Secure Hierarchical Data Aggregation in Wireless Sensor Networks

Julia Albath
Missouri University of Science and Technology
Department of Computer Science
julia.albath@acm.org

Sanjay Madria
Missouri University of Science and Technology
Department of Computer Science
madrias@mst.edu

***Abstract*—Communication in wireless sensor networks uses the majority of a sensor's limited energy. Using aggregation in wireless sensor network reduces the overall communication cost. Security in wireless sensor networks entails many different challenges. Traditional end-to-end security is not suitable for use with in-network aggregation. A corrupted sensor has access to the data and can falsify results. Additively homomorphic encryption allows for aggregation of encrypted values, with the result being the same as the result when unencrypted data was aggregated. Using public key cryptography, digital signatures can be used to achieve integrity. We propose a new algorithm using homomorphic encryption and additive digital signatures to achieve confidentiality, integrity and availability for in-network aggregation in wireless sensor networks. We prove that our digital signature algorithm which is based on the Elliptic Curve Digital Signature Algorithm (ECDSA) is as secure as ECDSA.**

I. Introduction

Wireless sensor networks (WSN) are self-organizing networks of small, battery powered sensors used to monitor the environment for events such as forest fires, pollutant levels or enemy troop movements. A large number of small, battery powered computing devices with built-in radios are spread over the area to be monitored. Upon activation, these sensors self-organize into a multi-hop network, which connects to the users via a powerful base station in order to achieve a common goal [1]. As each sensor surveys the area within its sensing range, the data is sent towards the base station along a multi-hop path. A WSN is able to remotely cover a large sensing area since these low-cost sensors organize into a multi-hop network without human assistance.

Since sensors are typically battery powered and a WSN contains thousands of sensors, replacing the batteries is not a possibility. In terms of energy usage, communication is much more expensive than any internal computations [2]. In-network aggregation, intermediate results are calculated along the multi-hop path whenever two or more messages are routed along the same path. Depending on the routing structure, energy savings may be by as much as eight times [3].

Security in WSNs includes confidentiality, integrity and availability. Confidentiality in sensor networks is accomplished by preventing outsiders from eavesdropping on transmissions. This is generally achieved by encrypting the relevant parts of a packet. Integrity in general means that the receiver is assured that the packet was not tampered with or the message altered in some way. By ensuring availability we mean that the data is available in a timely fashion so that it is useful to the user. Availability in sensor networks is of great concern to the user of the network. Unfortunately, many existing security primitives can not be used in sensor networks, either because the computing power of the sensors is too limited or the additional work created by the protocols causes excessive network traffic [4].

Sensors in the network can become corrupted due to the environment such as water, wind or sand acting on the sensor. In hostile environments, a sensor may deliberately be corrupted by an attacker. A corrupted sensor may appear to participate in the mission of the network but falsify sensor readings, improperly apply an aggregation function, exclude legitimate messages from the aggregate result or create a fictitious result. A sensor corrupted by an attacker may behave in this way in order to get the base station to accept an incorrect result that is favorable to the attacker. Hence in order to securely aggregate data in a sensor network, we must not only provide protection against eavesdroppers, but we should also prevent intermediate sensors from having access to the data.

Homomorphic encryption schemes are one possibility of ensuring secure aggregation, as they allow data aggregation to be performed on encrypted data. Encryption and decryption operations are computationally very expensive and time consuming. In link-layer cryptography [5], the data is encrypted by the sender, decrypted at intermediate nodes, the aggregation function is applied and the result is encrypted again before being sent to the next hop; this can lead to overflowing queues. In homomorphic encryption certain aggregation functions such as sum and average can be calculated on the encrypted data, reducing the workload of the sensors in the network significantly. The data is encrypted and sent toward the base station, while sensors along the path apply the aggregation function on the encrypted data. The base station receives the encrypted aggregate result and decrypts it. In section II we describe the scheme of homomorphic encryption in detail. Homomorphic encryption schemes provide security against eavesdroppers and protect the aggregate result from being known by intermediate, possibly corrupted sensors.

Integrity of the aggregate result can easily be achieved on a hop-by-hop basis in wireless sensor networks. Achieving end-to-end integrity while allowing for data aggregation provides

978-1-4244-2948-6/09/$25.00 ©2009 IEEE

us with new challenges. We need to clarify the meaning of integrity when data aggregation is applied. In aggregation integrity implies that any aggregate result is made up of only legitimate data without inclusions or additions, and that corrupted sensors can not interfere with operations of the aggregation. We want to be able to assure the base station that the aggregate result it receives is a fair representation of the network state.

In this paper we introduce a novel way to provide confidential and integrity preserving aggregation in wireless sensor networks. In section III we propose the use of homomorphic encryption in WSN in order to achieve:

- A solution for confidentially calculating the SUM and AVERAGE in a wireless sensor network. Our algorithm is present in Section IV.
- A solution for integrity preserving data aggregation in wireless sensor networks. We are using an additively digital signature algorithm based on ECDSA to achieve integrity of the aggregate result. We provide security analysis of the algorithm in Section V.

II. Background

In sensor networks, data aggregation provides energy savings. The lifetime of most networks is limited and it is important for protocols to be energy-efficient. Combining multiple payloads into one message or combining data by allowing for in-network calculation of aggregates leads to these energy savings [6].

In many proposed applications of wireless sensor networks, the networks generate large amounts of data in a continuous stream, making it difficult for a user to sift useful information from these masses of data [7]. Sensors may generate a reading of their environment every three seconds until the mission is completed; for a 100 sensor network, this would generate 300 messages every three seconds, or 6000 messages every minute. In-network aggregation can take this amount of data and combine it into one or more aggregated results every minute.

Calculating aggregate results such as SUM or AVERAGE is of special interest to sensor networks. Wireless sensor networks are designed to provide large amounts of data, which is a snapshot of the environment at one point in time. Combining several readings by calculating the AVERAGE or the SUM increases the accuracy of these readings [8].

Security in sensor networks requires new approaches due to the limitations of sensors and their limited computing power. Since a sensor network uses radio as the communication medium, all communications are inherently insecure. Anybody tuned to the same channel is able to eavesdrop on the transmissions. Many sensor network applications demand secure communications. Encryption is the preferred way to provide for a secure communication channel. Encryption ensures that only the sender and the intended receiver can read the message contents [9]. Traditional link-layer cryptography is an important part of an overall security strategy for sensor networks.

TinySec [5], an implementation of link-layer cryptography for TinyOS, has two operational modes: hop-by-hop (HBH) and end-to-end (ETE). ETE provides total security, as only the sender and the receiving base station are able to know the content of the message. Unfortunately, this also means that aggregation is not possible, because the intermediate nodes can not access the payload. TinySec uses the SkipJack cipher, which does not allow for calculating of aggregate functions such as SUM or AVERAGE on encrypted data. When TinySec is used in HBH mode instead, the message is decrypted at each hop, and aggregation is possible. Due to the amount of time required for the encryption and decryption operations, the queues in a WSN can overflow, leading to dropped packets. Other drawbacks are that TinySec is based on private key cryptography, which leads to problems such as key distribution, key management, and that digital signatures are not possible [10].

In private key cryptography, both parties use the same key. Deciding when and how two sensors agree on which key to use is a big challenge. A private key cryptography approach also means that a sensor needs to store one key for every other sensor it wishes to communicate with. In WSN, the topology of the network can change, and protocols need to be flexible enough to allow for two previously unassociated sensors to begin communicating securely. One possible solution is a network-wide key, but the obvious problem with this approach is that only one corrupted sensor is required to compromise communications in the entire network.

Since all parties in private key cryptography use the same key, digital signatures are not feasible. In order to achieve integrity Message Authentication Codes (MACs) are used. MACs are used to prove that the message has not been tampered with. Since sender and receiver share the same key, it can not be proven who sent the message. A public key cryptography approach addresses many of these problems [11]. Public key cryptography allows for the application of digital signatures. Digital signatures provide integrity and repudiation. Only the party in possession of the private key can create a particular signature. When a message with a signature is received, the corresponding public key is used to verify the signature. Once the signature is verified, the receiver can be certain that the integrity of the message has not been breached. The receiver is also certain that only the sender in possession of the private key could have created that signature.

Homomorphic encryption is a cryptographic technique which allows calculations to be performed on aggregate data. Specifically, a homomorphic encryption scheme allows the following property to hold:

$$enc(a \oplus b) = enc(a) \oplus enc(b).$$

This means that in order to calculate the SUM of two values, we can apply some function to their encrypted counterparts and then decrypt the result of the SUM operation. Clearly, considering the cost of encryption and decryption, homomorphic encryption is useful in wireless sensor networks, because homomorphic encryption would allow for the calculation of

SUM and AVERAGE on encrypted data. The data would be encrypted at the sensor node, the SUM or AVERAGE would be calculated as the aggregate result follows a path to the base station, and the final result would be decrypted at the base station. Any eavesdropper would be unable to gather information from the transmissions. Any corrupted sensor could not know the aggregate result. An example of an homomorphic cryptography scheme is the elliptic curve ElGamal system [12]. The EC ElGamal system is additively homomorphic because the following property holds:

$$enc(a + b) = enc(a) + enc(b)$$

.

Elliptic curve cryptography (ECC) employs the points on an elliptic curve over a finite field K. The required algorithms for elliptic curve cryptography can easily be implemented, even on small devices such as sensors [13]. Elliptic curve cryptography uses the analog of the discrete logarithm problem (DLP), also known as the elliptic curve discrete logarithm problem (EC-DLP). The DLP over elliptic curves is believed to be computationally much more difficult than DLP over finite fields of the same size [10].

Homomorphic encryption does not provide integrity. Since we are using public key elliptic curve cryptography, we will use digital signatures to provide integrity. Digital signature schemes are not homomorphic. That is two signatures generated on two different messages can not be combined to verify the sum of the messages. We propose the use of an encryption scheme which will allow for homomorphic signature generation and verification.

III. Approach

We propose to use the EC-ElGamal system for homomorphic encryption in wireless sensor networks. In [11] the authors evaluated several public key homomorphic encryption schemes for use on sensors. The authors in [14] use an implementation of the EC-ElGamal algorithms for homomorphic encryption and storage in sensor networks. The EC-ElGamal system is thus clearly suitable for wireless sensor networks. Instead of applying EC-EG (EC-ElGamal) for persistent data storage, we propose to use it for homomorphic data encryption during transmission. The work in [14] provides for data confidentiality only. We propose the use of elliptic curve digital signatures to provide message integrity and integrity of the aggregate in addition to data confidentiality.

We will now provide an explanation of the example in Figure 1. Each node generates a reading. The reading is signed with the aggregate signature protocol using the node's private key; this is shown as $Sig(x)$. Each node homomorphically encrypts the reading with the base station's public key; this is shown as $Enc(x)$ in Figure 1. The node sends the secured reading, the signature and its public key to its parent. After receiving messages from all its children, the parent combines the messages into one. The parent sums the secured readings, the signatures and the public keys. If the parent also contributes a reading, that reading is treated like any other reading. These are shown as $SUM-ENC$, $SUM-SIG$ and $SUM-KEY$ in Figure 1. This process is repeated by each parent along the path to the base station.

The base station decrypts the received message. The sum of the readings was homomorphically encrypted with the base stations public key. This allows the base station to decrypt the readings. Only the base station which is in possession of the matching private key is able to decrypt the readings. This is shown as $Dec(Enc(x))$ in the figure. Each node signed its messages, and these signatures were combined along the way. The base station can now verify the sum of the signatures given the sum of the public keys. The aggregate signature protocol ensures that only readings from legitimate sensors are included in the aggregate.

IV. Algorithmic Details

We first describe the details for the algorithm executed at the sensors. Each sensor is pre-loaded with the appropriate elliptic curve parameters, the base stations' public key and a network wide random integer. The integer is used to generate a new k at set intervals. This ensures that the signatures are additive and secure against attacks. At the start of each round, each sensor choses a private key and computes the appropriate public key. Choosing a private key is straightforward and requires the sensor to pick an integer in the field of the elliptic curve. The public key is generated by multiplying the base point T with the private key; the result is another point on the curve. A new public/private key pair is necessary during each round of processing because it would only take two signatures for a malicious node to determine another node's private key. Let's say that node A signs a message m_a. The signature would be $k^{-1}(m_a + z_a * r(x))$. Any other node would know k^{-1} as well as $r(x)$ that would leave two unknowns m_a,the message and z_a, the private key. Clearly, if another message is signed with the same private key, that signature would not be secure. We add another level of security by signing the message and then encrypting it before sending it to the next level. If a sensor signs the same message with the same key, another sensor would be able to determine the private key. In most sensor applications, it's likely that a sensor would generate the same message several times. Each sensor computes R, which is the base point T multiplied by the current random integer k. Additionally, each sensor computes the multiplicative inverse of $k \bmod p$. Each sensor can now generate its unique signature s_i. After the signature has been generated, the sensor proceeds to homomorphically encrypt its reading x_i. The sensor first maps its reading onto the elliptic curve. After the mapping the reading is encrypted using the EC-IES algorithm [15].

If the sensor receives messages from other nodes for forwarding, it combines them according to the algorithm. The signature scheme is designed such that all signatures can be combined via simple arithmetic. This makes the amount of work required from a parent very small and thus well suited for wireless sensor networks.

We will now describe the base station's algorithm. The base station receives the sum of the signatures, the sum of the

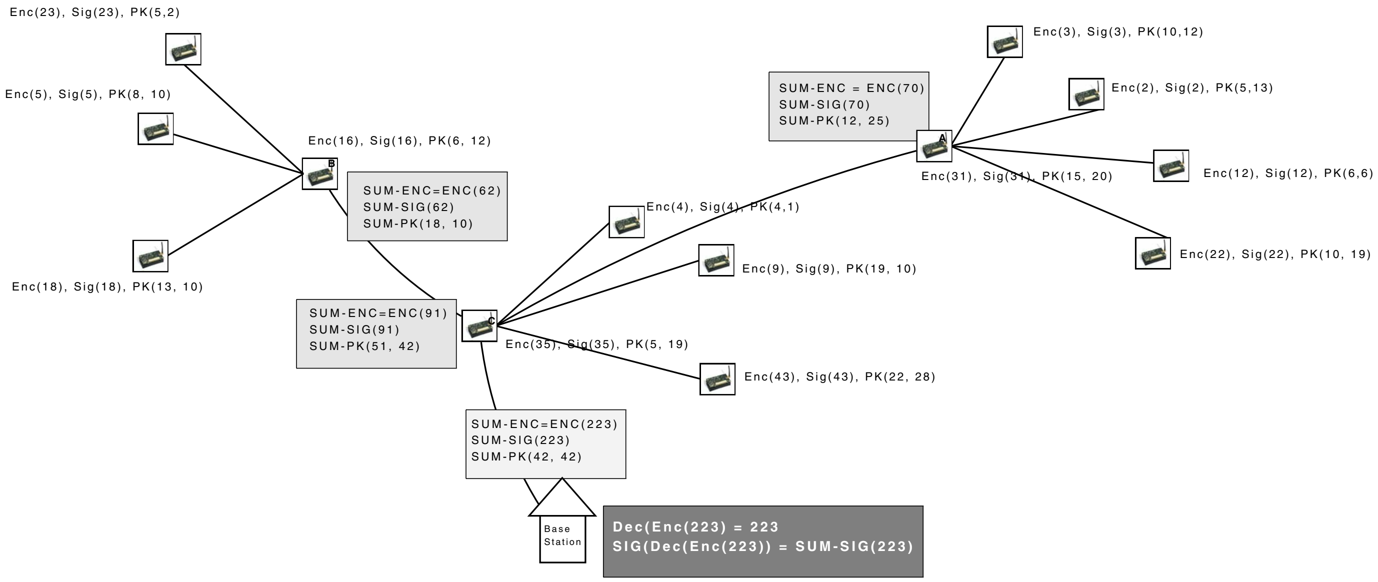
Enc(23), Sig(23), PK(5,2)
Enc(5), Sig(5), PK(8, 10)
Enc(16), Sig(16), PK(6, 12)
B
SUM-ENC=ENC(62)
SUM-SIG(62)
SUM-PK(18, 10)
Enc(18), Sig(18), PK(13, 10)
SUM-ENC=ENC(91)
SUM-SIG(91)
SUM-PK(51, 42)
C
Enc(4), Sig(4), PK(4,1)
Enc(9), Sig(9), PK(19, 10)
Enc(35), Sig(35), PK(5, 19)
Enc(43), Sig(43), PK(22, 28)
SUM-ENC=ENC(223)
SUM-SIG(223)
SUM-PK(42, 42)
Base Station
Dec(Enc(223)) = 223
SIG(Dec(Enc(223))) = SUM-SIG(223)
Enc(3), Sig(3), PK(10,12)
SUM-ENC = ENC(70)
SUM-SIG(70)
SUM-PK(12, 25)
A
Enc(2), Sig(2), PK(5,13)
Enc(31), Sig(31), PK(15, 20)
Enc(12), Sig(12), PK(6,6)
Enc(22), Sig(22), PK(10, 19)

Fig. 1. Homomorphic Encryption Example

```
Require: Elliptic Curve Parameters D = (q, FR, a, b, T, p , h), sensor reading $m_i$, private key $z_i$, base station public key $Q$, a network wide random integer $k$
Each sensor computes $z_i * T = (x, y)$, its public key.
Each sensor computes $R = (r(x), r(y)) = k * T$.
Each sensor computes $k^{-1} \bmod p$.
Each sensor computes $s_i = k^{-1}(m_i + z_i * r(x)) \bmod p$.
Each sensor's signature for the message $m_i$ is $s_i$.
Each sensor maps its reading $m_i$ onto the elliptic curve $D$.
Each sensor generates ciphertext $\overline{m_i} = enc(m_i)$
if Sensor is a parent then
    The sensor combines the signatures into $s = \sum s_i)$
    The sensor combines all ciphertexts into one ciphertext $\sum m_i$
end if
```

Fig. 2. HAgg: Sensor Algorithm

```
Require: Elliptic Curve Parameters D = (q, FR, a, b, T, p , h), sum of encrypted sensor readings $m = \sum \overline{m_i}$, sum of the signatures $s = \sum s_i$, base station private key $q_i$, sum of public keys $Z$, a network wide random integer $k$
Decrypt ciphertext $\sum \overline{m_i} = \sum m_i$
Map reading $m$ from the elliptic curve $D$ into plaintext.
Compute $R = (r(x), r(y)) = k * T$.
Compute $w = s^{-1} \bmod p$.
Compute $u_1 = mw \bmod p$.
Compute $u_2 = r(x)w \bmod p$.
Compute $X = u_1 T + u_2 Z$.
Compute $v = X(x) \bmod p$.
if $v == r$ then
    The signature verified
end if
```

Fig. 3. HAgg: Base Station Algorithm

appropriate public keys and the homomorphically encrypted aggregate result. The base station can now verify that the same sensors that contributed to the aggregate also signed their inputs and that signature is included in the combined signature. The base station first decrypts the aggregate result using its private key. Additionally, the base station needs to reverse the mapping from the point on the elliptic curve to the aggregate result. To verify the signature, the base station calculates a point on the curve using the received signature, the decrypted aggregate result and the integer k. If the x-coordinate of the point calculated is the same as r(x), the signature is verified. The base station is now assured that no data not generated by a legitimate sensor was included in the aggregate.

The algorithm described securely calculates the SUM of the readings in a wireless sensor network. In order to securely calculate the AVERAGE in a wireless sensor network, the base station needs a count of the number of points included in the SUM. With the knowledge of how many sensors contributed to the aggregate, the AVERAGE can be calculated.

V. Security Analysis

Our signature algorithm is an extension of the ECDSA. ECDSA is assumed to be secure. ECDSA has been shown to be secure under the assumption that the underlying group is generic and that a collision resistant hash function has been used.

Theorem 5.1: The signature produced by summing the individual signatures will only verify if the contributing individual signatures were produced by a valid node and the appropriate public key was included in the sum of public keys.
We will now prove that the combined signature will only verify if the individual signatures contributed by the nodes are

signatures generated by valid nodes and are valid signatures. The value k is a randomized, synchronized integer used by all nodes in the network. We do not need to send $r(x)$ with each signature, as the base station is able to compute $r(x)$. Therefore the unique part of each node's signature is s_i.

Lemma 5.2: The sum of the public keys equals the public key generated from the sum of the private keys.

Proof: We have the sum of the public keys, $Z = Z_A + Z_B + \ldots$. Let's say that the sum of the private keys, $z = z_A + z_B + \ldots$. Since $Z_A = z_A * T, Z_B = z_B * T, \ldots$ we know that

$$\begin{aligned} Z &= z_A * T + z_B * T + \ldots \\ &\equiv (z_A + z_B + \ldots) * T. \end{aligned}$$

Therefore $Z = zT$. In other words, the sum of the public keys equals the public key generated from the sum of the private keys. ■

Lemma 5.3: The sum of signatures produced creates a valid signature equivalent to a signature produced using the sum of the private keys on the sum of the messages.

Proof: We know that $m = m_A + m_B + \ldots$ and $s = s_A + s_B + \ldots$, as per Algorithm 2. Then

$$\begin{aligned} s &= k^{-1}(m_A + z_A r(x)) + k^{-1}(m_B + z_B r(x)) + \ldots \\ &\equiv k^{-1}((m_A + z_A r(x)) + (m_B + z_B r(x)) + \ldots) \\ &\equiv k^{-1}((m_A + m_B + \ldots) + (z_A + z_B + \ldots) r(x)) \\ &\equiv k^{-1}(m + (z_A + z_B + \ldots) r(x)). \end{aligned}$$

Using Lemma 5.2, it follows that $s = k^{-1}(m + zr(x))$. ■

We can now prove Theorem 5.1, that the signature verification of $v == r$ will only work if each signer contributed the signature and the matching public key to the aggregate.

Proof: From Lemma 5.3, we know that $s = k^{-1}(m + zr(x))$. Rearranging we get

$$\begin{aligned} k &\equiv s^{-1}(m + zr(x)) \\ &\equiv s^{-1}m + s^{-1}r(x)z \\ &\equiv wm + wr(x)z \\ &\equiv u_1 + u_2 z \mod p. \end{aligned}$$

We also know that $X = u_1 T + u_2 Z$ and that $Z = zT$; it follows that

$$\begin{aligned} X &= u_1 T + u_2 Z \\ &\equiv u_1 T + u_2 (zT) \\ &\equiv (u_1 + u_2 z) T \end{aligned}$$

Thus $(u_1 + u_2 z)T \equiv kT$ and $r == v$ as required for the signature verification. ■

VI. Related Work

Secure data aggregation schemes have been of interest to researchers. The earliest approaches focused on confidentiality of the data against a single aggregators. Algorithms which prevented or detected multiple aggregators colluding to deceive the base station were also introduced.

A new protocol for provably secure data aggregation in wireless sensor networks was proposed in [16]. The algorithm guarantees the detection of aggregate modification by the aggregator, except for those cases where the aggregator injects data into the aggregate. The algorithm supports any arbitrary tree structure and is resilient to any number of malicious nodes. The algorithm focuses on the use of the SUM operator, but would also work with MEDIAN, COUNT and AVERAGE. This algorithm forces a commitment from the adversary at intermediate nodes. Each sensor also verifies that its data was properly added to the aggregate. Our algorithm works with any single-path routing protocol, and will securely calculate the SUM and AVERAGE. Compared to the algorithm in [16], our proposed algorithm provides for a greater reduction in energy savings due to a reduced number of messages send.

The algorithms introduced in [17] achieve concealed data aggregation. Concealment means that the data and the aggregates are not readable for anyone who is not in possession of the proper key. The algorithm uses privacy homomorphism to achieve data hiding while still allowing for data aggregation. The algorithm provides for data confidentiality only; the authors refer to other papers regarding solutions that provide data integrity and authenticity. The algorithm uses symmetric keys, while our work uses a private/public key approach.

The protocol introduced in [14] is not meant to provide data authentication and message integrity during transmission. Rather it is meant to provide persistent, secure data storage in sensor networks. It provides for secure data replication to ensure data availability in case of node failure. It also introduces secure data aggregation due to restricted storage space. Our work uses a similar public/private key homomorphic encryption protocol to ensure secure data aggregation during transmission.

The work in [11] provides a survey of possible homomorphic public key encryption schemes suitable for wireless sensor networks. The authors provide a list of desirable properties of a homomorphic public key encryption scheme for wireless sensor networks and evaluate the various candidates based on that list. The authors conclude that EC-OU (Elliptic Curve Okamoto-Uchiyama) and EC-EG (Elliptic Curve ElGamal) are the two algorithms most suitable for use as homomorphic public key encryption schemes. For our implementation we have chosen EC-IES, a variant of EC-EG.

For the work in [18], the authors propose a new additively homomorphic stream cipher suitable for wireless sensor networks. The proposed algorithms use a symmetric key stream. In addition to the issues related to symmetric key use, the base station needs to know exactly which sensor did or did not contribute data to the aggregate. Without this information the base station will be unable to decrypt the result. Only data confidentiality is provided with this algorithm. Our algorithm provides both data confidentiality and message integrity without the limitations of symmetric key cryptography.

The authors in [19] provide a symmetric key algorithms which provides data confidentiality and integrity. The protocol provides security against data injection by attackers during aggregation as well as forwarding. The algorithm works by ensuring that a compromised node is will only reveal the readings of a small subset of the nodes in the network. Additionally, data injection attempts can be detected and the algorithm is highly resilient to node failure. Our algorithm on the other hand provides security using public/private key cryptography as well as confidentiality and integrity using digital signatures.

The proposed algorithm in [20] provides data confidentiality in WSNs. The authors show that in WSNs two types of data confidentiality are necessary: generic confidentiality and end-to-end confidentiality. Generic confidentiality means that any node not participating in the aggregation mechanism is not able to access the data. End-to-end confidentiality means that any node participating in the aggregation mechanism is unable to access the already aggregated data. The proposed protocol provides security for both types of confidentiality using symmetric key cryptography and multiple homomorphic encryption. Our algorithm on the other hand uses public key cryptography and provides both types of confidentiality as well as data integrity.

In [21] the authors propose an algorithm which provides for secure calculation of max/min and average on encrypted data. Most homomorphic encryption schemes are not secure against ciphertext only attacks when comparison functions such as max/min and average are calculated. The algorithm improves OPES which is computationally expensive. The protocol provides for data confidentiality during aggregate computation of max/min and average. On the other hand, our algorithm allows for the calculation of sum and provides for data integrity as well.

As far as we can determine, this is the first work which used additively digital signatures in wireless sensor networks.

VII. Conclusion and Future Work

In this paper a novel algorithm is presented to address the problem of secure data aggregation in wireless sensor networks. We apply a homomorphic encryption algorithm to the messages to achieve data confidentiality while allowing in-network aggregation. An additively digital signature algorithm based on ECDSA is used to achieve integrity of the aggregate. We showed that the signature algorithm is as secure as ECDSA. Future work will include implementing this algorithm in TinyOS/TOSSIM [22].

References

[1] L. Clare, G. Pottie, and J. R. Agre, "Self-organizing distributed sensor networks," *SPIE-The International Society for Optical Engineering*, pp. 229–237, 1999.

[2] R. W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM Press, 1999, pp. 174–185.

[3] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8*. Washington, DC, USA: IEEE Computer Society, 2000, p. 8020.

[4] J. Albath and S. Madria, "Practical algorithm for data security (pads) in wireless sensor networks," in *MobiDE '07: Proceedings of the 6th ACM international workshop on Data engineering for wireless and mobile access*. New York, NY, USA: ACM Press, 2007, pp. 9–16.

[5] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: A link layer security architecture for wireless sensor networks," in *Second ACM Conference on Embedded Networked Sensor Systems (SensSys 2004)*, November 2004.

[6] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," in *ICDCSW '02: Proceedings of the 22nd International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 575–578.

[7] K. Kalpakis, K. Dasgupta, and P. Namjoshi, "Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks," *Comput. Networks*, vol. 42, no. 6, pp. 697–716, 2003.

[8] C. Y. Chong, S. P. Kumar, and B. A. Hamilton, "Sensor networks: evolution, opportunities, and challenges," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247–1256, 2003.

[9] M. Anand, Z. Ives, and I. Lee, "Quantifying eavesdropping vulnerability in sensor networks," in *DMSN '05: Proceedings of the 2nd international workshop on Data management for sensor networks*. New York, NY, USA: ACM Press, 2005, pp. 3–9.

[10] A. J. Menezes, *Elliptic Curve Public Key Cryptosystems*. Norwell, MA, USA: Kluwer Academic Publishers, 1994.

[11] E. Mykletun, J. Girao, and D. Westhoff, "Public Key Based Cryptoschemes for Data Concealment in Wireless Sensor Networks," *IEEE International Conference on Communications ICC*, 2006.

[12] J. M. Adler, W. Dai, R. L. Green, and C. A. Neff, "Computational Details of the VoteHere Homomorphic Election System," 2000.

[13] D. J. Malan, M. Welsh, and M. D. Smith, "A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography," *IEEE SECON 2004. First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, pp. 71–80, 2004.

[14] J. Girao, D. Westhoff, E. Mykletun, and T. Araki, "Tinypeds: Tiny persistent encrypted data storage in asynchronous wireless sensor networks," *Ad Hoc Networks*, vol. 5, no. 7, pp. 1073–1089, September 2007.

[15] A. Liu and P. Ning, "Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks," in *7th International Conference on Information Processing in Sensor Networks (IPSN 2008)*, April 2008, pp. 245–256.

[16] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," in *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security*. New York, NY, USA: ACM Press, 2006, pp. 278–287.

[17] D. Westhoff, J. Girao, and M. Acharya, "Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution, and routing adaptation," *IEEE Transactions on Mobile Computing*, vol. 5, no. 10, pp. 1417–1431, 2006.

[18] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks," in *The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, July 2005, pp. 109–117.

[19] R. Di Pietro, P. Michiardi, and R. Molva, "Confidentiality and integrity for data aggregation in wsn using peer monitoring," Institut Eurecom, Tech. Rep., 2007.

[20] M. Oenen and R. Molva, "Secure data aggregation with multiple encryption," in *Proceedings of Fourth European Conference on Wireless Sensor Networks (EWSN 2007)*, 2007.

[21] L. Ertaul and V. Kedlaya, "Computing aggregation function minimum/maximum using homomorphic encryption schemes in wireless sensor networks (wsns)," in *ICWN*, H. R. Arabnia, V. A. Clincy, and L. T. Yang, Eds. CSREA Press, 2007, pp. 186–192.

[22] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim: accurate and scalable simulation of entire tinyos applications," in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2003, pp. 126–137.