring additional costs, the proposed schemes are still affordable for WSNs after evaluation. Considering a large WSN (over 100 nodes), this thesis also performed simulations on the proposed schemes. Communication cost increases linearly when the size of cipher text increases

References

[1] Chong, C.-Y.& Kumar, S. P. (2003). Sensor networks: Evolution, opportunities, and challenges, *Proceedings of the IEEE* **91**(8): 1247–1256
[2] Rashid, R. & Robertson, G. (1981). Accent: A communication oriented network operating system kernel, *Proc. of the 8th Symposium on Operating System Principles*, pp. 64–75.
[3] Myers, C., Oppenheim, A., Davis, R. & Dove, W. (1984). Knowledge-based speech analysis and enhancement, Proc. of the International Conference on Acoustics, Speech and Signal Processing.
[4] Kumar, S. & Shepherd, D. (2001). Sensit: Sensor information technology for the warfighter, Proc. of the 4th International Conference on Information Fusion (FUSION'01), pp. 3–9 (TuC1).
[5] ZigBee Alliance (n.d.). http://www.zigbee.org.
[6] 21 Ideas for the 21st Century (1999). BusinessWeek pp. 78–167.
[7] Ni, L.M. (2008). China's national research project onwireless sensor networks, Proc. of the 2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'08), p. 1
[8] Crossbow Technology (n.d.), http://www.xbow.com.
[9] Kumar, Vimal; Sanjay K. Madria (August 2012). "Secure Hierarchical Data Aggregation in Wireless Sensor Networks: Performance Evaluation and Analysis". MDM 12
[10] S. Roy, S. Setia, and S. Jajodia, "Attack-Resilient Hierarchical Data Aggregation in Sensor Networks," Proc. ACM Fourth Workshop Security of Ad Hoc and Sensor Networks, pp. 71-82, 2006.
[11] W. Zhang and G. Cao, "Group Rekeying for Filtering False Data in Sensor Networks: A Predistribution and Local Collaboration-Based Approach," IEEE INFOCOM, March 2005.
[12] S. Ozdemir, "Concealed Data Aggregation in Heterogeneous Sensor Networks Using Privacy Homomorphism," Proc. IEEE Int'l Conf. Pervasive Services, pp. 165-168, July 2007
[13] H. Chan, A. Perrig, and D. Song, "Secure Hierarchical In-Network Aggregation in Sensor Networks," Proc. ACM 13th Conf. Computer and Comm. Security, pp. 278-287, 2006.