Secure Group Association Management in Heterogeneous Distributed Sensor Networks

Al-Sakib Khan Pathan, *Student Member*, *IEEE*, and Choong Seon Hong

***Abstract*--A Heterogeneous Distributed Sensor Network (HDSN) is a type of distributed sensor network where sensors with different functional types participate at the same time. In this network model, the sensors are associated with different deployment groups but they cooperate with each other within and out of their respective groups. The heterogeneity of HDSN refers to the functional heterogeneity of the sensors participating in the network unlike the heterogeneity considered (e.g., considering transmission range, energy level, computation ability, sensing range) for traditional heterogeneous sensor networks. Taking this model into account, we propose a secure group association authentication mechanism using one-way accumulator which ensures that; before collaborating for a particular task, any pair of nodes in the same deployment group can verify the legitimacy of group association of each other. Secure addition and deletion of sensors are also supported in this approach. In addition, a policy-based sensor addition procedure is also suggested. For secure handling of disconnected node of a group, we use an efficient pairwise key derivation scheme. Side by side proposing our mechanisms, we also discuss the characteristics of HDSN, its scopes, applicability, challenges, and future. The efficiency of our management approach is demonstrated with performance evaluation and analysis.**

***Index Terms*—Accumulator, Association, Heterogeneous, Security.**

I. Introduction

A Wireless Sensor Network (WSN) is composed of hundreds or thousands of inexpensive, low-powered sensing devices with limited computational and communication resources. Typical task of the sensors is to sense certain parameters from their surrounding environments and to send the readings to a central entity called base station. The raw data collected from such a network are analyzed to extract important information about a particular area and are often used for taking important decisions. Considering today's advancements and achievements, the applicability of sensor network is very broad. With the capabilities of today's sensors, many applications can be benefited a lot. Now we have varieties of sensors that can monitor temperature, pressure, humidity, soil makeup, vehicular movement, noise level, lighting condition, the presence or absence of certain kinds of objects, mechanical stress level on attached objects, and other properties. These tiny devices could be used for surveillance in military and public-oriented applications, target tracking, environmental monitoring, patient monitoring in hospitals, disaster management and warning systems, and in so many other applications.

With the rapid advancements of wireless technologies and sophistication of sensing technologies, many innovative applications could be thought of using the smart sensors. Though most of the applications focus on collecting a specific type of data, for some applications it is necessary to acquire different types of data from the same geographical region. Again, some applications might need collaborative operations among the sensors before producing reports to send to the base station/sink. As an example, a volcano monitoring application may require thermal, seismic, and acoustic data from the same geographic location. Though only one type of data may be satisfactory for such an application, using various types of data could be more beneficial for extracting accurate and timely information. Say for example, the average temperature (already processed by a sub-set of nodes) of a certain region along with the seismic and acoustic readings can provide more precise information regarding an imminent event. Especially for disaster management and warning systems, military applications, and medical applications, use of multiple types of data can really be advantageous. To facilitate such types of applications that need more than one type of data, ExScal mote [1], [2] is designed by CrossBow Inc. and Ohio State University. This mote is basically an extension of the well-known MICA2 mote [3] which supports multiple sensors (i.e., sensing units) on the same radio board. However, instead of using this type of multipurpose node in the network, using different types of nodes in the same area could be more efficient considering the utilization of memory, processing, and energy resources of the sensors. We will provide more points in the next section to support this argument. The key point here is that whatever the configurations of the sensors are, heterogeneous data are often required for some applications which can increase the complexity of tasks in the network. Hence, efficient methods are required for dealing with all aspects in such types of applications.

Security is one of the critical aspects in any kind of sensor network. It is anticipated that in most application domains, sensor networks constitute an information source that is a mission critical system component and thus, require commensurate security protection. If an adversary can thwart the work of the network by perturbing the generated information, stopping production, or pilfering information,

Al-Sakib Khan Pathan is a PhD student at Networking Lab, Department of Computer Engineering, Kyung Hee University, Korea. (e-mail: spathan@networking.khu.ac.kr).

Choong Seon Hong is with the Department of Computer Engineering, Kyung Hee University, Korea (e-mail: cshong@khu.ac.kr).

then the usefulness of sensor networks is drastically curtailed. So, it should be made sure that the sensors that are participating in the data acquisition and supplying process are authentic and are included as legitimate entities in the network. To be specific, along with other supporting security mechanisms, it is required to verify the authenticity of the sensors before allowing them to participate in any collaborative task.

Considering all these factors, in the remainder of this article we first propose our model in Section II. We present our network assumptions and preliminaries in Section III and IV. Then, we propose our approach of secure group association management in Heterogeneous Distributed Sensor Networks in Section V. Performance analysis and discussions are presented in Section VI and finally, Section VII concludes the article with future scopes of research on this network model.

II. Background: Heterogeneous Distributed Sensor Network

A. What is Heterogeneous Distributed Sensor Network?

Typical '*heterogeneity*' in sensor network is considered based on the capabilities of the sensors in the network or more specifically based on the memory, processing capability, energy level, sensing range, and transmission range of the radio [5], [6]. Say for example, transmission range of a sensor depends on the level of energy. Larger transmission range requires more energy or vice versa. So, in most of the cases, the *heterogeneity* is defined considering the dissimilarities in the energy level, processing power, and transmission range. However, in our case this term refers to the dissimilarities in the functions of the sensors. Say for example; in a given network, type 1 sensors sense temperature, type 2 sensors sense seismic signals, type 3 sensors sense magnetism, and so on. In our model, different types of sensors (with different sensing units) may even have similar capabilities (in terms of other capabilities), but this fact doesn't help to give the network a '*homogeneous*' tag.

We term our model network as Heterogeneous Distributed Sensor Network (HDSN), where sensors of various functional capabilities form different functional groups [9]. There could be T (where, $1 < T \leq 6$) functional types of sensors in the deployed network. We primarily consider at most six types of nodes in the same HDSN as this could be enough for supporting any of today's applications and many innovative multi-purpose applications of sensor networks in the coming future. The value of T could also be set based on the application at hand. The network is called homogeneous when $T = 1$, that means there is only one type of sensor in the whole network. For deploying the entire network, first a number of different types of sensors are taken and they are assigned different ids based on their functional types.

Figure 1 shows a graphical model of a Heterogeneous Distributed Sensor Network (HDSN). In this figure, T=3. Like any other Distributed Sensor Network (DSN), it has a large number of sensors covering a large area. The deployment is dense so that a particular network region is covered perfectly. Also we assume that the sensors could frequently be added or deleted from the network. In the figure, we show three different sinks collecting data from three different deployment groups (DGs). A deployment group (DG) is composed of only one type of sensor and it spreads over the entire area of the network. The sinks shown in the model are inter-connected securely with each other. There could also be only one sink gathering all forms of data from different DGs. In such a case, the processing burden of the sink increases as it has to collect, classify, and process different types of incoming data simultaneously.

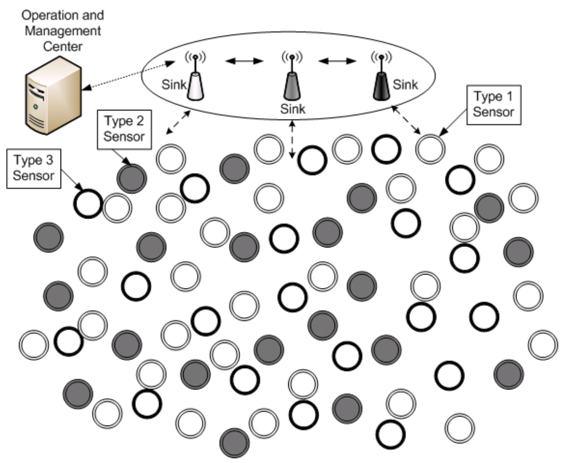


Fig. 1. An example of Heterogeneous Distributed Sensor Network (HDSN) deployment. Here, T=3. Three types of sensors are dispersed over the same target area. Same types of sensors form a deployment group (e.g., all the gray sensors (Type 2) form one network-wide deployment group (DG))

Each of the DGs covers the whole AOI (Area of Interest) and works independently. However, the data packets from a sensor in one DG could be relayed by the sensors of another DG. So, practically in this sample HDSN, there are three distributed sensor networks of different functionalities that are working individually but side-by-side cooperating with each other for data transmissions and network operations. However, for collaborating for a particular task, the neighboring sensors must be the members of the same deployment group. That means even if two dissimilar sensors are neighbors to each other, they can only help for forwarding each other's packets but cannot take part in the same collaborative task. Figure 2 illustrates this. In the figure, three nodes 1, 2, and 3 are neighbors of each other but each of them is from a different network-wide DG. They can just relay each other's packets if needed, but cannot participate in the same task.

B. Why HDSN?

Now, an important point of argument can be that; instead of using multiple nodes over the same area, similar benefit could be achieved by using multipurpose nodes (like ExScal motes) in the network. Sometimes (not always) it might even be more cost effective solution than our approach. However, considering all pros and cons, we have found that the use of large number of homogeneous multipurpose nodes (for collecting heterogeneous data) is less efficient than our approach considering some network settings. It cannot be said whether use of HDSN is always advantageous or not. Instead, the verdict depends on the network settings and given application requirements. Some sample cases where HDSN could be a preferable choice over other options are:

(i) In HDSN, a particular DG could be kept in sleep mode whenever necessary while other DGs can keep functioning (Say for example, in Figure 1, Type 1 sensors are kept in sleep mode for a certain period of time). As the deployment of sensors is dense, the connectivity of the whole network is supposed to be supportable even if a particular DG is in sleep mode. This feature can help for maximizing the lifetime of the overall network as in such case at least one DG could be preserved for a long time even if the energies of other types of nodes are exhausted. On the other hand, ExScal type nodes need continuous wake mode when any of the sensing units on the same radio board is functioning. This eventually causes continuous consumption of energy resource (i.e., battery). Putting this type of multipurpose node in sleep mode means all of the sensing units would become idle at the same time.

(ii) For assigning different security levels for different types of sensed data, HDSN is a good platform. Different security levels could be set for different DGs based on the requirements and/or functions of the sensors.

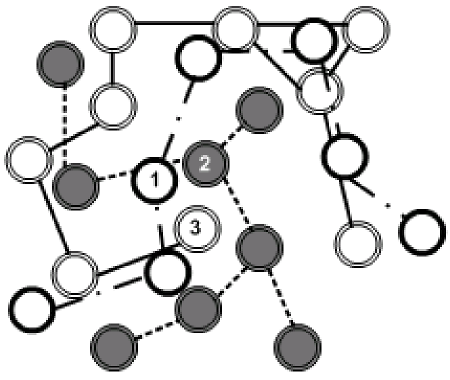


Fig. 2. Nodes in different DGs are neighbors to each other but are not able to take part in the same task. For a given collaborative task, two neighbors must be of the same functional type. The distinguishable connecting lines show the achievable associations between any pair of nodes of same type

(iii) HDSN offers a great advantage over multipurpose node-based network in case of physical capture attack. If a physical intruder intends to hamper the sensing of different parameters in a particular location, it needs to destroy all types of sensors in that location. For example, in our shown model the attacker needs to destroy at least three sensors from a particular section of the network. On the other hand, for a network consisted of multipurpose nodes, destroying one node is enough to destroy three sensing units assigned for a particular spot. The latter is a relatively easier task for a physical intruder.

(iv) In HDSN, different sinks associated with different DGs can gather and analyze specific type of data separately. They can even collaborate with each other after extracting the gist from the collected data. This facility is not available if ExScal type motes are used with a single base station. In many cases, multiple types of readings from multipurpose nodes can somewhat increase the complexity of tasks for the sink as it has to classify and reorder incoming data prior to using them. Use of multiple base stations in the same scenario cannot even help as various types of data packets are amalgamated in the incoming traffic for each base station.

III. Network Assumptions

We assume that in each DG, for each participating node there is an end-to-end path from the corresponding sink to the node. That means in a particular deployment group, G_i, *i*=1, 2, 3….T, (where T is the maximum number of DGs in the HDSN), for each node n_{G_i}, there is a path from the corresponding sink, $S_{G_i} \to n_{m_i} \to (n_{m_i} - 1) \to \cdots n_{G_i}$. A scheme like that is presented in [7] could be used for energy-efficient logical structuring of the network to get a sink rooted tree (SRT) for each deployment group (DG). In this case, n_{G_i} is the leaf node and there could be zero or more intermediate nodes along the end-to-end path.

We assume that each sink associated with each DG has enough processing power to do the initial calculations to initiate the network-wide groups in the network. The sensors deployed in the network have the computational, memory, communication, and power resources like the current generation of sensor nodes (e.g., MICA2 motes [3]). Once the sensors are deployed over the target area, they remain relatively static in their respective positions; that means the neighboring nodes move together (if mobility is allowed) or do not move at all.

Based on these network assumptions, our goal here is to propose a mechanism by which the nodes in a particular DG might securely recognize one another so that any adverse entity cannot in any way be included in the network within its operation time. This kind of group association verification could especially be required for performing some collaborative tasks in the network. For example, when the sink wants to know the average temperature of a certain region, the temperature sensors in that region might have to work together to measure the average of the temperature readings over that particular area. There could also be some sort of data aggregator or pre-processor which would be responsible for doing the primary calculations. In fact, our secure group association verification mechanism could well be used with other clustering mechanisms where there are some cluster heads present to collect these types of readings from a certain set of sensors.

IV. Building Blocks of our Security Management Scheme

In this section we introduce one-way accumulator and pseudoinverse matrix which are used as the building blocks for our security management scheme.

A. One Way Accumulator

From the definition of a one-way function we know that it is a function *F* with the property that; for a given x it is easy to compute $y = F(x)$. However, given F, y, it is computationally infeasible to determine x such as

$x = F^{-1}(y)$. Generally, one-way functions take a single argument. However, Benaloh and Mare [4] considered hash functions which take two arguments from comparably sized domains and produce a result of similar size. In other words, according to [4], a hash function is a function *F* with the property that, $F : A \times B \rightarrow C$ where, $|A| \approx |B| \approx |C|$. This view introduces the one-way hash function with a special *quasi-commutative* property which is termed as one-way accumulator (OWA). According to the definition, OWA is a one-way function, $f : X \times Y \rightarrow X$ with the *quasi-commutative* property such that, for all, $x \in X$ and for all, $y_1, y_2 \in Y$,

$$f(f(x, y_1), y_2) = f(f(x, y_2), y_1)$$

A family of one-way accumulators is a family of one-way hash functions each of which is *quasi-commutative*.

This property is not unusual. In fact, addition and multiplication modulo *n* both have this property as does exponentiation modulo *n* when written as, $e_n(x, y) = x^y \bmod n$. Modular exponentiation also satisfies the *quasi-commutative* property of one-way accumulator:

$$f(f(x, y_1), y_2) = f(f(x, y_2), y_1) = x^{y_1 y_2} \bmod n$$

This could be extended for a long sequence of y_j values (where, $j = 1, \cdots, m$).

The *quasi-commutative* property of one-way accumulators *f* ensures that if one starts with an initial value, $x \in X$, and a set of values $y_1, y_2, \cdots, y_m \in Y$, then the accumulated hash,

$$z = f(f(f(\cdots f(f(f(x, y_1), y_2), y_3), \cdots, y_{m-2}), y_{m-1}), y_m)$$

would be unchanged if the order of the y_js were permuted. This feature could be used for membership verification in a large set of entities. We adopt this feature of OWA for secure group association authentication in HDSN.

B. *Pseudoinverse Matrix*

The pseudoinverse matrix or generalised inverse matrix [10], [11] has a very nice property that could be used for cryptographic operations. It is well known that a nonsingular matrix over any field has a unique inverse. For a general matrix of dimension $k \times n$, there might exist more than one generalized inverse. This is denoted by, $M(k, n) = \{A$: *A* is a $k \times n$ matrix$\}$. Let $A \in M(k, n)$. If there exists a matrix $B \in M(n, k)$ such that, $ABA = A$ and $BAB = B$ then each of *A* and *B* is called a generalized inverse matrix (or pseudoinverse matrix) of the other. In this article, we use the notation A_g to denote the generalized inverse matrix of *A*. We use pseudoinverse matrix for the pairwise key derivation process presented later in the article.

It should be noted that $(A_g)_g = A$ is not always true. The set of all possible pseudoinverse matrices of A is denoted by $\{A_g\}$ and $|\{A_g\}|$ is the cardinality of $\{A_g\}$. Then we have:

Lemma 1: Let A_g be a pseudoinverse matrix of A. Then,

$$rank(A_g) = rank(A)$$

Lemma 2: Let $A \in M(k, n)$ with $rank(A) = k$. If A can be written as $A = [A_1; 0]$, where A_1 is a $k \times k$ nonsingular matrix then, $\{A_g\} = \{\begin{bmatrix} A_1^{-1} \\ Z \end{bmatrix} : Z \in M(n-k, k)$ is an arbitrary matrix$\}$

Proof: Let $B = \begin{bmatrix} X \\ Z \end{bmatrix} \in M(n, k)$. It is then easy to verify that both $ABA = A$ and $BAB = B$ hold if and only if $X = A_1^{-1}$.

V. Secure Group Association Management in HDSN

A. *Association of Sensors in the DGs*

The simplest way to maintain the group association information of a particular group of sensors could be storing the member ids (the ids of each participating sensor in that group) in each sensor node. However, the storage requirement for such member id list linearly increases with the increase of the number of sensors in that particular group. For the sensors with limited storage capabilities, this is not a good solution. Hence, we employ an efficient OWA-based scheme for managing the membership information of a group in such a way that it could well be supported by the storage and computation power of the modern-era sensors. Also based on this limited information, the sensors in a particular DG can securely verify and recognize each other.

B. *Calculating Partial Accumulated Hash Value (PHV)*

In case of one-way accumulator, if the values, $y_1, y_2, \cdots, y_m$ are associated with the users of any cryptosystem, the accumulated hash z of all of the y_js can be computed. A user holding a particular y_j can compute a partial accumulated hash z_j of all y_i with $i \neq j$. The holder y_j can then demonstrate that y_j was a part of the original hash by presenting z_j and y_j such that, $z = f(z_j, y_j)$. We use this partial accumulated hash values in the membership verification process. The following sub-sections present our scheme in details.

C. *Pre-Processing and Pre-storing of PHVs*

Before deployment of a group of sensors, the following steps are performed:

1. A unique id, y_j, $j = 1, \cdots, m$ is assigned for each sensor participating in a particular deployment group.

2. Two safe relatively prime numbers, p and $q = 2p + 1$ are generated.

3. n and $\phi(n)$ are computed as, $n = pq$ and Euler's totient function, $\phi(n) = (p-1)(q-1)$.

4. A random number x (as a seed) is generated which is same for every node in the group.

5. *PHV* for each node y_j is computed using the formula,

$$z_j = x^{\prod_{i=1,i\neq j}^{m} y_i} \bmod n$$

6. Now the values of z_j, n, $\phi(n)$, and corresponding y_j are stored in each sensor in the deployment group.

D. Secure Group Association Verification

After deployment of the sensors in the target area, if a node needs to verify the association of another node (whether they are in the same DG or not), the PHVs and the identities of the nodes are used. For example, let us suppose that two nodes n_p and n_q want to verify whether they are in the same group or not. For this membership verification, these two nodes exchange their pre-stored partial accumulated hash values z_p, z_q and their identities, y_p, y_q. Node, n_q calculates $z = f(z_p, y_p) = z_p^{y_p} \bmod n$, while the other node calculates, $z = f(z_q, y_q) = z_q^{y_q} \bmod n$ locally. If both of the locally computed one-way accumulator values match with each other, the nodes could be sure that, they are participating as the siblings in the same DG in the HDSN. Once the accumulator value is calculated and matched, it could be preserved in the node for successive node membership verification for a given collaborative task.

E. Secure Pairwise Key Derivation between Two Sensors from Two different DGs

Since for each DG, there is a sink rooted tree (as mentioned in the network assumption in section III), once a node in a particular DG finds and verifies its siblings, it can use them for forwarding its own readings. The problem arises when a node is disconnected from all other nodes of its same DG. It can happen due to the failure of an intermediate node of the same DG, or because of random (or, poor) deployment of the sensors. In such a case, though the stranded node cannot participate in the collaborative tasks, it needs to send its own readings to the sink/base station. In fact, it can even happen for a subset of nodes associated with a particular DG. To handle this issue, our mechanism uses a simple method of deriving pairwise keys between two neighboring sensors even if they are associated with two different DGs. Here we describe the secure pairwise key derivation method.

Let n_{G_X} be a node in deployment group, G_X and n_{G_Y} be a node in deployment group, G_Y. n_{G_X} has been somehow disconnected from its siblings but it has got n_{G_Y} as its neighbor, which is connected with the SRT of its own DG (i.e., G_Y). To derive a shared secret key between these two nodes, following operations are performed:

1. Node n_{G_X} randomly generates a matrix X with dimension $m \times n$ and its psedoinverse matrix, X_g. These matrices are kept secret in the node.

2. n_{G_X} calculates $X_g X$ and sends it to n_{G_Y}.

3. In turn, n_{G_Y} randomly generates another matrix Y with dimension $n \times k$, and finds out its pseudoinverse matrix Y_g. These matrices are also kept secret in node n_{G_Y}.

4. n_{G_Y} calculates $X_g XY$ and $X_g XYY_g$. Then it sends the resultant matrices to n_{G_X}.

5. Upon receiving the products of matrices from n_{G_Y}, n_{G_X} computes, $XX_g XYY_g = XYY_g$ and sends it back to n_{G_Y}.

6. Now, both the nodes n_{G_X} and n_{G_Y} can compute the common secret key. n_{G_X} gets it by calculating $X(X_g XY) = XY$ and n_{G_Y} gets it by calculating $(XYY_g)Y = XY$. Both of these outcomes (XY) are the same matrix with dimension $m \times k$.

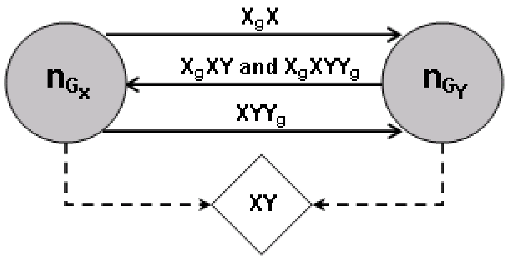


Fig. 3. Pairwise key derivation process between two nodes.

Figure 3 shows the communications between the two neighboring sensors in the pairwise key derivation process. Basically, the key XY is locally computed by each node and the entire method is executed without the intervention of any third party. The derived pairwise key now could be used for secure communications between two nodes. In our case, the node n_{G_X} encrypts all its readings using the key and sends it to n_{G_Y} which can use its own DG (i.e., G_Y) to forward it to the base station. Thus, in this process, even if the participating nodes are from different DGs, they can cooperate with each other for secure data handover within the network.

F. Addition of New Member in a Deployment Group

Addition of new sensors in a particular DG could be handled in two ways. At the time of deploying the sensors, all the sensors assigned for a group might not be used, rather some of them could be kept for later use (depends on the nature of application at hand). Say for example, total number of sensors in a group before deployment is λ. So, a certain portion, say η of these sensors could be deployed first for that particular group and the remaining, $(\lambda - \eta)$ sensors could be added later. In such a case, all the newly added sensors could still be able to prove their legitimacy of membership to other already deployed sensors in that group using our OWA-based verification scheme. This way of addition of new nodes is basically a policy-based management approach, where we handle the addition of new sensors by employing a good deployment strategy.

However, one-way accumulators allow addition of completely new sensors for a certain DG in the HDSN. For example, let us consider that, a new sensor has the id y_{new}, which is assigned from the base station. Mathematically, the new OWA is, $z_{new} = f(z, y_{new})$. To inform all the sensors in that particular DG about the newly added sensor, the base station uses the dedicated end-to-end path (according to our assumption) of each sensor. In turn, each sensor updates its PHV using the formula,

$$z_{j_{new}} = f(z_j, y_{new}) = z_j^{y_{new}} \bmod n$$

Before deployment of the new node, the base station calculates its PHV and stores it in its memory. To add a new set of sensors in a DG, the base station securely sends all the ids of the newly included sensors to the deployed sensors of that deployment group.

G. Deletion of Member from a Deployment Group

Any suspicious behavior detected by the intrusion detection system could convince the base station to purge any sensor from a particular deployment group in the HDSN. To purge an adverse node y_{adv}, the sink uses the secure end-to-end paths for the sensors in the DG to send the id of the deleted node. Getting the delete command, each of the remaining nodes calculates the stored PHV using the equation,

$$z_{u\_j} = z_j^{y_{adv}^{-1} \bmod \phi(n)} \bmod n$$

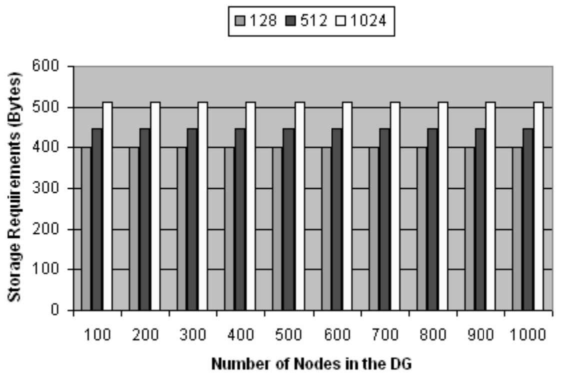
Euler's totient function $\phi(n)$ is used here for the modular operation to ensure that underflow doesn't occur and the purged id could not be reused by any adversary. In case of a node failure due to any unwanted incident like power outage (or other), it should be made sure that the node's id could not be used by any other entity or any attacker. So, to handle this, the same procedure for node purging is employed. And the sink is responsible for taking decision of purging. In some applications, where the clustering techniques are employed, it is possible to assign the charge of taking group-related decisions to the cluster head of the particular cluster (or sub-group). In this case, our scheme offers a decentralized node membership verification mechanism and reduces the burden of tasks of the corresponding sink.

VI. Performance Analysis and Discussion

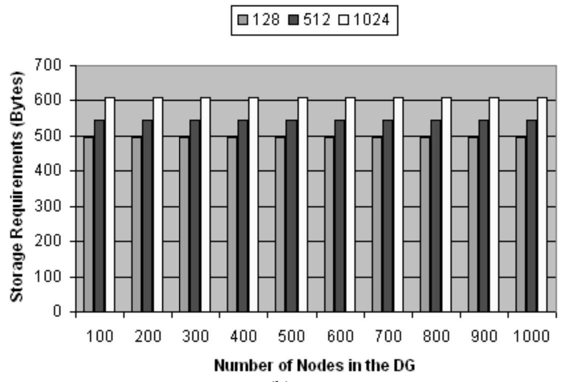
We analyzed our scheme in terms of security, processing complexity, and storage requirements. To understand the performance of our approach, in our system setting we varied the number of nodes in the deployment groups from 100 to 1000.

A. Storage Requirements

As we use similar type of method like RSA [8], like RSA cryptosystem, our scheme requires that the size of *n* should be sufficiently large, typically 1024 bits or more. We considered different lengths of z_j, n, $\phi(n)$ with three different lengths of the ids of the nodes, 128 bit, 512 bit, and 1024 bit. Each sensor node in a particular deployment group has to store a little information; only four values for the secure membership verification mechanism. Figure 4(a) and 4(b) show the storage requirements for a single sensor when our scheme is employed considering different lengths for the stored parameters. In Figure 4(a), lengths of z_j, n, $\phi(n)$ are considered as 1024 bits and in Figure 4(b), they are 1280 bits. We took MICA2 mote as a standard specification. Crossbow MICA2 mote [3] is a well-known sensor node with an ATmega128L 8-bit processor at 8 MHz, 128KB program memory (flash), and 512KB additional data flash memory. Usually it is powered by 2 AA sized batteries. Considering this configuration, our scheme requires only a small amount of memory while a large portion of memory remains available for other associated mechanisms to run smoothly.



(a)



(b)

Fig. 4. Storage requirements for different length node ids keeping the lengths of other parameters (a) 1024 bits (b) 1280 bits

An alternative method of membership management is to store the ids of the sensors participating in the same group. But, compared to our scheme, this approach is very inefficient. In Figure 5 we show that if simple membership list is stored for the same purpose, it requires much more storage than our OWA-based approach. In fact, for a small number of nodes (100 to 300), the storage requirements are very high which could in fact hamper proper functioning of other schemes running in parallel. While for different length ids, membership list technique requires huge amount of memory; in OWA-based approach, considering the lengths of other parameters even 1280 bits (which is good enough for the security) and id length 1024 bits, the memory requirement is fairly less. A great advantage of our approach is that the increase of the number of nodes in the DG doesn't affect the storage requirements for a sensor. Hence, a large number of nodes can be supported for a heterogeneous distributed sensor network. For larger lengths of n and other parameters, the storage requirement increases. However, still it is fairly affordable by today's sensors and comparatively much less than that of storing the whole membership list. In a nutshell, our approach is fairly scalable.

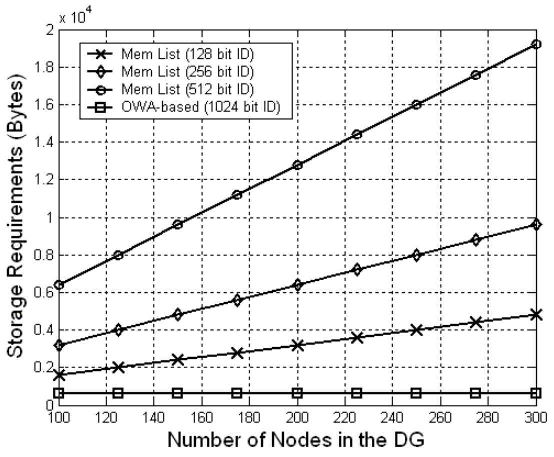


Fig. 5. Storage requirement for a single node in membership list storing mechanism (for different lengths of IDs) and our scheme (considering id 1024 bits and each of other values 1280 bits in length)

B. Computation Costs

Another advantage of our scheme is that, as the entire pre-processing step is done by the base station, the sensors do not need to bother about the calculations and no sensor resource is used for initializing the deployable groups (DGs). The sensors need to use the processing resource only for verification, sensor addition, or sensor deletion process. We found that, in the verification step for calculating the accumulator value, a node takes only about 3 milliseconds when 1024 bit id is used. This processing time is fair enough for such a scheme. Depending upon the size of the id, the processing time varies a little.

C. Security Analysis

Now, let us talk about the security level of such scheme. One-way accumulator uses one-way hash function which means that; given $x \in X$ and $y \in Y$, for a given $y' \in Y$, it is difficult to find an $x' \in X$ such that, $f(x,y) = f(x',y')$. So, an adversary that wants to forge a particular y' would face with the difficulty of constructing an x' with the property that, $z = f(x',y')$. Likewise, in our scheme, the use of arbitrary values for PHV and identity of node cannot pass the membership verification mechanism and the adversary cannot in any way be included in the group. A potential threat is that, if a dishonest member in the group tries to construct a false pair (x',y') such that, $z = f(x',y')$ by combining various node identities (y_j s) in one way or another. However, as mentioned earlier, this is not practical as the adverse node faces the difficulty of finding such a pair. Other methods of generating the pair might be possible. However, this could be handled by restricting the choice of the identities (set of y_j s) of nodes, which is dependent on the decision of the central entity and based on the requirements.

In the pre-processing stage, we use a rigid value of *n*. According to Benaloh and Mare [4], the advantage of using a rigid integer, $n = pq$ is that the group of squares (quadratic residues) modulo *n* that are relatively prime to *n* has the property that it has size, $n' = \frac{(p-1)}{2} \cdot \frac{(q-1)}{2}$ and the function, $e_n(x,y) = x^y \bmod n$ is a permutation of this group whenever y and n' are relatively prime. Thus, if the factorization of n is hidden, "random" exponentiations of an element of this group are extremely unlikely to produce elements of any proper subgroup. This means that repeated applications of $e_n(x,y)$ are extremely unlikely to reduce the size of the domain or produce random collisions. Although constructing rigid integers is somewhat harder than constructing ordinary, '*difficult to factor*' integers, it is still quite feasible.

D. Further Discussion

In this article, we have basically focused on ensuring secure membership of the nodes in the deployed groups in HDSN. Other security mechanisms can run side-by-side our scheme. In our approach, several groups of sensors could operate in the same heterogeneous distributed sensor network at the same time without hampering the operation of the nodes of other group. If required, there could be some other mechanisms of communications between the sinks of different groups or the nodes of different groups. This actually depends on the type of service required from the HDSN or the application at hand. At the time of deployment, if policy-based deployment is used, a certain portion of the sensors could be kept for future deployment. Yet, it would not affect anything in the deployed sensor group and all the other sensors in the DG can still verify each other's legitimacy of membership.

VII. Conclusions and Future Scopes of Research

In this article, first we have proposed a new deployment model of distributed sensor network termed HDSN. Based on the novel model, we then proposed a secure group association management scheme. Our scheme could be employed alongside other supplementary security mechanisms for HDSN. Our analysis shows that this scheme requires considerably very small storage and processing power, and is efficient enough to ensure secure membership of nodes in the deployment groups in HDSN. We believe that our work opens the door for research on other interesting issues in HDSN. For example, handling heterogeneous traffic, prioritized data, maintaining heterogeneous levels of security, quality of service of heterogeneous data, lifetime maximization of deployment groups, etc. could be some of the challenging research issues for HDSN. As our future work, we will analyze our scheme considering various deployment scenarios and will perform a detailed practical experimentation on the efficiency of our approach.

VIII. References

[1] L. Gu, D. Jia,, P. Vicaire, T., Yan, L. Luo, A. Tirumala, Q. Cao, T. He, J. A. Stankovic, T. Abdelzaher, and B. H. Krogh, "Lightweight detection and classification for wireless sensor networks in realistic environments," in *Proceedings of ACM SenSys 2005*, 2-4 November, San Diego, California, USA, 2005, pp. 205-217.

[2] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, D. Culler, "Design of a Wireless Sensor Network Platform for Detecting Rare, Random, and Ephemeral Events," in *Proceedings of the 3rd symposium on Information Processing in Sensor Networks (IPSN'05)*, LA, California, 2005, pp. 497-502.

[3] http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf, accessed 10 October, 2008.

[4] J. Benaloh and M. d. Mare, "One-Way Accumulators: A Decentralized Alternative to Digital Signatures," *LNCS 765*, Springer-Verlag, pp. 274-285, 1994.

[5] Ai, C., Hou, H., Li, Y., and Beyah, R., "Authentic Delay Bounded Event Detection in Heterogeneous Wireless Sensor Networks", *Ad Hoc Networks*, Elsevier, 2008 (in press)

[6] V. P. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, and N. Shroff, "A Minimum Cost Heterogeneous Sensor Network with a Lifetime Constraint," *IEEE Transactions on Mobile Computing*, Vol. 4, No. 1, pp. 4-15, January/February, 2005.

[7] A.-S. K. Pathan, and C. S. Hong, "SERP: Secure Energy-efficient Routing Protocol for Densely Deployed Wireless Sensor Networks," *Annals of Telecommunications*, Volume 63, Numbers 9-10, pp. 529-541, October 2008.

[8] M. Y. Rhee, *Internet Security Cryptographic principles, algorithms and protocols*, WILEY, ISBN 0-470-85285-2, 2003.

[9] A.-S. K. Pathan, G. Heo, and C. S. Hong, "A Secure Lightweight Approach of Node Membership Verification in Dense HDSN," Proceedings of the IEEE Military Communications Conference (IEEE MILCOM 2007), October 29-31, Orlando, Florida, USA.

[10] A.B. Israel and T. N. E. Greville, *Generalized inverses: theory and applications*, John Wiley & Sons, New York, 1974.

[11] T.L. Boullion and P.L. Odell, *Generalized Inverse Matrices*, Wiley-Interscience, New York, 1971.

IX. Biographies

Al-Sakib Khan Pathan is a researcher and PhD candidate at Networking Lab, Department of Computer Engineering in Kyung Hee University, South Korea. He received his B.Sc. degree in Computer Science and Information Technology from Islamic University of Technology, Bangladesh in 2003. Before joining Networking Lab, he was with the CSE Laboratories, North South University, Bangladesh. His research interest includes wireless networking, wireless sensor networks, network security, and e-services technologies. He was awarded as a 'Scholar Student of the Year' (for foreign MS/PhD students studying in South Korea) two times in 2006 and 2007. He also received IEEE ComSoc Seoul Chapter Best Paper Award in JCCI 2007 in South Korea and IEEE Student Travel Grant Award in MILCOM 2006 in USA. He has served (or, is serving) as a Technical Program Committee member in some international conferences like IEEE HPCC'09, IEEE IDCS'08, and IEEE WiMob'08. He also serves as a reviewer of a few renowned journals and magazines such as IEEE Transactions on Vehicular Technology (IEEE TVT), IEEE Transactions on Dependable and Secure Computing (IEEE TDSC), Computer Standards and Interfaces, Computer Communications, Journal of High Speed Networks (JHSN), and IEEE Communications Letters. He is a Student Member of IEEE, member of Korea Information Processing Society (KIPS), and Korean Institute of Information Scientists and Engineers (KIISE).

Choong Seon Hong received his B.S. and M.S. degrees in electronic engineering from Kyung Hee University, Seoul, Korea, in 1983, 1985, respectively. In 1988 he joined Korea Telecom (KT), where he worked on Broadband Networks as a member of the technical staff. From September 1993, he joined Keio University, Japan. He received the Ph.D. degree at Keio University in March 1997. He had worked for the Telecommunications Network Lab, KT as a senior member of technical staff and as a director of the networking research team until August 1999. Since September 1999, he has been working as a professor in the School of Electronics and Information, Kyung Hee University, South Korea. He has served as a Program Committee Member and an Organizing Committee Member for various international conferences such as NOMS, IM, APNOMS, E2EMON, CCNC, ADSN, ICPP, DIM, WISA, BcN, and TINA. His research interests include ad hoc networks, network security, and network management. He is a member of IEEE, IPSJ, KIPS, KICS, and KISS.