A Selective Survey on Key Distribution in Sensor Networks

Dr. Ghassan Chaddoud *

Abstract

Key management in Wireless Sensor Networks (WSNs) is an important issue due to the absence of trusted infrastructures, on one hand, and the limited resources of sensor nodes, on the other hand. This paper surveys some recent key management approaches in WSNs. It first identifies some of the problems that confront the key management. Then, it defines some criteria for viable solutions to key management problems. Next, it explores some of the proposed key management approaches, and analyzes them according to the presented criteria. Some open research issues are discussed.

Keywords: Sensor networks, key Management, Security of Wireless Sensor Networks.

* Atomic Energy Commission, P.B. 6091, Damascus

1. INTRODUCTION

A Wireless Sensor Network (WSN) [1], [4], [9] consists of a large number of tiny devices. Each device, called sensor node or, for short, sensor, is composed of sensing, data processing, memory, and short-range radio communication unit. A sensor is battery powered and has limited computation, storage, bandwidth, and energy resources. In most cases, sensors are randomly and densely deployed in fields with the aim of monitoring ambient conditions or collecting information or readings about certain events or motions. Typical applications include civilian and military environments. More specifically, they can be used for reading temperature, humidity, wind speed, pressure, and tracking objects. In all cases, data are sent, either directly or via other nodes, to an end user or a server, to be treated.

In typical application scenarios, sensors are deployed via aerial scattering in large number in unattended, inaccessible, and often adversarial environment. So, they are prone to many type of attacks [29], [47], [51], [59] For example, an adversary can listen to all traffic, inject packets, impersonate sensor nodes, provide misleading information, or replay older messages. Therefore, security services such as authentication and confidentiality are crucial for the right functioning of WSNs. In order to provide such security services between communicating nodes, keys management is a critical building block. Generally, the set of operations which includes key generation, setup or distribution, updating, and revocation forms the well known *key management operations*.

In WSNs, initiating secure communication between sensor nodes is a real challenge due mainly to the absence of trusted infrastructure. For instance, there does not exist an entity (e.g., a key server) available and reachable at any time by sensors. Even if the environment allows for the existence of such an entity, sensor nodes can not play this role because, as we will see below, they have very constrained capabilities. Therefore, bootstrapping sensor nodes with the required keys for securing their communications is a real issue. Besides, key updating, which is, as we believe, more difficult than initiating nodes with keys, is another issue. This renders key management a real problem and tackling this problem is a necessity for securing communication in WSNs.

In traditional networks, there exist many approaches for key management [12], [14], [31] based either on public key or shared key techniques. However, these solutions are not appropriate for use in their current states in WSNs due to the characteristics of sensor nodes and the operating environment as well. Even those proposed for wireless ad hoc networks [19], [25], [32], [39], [52], [57] are, as we will see later, far from being applicable to WSNs.

Recently, many works have been achieved on key management in WSNs. Most of these works either deal with only pre-deployment key initiating, this is the case for example of [11], [15], [16], [18], [36], or based on traditional methods such as [6], [12], [31], [41], [43]. However, neither one of these works presents a real solution to key management problem in WSNs due to the fact that the first set limits itself to certain applications and did not think about a global solution and the second set is too heavy for implementation on sensors.

We believe that understanding a problem is the half solution and key management in WSNs is not an exception. So, proposing solutions for key management should not be restricted to studying protocols and applications but it must cover the characteristics of sensor nodes, especially their capabilities, and application environment as well. This paper focuses mainly on identifying the issues of key management in WSNs, drawing some evaluation metrics, presenting some of the proposed key management protocols, and comparing these protocols with regard to the introduced evaluation metrics.

The paper is organized as follows. Section 2 will present the characteristics of sensor networks. Then, we introduce some evaluation metrics for key management in WSNs in Section 3. Next, we describe some of key management approaches in Section 4. Finally, we conclude in Section 5.

2. SENSOR NETWORK CHARACTERISTICS

In this section, we outline some unique specifications of sensor nodes. Then, we discuss the topology and architecture of sensor networks. We list next the most used communication patterns defined inside networks. We describe the trust relationship defining security restrictions. Then we explain why key management approaches of ad hoc networks can not be used in WSNs. Finally, we present the security services required to ensure secure communications in WSNs.

A. Sensor characteristics

A sensor node is the basic component in WSNs and this component is featured by some special characteristics that make it different from any node in traditional, especially wireless ad hoc, networks. The sensor node is designed for ease of deployment and to be low cost, compact, lightweight, and disposable [9]. Further, it is a very tiny device with a very limited energy capability. So, knowing the sensor's features prior to the design of any network protocol, especially those security-related functionality, is a priority.

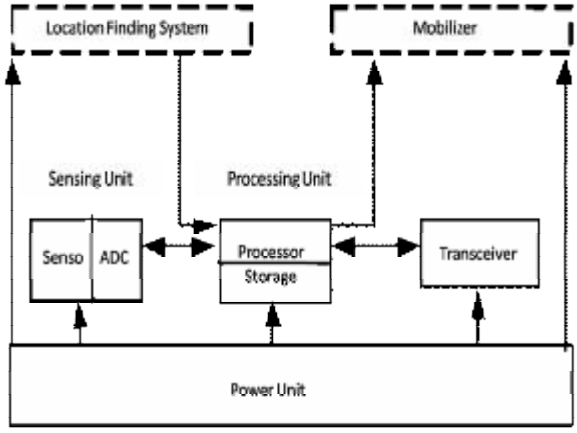

Figure 1. The components of senor node

A Sensor node consists of four basic components [4]: sensing, processing, transceiver, and power units as illustrated in figure 1 (redrawn from [4]). The sensing unit is made up of two subunits: sensor and Analogue to Digital Converter (ADC). The sensor subunit is responsible for sensing events (seismic, acoustic, magnetic, etc.) and produces analog signals that the ADC treats and hands over to the processing unit. The processing unit is equipped with a small storage unit (in the order of hundreds or thousands of bytes). It is responsible for the management of procedures that qualify the node for cooperation with other nodes in order to carry out the sensing task. Therefore, the processing unit processes data produced by either the sensing unit or other sensor nodes. The transceiver unit connects the node with other nodes. The power unit is the most important unit of a sensor node and it is, in general, a battery with a limited capacity. This explains the reason why any network functionality should take energy conservation into account.

These units are mostly common in sensor nodes, however, some nodes need to have some special capability such as mobility and location system which depend on the application. Therefore those sensors may be equipped with additional units.

Recently, many sensor node prototypes [8], [28] have been developed. These prototypes facilitated a lot the comprehension of nodes' capabilities in WSNs and were of such importance for developing the most fitted protocols. For instance, the *Smart Dust* mote prototype [28] is the most representative sensor node architecture. It is a 4 MHz Atmel AVR 8535 micro-controller with 8 KB instruction flash memory, 512 bytes RAM and 512 bytes EEPROM. The operating system used on this mote is TinyOS [17], which has 3500 bytes code space and 4500 bytes available code space.

From constrained processing and memory capacity of these nodes, it is legitimate to infer the following. Firstly, sensor memory can not fit any protocol. Therefore, not any protocol designed for traditional network can be used in WSNs. For example, this memory space might not be enough to even hold the parameters of an asymmetric primitive [46]. Therefore, the use of public key-based security services, such as Diffie-Hellman key agreement [14] and RSA

[50], is not suitable for WSNs. Secondly, sensor nodes can not be capable of carrying out extra task (except sensing, treating raw data and aggregation) such as playing the role of key management server. Thirdly, a node may vanish at any moment due to battery depletion. As a result, sensor nodes may not be able to participate in achieving a task for energy-related reasons. Moreover, re-deployment of new nodes must be envisioned in order to replace malfunctioning or vanished nodes

For these reasons, key management task can not be the responsibility of sensor nodes, and thus another means should be sought instead. Further, storage space needed to store keys should be within the limit of sensor storage units. Furthermore, protocols, especially security-related ones, should be carefully analyzed with regard to nodes' limits. Here it is not enough to talk about the traditional performance metrics such as complexity, rigidity, etc., instead, we have to keep in mind that each bit (of code, parameters, or data) needs to be treated, transmitted, or stored. In other words, it needs mainly memory space and energy.

B. Sensor networks topology

WSNs consist of a very large number (hundreds, thousands, or even hundreds of thousands) of sensors that are densely deployed in a scrutiny environment with the aim of carrying out specific tasks. In general, sensors are deployed either by physical installation (for example, along a road or field) or randomly (for example via aerial scattering) without any prior knowledge about their neighborhood. The latter case seems the most interesting and frequent, hence, will be considered in our study. If we consider this last case, sensors must possess self-organizing capability in order to cooperate with each other as well as in-network data processing for resources conservation.

Generally, sensors are disseminated with the aim of collecting readings about an event and then sending them back to a *sink* or *base station.* Intermediate nodes may process data collected or received from others. Sensors treat or aggregate raw readings before transmission in order to get rid of redundant data, hence, reduce transmission cost (energy and bandwidth) due to the fact that processing one bit is hundred of time cheaper than transmitting it [9]. Then, data are sent by multihop wireless communication. The *sink* is, if it is not the end user, a gateway to a server via Internet or Satellite (Fig. 2). The *sink* could be considered as a special node in WSNs with laptop-class capabilities.

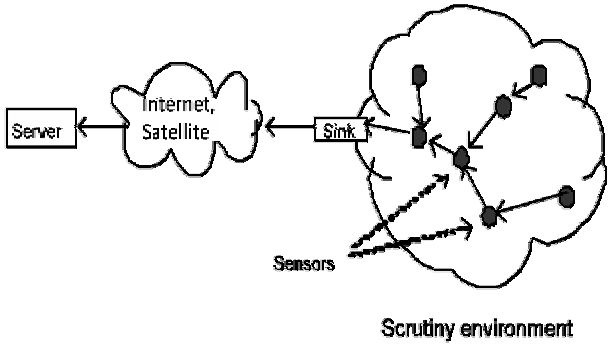

Figure 2. A representative WSN topology

Furthermore, depending on the application and deployment field, the sensor nodes or *sink* are either stationary or mobile. There might be four different configurations:

1) **Stationary nodes and stationary *sink***. This is the case for example of sensors deployed in forests for fire detection, where nodes are connected to a collection data server via a gateway (i.e., a *sink*).
2) **Stationary nodes and mobile *sink***. For example, sensors are deployed in a field and the *sink*, an embedded system, overflies the scrutiny area for collecting data.
3) **Mobile nodes and stationary *sink***. For instance, Sensors with mobility capability may be deployed in critical terrains and roam around a collection point.
4) **Mobile nodes and mobile *sink***. This case may happen when sensor nodes move along a *sink* which is an embedded system.

Therefore, it is legitimate to assume that a node may fall out of communication range of the

other nodes. Due to this assumption, node mobility can not be negligible. This means that node motion may make nodes not reachable at any time. In summary, WSN topology can not be assumed to be static due to change in node position, reachability, availability, and task details.

C. Communication models

Depending on the application, sensors report readings to the sink either periodically, occasionally (*i.e.*, when available), or as an answer to a query. The sink queries a group of sensors localized in a zone, or a sensor node broadcasts or unicasts messages to its neighbors. Readings propagate from one node to another until arrival to the destination, therefore, communication pattern in WSNs can fall into one of the following three types [2]:

- Many-to-one: Sensors send readings to a sink or an aggregation point.
- One-to-many: The sink broadcasts query or control information to a set of sensors.
- Local communication: Nodes broadcast messages to their neighbors with the aim of discovering or coordinating with each other.

Furthermore, readings are assumed to be treated by intermediate nodes, *i.e.*, in-network processing or aggregation. As a result, secured (*i.e.*, encrypted or authenticated) readings must be treated on node-to-node, but not end-to-end[1], basis. From key management perspective, this means node's keys must be shared and used mostly between neighboring sensor nodes.

D. Trust models

Similar to any wireless networks, communications in WSNs are not secure due to the used broadcast paradigm. In fact, an adversary can not only react actively or passively on traffic (means eavesdropping on messages, inject new messages, replay old messages, etc) but can also take control over a sensor node. This latter case is more serious than the former ones because when security services are applied, the adversary could misuse the keys in possession of the captured node. Further, the adversary becomes an integrated part of the security model. However, this case of captured node may be isolated when some one discover the compromise of a node.

The worst case is when the compromised node becomes captured and it has in possession keys that are used to secure communication amongst other nodes. This the case, for instance, of using random key pre-deployment schemes such as [15], [16], [36] (see Section 5). For this reason, the key management scheme must ensure network resiliency where resiliency is defined as the fraction of total network communications that are compromised, by the capture of some nodes, not including the communications in which the captured node is directly involved [11]. So, a sensor node should not be considered as a fully trusted entity. Even aggregators, the nodes that act as aggregation points, should not be trusted and hence, security measures must be taken with the aim of protecting aggregation process [13], [47].

Still, from functionality point of view, we have the right to consider that the sink or the base station is part of the WSN, even it is not a sensor node alike. Moreover, since that entity is a collection point of readings and a gateway to the outsider world, we believe that we can assume that the base station can be trusted in such a way that it can be responsible for key management operations.

E. Sensor networks vs. ad hoc wireless networks

Sensor networks are similar to ad hoc networks in nature; i) broadcast wireless communication, ii) sporadic nature of connectivity, iii) limited physical protection of each of the node, iv) lack of a centralized monitoring or management point, and v) limited resources. However, sensor networks present properties beyond those of ad hoc networks. In particular [29]:

- Limited resources: Ad hoc networks were generally considered to have limited

[1] Although most of the applications focus on node-to-node communications, some recent works tried to address the problem of globally addressable nodes [30], [42], [48], [49].

resources, but sensors are more constrained. Energy resources is the most serious restriction.

- In-network processing: Sensor networks are deployed for scrutiny purposes. Many nodes may sense the same event, and send reports upstream to the sink. Therefore, redundant messages can be avoided by in-network processing (*i.e.*, aggregation, duplicate elimination, etc.). This is necessary to reduce traffic and hence conserve energy and bandwidth resources.
- Node-to-node Communication model: Ad hoc networks support communication routing [27], [44], [53] between any pair of nodes, i.e., network-wide addressable nodes (*e.g.*, IP based addresses), whereas it is common for sensor networks to support location-based routing [2], [23], [24], [26], [33].

For these reasons, the security approaches that have been proposed for authentication and key management [5], [19], [25], [32], [39], [52], [57] in ad hoc networks are not applicable to sensor networks.

F. Security services

With regard to the aforementioned communication patterns, many security services are required:

- *Confidentiality*: Messages containing sensitive data such as readings and cryptographic keys need confidentiality which is ensured usually via encryption. Hence, secure channel (via encryption) must be thought of. The confidentiality must be ensured between neighboring nodes and between the base station and each node. Many-to-one model needs unicast shared keys. Moreover, group keys are required for broadcast readings.
- *Authentication*: Control -or management-related messages require authentication. Here we distinguish between two types of authentication: Mutual authentication (between two parties) and broadcast authentication. The first one can be realized through symmetric mechanism: the sender and receiver share a secret key to compute a Message Authentication Code (MAC) of all communicated data. This mechanism can not be applied in broadcast authentication because any node can impersonate the sender and forge messages to other receivers. In traditional networks, broadcast authentication is achieved via asymmetric mechanisms which can not be realizable in WSNs. So, other means should be sought. Perrig et al. designed a new symmetric mechanism μTesla [46] inspired by Tesla [45] and based on delayed key disclosure and one-way function key chains.
- *Integrity*: Integrity ensures the receiver that data is not altered in transit. It is usually achieved via data authentication.
- *Freshness*: Data freshness ensures that data is fresh or recent, as well as no adversary replayed old messages. It could be realized via the use of synchronization mechanism, nonces or counters.

With regard to the communication patterns presented in Section 2.3, there is a need to ensure mainly confidentiality and authentication for unicast and broadcast communication. Applying these services require mainly cryptographic keys (unicast and multicast or group keys) as well as the suitable security primitives. We emphasize that unicast keys can be used to perform authentication and confidentiality for unicast and broadcast, whereas group keys are used only for confidentiality; broadcast authentication can not be achieved via group keys because nodes are not trusted. Broadcast authentication is a serious issue and should be resolved independently from key management.

Therefore, WSNs security could be divided into three levels:

1) **Key management** for unicast and group keys that are required to ensure confidentiality and unicast authentication.
2) **Cryptographic primitives** that provide security services, mainly confidentiality and unicast authentication, and fit sensor node resources.

3) **Broadcast authentication**.

In the rest of this paper, we'll focus on the first level; key management.

3. EVALUATION CRITERIA

In general, key management protocols in traditional networks are assessed based on some evaluation metrics such as complexity, ability to ensure secrecy, and performance efficiency. However, protocols in WSNs should meet some criteria specific to sensor network characteristics (discussed earlier in Section 2). So, we propose the following criteria for key management evaluation in WSNs:

1) **Independency:** As mentioned earlier, sensors are not able to manage keys themselves so, this task must be in responsibility of an outsider entity, let us call it key server. Here we can distinguish between two types: offline and online server. The offline server initiates sensor nodes with the required keys prior to deployment. In the second type, the server distributes keys after deploying the nodes. We stress here that offline servers are not able to interact with sensors once deployed, whereas online servers can communicate with nodes at any time. However, with or without online server, nodes should be able to bootstrap secure communication and update keys. So, the independency can be seen as the ability of WSNs to operate without online servers.
2) **Resiliency:** Key management schemes must be resilient against node capture because nodes may possess keys shared with many others such as in random key schemes [11], [15], [18]. If an adversary takes control over multiple nodes in a region, the whole sensing task may be jeopardized.
3) **Dynamicity:** New nodes must be able to join the network or others must be able to leave or vanish. Extra nodes may join the network either to replace vanished nodes or to carry out extra task, or others may leave or vanish for cheating or battery depletion.
4) **Mobility:** Regarding WSNs, any key management protocol should be able to stand up to network dynamism where the nodes or sink are mobile. Node move produces a change in neighborhood and hence a change in shared keys with neighboring nodes. Unused keys in possession of the mobile node must be destroyed in order to prevent memory saturation. Besides, mobile nodes must be able to negotiate or discover shared keys with new neighbors.
5) **Memory-Fitness:** Memory space required to store keys or intermediate parameters should be as reduced as possible. We have to keep in mind that in addition to the keys, the operating system, security primitives, and network application must be within the sensor's memory capacity.
6) **Mutual Authentication:** In key set up, nodes should authenticate each other in order to prevent unauthorized entities from gaining access to the network.
7) **Energy-Awareness:** Since all sensors are at the risk of battery depletion, sensor nodes must not be busy all the time carrying out key management-related operations, especially when they are directly concerned by the key itself. So, protocols should adopt their behavior in a similar manner to some of energy aware routing protocols such as [3].

In the remainder of this paper, we will present and analyze some of the proposed key management approaches and compare them with regard to these criteria.

4. KEY MANAGEMENT: STATE OF THE ART

There exist three types of key distribution methods [15]:

1) *Pre-deployment scheme:* Key information is distributed to sensors prior to deployment.

2) *Trusted-server scheme:* This type depends on a trusted third party that is used as key management server.

3) *Self-enforcing scheme:* Schemes depend on asymmetric cryptography such as key agreement using public key certificate.

With regard to the first type, if neighborhood is known in advance (this is the case of physical installation) prior to deployment, keys can be loaded into sensors and pre-distribution schemes work fine. However, since in most of applications, sensors are disseminated randomly, knowing the set of neighbors deterministically might not be feasible. This type is called probabilistic, *i.e.*, two nodes can establish a secure connection between them with a given probability.

The second type is not very suitable for sensor networks because there is usually no trusted infrastructure in WSNs because, as mentioned earlier, there does not exist a sensor node that has the capacity to play the role of key server. However, This type can still be used when the trusted server is an outsider entity connected directly to the WSN. Most of the proposed approaches relied on the *base station* or *sink* to be responsible for key management operations.

As for the third type, Self-enforcing scheme is not very convenient for WSNs due to the characteristics of sensor nodes aforementioned in Section 2. Further, except some works [20], [40] which concentrated on the design of public key cryptographic primitives that suit sensor nodes and belong to the third level, Cryptographic primitives, this type has not been approached yet as a base for key management in WSNs.

Most of the work done in this field focused on the first and second types.

5. PRE-DEPLOYMENT SCHEMES

The naive solution for secure communication is to distribute a key, called a master or mission key, to all sensors prior to deployment: any pair of nodes can use this key to achieve key agreement and obtain a new pairwise key. This scheme suits well the limited resources of WSNs, however, it does not exhibit any network resiliency [18]. If any node is captured, the entire network security will be compromised.

Another pre-distribution scheme is to let each node to store $N-1$ (N, network size, *i.e.*, number of sensors) secret pairwise keys, each is known only to this node and to one of the other $N-1$ ones. The resiliency of the scheme is perfect because compromising one node does not affect communication of uncompromising nodes and it has a zero energy cost and latency as well. However, adding new nodes after deployment is difficult because existing nodes do not have the keys of new nodes. Besides, it does not suit sensors due to the large amount of memory needed to store the $N-1$ keys.

The alternative is a compromise between this two schemes, and it is based on a probability distribution, *i.e.*, give nodes a certain number of keys, later a node can communicate securely with a neighbor with a given probability p. This probability determines the number of keys given to each node.

In the following, we can distinguish between two categories: Random Key distribution and Pairwise Key distribution. Each decomposes of three phases: key setup, key establishment, and path key establishment. First, in the pre-deployment phase, which is the key setup, an offline server distributes keys or information about keys to the sensors. After deployment, each pair of neighboring nodes try to create a pairwise or a shared key between them in a direct way if it is possible. Otherwise, in the path key phase, they try to establish shared keys by the intermediate of other nodes.

A. Random Key Pre-distribution

1) *Basic Scheme*: Eschenauer and Gligor [18] were the first to propose a random key pre-distribution scheme for key management in WSNs. Their scheme is based on probabilistic key sharing between the nodes of a random graph. The scheme consists of the following phases: *key initialization*, *shared-key discovery*, and *path-key establishment*. Contrary to the former phase which is achieved offline (*i.e.*, prior to deployment), the latter two phases are accomplished online, after sensors deployment.

Key initialization: The scheme generates a pool P of $|P|$ keys, where $|P|$ is the number of

keys in *P*. Then, for each node, *m* keys are randomly picked, without replacement, from *P* and loaded into the node's memory. The set of *m* keys given to the node are qualified by its key ring. |*P*| and *m* are selected in such a way that two nodes share at least one key in a given probability *p*.

Shared key discovery: This phase takes place right after nodes deployment. During this phase, a node discovers its neighbors, within its wireless communication range, with whom it shares keys. This means that the intersection of their key rings is not an empty set. This phase establishes the topology of the network where a link exists between two neighboring nodes if they share at least one key and this key is used to secure communications between them. Moreover, once the shared-key discovery is finished, a connected graph *G*(*n*, *p*) (where *n* is the number of nodes and *p* is the probability that a link exists between two nodes) of secure links is formed. We stress here that a shared key is not pairwise because it may be owned by other nodes.

Path-key establishment: This phase is used to establish secure paths between neighboring nodes which do not share keys in common but they are connected via one or more nodes with whom they have a secure link. For instance, if A, B, and C are 3 neighboring nodes and there exist secure links between A and C and between B and C and there is no link between A and B. A and B can use C to establish a path key between them.

This scheme allows for an external server (*i.e.,* offline server) to add and revoke nodes after deployment. Deployment of new nodes happens in the same way as in the initial node deployment. The offline server provides the new sensor with a key ring picked from *P*. Then, once the node is in field, it integrates the network by carrying out phases 2 and 3. As for the revocation, it is reduced to revoking all keys in its key ring from other nodes' ring. This is achieved by a controller node which is an external node with a large communication range. First, the controller unicasts to each sensor node a signature key encrypted with pairwise keys shared with the controller. Then it broadcasts a signed message containing the list of identifiers of keys to be revoked. Next the sensor nodes locate these keys in their ring and remove them.

As we remark, the scheme depends on an offline server and controller nodes. The former one is responsible for initializing nodes with key rings before and after (*i.e.,* node join) deployment. The latter one is responsible for node revocation.

This scheme seems to be appeal because it allows sensors to share keys with neighbors without additional computational overhead for key sharing (during shared key discovery and path-key establishment phases). However, the communication overhead required to realize key sharing has a negligible effect on sensor resources because this operation is not carried out only one time. Sensors need to do it permanently in order for the graph to be kept connected. Establishing shared keys with neighbors should take place in many cases such as node join, node revocation, node's battery depletion, especially when nodes are mobile. It is true that the former cases are not very frequent, but the latter one depends on node mobility. Moreover, mobility means the connected graph is in constant change. Hence, not only nodes recently connected with the mobile node have to perform key sharing but also the nodes which were connected with this node must carry out key sharing. Node memory would be saturated quickly if they do not remove the shared keys not belonging to the key ring (keys that were established via path key establishment).

Further, the simulation shows [18] that for a pool of 10,000 keys, only 50% of the keys are used to secure links, 30% are used to secure one link, 10% are used to secure two links, and only 5% are used to secure 3 links. So, a half of the memory space used to store the key ring is wasted. For example, if the key ring is 100 keys and each key is 16 bytes this means that 800 bytes is wasted. This wasted part is not insignificant of the sensor memory, knowing that there is only 4500 bytes in SmartDust mote [28] of available memory space. Surely the case is worst when the key ring size is bigger.

Furthermore, the comprise of a node leads to compromising links between other nodes. The number of affected nodes or links depends on the size of key ring and key pool. For example, according always to the simulation results [18], the compromise of one key leads to the compromise of another link with a probability 0.3, of 2 other links with probability 0.1, and so on. This means that the scheme is not enough resilient because the capture of 0.5% of nodes leads to the compromise of approximately of 10% of the links. And thus the scheme is not enough resilient against attackers. To improve the resiliency of this scheme, Chan et al. [11] imposed the use of a combination of many keys instead of one to secure links.

2) *q-composite key random pre-distribution*: Chan et al. proposed in [11] another random Pre-distribution scheme called *q-composite key random pre-distribution scheme*. This scheme is similar to the basic scheme in the way that it is initialized offline, online shared key discovery and path-key establishment phases. However, the only difference emanates from the way they define a link or a secure link and the pool size. As we noticed in the basic scheme, a secure link is defined between 2 neighboring nodes if they share at least one key, whereas in this scheme a secure link is defined as if they share at least q keys, where $q > 1$. The key K used to secure link between two nodes is defined as follows: let $q' > q$ is the actual number of keys shared between the two nodes, $k_1, k_2, ..., k_{q'}$ are the actual keys shared between them, and h is a hash function, then K is computed as follows: $K = h(k_1||k_2||...||k_{q'})$.

This scheme improves network resiliency against node capture for only a small number of compromised nodes. In fact, the security of the network depends on q. The bigger value of q the stronger security we have. However, increasing q depends on either decreasing $|P|$, the pool size, or increasing $|m|$, the key ring size, for a given network size. The first one means starting from a certain number of compromised nodes, the whole network security may become at risk. The second one means that key ring occupies more space in sensor's memory.

3) *Knowledge-based key pre-distribution*:

Knowledge- based key pre-distribution scheme [15] is based on the basic scheme in the way that key distribution is done prior to sensors deployment. However, it improves the performance of the basic scheme by reducing the quantity of keys or the key ring size via the use of pieces of information available prior to network deployment. This piece of information is called *node deployment knowledge*. Node deployment knowledge gives an idea about where a node is more likely to reside after deployment. In other words, it gives an idea about the neighborhood of nodes. Keys that are assigned to a node should be shared only with possible neighboring nodes. In this way, a node does not share keys with others that are far away from the node's actual position.

To understand how the deployment knowledge is used in this scheme, let us go back to the basic scheme. In this last one, sensors are scattered randomly which means that there is no knowledge about their actual position or the sensor node may be anywhere in the scrutiny field. In other words, sensors are deployed according to a non-uniform distribution. For instance, in a two-dimensional rectangular region with a size $X * Y$ and the upper left corner is its origin, the *pdf* (probability distribution function) for the location of a node i is given by $f_i(x, y) = 1/xy$, where $x \in [0, X]$ and $y \in [0, Y]$.

As for the knowledge-based scheme, it assumes that the rectangular area is divided into subregions, in each a subgroup of sensors are deployed. The distance between the *deployment point*, the point where sensors are spread, and the actual position of the sensor node i is determined by the *pdf* correspondent to that subregion. We stress here that the *pdf* may be identical for all subregions or differ from one subregion to another. Keys assigned to sensors in a subregion c are picked, without replacement, from a subset S_c of the key pool P. S_c is chosen in such a way that keys assigned to a sensor in the subregion c are common only with sensors situated in neighboring subregions.

Figure 3 (redrawn from [15]), illustrates sensor deployment over a rectangular region 600*600, and the key subset correspondent to each subregion. Each dot, black point, in figure 3

(a) represents the deployment point of a subset of sensors. Figure 3 (b) explains how key subset for subregion E are chosen as a combination of the subsets of neighboring subregion's key pools. If S_i is the key pool of subregion *i*, key pool of E, S_E, is composed of *a* keys of each of S_B, S_F, S_D, and S_H, and *b* keys of S_A, S_C, S_G, and S_I , where *a* and *b* are overlapping factors. Hence, sensors in subregion F share keys only with their direct neighbors in subregions A, B, C, F, I, H, G, and D.

Comparing this scheme with the basic and q-composite schemes, the knowledge-based scheme improves memory usage and network's resilience against node capture. However, this scheme has many drawbacks. To start with, it is developed only for aerial dissemination in a strict order of sets of sensors. The question is: is it really feasible to realize such a scattering in such an order? Secondly, it is based on static nodes, so it is not suitable in dynamic environment where nodes may need to move around. Third, it is fragile to any type of oriented attack, *i.e.*, if an attacker takes control over a small number of nodes in a subregion, there is a big possibility that the whole key pool of that subregion be under control of the attacker.

Similar to this scheme, Liu and Ning developed another deployment knowledge scheme [37] but instead of using secret keys, they use secret polynomials.

B. Random pairwise scheme

1) Random Pairwise Key Pre-distribution: Chan et *al.* proposed this scheme in [11] in order to reduce the amount of unused keys stored in a node, ensure a node-to-node authentication[2], and improve network resiliency. Similar to random pre-distribution scheme, this scheme consists of three phases: *pre-deployment initialization*, *key setup*, and *multi-hop range extension.* Let *m* be the node's key ring size and *p* the probability of any two nodes being able to communicate securely.

Pre-deployment initialization: This scheme generates $n = m/p$ node identities. Then for each node, it randomly picks m distinct node identities. Next, it generates to each pair of identities a pairwise key. Finally, the key is stored into the memory of both nodes along with the identity of the other node which knows this key.

Key setup: In this phase, nodes broadcast their ID to their neighbors. Each node in the neighborhood looks up the received keys correspondent to received IDs in its key ring. If found, then a cryptographic hand shake is then carried out to verify their knowledge of the key.

Multi-hop range extension: The purpose of this phase is to extend the key setup beyond the effective communication range of nodes. When a node receives a key setup message, the node re-broadcasts it for a given number of hops. This means that nodes can find nodes sharing keys out of their neighborhood (*i.e.*, out of their communication range).

This scheme presents many advantages over random pre-distribution schemes presented thus far. For example, it ensures resiliency and mutual authentication. Indeed, captured nodes reveal

[2] Any node must be sure about the identity of the nodes that it is communicating with

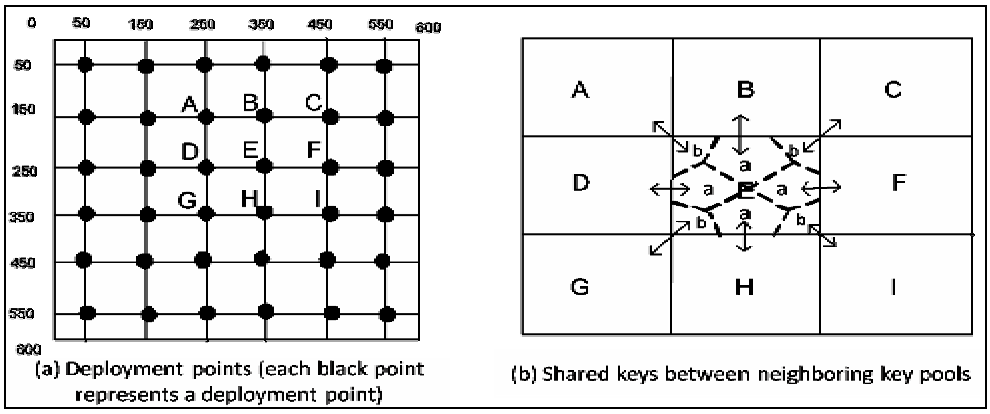

Figure 3. Knowledge-based key pre-distribution over a rectangular region 600*600

nothing about the links that are not directly involved in. Thus, it presents a perfect resiliency against node capture. In addition, it allows for nodes connected by secure links to reciprocally verify their identity.

However, it does not present any improvement to memory usage over the first schemes; a node still stores unused keys, *i.e.*, keys shared with nodes that are not in neighborhood and hence it wastes memory space.

In spite of the fact that it can not be considered a mature solution for key management in WSNs, it is still an alternative for key bootstrapping.

2) Matrix-based pairwise scheme: This scheme relies on the basic scheme [18] and Blom's key pre-distribution scheme [7].

Blom's key pre-distribution Scheme: Blom scheme allows for any pair of nodes in a network to find a pairwise key between them. It works as follows: first a server creates a $(\lambda + 1) * N$ matrix G over a finite field $GF(q)$, where N is the size of the network, q is a large prime number, and λ is a threshold. λ defines the following property: as long as an adversary compromises λ nodes or fewer, non-compromised nodes are perfectly secure; when an adversary compromises more than λ nodes, all pairwise keys of the entire network will be compromised. The former property is called λ *–secure property*. G can be made public. Then, the server constructs a random symmetric $(\lambda + 1) * (\lambda + 1)$ matrix D and computes a $N * (\lambda + 1)$ matrix $A = (D.G)^T$, where $(D.G)^T$ is the transpose of $D.G$. D should be kept secret. Let $K = A.G$. We should notice that K is symmetric. Let $A(i)$ be the i^{th} row in A and $G(i)$ the i^{th} column in G.

In a pre-deployment phase, the server loads $A(i)$ and $G(i)$ to node i. Remind that $A(i)$ is a secret value known only to i and G is public. Later, two nodes i and j can compute a pairwise key if they exchange the value of their respective column in G. For instance, i sends to j $G(i)$ and j sends to i $G(j)$. Then any one can compute their pairwise key by evaluating $k=A(i) * G(j)= A(j) * G(i)$. If G is a Vandermonde Matrix[3] then each node needs to store $(\lambda + 1)$ elements.

The Matrix-based pairwise scheme is no more than a random key pre-distribution scheme but instead of using a pool of keys, it uses a set of *key spaces*. A key space is by definition a tuple (D, G), where matrices D and G are as defined in Blom's scheme. And each node randomly picks, without replacement, (D_τ, G_τ), where $\tau \leq W$. W is number of the key spaces. Any two nodes can compute a pairwise key if they pick the same key space. Moreover, each node needs to store $(\lambda + 1)\tau$ elements.

This scheme seems attractive because it offers the same $(N - 1)$ pairwise key scheme property for much less storage capacity. It needs only $(\lambda +$

[3] A Vandermonde Matrix is an $n * n$ matrix where the j^{th} column is a vector $(x_1^{j-1}, x_2^{j-1}, ..., x_n^{j-1})^T$ for $j = 1, 2, ..., n$, where $x_1, x_2, ..., x_n$, is a set of elements.

1)τ memory units (a memory unit is the space required to store an element of $GF(q)$). Hence, there is no memory wasted by storing unused keys. Further, required memory space is independent from network size; it depends only on τ. This means that the security property is independent from memory storage whereas in the other schemes it depends on memory storage and, hence, for a given probability, the network size is limited by the memory space dedicated to key storage. Huang et al. [22], and Yu and Yong [55], [56] reduced the required memory space by using deployment knowledge.

However, due to the high number of multiplication operations modulo q needed to compute the pairwise keys for each neighboring node, this scheme needs more computational power than the previous ones. This might not be an issue if the keys are computed once in the case of static nodes. Nevertheless, this becomes a real issue if nodes are mobile.

3) Polynomial pool-based pairwise scheme:

The main idea is to establish a pairwise key between sensors on the basis of a polynomial-based key pre-distribution protocol which works as follows: First, the setup server generates a bivariate t-degree polynomial $f(x, y) = \Sigma^{t}_{i,j=0}\, a_{ij}\, x^i\, y^j$ over a finite field F_q , where q is a prime number large enough to accommodate cryptographic keys, and $f(x, y) = f(y, x)$. Then the server attributes to each sensor i *a polynomial share* $f(i, y)$, which is a secret value known only to i. Any two nodes can compute their pairwise key by evaluating their shares with the value of the other peer (*i.e.*, node j computes $f(j, i)$ and i computes $f(i, j)$). So i and j are able to find the same key. In this protocol, each node needs to store a t-degree polynomial $f(i, y)$, *i.e.*, $(t + 1)$ memory units.

Based on the polynomial-based pre-distribution and similarly to matrix-based scheme, Liu and Ning [36] developed a scheme for initializing nodes with pairwise keys. Instead of working on a pool of key spaces, the new scheme works on a pool of bivariate polynomials to help establishing pairwise keys between sensors. Indeed, in the setup phase the server first creates a pool F of bivariate t-degree polynomials over a finite field F_q. Then for each node i, the server picks a subset of polynomials $F_i \in F$ and assigns to the node i, the correspondent shares. So, replacing key spaces by bivariate t-degree polynomials, we get the same matrix based scheme.

This scheme proposes two ways to assign the subset of polynomials to the nodes. The first one consists of randomly picking F_i, hence it is called *random subset assignment*. In this way each node needs $F_i * (t + 1)$ of storage space. The second way of assignment is called *grid based assignment*. This way of assignment consists of creating an $m * m$ grid of $2m$ bivariate polynomials $\{ f_i^c(x, y), f_i^r(x, y)\}_{i=1,\ldots, m-1}$, where $m = N^2$ and N is the network size. Each row i is assigned with a polynomial $f_i^r(x, y)$ and each column j is assigned with a polynomial $f_j^c(x, y)$. Moreover, each node is assigned to an intersection $< i, j >$ in the grid. Next, the node in $< i, j >$ is given the shares $f_i^c(x, y)$ and $f_j^r(x, y)$. Thus, each node needs to store $2(t + 1)$ in its memory. The grid based assignment guarantees that any node can easily know if it is possible to establish a pairwise key with a given node.

The polynomial pool-based scheme needs to evaluate the bivariate polynomial at a given point. Thus, nodes have to carry out t modular multiplications and additions in the finite field F_q, where q should be bigger than 64 bits which is usually much larger than word size in sensor processor. Furthermore, nodes may require to regularly update the new state of nodes in its neighborhood.

C. Analysis

In general, the schemes presented this far require to be initialized by offline servers and need external servers to participate in node revocations. Moreover, they do not scale well because they impose constraints on network size due to the fact that avoiding the increased number of compromised nodes requires the augmentation of either the key ring (in random schemes) or key spaces (in pairwise schemes). This means augmenting the storage space. [58] and [38] improved memory usage by exploiting a

group-based pre-deployment model where nodes are supposed to be deployed in groups.

Further, the random schemes are not resilient against node capture because keys are held by many nodes. As for the pairwise scheme, the matrix- and polynomial-based methods are resilient as far as the number of compromised nodes is equal or less than a threshold λ (in matrix-based) or τ (polynomial-based), whereas the basic pairwise scheme is perfectly resilient against node capture but it wastes memory usage.

Furthermore, in these pre-deployment schemes, authentication is still a real issue because in most cases it is based on shared keys. For instance, in random pre-deployment, a key may be held by many nodes, so nothing guarantees that a node deals really with a non-compromised one.

Still, these schemes are attractive for secure communications initialization amongst sensor nodes. But, they are not a mature solution for key management.

6. Comparison

Table 1 summarizes the main features of the protocols presented thus far with regard to the criteria introduced in Section 3. The *q*-composite scheme is omitted because it has the same properties as the basic scheme. Also, we let down the polynomial-based scheme because it shows the same behavior of the matrix-based scheme. We regrouped the three trusted third party-based schemes, namely Kerberos, certificate-based, and identity-based in only one called Trusted-server.

Pre-deployment schemes depend on external controllers, which would not be available online when needed, for performing mainly node revocation. So, the efficiency of these approaches depends on the availability of such entities. As for knowledge-based scheme, it does not support neither node mobility nor dynamicity. The trusted-server scheme can not work without an online server (*i.e.*, key server).

Random pair-wise and trusted-server scheme present perfect resiliency and mutual authentication, whereas the others do not. Also, all methods can cope with network dynamicity, but knowledge-based scheme. In the latter one, new node deployment is not possible due to the specificity of the initial dissemination. Moreover, it will not be able to deal with node mobility and does not allow for mutual authentication. However it improves memory usage over other pre-deployment methods.

Although trusted-server schemes are best evaluated according to our evaluation criteria they are not ideal for key management because their efficiency depend on many factors such as the availability of the online servers, the number of hops between key servers and nodes, and network density. So, we believe that these schemes must be used carefully and after taking these factors into account.

Remark: We omitted the criterion *Energy-Awareness* from Table 1 because, to our knowledge, there does not exist any key management work which considers energy conservation in WSNs.

	Independency	Resiliency	Dynamicity	Mobility	Memory	Mutual Auth.
Basic scheme	Ext.controller	No	Offline	Yes	No	No
Knowledge-based	-	No	No	No	Yes	No
Random pairwise	Ext.controller	Yes	Offline	Yes	No	Yes
Matrix-based	Ext.controller	No	Offline	Yes	No	No
Trusted-server	Online	Yes	Yes	Yes	Yes	Yes

Table 1. Comparison of WSN key management protocols. In the column *Independency*, "Ext.controller" means that methods need external controllers for performing updating operations, "Online" tells that methods rely on online servers, and "-" means that the scheme is independent from any other entity or does not carry out post-deployment operations. When schemes are resilient, they are tagged by *Yes* in the column *Resiliency*, otherwise they are tagged by *No*. In the *Dynamicity* column, "Offline" means that an offline server manages node join and revocation operations, "No" means that scheme can not perform update operations, "Yes" means that update operation is supported without any restriction. When a scheme can handle node mobility, "Yes" appears in the column *Mobility*". In the columns *Memory* and *Mutual Auth*, "Yes" means that schemes can deal with memory constraints and ensure mutual authentication.

7. Conclusion and Future Work

In this paper, we discussed the problems that key management encounter in WSNs. Most of the problems are related to the constrained sensor resources and the absence of trusted infrastructure. Also we introduced some criteria that are, in our view, best suited for evaluating key management methods. Then, we explored some of the proposed solutions to key management problems. The solutions are either pre-deployment or trusted-server schemes. As we have seen that according to the introduced criteria neither one could be considered a mature solution for key management. Despite this, these methods would be used as a base to develop more elaborated solutions.

For instance, knowledge-based pre-deployment schemes improve memory usage, but they do not cope with mobility. So, it can be developed through the distribution of information based on anticipated or predicted type of movements based mainly on empirical distribution functions. To explain this, the proposed scheme uses Guassian distribution with constant standard variation and assumes that nodes are static. If we study the standard variation as a function to time based on some predicted or empirical types of mobility, this would improve memory usage and allows nodes to move according to certain trajectories.

Further, most of the proposed schemes focus only on pairwise keys and did not consider other solutions especially the need for group keys into account. The emergence of new techniques in routing that are either hierarchical [21], [35] or location-based [34], [54], reveals the non-competitiveness of the current schemes. Hence, the need to cluster or group key scheme is a must.

Furthermore, hybrid methods based on pre-deployment and trusted third parties can be an alternative. It is possible to use the first type to initiate nodes with the required keys and use trusted party based scheme for updating keys and shares, adding or revoking nodes.

References:

[1]T. Abdelzaer et al. *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC Press, July 2004.

[2]K. Akkaya and Y. Mohamed. A Survey on Routing Protocols for Wireless Sensor Networks. *Journal of Ad hoc Networks*, 2003.

[3]K. Akkaya and M. Younis. An Energy-Aware QoS Routing Protocol for Wireless Sensor Networks. In *Proceedings of the IEEE Workshop on Mobile and Wireless Networks (MWN 2003)*, Rhode Island, USA, May 2003.

[4]I.F. Akyildiz, Y. Sankarasubramaniam, and E. Cayirci. Wireless Sensor Networks: a Survey. *IEEE Communications Magazine*, 40(8):102–114, August 2002.

[5]S. Basagni, K. Herrin, E. Rosti, and D. Bruschi. Secure pebblenets. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001)*, pages 156–163, October 2001.

[6]M. Bellare and P. Rogaway. Entity Authentication and Key Distribution. In *Advances in Cryptology: Proceedings of Crypto93, LNCS 773*, pages 232–249. Springer-Verlag, 1994.

[7]R. Blom. An optimal class for symmetric key generation systems. In Norbert cot Thomas Beth and Ingemar Ingemarsson, editors, *Advances in Cryptology: Proceedings of EUROCRYPT 84, LNCS 209*, pages 335–338. Spinger-Verlag, 1985.

[8]K. et al. Bult. Wireless Integrated Microsensors. In *Proceedings of Conference on Sensors and Systems*, Anaheim, USA, April 1996.

[9]D. Carman, P. Kruus, and B. Matt. Constraints and Approaches for Distributed Sensor Network Security. Technical Report 00-010, NAI Labs, September 2000.

[10]H. Chan and A. Perrig. PIKE: Peer Intermediaries for Key Establishment in Sensor Networks. March 2005.

[11]H. Chan, A. Perrig, and Song d. Random Key Pre-distribution Schemes for Sensor Networks. In *IEEE Symposium on Security and Privacy, Berkeley, California*, Berkeley, California, USA, May 2003.

[12]D. Davis and R. Swick. Network Security via Private Key Certificates. *ACM Operating Systems Review*, 24(4):64–67, October 1990.

[13]J. Deng, R. Han, and S. Mishra. Security Support for In-Network Processing in Wireless Sensor Networks. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, October 2003.

[14]W. Diffie and M.E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, November 1976.W. Du, J.

[15]Deng, Y. S. Han, S. Chen, and P. K. Varshney. A Key Management Scheme for Wireless Networks Using Deployment Knowledge. In *The 23rd Conference of the IEEE Communications Society (Infocom)*, Hong Kong, March 2004.

[16]W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, Washington, DC, USA, October 2003.

[17]A. Easwaran. TinyOS Presentation, 2003. http://www.cis.upenn.edu.

[18]L. Eschenauer and V. Gligor. A Key-Management Scheme for Distributed Sensor Networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, Washington, DC, USA, November 2002.

[19]K. Fokine. Key Management in Ad Hoc Networks, September 2002. Mastre Thesis, Linkoping University, Sweden.

[20]G. Gaubatz, J. Kaps, and B. Sunar. Public Key Cryptography in Sensor Networks - Revised. In *Proceedings of the First European Workshop on Security in Ad hoc and Sensor Networks (ESAS)*, October 2004.

[21]W. Heinzelman, A. Chandrakasan, and H.

Balakrishnan. Energy-Efficient Communication Protocol for Wireless Sensor Networks. In *Proceedings of the Hawaii Int'l conference on System Sciences*, Hawaii, USA, January 2000.

[22]D. Huang, M. Mehta, D. Medhi, and H. Lein. Location-Aware Key Management Scheme for Wireless Sensor Networks, 2004. *In ACM Workshop on Security of Ad hoc and Sensor Networks (SASN*).

[23]Q. Huang, C. Lu, and G. Roman. Mobicast: Just-in-Time Multicast for Sensor Networks under Spatiotemporel Constraints. In *International Workshop on Information Processing in Sensor Networks (IPSN 2003), LNCS 2634, Sringer-Verlag*, April 2003.

[24]Q. Huang, C. Lu, and G. Roman. Spatiotemporel Multicast in Sensor Networks. In *the First ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, November 2003.

[25]J. P. Hubaux, L. Buttyan, and S. Capkun. The Quest for Security in Mobile Ad hoc Networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking and computing*, pages 146 – 155, Long Beach, CA, USA, 2001. ACM Press.

[26]ntanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *Proceedings of sixth Annual International Conference on Mobile Computing and Networks (MOBICOM'01)*, Boston, USA, August 2000. ACM/IEEE.

[27]. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[28]Kahn, R. Katz, and Pister. Smart Dust: Wireless Networks of Millimeter-scale Sensor Nodes, August 1999. *Highlight Article in 1999 Electronics Research Laboratory, Reseach Summary*.

[29]arlof and D. Wagner. Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures, May 2003. *In the First Int'l Workshops on Sensor Network Protocols and applications*.

[30]arp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *Proceedings of the Sixth International Conference on Mobile Computing an Networking (MobiCom 2000*, August 2000.

[31]Kohl and Neuman C. The Kerberos Network Authentication Service (v5), September 1993. IETF, RFC 1510.

[32]Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks. In *Proceedings of the 9th International Conference on Network Protocols (ICNP'01)*, Mission Inn, Riverside, California, USA, November 2001. The IEEE Computer Society.

[33]B. Krishnamachari, D. Estrin, and S. Wisker. Modelling Data-Centric in Wireless Sensor Networks. Computer Engineering Technical Report, CENG 02-14, University of Southern California, 2002.

[34]L. Li and J. Y. Halpern. Minimum energy mobile wireless networks. In *Proceedings of the IEEE Int'l Conference on Communications (ICC'01)*, Helsenki, Finland, June 2001/

[35]S. Lindsey and C. Raghavendra. PEGASIS: Power Efficient GAthering in Sensor Information Systems. In *Proceedins of the IEEE Aerospace Conference*, Montana, USA, March 2002.

[36]D. Liu and P. Ning. Establishing Pairwise Keys in Distributed Sensor Networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, Washington, DC, USA, October 2003.

[37]D. Liu and P. Ning. Location-Based Pairwise Key Establishments for Static Sensor Networks. In *the Proceedings of the*

1st ACM workshop on Security of ad hoc and sensor networks, October 2003.

[38]D. Liu, P. Ning, and W. Du. Group-Based Key Pre-distribution in Wireless Sensor Networks. In *Proceedings of the ACM Workshop on Wireless Security (WISE)*, September 2005.

[39]H. Luo and S. Lu. Ubiquitous and Robust Authentication Services for Ad hoc Wireless Networks. Technical Report 200030, University of California (UCLA), October 2000.

[40]D. Malan, M. Welsh, and M. D. Smith. A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography. In *Proceedings of the First IEEE International Conference on Sensor and Ad hoc Communications and Neworks*, Santa Clara, California, October 2004.

[41]R. Needham and M. Schroeder. Using Encryption for Authentication in Large Networks of Computers. *ACM Operating Systems Review*,21(12):993–999, December 1978.

[42]J. Newsome and D. Song. GEM: Graph EMbedded for Routing and Data-Centric in Sensor Networks Without Geographic Information, in the *Proceedings of the first International Conference on Embedded Networked Sensor Systems*, New York, 2003.

[43]Otway and O. Rees. Efficient and Timely Mutual Authentication. *ACM Operating Systems Review*, 21(1):8–10, 1987.

[44]C. Perkins and E. Royer. Ad hoc On-Demand Distance Vector Routing, November 1997. *in MILCOM'97 panel on ad hoc networks*.

[45]A. Perrig, R. Canetti, D. Song, and J. D. Tygar. Efficient and Secure Source Authentication for multicast. In *Network and Distributed System Security Symposium, NDSS '01*, pages 35–46, 2001.

[46]A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security Protocols for Sensor Networks. *Wireless Networks Journal (WINET)*, 8(5):521–534, September 2002.

[47]D. Przydatek, B. Song and A. Perrig. SIA: Secure Information Aggregation in Sensor Networks. In *the First ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, November 2003.

[48]A. Rao. Geographic Routing without Location Information. In *Proceedings of the ninth Annual Internationl Conference on Computing and Networking*, March 2003.

[49]S. Ratnasamy, B. Karp, L. Yin, and F. Yu. GHT: A Geographic Hash Table for Data-Centric Storage. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, September 2002.

[50]R. Rivest, A. Shamir, and L. Adelman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of ACM*, 21(2):120–126, February 1978.

[51]S. Slijepcevic and al. On Communication Security in Wireless Ad-hoc Sensor Networks, 2002. *In the 11th IEEE Int'l Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*.

[52]V. Varadharajan. Security for Cluster Based Ad-hoc Networks. Technical report, LORIA/INRIA-Lorraine, July 2002.

[53]D. Vincent, V. Park, and M. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *IEEE Infocom (3)*, Kobe, Japan, April 1997.

[54]Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the 7th annual ACM/IEEE Int'l Conference on Mobile Computing and Networking (MobiCom'01)*, Rome, Italy, July 2001.

[55]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.

[56]Z. Yu and Y. Guan. A Robust Group-based

Key Management Scheme for Wireless Sensr Networks. In *Proceedings of IEEE Wireless Communication Networking Conference*, March 2005.

[57]L. Zhou and J. Haas. Securing Ad Hoc Networks. *IEEE Networks Magazine*, 13(6), November/December 1999.

[58]L. Zhou, J. Ni, and C. V. Ravishankar. Efficient Key Establishment for Group-Based Wireless Sensor Deployment. September 2005.

[59]S. Zhu, S. Setia, and S. Jajodia. LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. In *the 10th ACM Conference on Computer and Communications Security (CCS '03)*, October 2003.

Received 9/1/2011.