

Group-Based Secure Communication for Large-Scale Wireless Sensor Networks

Kashif Kifayat, Madjid Merabti, Qi Shi, David Llewellyn-Jones

School of Computing & Mathematical Sciences, Liverpool John Moores University,
Byrom Street, L3 1AF, Liverpool, UK

K.Kifayat@2004.ljmu.ac.uk, {M.Merabti, Q.Shi, D.Llewellyn-Jones}@ljmu.ac.uk

Abstract: Achieving security in resource-constrained wireless sensor networks is a challenging research task. Many key management schemes have been developed recently to provide secure communication between source and destination in wireless sensor networks. A serious threat highlighted in all of these schemes is that of node capture attacks, where an adversary gains full control over a sensor node through direct physical access. The compromised sensor node can be an aggregator node, cluster head node or a normal sensor node. This can lead an adversary to compromise the communication of an entire sensor network which creates a high risk for data confidentiality. Solving this problem with limited resources is a major challenge.

Ignoring the security issues related to data aggregation can bring large damage to a sensor network. Nonetheless data aggregation brings significant advantages in terms of reducing the amount of data transmitted between source and destination and thereby conserving energy. However data aggregation schemes are generally designed without considering the possible security consequences relating to data confidentiality.

To deal with the above issues we therefore propose a novel and distinct Structure and Density Independent Group Based Key Management Protocol which forms part of our broader work looking at security issues in Wireless Sensor Networks. This protocol provides better secure communication, secure data aggregation, data confidentiality, and resilience against node capture and replication attacks. The protocol has been evaluated using different topologies both with and without group structures and compared against existing schemes. Evaluation results show a significant improvement in resilience against node capture attacks, confidentiality, memory overhead and connectivity.

Keywords: security, key management, node capture attack, replication attack, secure data aggregation, wireless sensor networks

1. Introduction

Wireless Sensor Networks (WSNs) consist of a large number of low-cost, low-power, and multifunctional sensor nodes that communicate over short distances through wireless links [1]. The small size of sensors, their sensing operations, low cost and networking behaviour increase the benefits of using WSNs in our daily life, enabling them to provide significant advantage for many applications that would not have been possible in the past. The unique properties of WSNs increase flexibility and reduce user involvement in operational tasks. Continuous growth in the use of WSNs in sensitive applications such as military or hostile environments and also generally has resulted in a requirement for effective security mechanisms in the system design [18]. However establishing efficient communication in terms of routing, security etc. in such WSNs is not easy since the sensor nodes have limited processing power, storage, bandwidth and energy.

As for energy efficiency, most of a sensor node's energy is consumed during computation as well as the sending and receiving of data packets. Sending one bit requires the same amount of energy as executing 50 to 150 instructions on a sensor node [22]. Therefore reducing network traffic is also important to save sensors' battery power in any WSN communication protocol. To minimize the number of transmissions from thousands of sensor nodes towards a sink, a well known approach is to use in-network aggregation. The energy savings of performing in-network aggregation have been shown to be significant and are crucial for energy-constrained sensor networks [24-26].

Alongside energy efficient communication protocols we require a balanced security solution to guard against possible security threats in WSNs. Furthermore it is necessary to investigate all the areas and applications in WSNs where security is vitally required. For example, there are serious issues connected with the use of in-network data aggregation [23]. Although previous work [24-27] does provide in-network data aggregation to reduce energy costs, these schemes assume that every node is honest which may not be suitable in terms of security. Present research in WSNs security has found different types of new attacks e.g. node capture attack which can be harmful for network communication, in-network data aggregation, routing and so on.

Aggregation becomes especially challenging if end-to-end confidentiality between sensors and the sink is required [28]. It's interesting to note that Wireless Sensor Networks produce the same security challenges as traditional networks (LAN, WAN, MAN and etc.) but with the additional difficulties of the limited resources of the sensor nodes. As a result, we are unable to use traditional techniques for WSNs. A challenging and distinct security problem in WSN is that of the *node capture attack* (NCA), where an adversary gains full control over sensor nodes through direct physical access to a node. This can lead to a compromise in the communication of an entire sensor network. The compromised sensor node can be an aggregator node, cluster head node or a normal sensor node.

Therefore we should consider such threats as a high risk for communication and data confidentiality/security. Providing secure communication alongside resilience against new attacks such as node capture attacks using limited resources is a major challenge.

In terms of solutions to these security challenges researchers have proposed various key management schemes [2-9, 14] for secure communication, secure data aggregation [28-34] and resilience against possible attacks. These schemes try to provide better resilience against node capture attacks, but

still there is a chance of the entire network being compromised. For example in probabilistic key pre-distribution schemes [2, 5] compromising a few nodes can lead to the entire network communications being compromised.

Another important concern in terms of security solutions for WSNs is that all these proposed solutions are specifically designed for single security problems. They do not consider the consequences of multiple security problems occurring in the sensor network simultaneously. We therefore need a complete security package which can handle more than one problem in a given application. For example an application might require resilience against node capture attack, replication attacks and maintain high data confidentiality.

Furthermore as part of research on node capture attack the evolutionary results in [17] highlighted three main factors which can help adversaries during node capture attacks to compromise the communication of the entire sensor network. The first factor is that node capture attacks can be a threat only if sensor nodes within the network share a key or keys with neighbouring nodes used to encrypt or decrypt data. Consequently the greater the level of key-sharing between neighbouring nodes the greater the threat to communication confidentiality. Most existing solutions suffer from the drawback of having nodes share multiple keys. The second important factor is the structure (topology) of the network. In general, the fewer the communication links between sensor nodes, the greater the possibility that an attacker can entirely block the communication paths between a source and destination. For example, node capture attacks are generally more effective in tree topologies than mesh topologies because in the former there is only one route from child to parent. If the parent node is compromised, the entire communication from its child nodes downward will potentially be compromised [17]. The last factor which has a direct influence on node capture attacks is the density of the network, having a similar effect as the second factor.

On the basis of DGKE [17] evaluation results, three design factors were ascertained as being vital for a key management protocol to provide high resilience against node capture attacks. First, encryption should happen only at the source node and data should be decrypted at the destination node (group leader or sink) to provide better confidentiality. The second factor is the selection of an appropriate topology, since this can help to provide resilience against node capture attacks. For example, DGKE has better performance results against attacks when a random mesh is used as compared to other topologies. Finally we need to select a suitable density according to applications in order to overcome the risk of serious damage to the sensor network resulting from a node capture attack.

In contrast with the risks described earlier in this section we have proposed a novel Structure and Density Independent Group Based Key Management (SADI-GKM) Protocol for large-scale wireless sensor networks. This protocol is designed to provide multiple secure services such as better secure communication, secure data aggregation, confidentiality, resilience against node capture and replication attacks, and protection against malicious nodes sending arbitrary encrypted data in an aggregated form. SADI-GKM provides these services using less memory, reduces processing and provides high connectivity as compared to existing schemes. The structure and density

independence helps to reduce computation cost in maintaining the network topology when new nodes join or leave a group. SADI-GKM can be categorised under pre-deployment security solutions for WSNs.

Moreover SADI-GKM provides two different levels of confidentiality services (cases) at the group leader node according to the available resources and confidentiality requirements of different applications:

Case 1: Data can only be decrypted at the group leader node, aggregated and re-encrypted using a master key before being sent on to the sink.

Case 2: Data can be aggregated in encrypted form without decryption and subsequently sent on to the sink.

In both cases data aggregation is performed only at the group leader nodes. For this protocol, the sensor network is split into logical groups of nodes. Every group has a group leader node and member nodes. All member nodes in each group have a distinct key produced using the master key of the group leader and a node ID. These distinct keys at every node will be used for encryption only to maintain data confidentiality between member nodes during communication with each other, inside the group. During the transmission phase encrypted data is transferred in an ad hoc manner to group leader nodes through other member nodes with authentication. The encrypted data from all member nodes in a group can only be decrypted and aggregated at the group leader node. The aggregated data is further encrypted at the group leader node and sent towards the sink.

Suppose that using the SADI-GKM Protocol an adversary has compromised a group leader node (aggregator node). Although our proposed protocol provides several merits such as less memory usage, the communication and data confidentiality of the entire group can be compromised since the compromised group leader node can decrypt data from all its member nodes. Therefore our current requirement is that encrypted data from member nodes does not need to be decrypted for the aggregation. Instead, the aggregation is performed with encrypted values and only the sink can decrypt the result. However we do consider secure group leader selection in our future work in case of any damage to the group leader node.

In this paper we extend the results of our earlier paper [21], introducing a number of new results relating to the applicability and energy overhead of our protocol. We organize the paper as follows. Section 2 gives a brief introduction to existing key management schemes. In Sections 3 and 4 we consider the important issues of confidentiality, aggregation and homomorphic encryption that we will use in the remainder of the paper. Section 5 describes our design goals and assumptions. Section 6 introduces the proposed scheme in detail, which is then analysed carefully in Section 7. Final conclusions and future work are given in Section 8.

2. Related work

As we are considering more than one issue in our proposed protocol we therefore consider related work in the areas of both key management and secure data aggregation.

2.1 Key management

A number of key management schemes have been proposed for resource constrained sensor networks. Eschenauer and

Gilger [2] proposed a probabilistic key pre-distribution scheme. They use a large pool of keys with their associated key identifiers. Every sensor node is pre-equipped with a fixed number of keys and identifiers randomly chosen from the key pool. After deployment every pair of nodes within their wireless communication range establishes their common keys. If they share any common keys, they can pick one of the keys as their secret key. Path-key establishment can take place in case there is no common key between a pair of nodes. Chan *et al.* [5] extended the previous idea to overcome the difficulties that occur when a pair of nodes shares no common key. These schemes suffer from two major drawbacks, making them inappropriate for many applications. First they require that the deployment density is high enough to ensure connectivity. Second the compromise of sets of keys or key spaces can lead to compromise of the entire network.

PIKE [6] addresses the problem of the high density requirement of random key pre-distribution schemes [5]. In PIKE, each sensor node is equipped with an ID of the form (i,j) , corresponding to a location on a $\sqrt{n} \times \sqrt{n}$ grid, where n is the network size. Each sensor is also preloaded with a number of pairwise keys, each of which is shared with a sensor that corresponds to a location on the same row or the same column of the grid. Now any pair of sensors that does not share a preloaded pairwise key can use one or more peer sensors as trusted intermediaries to establish a path key. PIKE requires network-wide communication to establish path keys, each of which requires $O(\sqrt{n})$ communication overhead. This is a relatively high communication overhead, making it unsuitable for large sensor networks.

The BROadcast Session Key (BROSK) [10] negotiation protocol stores a single master key for the entire network in every sensor node. A pair of nodes (S_i, S_j) exchange random nonce values N_i and N_j . The master key K_m is used to establish a session key $K_{i,j} = \text{MAC}(K_m | N_i | N_j)$, where “|” is used for concatenation and MAC refers to the Message Authentication Code derived by applying an authentication scheme, together with a secret key, to a message [20].

Roberto, Luigi, and Sushil [11] proposed a key management protocol for large sensor networks. The protocol is composed of two main phases. In the first phase a new session key is generated, while in the second phase the new session key is distributed to all of the sensors in the WSN. In the first phase, each sensor autonomously generates a session key, with the algorithm driving this generation ensuring that each sensor generates the same key. The second phase focuses on ensuring that each sensor holds an appropriate set of cryptographic keys. This second phase requires synchronization, and such session-based solutions might therefore suffer from time synchronization problems.

Researchers have also proposed various group-based key management solutions. The group-based key pre-distribution scheme of Dongg *et al.* [4] has two parts: Group-based EG and Group-based PB. The Group-based EG scheme’s key pool size in each instance is 500. Each sensor node randomly selects 50 keys from its in-group instance and 50 keys from its cross-group instance. The Group-based PB scheme instance includes a 49-degree bivariate polynomial. Each sensor node gets assigned the polynomial shares from its in-group instance and cross-group instance. Eltoweissy *et al.* [7] have proposed a scheme for group key management in

large-scale WSNs. Their proposed scheme is based on Exclusion Basic Systems (EBS). The use of EBS is for assigning and managing keys. They assume that all nodes are pre-initialized, prior to deployment, with an identical state that mainly consists of a set of training parameters and a number of keys. A key server also has one or more session keys known to subsets of the group. All group members aware of a particular session key constitute a secure communication group. Members in a secure communication group use the session key corresponding to the group for encrypting messages exchanged among group members.

Pre-deployment knowledge of sensor nodes has been used to improve the performance of many key pre-distribution protocols [8, 9, 10, 11].

DGKE is a group based key management solution where every sensor node shares unique keys with its neighbouring nodes used for encryption and decryption. The number of keys depends on the density of the sensor nodes. Therefore DGKE has shown improvements in memory overhead and in resilience against node capture attack as compared to other protocols [4, 6]. However, DGKE also has a number of drawbacks in similarity to other protocols [4, 6].

Whilst all of these existing solutions provide resilience against node capture attacks to a certain degree, the compromise of a certain proportion of the nodes will lead to a compromise in communication across the entire network. This is a consequence of the fact that every sensor node is equipped with n keys shared with its neighbouring nodes. Moreover, the disordered structure of WSNs makes any proposed solution for key management in indoor sensor networks difficult to implement in outdoor applications. There exist some good solutions for key management, but when the network expands, or the structure of the network changes due perhaps to a change in the environment, these methods often fail.

2.2 Secure data aggregation

In the initial stages of sensor networks research many data aggregation protocols [25-27, 29] have been proposed but none of them were designed with the consideration of possible security threats but further research highlighted the importance of security.

Hu and Evans [30] proposed a secure hop-by-hop data aggregation scheme. In this scheme individual packets are aggregated in such a way that a sink can detect non-authorized inputs. The proposed solution introduces a significant bandwidth overhead per packet. They also assume that only leaf nodes with a tree-like network topology sense data, whereas the intermediate nodes do not have their own data readings. Jadia and Muthuria [31] extended the Hu and Evans approach by integrating confidentiality, but considered only a single malicious node.

Several secure aggregation algorithms have been proposed for the scenario of a single data aggregator. Przydatek, *et al.* [32] proposed Secure Information Aggregation (SIA) to detect forged aggregation values from all sensor nodes in a 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. They assume that the bandwidth between a remote user and an aggregator is a bottleneck. Therefore their protocol is intended for reducing this

bandwidth overhead while providing a means to detect with high probability if the aggregator is compromised. Yangs, *et al.* [33] describe a probabilistic aggregation algorithm which subdivides an aggregation tree into sub trees, each of which reports its aggregates directly to the sink. Outliers among the sub trees are then probed for inconsistencies [38]. A number of aggregation algorithms have been proposed to ensure the secrecy of the data against intermediate aggregators. Such algorithms have been proposed by Girao *et al.* [34], Castelluccia *et al.* [28], and Cam *et al.* [29].

3. Confidentiality and Data aggregation

Confidentiality can be maintained between source and destination using different solutions according to requirements and available resources. In our further discussion we describe three possible solutions to achieve end-to-end (source to destination) confidentiality in large scale sensor networks.

The first option is that each sensor stores a unique key shared only with the sink, and sends encrypted data through other sensors to the sink without decryption at any non-sink node. This end-to-end confidentiality can be achieved but there are some drawbacks. Because all packets are forwarded towards the sink, a lot of bandwidth is consumed. It's also very burdensome when a sensor network is busy to recover large amounts of data from every single node in the case of data loss. Finally, there is an extreme imbalance between sensors in terms of the amount of data communicated, e.g., sensors closer to the sink will lose energy more quickly.

The second option is hop-by-hop secure data aggregation. This type of scheme is only limited to specific topologies such as a tree topology, but it nonetheless does the job.

The third option which we recommend is where an aggregator node has an appropriate position, such as a group leader node, which can assist in securely aggregating the data within its group. This will also help to reduce the amount of data to be transferred to the sink and support network structure independence. It is possible that the group leader node is not in the best position to perform the role. The issue of how to optimally select a node in a group for the role of data aggregation in relation to several factors such as minimal energy consumptions is beyond the scope of this paper and will be studied in our future work.

4. Homomorphic Encryption

Homomorphic encryption is a semantically-secure encryption which, in addition to standard guarantees, has additional properties. In particular the sum of any two encrypted values is equal to the encrypted sum of the values. There are several efficient homomorphic cryptosystems such as Unpadded RSA, El-Gamal, Goldwasser-Micali, and Benaloh and Paillier [37].

Consider Unpadded RSA as an example, where we use the notation $E_k(x)$ to denote the encryption of a message x with a key k . Suppose that a public RSA key is expressed as $pk = (e, m)$. Then the encryption of a message x with key pk is signified as $E_{pk}(x) = x^e \bmod m$. In this case we have the homomorphic property:

$$\begin{aligned} (E_{pk}(x_1) E_{pk}(x_2)) \bmod m &= (x_1^e x_2^e) \bmod m \\ &= (x_1 x_2)^e \bmod m \end{aligned}$$

$$= E_{pk}(x_1 x_2)$$

For addition and average calculations in Wireless Sensor Networks, we can use the following homomorphic encryption algorithm.

Encryption:

1. Represent a message (sensed data) as an integer d with $0 < d < Z$, where Z is a large integer.
2. Let $s = (k, Z)$ be a shared secret key with $0 < k < Z$.
3. Define $c = E_s(d) = (d + k) \bmod Z$.

Decryption:

1. Compute $E_s^{-1}(c) = (c - k) \bmod Z = d$.

Addition of Cipher texts:

1. Let $s_1 = (k_1, Z)$ and $s_2 = (k_2, Z)$ be two secret keys, and d_1 and d_2 ($0 < (d_1 + d_2) < Z$) be two messages. Compute $c_1 = E_{s_1}(d_1) = (d_1 + k_1) \bmod Z$ and $c_2 = E_{s_2}(d_2) = (d_2 + k_2) \bmod Z$. This leads to $(c_1 + c_2) \bmod Z = ((d_1 + d_2) + (k_1 + k_2)) \bmod Z = E_{s_{1,2}}(d_1 + d_2)$, with key $s_{1,2} = (k, Z)$ and $k = (k_1 + k_2) \bmod Z$.
2. For decryption, we have:
 $E_{s_{1,2}}^{-1}(c_1 + c_2) = ((c_1 + c_2) - k) \bmod Z = d_1 + d_2$.

Note that if n different ciphers c_i are added, then Z must be larger than $\sum_{i=1}^n d_i$, and otherwise the correctness of recovered data is not provided. In practice, if $p = \max(d_i)$, then Z should be selected as $Z = 2^{\lceil \log_2(p * n) \rceil}$ [28].

5. Assumptions and Design goals

In order to apply the SADI-GKM technique to a Wireless Sensor Network, a number of assumptions are necessary in order to provide a consistent framework within which to work. It will also be useful for us to consider the requirements of our design. These are outlined in the following subsections.

5.1 Network and security assumption

We assume a static and synchronized sensor network. The sink, acting as a key server knowing all the groups' master keys, is assumed to be a PC, laptop or computer with long lasting power. The group leader node acts as aggregator and router node for communication with other groups. To avoid the need to consider routing issues between the group leader nodes and sink, we currently also assume they have a longer transmission range compared to normal sensor nodes. The sensor nodes can be deployed via aerial scattering in outdoor applications. However in indoor applications sensor nodes can be installed manually and the immediate neighbouring nodes will be known in advance according to application requirements.

We assume an adversary can eavesdrop on all traffic, inject packets, or replay older messages. If a node is physically compromised, all the information it holds will be known to the attacker. However, the base station cannot be compromised.

5.2 Design goals

SADI-GKM is designed to support secure communication in large scale sensor networks; therefore, it provides the basic

security services such as confidentiality and authentication. In addition, SADI-GKM also meets several security and performance requirements that are more challenging in sensor networks.

Supporting multiple topologies: The independent nature of SADI-GKM will be useful for different applications. The current situation is that researchers have proposed many distinct solutions for every different application. In addition ad hoc sensor network topologies change frequently due to nodes joining and leaving, especially in mobile sensor networks. A topology independent solution could significantly decrease energy consumption.

Survivability: Due to the unattended nature of sensor networks, an attacker could launch various security attacks and even compromise sensor nodes without being detected. Therefore, a sensor network should be robust against security attacks e.g. node capture attacks and replication attacks. Even if an attacker succeeds, its impact should be minimized. For example, the compromise of a single sensor node should not break the security of the entire sensor network.

Supporting secure in-network processing: Security mechanisms should permit in-network processing operations in secure way such as secure data aggregation. In-network processing significantly reduces energy consumption in sensor networks.

Forward and backward secrecy: The proposed key management scheme should provide forward and backward secrecy when nodes join or leave a group [18].

Memory overhead: Only a small number of keys should be necessary while supporting a high level of security.

Connectivity: With a small number of keys, the probability that two sensors share at least one common key during any given time-interval should be kept as high as possible to maintain connectivity [12].

6. Structure and Density Independent Group Based Key Management Protocol

Generally Wireless Sensor Networks are scalable networks, and it is not uncommon for them to incorporate thousands or even millions of sensor nodes. Researchers have proposed a number of different network management protocols and schemes for large scale WSNs. A common idea proposed for management of large-scale networks is that of splitting them into regions or small groups of nodes (logically or physically) using clustering, geographical division, topology *etc.* To deal with the scalability issue we therefore organise large-scale sensor networks into different small groups of nodes with unique IDs. However for better understanding of our protocol we first consider key management inside a group and secondly demonstrate key management between groups.

As for deployment, a Gaussian distribution can be used to establish random group deployment of sensor nodes in some outdoor applications [8]. For indoor applications sensor nodes can easily be deployed in groups. However, we do not intend to consider the issue of node deployment in detail at this stage. The notations to be used for the protocol presentation are summarised as follow:

GI	Group ID
NI	Node ID in group

ID	Node ID in network (concatenation of GI and NI)
M_{kGI}	Symmetric master key shared between sink SI and group leader GI
M_i	Encrypted data of node ID_i
M_{GI}	Encrypted data of group leader node GI
K_{SI, ID_i}	Symmetric key shared between node ID_i and the sink SI . This key is use for the encryption and decryption of data collected by ID_i
K_{GI, ID_i}	Symmetric key shared between node ID_i and its group leader node GI
V_{GI}	Key/secret shared among all the sensor nodes in group GI
U_i	Key/secret shared among all group leaders
TS_{ID_i}	Time stamp for node ID_i
TS_{GI}	Time stamp for group leader node GI
d_i	Sensed data by node ID_i

6.1 Key pre-distribution phase

In first phase of key pre-distribution the following steps are performed. This occurs before deploying the sensor nodes in groups.

- Assign a global unique ID to each sensor node. This global ID is comprised of a group ID, GI and an ID within the group, NI (see Figure 1).
- Assign a unique master key M_{kGI} to every group leader and member nodes in the sensor network.
- Assign a unique value V_{GI} to each group of nodes, used to generate keys for authentication inside the group.
- Assign a unique value U_i to every group leader node.
- Generate and store a unique key K_{GI, ID_i} using the master key M_{kGI} and node ID for every group of nodes.
- Assign unique key K_{SI, ID_i} to every sensor node ID_i (Note: this key need only be assigned if we require the aggregation of data in encrypted form at group leader nodes).

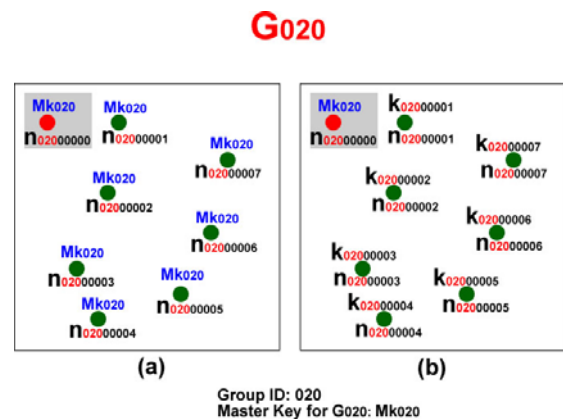


Figure 1. Organization of sensor groups.

A node has global ID of the form **02000002**, with the initial three digits representing GI (group ID) and the last five digits representing NI (local node ID in a group GI) as shown in Figure 1. Algorithm 1 shows the complete pre-distribution process. According to steps 4 and 5 there will ultimately be no master key stored in any other sensor nodes. These steps should be performed by every node in every group except for the group leader. It is important to emphasise that all nodes in all groups will use these different

keys K_{GI, ID_i} for encryption purposes only in order to provide better confidentiality. The reason for keeping the master key M_{kGI} at the group leader is to allow it to decrypt the encrypted data received from other member nodes. In every single group data will be encrypted at the source node and is only decrypted at the group leader node for data aggregation purposes, as will be explained in later sections.

- Pre deployment -
Step 1: Node ID formation
Concatenation of Group ID and Node ID forms a unique ID for each node in the Network
 $ID \leftarrow \text{join}(GI, NI)$
Step 2: Master Key assignment to node N_{ID} in each group GI
 $N_{ID} \leftarrow M_{kGI}$
- Key generation -
Step 3: Key generation for node N_{ID}
If $ID \triangleleft \text{join}(GI, 00)$ then
 $K_{GI, ID_i} \leftarrow \text{hash}(M_{kGI}, ID)$
Else
 $K_{GI, ID_i} \leftarrow M_{kGI}$
End if
Step 4: Replace master key M_{kGI} with K_{GI, ID_i} if node is not a group leader node
 $Mk_{GI} \leftarrow K_{GI, ID_i}$
Step 5: Store $K_{GI, ID_i}, V_{GI}, K_{SI, ID_i}$

Algorithm 1. Pre-distribution phase.

6.2 Data transmission phase

We first discuss the functions of each non-group-leader node, which are defined in Algorithm 2 as part of the proposed protocol. Suppose that after successful deployment of the sensor nodes, an event occurs. In the step 1 of the *Send* function in Algorithm 2, the event source node with its identity ID_i will collect and encrypt the event data d_i with its key K_{SI, ID_i} shared with the sink to produce C_i . To enable data freshness checking by the group leader of node ID_i for the detection of a replication attack, we produce a value M_i by re-encrypting C_i with a time stamp TS_{ID_i} using the key K_{GI, ID_i} shared between node ID_i and its group leader GI , where “ C_i, TS_{ID_i} ” in $E_{K_{GI, ID_i}}(C_i, TS_{ID_i})$ signifies the concatenation of C_i and TS_{ID_i} .

In step 2, node ID_i produces an authentication code X_i by hashing M_i , the key/secret V_{GI} shared among all the group members, ID_i and TS_{ID_i} , for communication with neighbouring nodes. After finding the correct neighbouring node, node ID_i sends out (X_i, M_i, ID_i, TS_i) . The value X_i will be used by the receiver node to authenticate the received information and TS_{ID_i} is used to avoid replication attacks.

When the neighbouring node receives the information, it produces an authentication code X_j by hashing the received items (M_i, ID_i, TS_{ID_i}) with shared key V_{GI} , and compares it against X_i for the purposes of authentication, as illustrated in the *Receive* function of Algorithm 2. If successful, the received data are forwarded to the next neighbour. In this way, we avoid encryption and decryption at every single hop to reduce the usage of limited resources available to the nodes.

- Group member nodes -
Send () { // Works inside a group
Step 1: Collect, encrypt and assign sensed data to variable M_i
 $d_i \leftarrow \text{Sensed_data}()$
 $C_i \leftarrow E_{K_{SI, ID_i}}(d_i)$
 $M_i \leftarrow E_{K_{GI, ID_i}}(C_i, TS_{ID_i})$
Step 2: Send out encrypted data M_i , hash value X_i , source node ID_i , and time stamp TS_{ID_i}
 $X_i \leftarrow \text{hash}(M_i, V_{GI}, ID_i, TS_{ID_i})$
Return $(X_i, M_i, ID_i, TS_{ID_i})$
}
Receive (X_i, M_i, ID_i, TS_i) {
Step 1: Authenticate the sender node
 $X'_i \leftarrow \text{hash}(M_i, V_{GI}, ID_i, TS_i)$
If $X'_i = X_i$ and TS_i is fresh then
Step 2: Forward received data to next neighbour toward group leader
Return (X_i, M_i, ID_i, TS_i)
Else
Abort and report to group leader GI
}

Algorithm 2. Group member nodes.

Second, each group leader node has additional responsibilities, including secure data aggregation, integrity checking and authentic communication with other group leaders. Algorithm 3 illustrates the functions performed by each group leader.

As shown in the *Receive* function of Algorithm 3, when a group leader node receives data, it will first check whether the data is from another group leader node or a member node, based on two calculated hashes X_{GI} and X_i . Here, secrets V_{GI} and U_i used for the calculation of the hashes are stored in the node prior to its deployment. In the case where the data is from a member node (i.e. $X_i = Y_i$), the leader node applies its master key M_{kGI} and received identity ID_i to compute shared key K_{GI, ID_i} for the decryption of M_i to recover C_i and TS_{ID_i} . It saves C_i if TS_{ID_i} is fresh, and discards C_i otherwise. The successful decryption of M_i can also be used to authenticate source node ID_i and check data integrity. After the data from all expected group member nodes have been received, the leader node invokes the *Send* function of Algorithm 3.

If the data is from another group leader (i.e. $X_{GI} = Y_i$) and TS_i is fresh, then the received data are forwarded to the next appropriate group leader on the route to the sink. For the communication between groups, we adopt the assumption used in LEACH [19], namely, each group leader node has a larger communication range than an ordinary one so that neighbouring group leader nodes can communicate with one another directly. This assumption will be relaxed in our future work to allow non-group-leader nodes to mediate communications between group leaders in an authentic manner.

- Group Leader Nodes –**Send () {****Step 1:** Compute aggregated data D_{GI} from encrypted data, C_1, \dots, C_n , received from group member nodes

$$D_{GI} \leftarrow \left(\sum_{i=1}^n C_i \right) \bmod Z \quad 0 < D < Z$$

Step 2: Encrypt D_{GI} and TS_{GI} to produce M_{GI}

$$M_{GI} \leftarrow E_{M_{k_{GI}}}(D_{GI}, TS_{GI})$$

Step 3: Send out encrypted data M_{GI} , hash value Y_{GI} , group identity GI , and time stamp TS_{GI}

$$Y_{GI} \leftarrow \text{hash}(M_{GI}, U_i, GI, TS_{GI})$$

Return $(Y_{GI}, M_{GI}, GI, TS_{GI})$ **}****Receive (Y_i, M_i, ID_i, TS_i) {****Step 1:** Check if the data is coming from a member node or another group leader

$$X_{GI} \leftarrow \text{hash}(M_i, U_i, ID_i, TS_i)$$

$$X_i \leftarrow \text{hash}(M_i, V_{GI}, ID_i, TS_i)$$

If $X_i = Y_i$ then // Data from a member node**Step 2:** Decrypt M_i using key K_{GI, ID_i} where $K_{GI, ID_i} = \text{hash}(M_{k_{GI}}, ID_i)$, to recover C_i and TS_{ID_i}

$$(C_i, TS_{ID_i}) \leftarrow E^{-1}_{K_{GI, ID_i}}(M_i)$$

If TS_{ID_i} is fresh thenSave C_i If C_i is the last value received then**Step 3:** Send out aggregated data

Send ()

Else

Discard C_i and report to sinkElse if $X_{GI} = Y_i$ and TS_i is fresh then**Step 4:** Forward the data received from a group leader to another group leader towards the sinkReturn (Y_i, M_i, ID_i, TS_i)

Else

Abort and report to sink

}**Algorithm 3. Group leader nodes.**

For the *Send* function of Algorithm 3, the encrypted data items $\{C_1, \dots, C_n\}$ are first aggregated to produce D_{GI} using homomorphic encryption as described in section 4. We note the value of Z must be chosen large enough to prevent overflow. As with Algorithm 2, it is necessary to encrypt D_{GI} and a time stamp TS_{GI} with the key $M_{k_{GI}}$, shared with the sink, for the purposes of data authentication and freshness checking. The *Send* function also uses the secret U_i , shared among all the group leader nodes, and a time stamp TS_{GI} to produce an authentication code Y_{GI} to deter replay attacks. Finally, the data items $(Y_{GI}, M_{GI}, GI, TS_{GI})$ are sent to the first neighbouring group leader node on the route to the sink.

Third, Algorithm 4 explains the decryption process performed by the sink. We assume the sink has all the keys $U_i, M_{k_{GI}}$ and K_{SI, ID_i} shared with the group leaders and sensor nodes. Once the data has been received at the sink node it will go through a process to verify the source group leader

(i.e. by checking that $Y_{sink} = Y_{GI}$ and TS_{GI} is fresh). However the Y_{GI} can be forged if any group leader node is compromised. Step 2 ensures the time stamp and data have not been altered by a compromised group leader (i.e. $E^{-1}_{M_{k_{GI}}}(M_i)$ and $TS'_{GI} = TS_{GI}$) otherwise discard received information and take appropriate action. If step 1 and 2 are successful then step 3 will be performed to retrieve the aggregated data as shown in algorithm 4.

- Sink Node -**Receive $(Y_{GI}, M_{GI}, GI, TS_{GI})$ {****Step1:** Authenticate the sender group leader

$$Y_{sink} \leftarrow \text{hash}(M_{GI}, U_i, GI, TS_{GI})$$

If $Y_{sink} = Y_{GI}$ and TS_{GI} is fresh then**Step2:** Decrypt M_{GI} using master key $M_{k_{GI}}$ to recover D_{GI} and TS_{GI} .

$$(D_{GI}, TS'_{GI}) \leftarrow E^{-1}_{M_{k_{GI}}}(M_{GI})$$

If $TS'_{GI} = TS_{GI}$ then**Step3:** Get the aggregated data

$$S_x \leftarrow (D_{GI} - \left(\sum_{i=1}^n K_{SI, ID_i} \right)) \bmod Z$$

Else

Abort and take action

}**Algorithm 4. Sink node.**

The sink can calculate the average μ as $\mu = S_x/n$. To calculate the variance σ^2 every sensor node needs to send the square of each sensed value to the group leader so the variance can be calculated as $\sigma^2 = S_x/n - \mu^2$ where S_x is the sum of the squared value of each sensor node. If the group leader node is interested to find the minimum or maximum values from receive encrypted values, we can use the Order Preserving Encryption Scheme for Numeric Data (OPES) [35]. This scheme allows comparisons to be directly applied to encrypted data. Existing results have shown that homomorphic encryption is a cost effective solution for secure data aggregation in sensor networks [36].

7. Performance Evaluation and Simulation

In this section, we analyze our proposed scheme in detail. For analysis, we adopt similar methods to those described in Du *et al.* [8]. We evaluate our proposed scheme against the following criteria that represent desirable characteristics in a key distribution scheme for WSNs: stronger resilience against node capture, forward and backward secrecy, resilience against replication attack, secure data aggregation, memory overhead, communication overhead and connectivity. We use the same radio model as LEACH [19]. Currently, there is a great deal of research in the area of low-energy radios. Different assumptions about the radio characteristics, including energy dissipation in transmit and receive modes, will change the advantages of different protocols. In our work, we assume a simple model where the radio dissipates $E_{elec} = 50$ nJ/bit to run the transmitter or receiver circuitry and $\mathcal{E}_{amp} = 100$ pJ/bit/m² for the transmit amplifier to achieve an acceptable E_b/N_o (see Figure 2 and Table 1). We also assume an r^2 energy loss due to channel

transmission. Thus, to transmit a k -bit message a distance d using this radio model, the radio would expend the following energy [19].

$$\begin{aligned} E_{Tx}(k, d) &= E_{Tx-elec}(k) + E_{Tx-amp}(k, d) \\ E_{Tx}(k, d) &= (E_{elec} \times k) + (\mathcal{E}_{amp} \times k \times d^2) \end{aligned} \quad (1)$$

To receive this message, the radio expends the following.

$$\begin{aligned} E_{Rx} &= E_{Rx-elec}(k) \\ E_{Rx}(k) &= E_{elec} \times k \end{aligned} \quad (2)$$

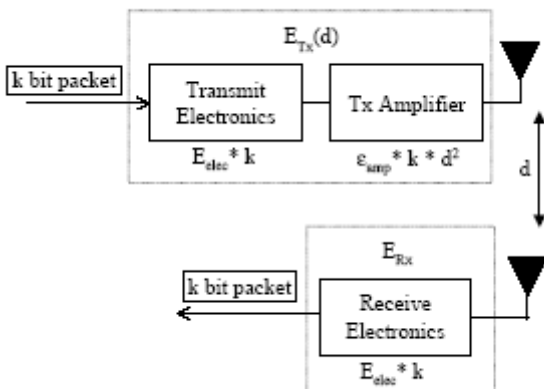


Figure 2. First order radio model [19].

Operation	Energy Dissipated
Transmitter Electronics ($E_{Tx-elec}$)	50 nJ/bit
Receiver Electronics ($E_{Rx-elec}$) ($E_{Tx-elec} = E_{Rx-elec} = E_{elec}$)	
Transmit Amplifier (\mathcal{E}_{amp})	100 pJ/bit/m ²

Table 1: Radio characteristics [19].

Furthermore in our implementation we have used adaptive routing, Blowfish and Homomorphic algorithms for encryption and decryption of data. During simulations we have used different sizes of packets ranging from 24bits to 400bits according to the size of sensed or encrypted data.

We have considered the energy consumption of processing (encryption, decryption and aggregation), and that of sending and receiving packets. Further detail is given in section 7.2.

In the next section we focus mainly on node capture attacks and secure data aggregation providing detailed information about our proposed protocol's resilience. In the evaluation we make no assumptions about the topology used for communication inside a group, as our proposed solution is structure and density independent. As part of the comparison, we have simulated the SADI-GKM protocol using three different topologies: tree, grid, and random mesh.

7.1 Node capture attacks

In this section we check the performance of our SADI-GKM Protocol against node capture attacks. DGKE has shown differing resilience to node capture attacks for different topologies, highlighting its topology dependence [16]. However, in contrast to this we are able to show that our SADI-GKM Protocol generates similar results with different

topologies. A summary of these results is shown in Figure 3, illustrating the structure independence of the protocol.

As further performance analysis against node capture attacks we implemented our protocol using grid and mesh topologies and compared it against existing grid and mesh-based schemes. Finally we present the performance of our proposed protocol in various different scenarios.

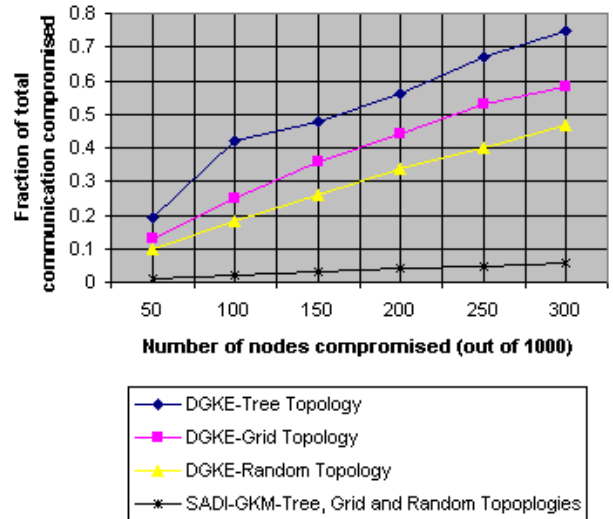


Figure 3. Node capture attack on DGKE and SADI-GKM at different topologies.

7.1.1 SADI-GKM Protocol performance using grid topology

To evaluate the performance of our scheme using grid topologies, we compare our results with PIKE 2D [6] and DGKE grid-based key management protocols.

We consider PIKE 2D since it is a well established and widely known protocol that uses a unit distance bidirectional communication model. Consequently, we were able to use the same simulation and network settings for our SADI-GKM Protocol, PIKE 2D and DGKE. PIKE and DGKE were simulated on a flat, square deployment field. Although we note that there are other functional differences between the various protocols, our focus of concern is primarily that of security and our objective is therefore to compare protocols based on this metric, rather than other aspects of their functionality. We have used the following configuration during the simulations. The sensor network is based on 5000 sensor nodes. The link density at each sensor node is 2-4 sensor nodes. We take the top right, top left, bottom right and bottom left edge nodes to be group leaders for different groups. During the simulation, if a group leader is compromised, we assume the entire group communication is compromised. The simulations involved 50 groups, with each group comprised of 100 sensor nodes. The results are based on averaged runs of 500 simulations. The detail explanation of simulation with code is given in [22].

The results – illustrated in Figure 4 – show that our SADI-GKM Protocol provides better resilience against node capture attacks. Through a comparison of the three algorithms, we can see that this improved performance is primarily as a result of fewer keys being shared on average between nodes in our proposed protocol as compared to DGKE and PIKE 2D. This reinforces the first observation of DGKE that sharing keys with neighbouring nodes for data

encryption and decryption helps node capture attacks to compromise the entire network.

For these results, the proportion of compromised communication is based on the number of compromised links as a fraction of the total links in the simulation.

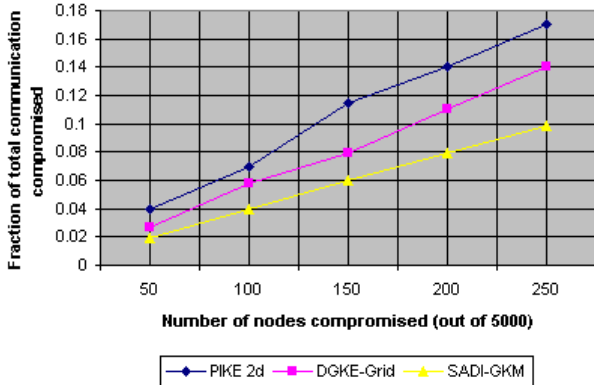


Figure 4. Comparison of probability of total communication compromise in grid topology.

7.1.2 SADI-GKM Protocol performance using random mesh topology

Evaluating the performance of the SADI-GKM Protocol using a random topology (random mesh), we compared our results with the group-based key management protocol Group-based EG [4] and DGKE random mesh. We chose the Group-based EG scheme since the protocol organises the sensor network into groups, each group being organised using a random mesh topology. The Group-based EG scheme assumes a key pool size of 10,000; this key pool is divided into 200 smaller, equal-sized key pools for every group (500 keys per group in each of the smaller key pools). We have used similar simulation and network settings for Group-based EG, DGKE and our proposed protocol.

The simulation settings are different from the PIKE 2D and DGKE grid, as the Group-based EG scheme uses a 10,000 sensor node network for the simulation and the link density of every sensor node is random.

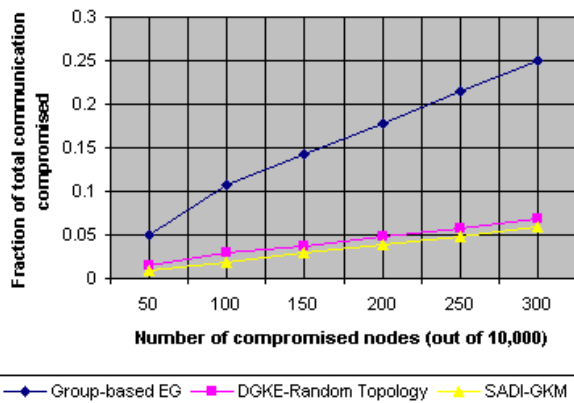


Figure 5. Comparison of the probability of total communication compromise in mesh topology.

To compare our approach with Group-based EG and DGKE we have used a similar configuration. We assume there is a total of 10,000 sensor nodes deployed in a 1000×1000m area. These sensor nodes are divided into 100 deployment groups with 100 sensor nodes in each group. The internal

communication structure of every group will be distinct from any other group. We assume the radio range is $R = 40m$. Every sensor node will find its neighbouring nodes within this 40 metre radio range, thereby dictating the link density. The results are based on averaged runs of 500 simulations. Figure 5 shows that the SADI-GKM Protocol clearly has better resilience against node capture attacks in mesh topologies.

Again, this is partly a result of the way keys are distributed between nodes. As we have already noted, the Group-based EG scheme utilises a key pool in order to allow keys to be shared between nodes. This has certain advantages in terms of memory usage, where a node is unable to hold a unique key for every other node in the network. It also marginally increases security during the network deployment stage, avoiding the need to use a master key for all nodes in a group. However, the memory benefits for group-based systems are less pronounced, and the consequence of using a key pool is greater key sharing across multiple nodes. This compares against our own scheme, where key sharing is restricted to pairs of nodes, providing the increased resilience shown by the simulation results.

Finally we compare the SADI-GKM Protocol random mesh results with other similar existing schemes. The State-Based Key Management scheme [12] improves on the results of others as shown in Figure 6.

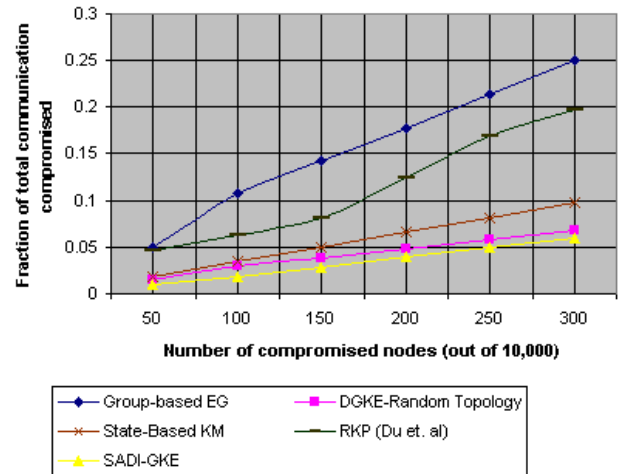


Figure 6. Comparison of probability of total communication compromise for various protocols.

Although our proposed solution is a group based solution, we have only considered peer to peer communication inside the groups. In contrast to our solution, the other schemes compared against provide the added functionality of peer to peer communication for the entire sensor network. However as we shall see in the following section this comes at a cost in terms of scalability, memory overhead and processing. In situations where such functionality is not needed, our solution therefore provides important benefits.

7.1.3 Resilience against node capture attacks with or without using groups

In this section we have undertaken an analysis as to whether sensor networks are more robust against node capture attacks either with or without using groups. We have implemented the SADI-GKM Protocol in both group-based and non-group sensor networks using a random mesh topology. Figure 7 presents the effect of node capture attacks on sensor

networks both with and without groups using SADI-GKM and the probabilities of group leader compromise using different topologies are shown in Figure 8.

We can see from figure 8 that the probability of group leader compromise is similar across different topologies. The effect of grouping therefore remains the same using different topologies in the case of physical compromise of the group leader nodes.

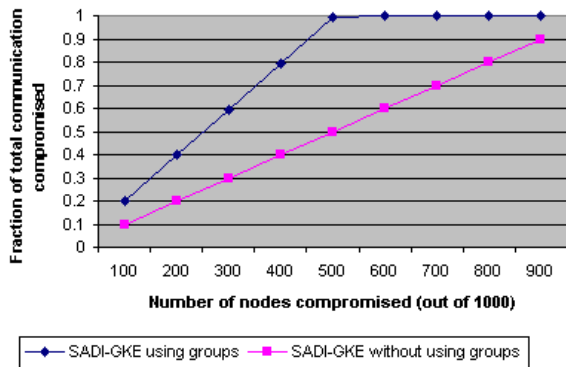


Figure 7. Node capture attack in group and non-group SADI-GKM.

The non-group based solutions' results are better than those for the group-based solutions. An important reason for using groups is that of scalability; otherwise our proposed protocol produces better results without using groups. Therefore in small-scale networks we can use our proposed protocol with greater efficiency. The final conclusion is that our solution can easily be used in small and large-scale sensor networks with high resilience against node capture attacks compared to the other solutions that we have tested against.

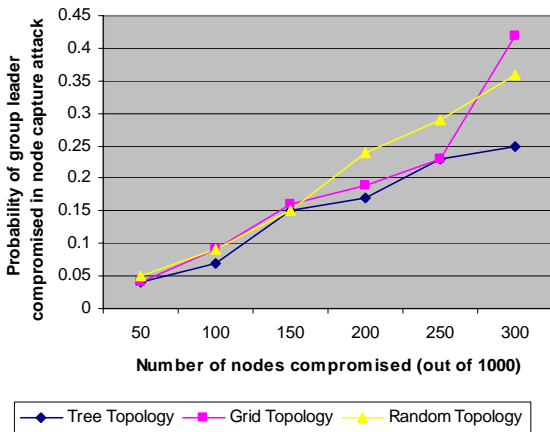


Figure 8. Probability to compromise Group leader in different topologies.

Our simulations show the influence of these three parameters, which were highlighted initially by DGKE:

- The structure (topology) of the sensor network.
- The density of the sensor nodes.
- Sharing keys with neighbouring nodes used for encryption and decryption.

In our proposed protocol we have avoided the influence of these three parameters, thereby obtaining significantly improved results in terms of resilience against node capture

attack, memory overhead, processing, connectivity, and forward and backward secrecy.

7.2 Secure data aggregation and communication overhead

In this section we present the energy consumption during implementation of SADI-GKM with Adaptive routing for sensor networks. The simulations involved groups of 100 nodes using a grid topology with one group leader node. The initial energy of all sensor nodes is set to 1 Joule. We assume that all sensor nodes in the group will continually sense and send information to the group leader and that the group leader will aggregate this data and send it towards the sink. Once all sensor nodes have successfully sent information to the group leader it will have completed one cycle (100 events). Our simulation results are based on 20 to 700 cycles in a group of nodes.

During evaluation we have considered a number of different cases in order to establish the energy requirements for various levels of security functionality. The general structure of the network for these various cases – as defined by the SADI-GKM protocol – is shown in Figure 9.

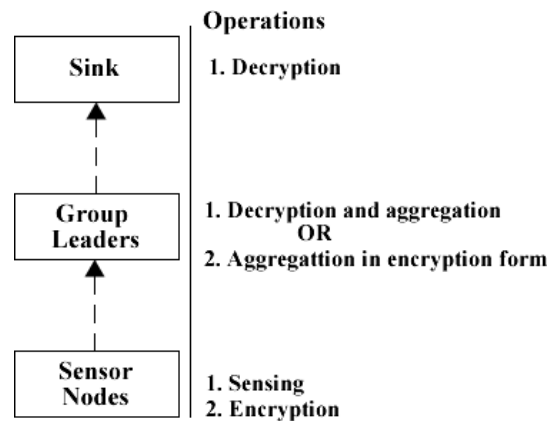


Figure 9. Overview of functionality provided in various cases by nodes in the SADI-GKM protocol.

As described in section 1, SADI-GKM considers two different cases in terms of services at the group leader node. In Case 1 the group leader aggregates data after decryption and in Case 2 the group leader aggregates encrypted data without decryption. Both of these services afford different levels of security and cost.

We have implemented Case 1 in three different forms. In the first form (no security) we remove all security features in order to establish the exact cost of SADI-GKM during communication. Therefore the cost of this “no security” case is solely the cost of routing and aggregation at the group leader, as shown in Figure 10. The size of the packet for this first form is 24bits for each event (8bits for sensed temperature data and 16bits for source node ID_i).

In the second form every sensor node in the group encrypts its sensed data using the key K_{GL, ID_i} as describe in Section 6. Furthermore the group leader will decrypt all received data and calculate the final aggregated result. The aggregated result will then be re-encrypted using the master key M_{KGI} and sent to the sink. In this second form we use the Blowfish algorithm for encryption and decryption, which increases the size of the packets to 208bits (192bits of encrypted data and

16bits of source node ID_i). The third form is similar to the second form but in this case we use homomorphic encryption (as discussed in Section 4) instead of Blowfish for the encryption and decryption of data. The simulation results for all three forms are shown in Figure 10. We can clearly see that using Blowfish encryption a group leader will lose all energy after around 440 cycles.

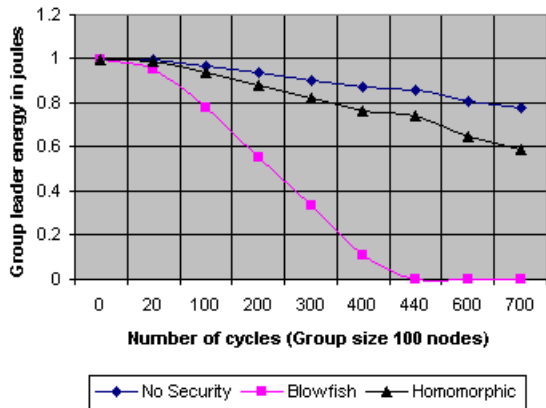


Figure 10. Cost of SADI-GKM using group of 100 nodes grid.

Figure 10 clearly shows how the load on the group leader increases as we apply increased security (confidentiality) measures.

In Case 2 every sensor node ID_i has two keys: K_{GI, ID_i} shared with the group leader node and K_{SI, ID_i} shared with the sink. In the first step the sensor node encrypts the data using the key K_{SI, ID_i} . In addition the encrypted data is re-encrypted along with a time stamp using K_{GI, ID_i} . As a result encryption happens twice at the source node. This additional encrypted data is used to provide resilience against replication attacks. Once the group leader receives a packet it will decrypt it using the master key M_{kGI} and the source node ID_i . Subsequently the group leader aggregates the encrypted data and sends it on to the sink re-encrypted using the master key M_{kGI} .

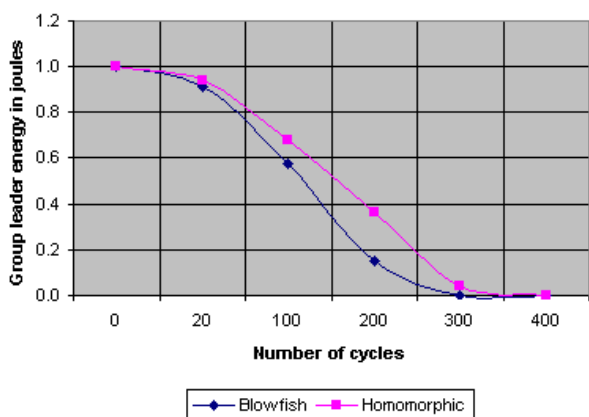


Figure 11: Cost of secure data aggregation using Blowfish and Homomorphic encryption.

We have also implemented Case 2 using homomorphic and Blowfish encryption algorithms. However, as described earlier, encryption happens twice at the source node using the key shared with the sink (K_{SI, ID_i}) and the key shared with the group leader (K_{GI, ID_i}). Consequently in the first form we use Blowfish twice for encryption (using K_{SI, ID_i} , K_{GI, ID_i}) and in second form we use the homomorphic algorithm (using

K_{SI, ID_i}) and Blowfish (using K_{GI, ID_i}) together. The reason for encrypting the data a second time using Blowfish as well as with the homomorphic algorithm, is that we need to encrypt the time stamp along with the encrypted data at the source node for resilience against replication attacks. The sizes of the packet in Case 2 are 208bits (192bits of double encrypted data and 16bits of source node ID_i) and 400bits (384bits of double encrypted data and 16bits of source node ID_i). The results for these cases are shown in Figure 11.

Figure 12 shows the energy consumption of the group leader in Case 1 and Case 2 with increasing group size using SADI-GKM. In all cases the result is established for a 100 cycle run. We can see from this how the load on the group leader increases as the group size increases. This is a clear consequence of the increased data that must be aggregated as the group size increases. However, it also highlights how the application of functionality to prevent replication attacks in Case 2 has an impact on the energy used by the group leader, in comparison to that of Case 1.

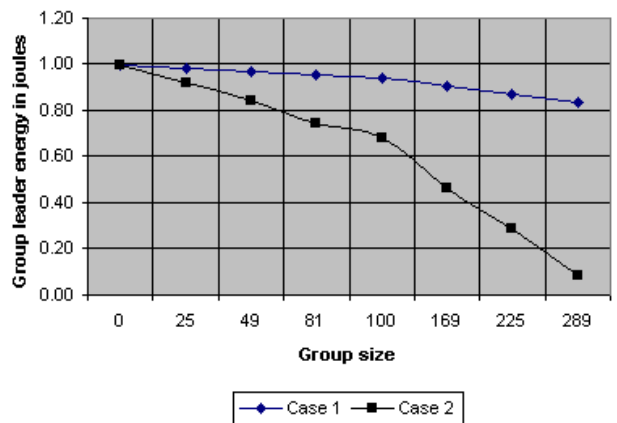


Figure 12. Cost of SADI-GKM using different group sizes (100 cycles for each group).

7.3 Forward and backward secrecy

Our proposed key management protocol is able to provide forward and backward secrecy due to the fact that every sensor node in each group has a different key for data encryption. We can establish this as follows.

Joining a Group: When a node intends to join a group GI , it will follow a similar process to that of “Pre-deployment” as explained in section 4.1. The new node will be supplied with a fresh key K_{GI, ID_i} for encryption, the value V_{GI} for authentication and a node ID_i . The new nodes must be deployed within the target group according to the position of the group in the sensor network. Once a node has been successfully deployed within the group it will be allowed to communicate with member nodes in that group. Backward secrecy is achieved since all nodes in the group have distinct encryption keys from the new node. Consequently all nodes in the group are independent in terms of data secrecy. The data can only be encrypted at the source node using its unique key and can thereafter not be decrypted by any other member node except the group leader of the same group.

Leaving a Group: Suppose a node dies due to power loss or enemy capture. This action will have no effect on the other nodes in the group in terms of data secrecy. Suppose the enemy has captured the sensor node and has successfully retrieved the keys K_{GI, ID_i} , V_{GI} , and the node’s ID. Although the enemy node may be able to launch a limited replication

attack using V_{GI} , it has no access to the data from other nodes during communication, and forward secrecy is therefore achieved. We deal with the issue of the potential for replication attacks in the next section

7.3 Replication attacks

To prevent replication attacks we have used time stamps at two steps in our algorithms. First when the source node encrypts a time stamp with the collected sensed data, to prevent replication attacks between the source node and destination (the group leader or sink). Second when the sender node sends the times stamp TS_{ID_i} with the value X_i , so that the receiver node can authenticate the sender node, protecting against replication attacks at each hop.

We will explain this process using an example. Suppose node B is an enemy node and acts as if it is a normal group member. Now node A sends information to node B . At the first step the source node A encrypts its sensed data along with a time stamp using the unique key K_{GI, ID_i} . At the second step the encrypted data M_i , the values V_{GI} , ID_i and the time stamp TS_{ID_i} are hashed together and assigned to X_i . As we know V_{GI} is shared between all member nodes and K_{GI, ID_i} is different for every node in the group. Finally the source node A sends X_i , M_i , ID_i and TS_i to node B (the enemy node). To launch a replication attack, node B will act as node A , using node A 's ID. In other words, node B starts forwarding the same encrypted data M_i using ID_i from node A but using a new time stamp. We assume that Node B has value V_{GI} , Node A 's ID_i . Although this is sufficient to convince intermediate nodes, the group leader can finally detect this attack by checking the data freshness, since the attacker node B is unable to produce a fresh M_i since it is not party to node A 's encryption key.

7.5 Connectivity

A particular advantage of our scheme is that we are able to achieve absolute unconditional connectivity in any size of group. As previously mentioned, every node keeps keys according to local node density. Li *et al.* [14] also achieve 100% connectivity [15] but their 100% connectivity is conditional. As they explain, their scheme will achieve full connectivity only when 55 keys are assigned in each group. The size of group doesn't affect performance of our scheme either, unlike solutions such as the key pool based idea and other group-based schemes.

7.6 Memory overhead

In this section we will compare our proposed SADI-GKM Protocol with Group-based pre-distribution scheme [4], PIKE [6] and DGKE [16]. The memory overhead for PIKE-2D is $\lceil \sqrt{n} \rceil + 1$ where n is the total number of nodes and DGKE requires only $n_k \leq d$ keys where n_k is the number of keys and d is the density of sensor nodes. The SADI-GKM Protocol is more flexible in terms of memory overhead since every sensor node in the network has two keys: one for encryption and another for authentication. Group size doesn't affect memory overhead in our scheme, whereas for PIKE and Group-based EG increasing the size or number of groups causes the number of keys in each node to increase. Group-based EG stores 50 keys on average in every sensor node in a group on the basis of a 40m radio range. In DGKE

the number of keys depends only on the number of neighbours a node has but our proposed protocol is density independent. The number of neighbouring nodes has no effect on memory use.

8. Conclusions and Future work

In this paper we have proposed a novel and distinct Structure and Density Independent Group Based Key Management Protocol. As an extension of our earlier work, we have shown that our protocol provides better secure communication, secure data aggregation, confidentiality, and resilience against node capture and replication attacks using reduced resources. The protocol has been evaluated using different topologies both with and without group structures, different encryption algorithms and compared against existing schemes. Evaluation results show a significant improvement in resilience against node capture attacks, confidentiality, memory overhead and connectivity.

By considering the three parameters identified for DGKE – as presented in the introduction – we have improved the performance of our solution. Our solution can be implemented in any small or large-scale network.

In future work we intend to extend our protocol to deal with security issues related to mobility in sensor networks, as this feature impacts significantly on key management. Additionally, we will generalise the protocol to include other confidentiality-preserving computations such as that for determining standard deviations.

References

- [1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks", IEEE Communications Magazine, Vol. 40, No. 8, pp. 102-116, August 2002
- [2] L. Eschenauer and V. Gligor, "A Key Management Scheme for Distributed Sensor Networks", CCS 2002
- [3] A. Hac, "Wireless Sensor Networks", John Wiley & Sons, 2003, ISBN 0-470-86736-1
- [4] D. Liu, P. Ning, W. Du, "Group-Based Key Pre-Distribution in Wireless Sensor Networks," in Proceedings of 2005 ACM Workshop on Wireless Security (WiSe 2005), September 2005.
- [5] H. Chan, A. Perrig and D. Song, "Random Key Predistribution Schemes for Sensor Networks". In 2003 IEEE Symposium on Research in Security and Privacy. pp 197-213.
- [6] H. Chan, A. Perrig. "PIKE: Peer intermediaries for key establishment in sensor networks". In INFOCOM, 2005.
- [7] M. Eltoweissy, A. Wadaa, S. Olariu, L. Wilson, "Group key management scheme for large-scale sensor networks", Elsevier Ad Hoc Networks 2004.
- [8] W. Du, J. Deng, Y. S. Han, S. Chen, and P. Varshney. "A key management scheme for wireless sensor networks using deployment knowledge". In Proceedings of IEEE INFOCOM'04, March 2004.
- [9] D. Huang, M. Mehta, D. Medhi, and L. Harn. "Location-aware key management scheme for wireless sensor networks". In Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks (SASN '04), pp 29 – 42, October 2004.

- [10] D. Liu and P. Ning. "Location-based pairwise key establishments for static sensor networks". In 2003 ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN '03), pp 72–82, October 2003.
- [11] Z. Yu and Y. Guan. "A key pre-distribution scheme using deployment knowledge for wireless sensor networks". In Proceedings of ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), April 2005.
- [12] J. Park, Z. Kim and K. Kim, "State-Based Key Management Scheme for Wireless Sensor Networks", Proc. of WSNS2005, pp. 819-825, Nov. 7, 2005, Washington, DC, USA.
- [13] A. Becher, Z. Benenson, and M. Dornseif. "Tampering with motes: Real-world physical attacks on wireless sensor networks". In 3rd International Conference on Security in Pervasive Computing (SPC), April 2006.
- [14] L. Zhou, J. Ni, C. V. Ravishankar, "Efficient Key Establishment for Group-Based Wireless Sensor Deployments" WiSe 2005, September 2, 2005 .
- [15] W. Du, J. Deng, Y. S. Han. "A pair wise key predistribution schemes for wireless sensor networks". In ACM Conference on Computer and Communications security (CCS), 2003.
- [16] K. Kifayat, M. Merabti, Q. Shi, D. Llewellyn-Jones, "Dynamic Group-Based Key Establishment for Large-scale Wireless Sensor Networks", 1st International Conference on Communications and Networking (ChinaCom 2006), Beijing China.
- [17] K. Kifayat, M. Merabti, Q. Shi, D. Llewellyn-Jones, "Application Independent Dynamic Group-Based Key Establishment for Large-scale Wireless Sensor Networks", China Communication Journal, Special issue on Communication and Information Security in Feb, 2007
- [18] S. A. Çamtepe, B. Yener, "Key Distribution Mechanisms for Wireless Sensor Networks: a Survey," TR-05-07 Rensselaer Polytechnic Institute, Computer Science Department, March 2005
- [19] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. "Energy-Efficient Communication Protocol for Wireless Microsensor Networks." In Proc. Of Hawaiian Int'l Conf. On Systems Science, January 2000.
- [20] <http://www.rsasecurity.com/rsalabs/> [Access Data: 11/10/2006]
- [21] K. Kifayat, M. Merabti, Q. Shi, D. Llewellyn-Jones, "Applying Secure Data Aggregation Techniques for a Structure and Density Independent Group Based Key Management Protocol", The Third IEEE International Symposium on Information Assurance and Security (IAS 2007), Manchester, UK, 29-31 August 2007.
- [22] K. Piotrowski, P. Langendoerfer, and S. Peter. "How public key cryptography influences wireless sensor node lifetime". In Proceedings of the 4th ACM Workshop on Security of ad hoc and Sensor Networks, SASN 2006, 2006.
- [23] S. Peter, K. Piotrowski, P. Langendoerfer. "On Concealed Data Aggregation for Wireless Sensor Networks, Consumer Communications and Networking Conference", IEEE CCNC 2007, Las Vegas Nevada, USA, January 2007.
- [24] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. "Impact of network density on data aggregation in wireless sensor networks". In Proceedings of the 22nd International Conference on Distributed Computing Systems, 2002.
- [25] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. "TAG: a tiny aggregation service for ad-hoc sensor networks". SIGOPS Oper. Syst. Rev., 36(SI): pp.131–146, 2002.
- [26] Y. Yao and J. Gehrke. "The COUGAR approach to in-network query processing in sensor networks". SIGMOD Rec., 31(3): pp. 9–18, 2002.
- [27] A. Deshpande, S. Nath, P. B. Gibbons, and Srinivasan Seshan. "Cache-and-query for wide area sensor databases". In SIGMOD 2003, 2003.
- [28] C. Castelluccia, E. Mykletun, and G. Tsudik. "Efficient aggregation of encrypted data in wireless sensor networks". In Proceedings of The Second Annual International Conference on Mobile and Ubiquitous Systems, 2005.
- [29] H. Cam, S. Ozdemir, P. Nair, D. Muthuavinashiappan, and H. O. Sanli. "Energy-efficient secure pattern based data aggregation for wireless sensor networks". Computer Communications, 29:446–455, 2006.
- [30] L. Hu and David Evans, "Secure aggregation for wireless networks," in Workshop on Security and Assurance in Ad hoc Networks, January 2003.
- [31] P. Jadia and A. Mathuria. "Efficient secure aggregation in sensor networks". In Proceedings of the 11th International Conference on High Performance Computing, 2004.
- [32] B. Przydatek, D. Song, and A. Perrig. "SIA: Secure information aggregation in sensor networks". In Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, 2003.
- [33] Y. Yang, X. Wang, S. Zhu, and G. Cao. "SDAP: A Secure Hop-by-Hop Data Aggregation Protocol for Sensor Networks". ACM MOBIHOC'06, May 2006.
- [34] J. Girao, M. Schneider, and D. Westhoff. "CDA: Concealed data aggregation in wireless sensor networks". In Proceedings of the ACM Workshop on Wireless Security, 2004.
- [35] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. "Order preserving encryption for numeric data". ACM SIGMOD, June 2004.
- [36] M. Acharya, J. Girao, D. Westhoff, "Secure Comparison of Encrypted Data in Wireless Sensor Networks" (WiOpt 2005), Riva del Garda, Trentino, Italy, April 2005
- [37] [Access Date: 01/03/2007, Available at: http://en.wikipedia.org/wiki/Homomorphic_encryption]
- [38] H. Chan, A. Perrig, D. Song. "Secure hierarchical in-network aggregation in sensor networks", Proceedings of the 13th ACM conference on Computer and communications security, Alexandria, Virginia, USA, 2006.

Author Biographies

Kashif Kifayat received his MSc degree in Computer Science from the University of Liverpool, UK and BSc in Computer Science from the University of Peshawar, Pakistan. He is now working towards his PhD degree in Computer Science at Liverpool John Moores University, UK.

His research interests include security, scaleable and mobile challenges in Wireless Sensor Networks.

Professor Madjid Merabti is Director of the School of Computing and Mathematical Sciences, Liverpool John Moores University, UK. A graduate of Lancaster University, he has over 20 years experience in conducting research and teaching in Distributed Multimedia Systems (Networks, Operating Systems and Computer Security). Professor Merabti leads the Distributed Multimedia Systems and Security Research Group, which has a number of government and industry supported research projects. He is an Associate Editor of IEEE Transactions on Multimedia and IEEE Computer Communications, a Co-Editor in Chief of Pervasive Computing and Communications, and a member of the editorial board for the Computer Communications Journal. He is a member and chair of a number of conference TPCs and is chair of the PostGraduate Networking Symposium series (PGNet) for UK PhD students.

Dr. Qi Shi received his PhD in Computing from Dalian University of Technology, P.R.C. He worked as a research associate for the Department of Computer Science at the University of York in the UK. Dr Shi then joined the School of Computing and Mathematical Sciences at Liverpool John Moores University in the UK, and he is currently a Reader in Computer Security. His research interests include network security, security protocol design, formal security models, intrusion detection, and ubiquitous computing security. He is supervising a number of research projects in these research areas.

Dr. David Llewellyn-Jones is a Research Fellow at Liverpool John Moores University. He received his PhD in Model Theory from the University of Birmingham in 2002. In March 2003 he joined the Distributed Multimedia Systems and Security Research Group to work on an EPSRC funded project looking at Secure Component Composition for Personal Ubiquitous Computing. His research interests include computing and network security, component composition, security frameworks in Ubiquitous Computing environments and Sensor Network security.