ISSN No. 0976-5697

Volume 2, No. 4, July-August 2011

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Well-Organized and Reliably Secure Dynamic Groups of Data Aggregation in Wireless Sensor Networks

C. Balarengadurai
Associate professor
Department of Computer Science & Engineering
Aurora's Engineering College, Hyderabad, India
balamtech@yahoo.co.in

Abstract-Wireless sensor networks (WSNs) is an infrastructure comprised of measuring, computing and communication elements that gives an administrator the ability to instrument, observe and react to events. We present a new encryption mode of operation that allows nodes of a network to exchange messages securely without sharing a common key or using public key cryptography. We show that our proposal can be used in wireless sensor networks to send encrypted packets to very dynamic sets of nodes without having to establish and maintain group keys. The Security of the scheme is based on the indistinguishability property of Pseudo Random Function (PRF), a standard cryptographic primitive. We show that aggregation based on this scheme can be used to efficiently compute statistical values, such as mean, variance, and stand deviation of sensed data, while achieving significant bandwidth savings. As a result, a node can send encrypted data to all the nodes within a given geographical area, without having to identify the destination nodes in advance.

Keywords: Wireless Sensor Network, Security, Vulnerabilities, aggregation, Security Mechanism, Pseudo Random Function

I. INTRODUCTION

Wireless sensor networks are often deployed in public and unattended environments. Data transmission is very energy-consuming, in order to maximize sensor lifetime, it is essential to minimize the sheer number of bits sent by each sensor device. One natural approach involves aggregating sensor data as it propagates along the path from the sensors to the so-called sink a special node that collects sensed data. For an intermediate sensor (one that receives and forwards data), this would entail:1) sharing a key with each neighboring sensor, (2) decrypting encrypted messages received from each child, (3) aggregating all received values, and (4) encrypting the result for transmission to its parent. Though viable, this approach is fairly expensive since each value has to be decrypted before aggregation. It also complicates key management since each node must share a key with each of its neighbors. Furthermore, hop-by-hop encryption assumes that all sensors are trusted with the authenticity and privacy of other sensors' data. One common feature too many sensor networks is the need to communicate secretly with arbitrary sub-groups of sensors.

To minimize trust assumptions, we assume that each of the *n* sensors shares a distinct long-term encryption key with the sink. This key is originally derived, using a Pseudo-Random Function (PRF), from the master secret known only to the sink. The group members might actually be defined on the basis of some criteria, such as location, proximity to an object, temperature range or any other environmental property. Existing group keying solutions are not applicable to small and dynamic groups. Most of them assume that the group is rather stable, revocation is a rare event, and that the size of the group is quite close to the entire nodes population. In this paper, we focus on efficient, bandwidth-conserving privacy in WSNs. More specifically, we blend inexpensive encryption techniques with simple aggregation methods to achieve very efficient aggregation of encrypted data. We propose a new scheme that allows two nodes of a network to exchange messages securely (i.e. encrypted and authenticated) without sharing a common key or using public key cryptography. Our scheme also solves the short-lived group encryption problem described previously. It allows a node to send encrypted packets to very dynamic sets of nodes without having to establish or update group keys. As a result, a node can, for example, send encrypted data to all the nodes within a given geographical area, without having to identify destination nodes in advance. Finally we show that our proposal can be used to securely aggregate encrypted data. The proposed scheme is based on stream ciphers and does not rely on any public key cryptography algorithms.

II. HOMOMORPHIC ENCRYPTION SCHEME

Assume there is a sink and *n* nodes in the system. In the following description, *f* is a PRF for key stream generation and *h* is a length-matching hash function. The details of the proposed aggregate encryption scheme are as follows.

In the proposed scheme, a PRF is used: (1) by the sink to generate the encryption keys ek_i 's from the root key *K*, and (2) by sensor node *i* to generate the
key stream $f_{ek_i}(r)$ from ek_i and the nonce *r*. It is not necessary to use two different PRF schemes for the instantiations. The same PRF scheme can be used for these purposes.

Assume the modulus is M.

Key Generation:

(1) Randomly pick $K \in \{0,1\}^{\lambda}$ and set it as the decryption key for the sink.
(2) For each $i \in [1,n]$, set encryption key for node i as $ek_i = f_K(i)$.

Encryption:

(1) Given encryption key ek_i, plaintext data m_i and nonce r, output $c_i = Enc_{ek_i}(m_i) = m_i + h(f_{ek_i}(r)) \bmod M$.
(2) Set header $hdr_i = \{i\}$.
(3) Output (hdr_i, c_i) as ciphertext.

Decryption:

(1) Given ciphertext (hdr, c) and nonce r used in encryption, generate $ek_i = f_K(i), \forall i \in hdr$.
(2) Compute $x = Dec_K(c) = (c - \sum_{i \in hdr} h(f_{ek_i}(r))) \bmod M$ (where $K = \sum_{i \in hdr} h(f_{ek_i}(r))$), and output plaintext aggregate x.

Addition of Ciphertexts:

(1) Given two CDA ciphertexts (hdr_i, c_i) and (hdr_j, c_j), compute $c_l = (c_i + c_j) \bmod M$
(2) Set $hdr_l = hdr_i \cup hdr_j$.
(3) Output (hdr_l, c_l).

A. *Protocol Description*

It is assumed that a node N0, wants to privately send a message m to node N_D. All the nodes along the path from N0 to N_D are denoted Ni, where N0 is the source, N1 the first node on the path,…, and N_D the final destination, Furthermore, each node on the path only knows the next hop to the destination node N_D. The protocol executed by each node Ni, on the path from N0 to N_D is described as follows:

```
if $(i == 0)$ then
  compute $C_0 = Enc(k_{s,1}, Enc(kn_{s,1}, m))$
  forward $C_0$ to $N_1$.
end if
if $1 \leq i \leq (D-1)$ then
  compute $t_i = Dec(kn_{i-2,i-1}, Dec(k_{i-1,i}, c_{i-1}))$
  compute $C_i = Enc(k_{i,i+1}, Enc(kn_{i,i+1}, t_i))$
  forward $C_i$ to $N_{i+1}$
end if
if $(i == D-1)$ then
  compute $t_i = Dec(kn_{i-2,i-1}, Dec(k_{i-1,i}, c_{i-1})$
  compute $C_i = Enc(k_{i,i+1}, t_i)$
  forward $C_i$ to $N_{i+1}$
end if
if $(i == D)$ then
  compute $m' = Dec(kn_{D-2,D-1}, Dec(k_{D-1,D}, c_{D-1})$
end if
```

III. DATA AGGREGATION IN WSN

Typically, there are three types of nodes in WSN: normal sensor nodes, aggregators, and a querier. The aggregators collect data from a subset of the network aggregate the data using a suitable aggregation function and then transmit the aggregated result to an upper aggregator or to the querier who generates the query. The querier is entrusted with the task of processing the received sensor data and derives meaningful information reflecting the events in the target field. It can be the base station or sometimes an external user who has permission to interact with the network depending of the network architecture.

In Figure 1 contains 16 sensor nodes and uses SUM function to minimize the energy consumption by reducing the number of bits reported to the base station, Node 7, 10-6,8,9 are aggregators that perform sensing and aggregating at the same time. In this example 16 packets is transmitted to the base station. However the number of traveling packets would increase to 50 packets if no data aggregation exists. This number of packets has been computed for one query.

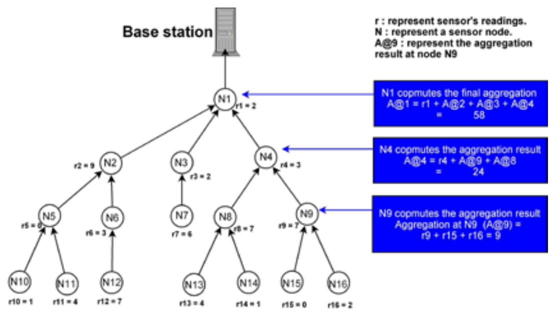
Base station
r : represent sensor's readings.
N : represent a sensor node.
A@9 : represent the aggregation result at node N9
N1 copmutes the final aggregation A@1 = r1 + A@2 + A@3 + A@4 = 58
N4 copmutes the aggregation result A@4 = r4 + A@9 + A@8 = 24
N9 copmutes the aggregation result Aggregation at N9 (A@9) = r9 + r15 + r16 = 9
N1
r1 = 2
N2
r2 = 9
N3
r3 = 2
N4
r4 = 3
N5
r5 = 0
N6
r6 = 3
N7
r7 = 6
N8
r8 = 7
N9
r9 = 7
N10
r10 = 1
N11
r11 = 4
N12
r12 = 7
N13
r13 = 4
N14
r14 = 1
N15
r15 = 0
N16
r16 = 2

Figure 1: An aggregation scenario using sum function.

Additive aggregation can be also used to compute the variance, standard deviation and any other moments on the measured data. For example, in case of variance, each aggregating node not only computes the sum,

$$S = \sum_{i=1}^{k} x_i,$$

of the individual values sent by its *k* children, but also the sum of their squares:

$$V = \sum_{i=1}^{k} x_i^2.$$

Eventually, the sink obtains two values: the sum of the actual samples, which it can use to compute the mean and the sum of the squares, which it can use to compute the variance:

$$Var = E(x^2) - E(x)^2; \text{ where}$$

$$E(x^2) = \left(\sum_{i=1}^{n} x_i^2\right)/n \text{ and } E(x) = \left(\sum_{i=1}^{n} x_i\right)/n.$$

IV. AGGREGATION OF ENCRYTED DATA

As previously noted, efficient aggregation in WSNs becomes challenging when end-to-end privacy of data is required. One solution is to disregard aggregation altogether in favor of privacy, for each sensor to encrypt and forward upstream its individual measurement. The sink, upon receiving as many packets as there are responding sensors, proceeds to decrypt all ciphertext and sum them in order to compute the desired statistical measurements. We denote this approach as *No-Agg*. A variant of this scheme consists of having the intermediate nodes concatenate the packets they receive from their children into a smaller number of packets in order to avoid the overhead due to the headers.

We denote this variant as *CON*. The CON scheme performs better than the *No-Agg* scheme but it remains quite costly. A second approach, that does not achieve end-to-end privacy but does aggregate data, is the *hop-by-hop (HBH)* encryption method, which is also used for comparison between aggregation methods in Girao et al. [2004]. In HBH all nodes create pair-wise keys with their parents and children at bootstrapping phase. When answering a query, a node decrypts any packets received from downstream, aggregates the plaintext data with its own, encrypts the aggregated result and forwards to its parent. This approach is obviously more bandwidth efficient than No-Agg since no packet is sent twice. In particular, nodes close to the sink become attractive attack targets since the aggregated values they handle represent large portions of the overall data in the WSN. We propose an end-to-end privacy preserving aggregation approach denoted as *AGG (aggregation).* Since this scheme is additively homomorphic, values can be added (aggregated) as they are forwarded towards the sink. The sink can easily retrieve from the aggregates it receives the sum of the samples, and derives statistical information. AGG retains the attractive properties of both the No-Agg (end-to-end privacy) and HBH (energy efficient) schemes.

V. OVERHEAD ANALYSIS

We now compare the bandwidth consumption of the proposed AGG protocol with the No-Agg (forwarding individual data packets), CON (concatenating and forwarding data packet), and HBH(hop-by-hop encryption and aggregation) The overall bandwidth in the WSN and the number of bits sent by individual nodes are considered for different WSN tree-like topologies. We next describe the network model used in the measurements. The comparison is for two cases: (1) average value only, and (2) both average and variance values.

A. Network Model

We assume a multilevel network tree with a multitude of sensors and one sink. To simplify our discussion, we assume a balanced k-ary tree, as shown in Figure 3. Let t denote the range of possible measurement values (e.g., if a sensor measures temperatures between 0 and 120 Fahrenheit, then t = 121). We also assume, for simplicity, that only the leaves of the tree are sensors and that the intermediate nodes are just forwarding nodes. We analyze bandwidth in this WSN model from two perspectives: (1) number of bits sent per node at different levels in a 3-ary tree, and (2) total number of bits transmitted in theWSNfor 3-ary trees of various heights. These measurements are performed for the four models: No-Agg, CON, HBH, and AGG. Next, we show how to compute the number of bits (header and payload) sent per node. The packet header is56 and maximum supported data payload is 232 bits, respectively. If a data payload is larger than 232 bits, it is sent over several packets. For example, the transmission of a data payload of 300 bits results in the transmission of 2 packets: one of size 288 bits (232 + 56) and another of size 124 bits (68+ 56).The total cost is then equal to 312 bits.

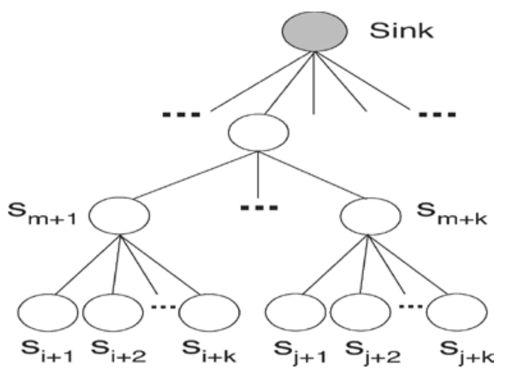

Figure 2. Multi-level WSN model with nodes of degree *k*.

For No-Agg, a node only needs $log(t)$ bits to encode its data. Also, all internal nodes forward packets sent to them by their children, and the number of packets received grows exponentially in k as we move higher in the tree closer to the sink. The CON scheme reduces the required bandwidth by reducing the overhead due the headers, but still has an exponential growth. Note that with the CON scheme, each intermediate node needs to append to the concatenate packet the IDs of its children that did not reply to the query. These IDs must be propagated to the sink along with the aggregate. In HBH, the number of sent bits depends on the node's level in the WSN tree. Leaf nodes only send $log(t)$ bits (as in No-Agg), while higher-up nodes receive aggregated data and therefore send more bits. Additionally, when the variance is also requested, the aggregating nodes need to keep track of this value separately, and use approximately $log(n't)$ bits to encode it (where n' is the number of node-values aggregated so far). Finally, in AGG, the number of bits sent by a node depends on the size of the modulus M. Its size corresponds to the maximum possible aggregate value, which is $M = n$t, that is, all sensors report the largest possible reading. Therefore, in encoding the average, each node uses $log\ (M) = log\ (t) + log\ (n)$ bits. If variance is desired, a node sends an additional ciphertext corresponding to x^{2}. This requires extra $log\ (n \ast t^{2}{}_{2}) = 2 \ast log\ (t) + log\ (n)$ bits. Also, each aggregator needs to append to the aggregate, the IDs of its children that did not reply to the query. These IDs must be propagated to the sink along with the aggregate.

B. Numerical Results

We now compare the performance of the four schemes. *Forwarding Cost per node (fairness).* Tables I, II, and III show the number of bits sent per node at each level in a 3-degree tree of height (The sink is at level 0.) 7 when t = 128 for the different schemes.

Table I. Number of bits sent per node for each level with the No-Agg and CON Schemes

Levels	Nodes	A (0%)	A (10%)	A (30%)	CON (0%)	CON (10%)	CON (30%)
1	3	45927	41334	32149	6335	6811	7708
2	9	15309	13778	10716	21149	2270	2564
3	27	5103	4593	3572	735	775	2564
4	81	1701	1531	1191	245	258	285
5	243	567	510	397	119	123	132

6	729	189	170	132	77	78	81
7	2187	63	57	44	63	63	64

Table II. Number of bits sent per node for each level with the HBH Schemes

Levels	Nodes	HBH A (0%)	HBH A (10%)	HBH A (30%)	HBH AV (0%)	HBH AV (10%)	HBH AV (30%)
1	3	73	72	72	96	96	96
2	9	71	71	70	93	92	92
3	27	69	69	69	90	89	89
4	81	68	68	67	96	86	85
5	243	66	66	66	83	83	82
6	729	64	64	64	90	80	79
7	2187	63	62	61	63	62	61

Table III. Number of bits sent per node for each level with the AGG Schemes

Levels	Nodes	Agg A (0%)	Agg A (10%)	Agg A (30%)	Agg AV (0%)	Agg AV (10%)	Agg AV (30%)
1	3	75	1117	3315	100	1142	3340
2	9	75	422	1117	100	447	1142
3	27	75	172	442	100	197	448
4	81	75	107	172	100	132	197
5	243	75	85	108	100	111	132
6	729	75	78	85	100	103	110
7	2187	75	67	52	100	91	71

We considered three scenarios: (1) all nodes reply when only 0% average and variance are compute, (2) 90% of the nodes reply when only 10% average and variance are compute and (3) 70% of the nodes reply when only 30% average and variance are compute. We consider both approaches to be quite impractical. The *CON* scheme reduces the required bandwidth by a factor of 4, but it is still unfair. The nodes that are close to the sink need to forward many more its that the other nodes. Note that with the *No-Agg* scheme, the required bandwidth decreases as the number of no responding sensors increases. This is the result of the network having fewer messages to forward. In contrast, with the *CON* scheme, the bandwidth increases because the IDs of the no responding nodes must be appended to the concatenated messages. Table II shows a steady increase in bits-per-node for HBH, for both the average only (HBH-A), and average-plus-variance (HBH-AV) cases. Note a relatively dramatic increase in bits transmitted between nodes at level 7 and 6 for HBH-AV.

With *AGG*, when all the sensors are replying, a constant number of bits is sent by each node at every level in the tree. However, this number is larger than the maximum in any HBH approach, due to the size of the modulus *M*. As previously discussed, the number of bits sent by leaves is larger with the aggregation methods (AGG-A: $56 + log(t) + log(n) = 75$ bits, AGG-AV: $56 + 3 * log(t) + 2 * log(n) = 100$ bits) than when no aggregation is used ($56 + log(t) = 63$ bits). However, aggregation distributes the load evenly over all nodes, regardless of their distance to the sink. We believe this to be a major advantage in WSNs.

C. Bandwidth Gain

Tables IVand V shows the bandwidth transmission gains of HBH, CON, and AGG over No-Agg, assuming 3-degree WSNs of various heights. We consider gains for two cases: (1) only the average, and (2) both the average and variance These gains are obtained from the respective total bandwidth costs: *CHBH*, *CAGG*, *CNo−Ag g* and *CCON*, by adding, for each scheme, the total number of bits forwarded by each node. The bandwidth gain of HBH, AGG, and CON are defined as *CNo−Ag g* /*CHBH*, *CNo−Ag g* /*CAGG*, and *CNo−Ag g* /*CCON*, respectively.

Table IV. WSN bandwidth performance gain of the AGG and HBH schemes when aggregating the average for a 3-tree and t=128

Levels	Nodes	Agg (0%)	Agg (30%)	HBH (0%)	HBH (30%)	CON (0%)	CON (30%)
3	40	1.75	1.48	2.05	1.48	1.85	1.31
4	121	22.27	1.86	2.7	1.92	2.27	1.59
5	364	2.8	2.18	3.31	2.37	3.62	1.8
7	3280	3.92	2.71	4.61	3.3	3.2	2.08

Table V. WSN bandwidth performance gain of the AGG and HBH schemes when aggregating the average and variance for a 3-tree and t=128

Levels	Nodes	Agg (0%)	Agg (30%)	HBH (0%)	HBH (30%)	CON (0%)	CON (30%)
3	40	1.30	1.11	1.91	1.37	1.85	1.31
4	121	1.69	1.4	2.47	1.77	2.27	1.59
5	364	2.1	1.67	3.05	2.19	3.62	1.8
7	3280	2.92	2.14	4.25	3.1	3.2	2.08

VI. COMPUTATION COSTS

We now discuss computation costs for the proposed scheme and issues related to implementation. Let t_{add} and $t_{mul\ ti}$ denote the respective costs of addition and multiplication operation mod *M*. Let $t_{pr\ f}$ and t_h denote the costs of a PRF(indistinguishability property of a pseudorandom function) and a length-matching hash function, respectively. Let t_{ce} and t_{cd} denote the costs of one encryption and one decryption with a cipher used in the HBH scheme. Overall computation costs for proposed protocols are shown in Table VI, assuming *L* reporting nodes (|*hdr*| = *L*). For the aggregation operation, our calculations assume that each aggregation involves only two inputs. AGG places all decryption tasks at the sink, while HBH distributes the decryption cost over all non-leaf nodes in the network. Thus in HBH, a sensor may need to perform more computation than the sink. Since the sink is usually a more powerful device, AGG is clearly preferable.

Table VI Computation Cost Comparison

	HBH Encryption (HBH)	Aggregate Encryption (AGG)
Encryption	t_{ec}	$t_{prf}+t_k+t_{add}$
Decryption	t_{cd}	$2L.t_{prf}+L.t_h+L.t_{add}$
Aggregation (per 2inputs)	$2._{tcd}+t_{ec}+t_{add}$	t_{add}

In AGG (Aggregation), to encrypt its value, a node performs one PRF invocation, one length matching hash, and one mod *M* addition. It also performs one extra addition for aggregation. We thus consider the cost of evaluating *h* to be negligible in the calculation of overall computation cost for encryption. As a result, the cost of encryption is dominated by a single PRF invocation.

VII. CONCLUSION

This article proposes a new homomorphic encryption scheme that allows intermediate sensors to aggregate encrypted data of their children without having to decrypt. We show that, if key streams are derived from a good PRF, our scheme can achieve semantic security against any node collusion of size less than the total number of nodes. However it provides a much stronger level of privacy comparable to that provided by end-to-end encryption with no aggregation. Finally, we augmented our scheme to provide end-to-end aggregate authentication. Without knowledge of a group key, an external attacker cannot tamper with any aggregate, without being detected. In conclusion, we offer efficient and provably secure techniques for end-to-end privacy and authenticity, with reasonably good security assurances, in WSNs.

VIII. REFERENCES

[1] M. Goodrich, "Leap-frog packet linking and diverse key distributions for improved integrity in network broadcasts," in IEEESecurity and Privacy, May 2005.

[2] Sandro Rafaeli and David Hutchison, "A survey of key management for secure group communication," ACM Computing Surveys, vol. 35, pp. 309–329, 2003.

[3] David Wagner, "Attacks on the hash-then-encrypt (for streamcipher)," http://www.cs.berkeley.edu/ daw/my-posts/mdc-broken2.

[4] Girao J., Westhoff, D., and Schneider, M.2004." CDA: Concealed data aggregation in wireless sensor networks". In Proceedings of the ACM Conference on Web Information Systems (WiSe).

[5] Claude Castelluccia, Einar Mykletun, and Gene Tsudik, "Efficientaggregation of encryption data in wireless sensor networks," in IEEE Mobiquitous, 2005.

[6] F. L. LEWIS, "Wireless Sensor Networks", Technologies, Protocols, and Applications ed. D.J. Cook and S.K. Das, John Wiley, New York, 2004.

[7] Claude Castelluccia, "Authenticated interleaved encryption,"Cryptology ePrint Archive, Report 2006/416, 2006, http://eprint.iacr.org

[8] K. Sun, P. Ning, C. Wang, A. Liu, and Y. Zhou. TinySeRSync: Secure and resilient time synchronization in wireless sensor networks. In ACM CCS, November 2006.

[9] S. Misra, G. Xue, and S. Bhardwaj, "Secure and Robust Localization in a Wireless Ad Hoc Environment," IEEE Trans.Vehicular Technology, vol. 58, no. 3, pp. 1480-1489, Mar. 2008.

[10] Taban, G.; Gligor, V.D. Privacy-preserving integrity-assured data aggregation in sensor networks. In Proceeding of International Symposium on Secure Computing, SecureCom, Vancouver, Canada, August 29–31, 2009; pp. 168–175.