Int. J. Communications, Network and System Sciences, 2016, 9, 451-470
http://www.scirp.org/journal/ijcns
ISSN Online: 1913-3723
ISSN Print: 1913-3715

Trusted Third Party Authentication Protocol Development for Clustered Wireless Sensor Networks

Ahmad Alshanty, Ibrahim Erşan

Department of Computer Engineering, Girne American University, Girne, Cyprus
Email: ahmadshanticom@yahoo.com, ibrahimersan@gau.edu.tr

How to cite this paper: Alshanty, A. and Erşan, I. (2016) Trusted Third Party Authentication Protocol Development for Clustered Wireless Sensor Networks. *Int. J. Communications, Network and System Sciences*, **9**, 451-470.
http://dx.doi.org/10.4236/ijcns.2016.911037

Received: May 3, 2016
Accepted: November 8, 2016
Published: November 11, 2016

Abstract

Wireless Sensor Networks (WSNs) need effective security mechanisms because these networks are deployed in hostel unattended environments. There are many parameters which affect selecting the security mechanism as its speed and energy consumption. This paper presents a combined security system for WSN that enhances the security of the network and it is energy consumption. This system combines three mechanisms, asymmetric, trusted third party, and pre-distribution. The performance evaluation demonstrates that the combined protocol can enlarge the life time for WSNs, and enhance its security. This integrated protocol can provide sufficient security no matter how many sensors are compromised. Fixed key storage overhead, full network connectivity, and low communication overhead can also be achieved. Consequently, it enhances the immunity against different types of attacks. Moreover, this protocol is offered a high level of security for WSN considering its limited recourses. So it creates a balance between both security and constrained resources.

Keywords

Component, Formatting, Style, Styling

1. Introduction

Wireless Sensor Networks (WSNs) have become most interesting research area, because it is a useful inherent characteristic such as small volume, scalability of nodes, and easy to use. Large scale of WSNs is envisioned to be widely applied in various applications such as object tracking, environment monitoring and data gathering in the near future. Typically, a WSN is composed of a large number of sensor nodes; each sensor node is a small, inexpensive wireless device with limited battery power, memory storage, data processing capacity and short radio transmission range. The main characteristics of WSN include: wireless nature of communication, resource limitation on sensor nodes, lack of fixed infrastructure, unknown network topology prior to deployment, and high risk of physical attacks on unattended sensors. In order to design a practical WSN, many considerations should be taken into the account. Security is one of the most important issues, since it is going to be deployed in unattended environment. The task of securing WSNs is an open research. A solution must strike a tradeoff between the security provided and the consumption of energy, computing, and communication resources in the nodes.

Depending on applications used for WSNs, security is the biggest challenge in WSNs and security aspect is essential for WSNs before designing WSNs [1]. The energy-constrained nature of the sensor networks makes the task of incorporating security as challenging problem. [2] argued that, most of the well-known security mechanisms introduce significant overhead and require a lot of computation and communication resource. Generally security protocols would provide the WSN with three capabilities: encryption, authentication, and key management. Key management is the process by which cryptographic keys are generated, stored, protected, transferred, loaded, used, and destroyed. It is critical to meet the security goals of confidentiality, integrity and authentication to prevent the nodes being compromised by an adversary. Accordingly, key management is a crucial part of security in WSN and has been densely researched recently. For WSN, key management protocols can be classified into four categories: symmetric key protocols, asymmetric key protocols, trusted third party protocols, and key pre-distribution protocols [3].

Asymmetric cryptographic is offered a high level of security for WSN, where each communication entity has its own public key and private key pairs. However, using asymmetric cryptographic for WSN is not practical, because it is related algorithms to finalize and generate the required keys which have expensive computations. Also it needs a large memory to store the public keys. This will dramatically reduce the scalability of WSN. In symmetric cryptographic, the communication entities use a pre-shared symmetric key to negotiate a temporary session key. So it does not offer a high level of security, since all nodes should share the same private key. If an attacker compromises one node and extracts, it is master key. All nodes and the exchanged messages between them will be compromised. Trusted third party protocol is suffering from both lack of scalability and single point failure. Because the communication entities can achieve mutual authentication and secure communication through the single trusted third-party's assistance. Finally, adopting pre-distribution where the required secret keys are pre-loaded before deploying the nodes has many limitations: it has no immunity against node's capture attack, and it does not enhance the resilience and scalability of the WSN. Actually, the number of stored keys will be increased by increasing the number of deployed nodes. Moreover, if the number of compromised nodes is increased above a specific threshold, then the number of uncompromised nodes will be affected as well.

In order to solve the problem of key management and authentication in WSN, and addresses these aforementioned limitations, a new protocol is proposed in this thesis. This protocol is a combination of three different techniques, which are asymmetric, trusted third party, and pre-distribution. It is adopted the asymmetric cryptographic scheme to generate independent session keys. This scheme guarantees that two communicating parties can establish a unique session key between them. Furthermore, it is using key pre-distribution mechanism for a large-scale WSN. Based on a hierarchical clustered network model and trusted third party. Comparing with existing protocols, this integrated protocol can provide sufficient security no matter how many sensors are compromised, Fixed key storage overhead, full network connectivity, and low communication overhead can also be achieved. Consequently, it is enhanced the immunity against different types of attacks. Moreover, this protocol is offered a high level of security for WSN with considering its limited recourses. So it creates a balance between both security and constrained resources. By combing different security techniques and making use of their advantages and overcoming their limitations.

For this protocol, WSN is assumed to be composed of a number of distributed head clusters with heads. The base station or the trusted third party will be responsible about both: organizing the nodes into clusters and determining the cluster head for each one. Furthermore, the base station will apply an election algorithm, to allow a cluster head to take over instead of another damaged one to avoid single point failure. It is also assumed that, this protocol does not use node to node communication. Nodes can communicate only with their cluster heads. Therefore, each node needs only one hop to reach its closest cluster head. This reduces the network traffic and enhances the scalability. The protocol takes advantage of asymmetric cryptographic to solve the pre-distribution limitations. Each node of WSN will be pre-loaded with two keys (Private and Public). In this case the number of stored keys at each node is fixed, and it will not be increased with the increased number of nodes. So each node has to store only its cluster head's public key, instead of storing the number of public keys equals the number of deployed nodes. This will save the storage capacity for each node and maintain the scalability of WSN. Furthermore, these stored keys will be used to generate a new session key, so for each new link a new session key will be generated. If an attacker could compromise a node the other nodes will not be affected, since each one of them has a different session key.

2. Related Works

There are many base station participation schemes based symmetric key. [4] presented a base station participation scheme called SPINS. In base station participation scheme, each sensor has a share key with the base station. When two sensors need a pairwise key, the base station can send the pairwise key encrypted with the shared keys, respectively. This scheme has perfect resilience. The shorting coming is that it does not have good scalability as the base station needs to send the keys to related sensors. Both [4] then [5] presented a suite of security protocols optimized for sensor networks: SPINS and SNAKE, these two protocols have sets of steps to establish the security of sensor network, to establish the shared-session key among sensor nodes.

SPINS is a security suite for sensor networks. It includes two protocols, SNEP and TESLA. The former is for confidentiality, two-party data authentication, integrity, and freshness and the latter provides authentication for data broadcasting. Figure 1 shows the key negotiation protocol. Assume that node A wants to establish a shared session key (SKAB) with node B through a trusted third party S, the central key distribution center (KDC). This is a server that can perform authentication and key distribution. Node A will send a request message to node B. Node B receives this message and sends a message to the key server. Key server S will perform the authentication and generate the shared-session key and send the key back to node A and node B respectively. The use of the central key server limits the scalability of the sensor networks.

Following is the convention to describe these two protocols.

A | B: data A concatenates with data B.

{A}KAS: encryption of data A by key KAS.

MAC K [A]: MAC (message authentication code) of data A created by key K.

NA: the nonce generated by node A. Nonce is a one-time random bit-string, usually used to achieve freshness.

IDA: the name of node A.

SNAKE is a protocol that can negotiate the session key in an ad-hoc scheme. Nodes do not need a key server to perform the key management as shown in Figure 2. First, node A will send a request to node B. Node B will reply a message as a challenge to node A. When node A receives this message, it will prove its authenticity and send the message back to node B. This is a mutual challenge and authentication procedure. After this three handshaking and mutual authentication procedures, node A and node B will use KAB as their shared-session key where KAB = MACK [NA|NB].

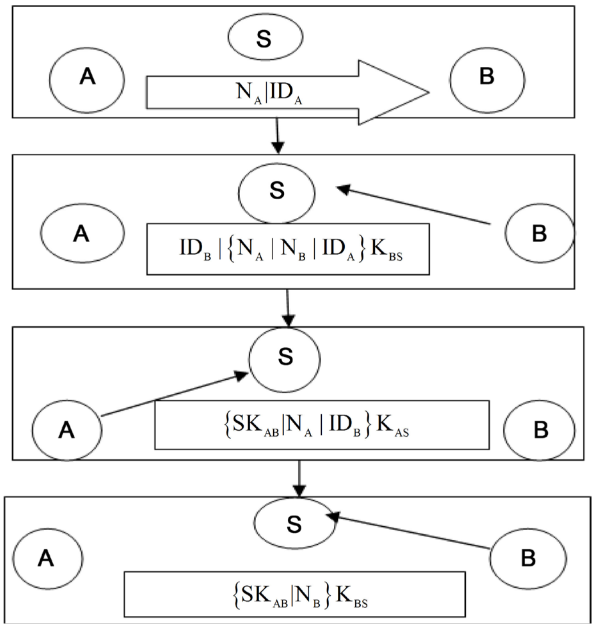

Figure 1. Key negotiation in SPINS.

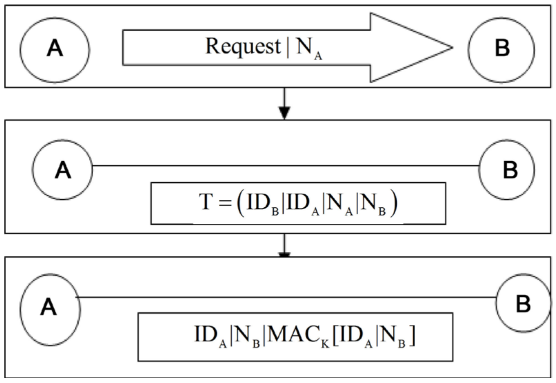

Figure 2. Key negotiation in SNAKE.

Most of the proposed security protocols in sensor networks are based on point-to-point handshaking procedures to negotiate link -dependent keys such as above two protocols. But this will influence the scalability of the network. According to [6], most of the security algorithms are based on sharing a secret key between two parties. They use this shared-secret key to verify each other and even use the key to encrypt and decrypt the data. In a distributed sensor network, constructing and negotiating this secret key is very hard, because of their limited resources. When the sensor network has been working for a while, nodes might run out of session keys. It is insecure to reuse the same key for data transmission and will be easily compromised. Therefore nodes in sensor network need to renegotiate new session keys. However, this protocol does not include a key renegotiate mechanism. Furthermore, transmission of data consumes energy. Therefore the more the transmissions in the network, the more energy will be consumed. Consequently, SPINS needs four data transmissions to finish the key negotiation process. [7] shows that data transmission takes 71% of the time. In SPINS four data transmission are needed while as SNAKE needs three data transmissions. In terms of security, both SPINS and SNAKE do not provide a solution for denial of service (DoS) attacks when the malicious node keeps sending the request to negotiate a session key. Both protocols can achieve authentication requirement. But they cannot detect or prevent the DoS attacks, because one adversary can easily trigger a REPLAY attack and exhaust the energy in the sensor nodes.

In SPINS, the malicious node can simply send the request message for key negotiation continuously, and Node B will keep asking the server about session key with the malicious node. Therefore node B will eventually run out of the energy. However, the base-station may have the ability to detect and try to prevent this attack.

In SNAKE, DoS attacks can be triggered by the same mechanism and SNAKE does not provide the detection of DoS when a malicious node tries to send the message to request key negotiation. In SNAKE there is no base-station to perform attack detection for sensor nodes, every node has to detect this attack by itself and this function is a heavy burden for resource constrained sensor nodes [4]. In SPINS, the base station plays an essential role in the key establishment process. It is a lightweight cryptography protocol but has limited scalability and depends heavily on the base station [7].

[8] and [9] proposed a broadcast session key (BROSK) negotiation protocol. BROSK assumes a master key shared by all the nodes in the network. To establish a session key between two nodes for example A and B. Bothe sensor nodes A and B broadcast a key negotiation messages (M1 = IDA|NA|[IDA|NA]K) and (M2 = IDB|NB|[IDB|NB]K) respectively. In order to produce a shared session key (KAB) where (KAB) = [NA|NB]K as shown in Figure 3. BROSK is a scalable and energy-efficient protocol. Compared to SPINS, BROSK can be considered a more recent WSN key negotiation protocol [10]. In this scheme, there is not a trusted party or server and each node directly negotiates a session key with its neighbors by broadcasting a key negotiation message. Once a node receives this message, it can construct the shared-session key by generating the MAC of two nonces. For example, all nodes from B to D will receive the broadcast message from node A. Node A will also receive the broadcast message from node B and others. They then use KAB as their shared-session key. On the other hand, in the BROSK proposal, no mention is found about what is done with the master key once the broadcasting process has finished, so it is assumed that the master key is not erased or processed in a special way. If this assumption is true, the scheme would be vulnerable to physical intrusion, in the same way as SPINS [11].

3. The Proposed Protocol for Authenticated Wireless Sensor Networks and Key Management

This protocol is based on a combination of three different techniques. These techniques are asymmetric encryption, trusted third party, and pre-distribution. It adopts the asymmetric cryptographic scheme to generate independent session keys. This scheme guarantees that two communicating parties can establish a unique session key between them. Furthermore, it is using key pre-distribution mechanism for a large-scale WSN, based on a hierarchical clustered network model and trusted third party. Comparing with existing protocols, this integrated protocol can provide sufficient security no matter how many sensors are compromised, Fixed key storage overhead, full network connectivity, and low communication overhead can also be achieved. Consequently, it enhances the immunity against different types of attacks. Moreover, this protocol offers a

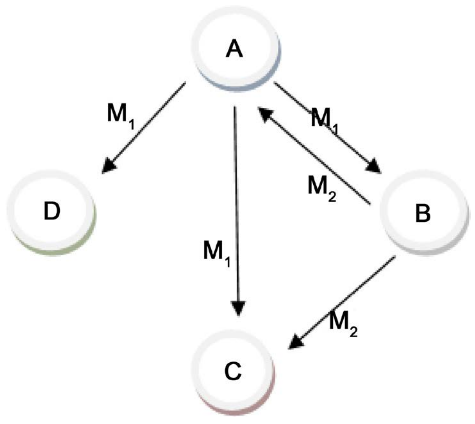

Figure 3. The key negotiation in BROSK protocol.

high level of security for WSN with considering it is limited recourses. So it creates a balance between both security and constrained resources, by combining different security techniques and making use of their advantages and overcoming their limitations.

WSN is assumed to be composed of a number of distributed head clusters with heads. The base station or the trusted third party will be responsible about both organizing the nodes into clusters and determining the cluster head for each one. Furthermore, the base station will apply an election algorithm, to allow a cluster head to take over instead of another damaged one to avoid single point failure. Actually, this protocol does not use node to node communication. Nodes can communicate only with their cluster heads. Therefore, each node needs only one hop to reach its closest cluster head. Thus reduces the network traffic and enhances the scalability. The protocol takes advantage of asymmetric cryptographic to solve the pre-distribution limitations. Each node of WSN will be pre-loaded with two keys (Private and Public). In this case the number of stored keys at each node is fixed, and it will not be increased with the increased number of nodes. So each node has to store only its cluster head's public key, instead of storing the number of public keys equals the number of deployed nodes. This will save the storage capacity for each node and maintain the scalability of WSN. Furthermore, these stored keys will be used to generate a new session key, so for each new link a new session key will be generated. If an attacker could compromise a node the other nodes will not be affected, since each one of them has a different session key.

Compared to previous investigations; the proposed protocol possesses the following features. 1) Scalability. No matter how many sensor nodes and how many clusters are in a WSN, the proposed protocol can use constant communication rounds to establish all cluster keys. Therefore, it is especially suitable for resource-constrained large-scale WSNs. 2) Applicability. In the proposed protocol, all sensor nodes can be deployed randomly and establish cluster keys without knowing the topology of the whole network in advance. 3) Flexibility. This protocol also presents dynamic insert and remove protocols. The dynamic insert protocol allows newly deployed sensor nodes to join an existing WSN while the dynamic remove protocol can delete compromised sensor nodes from a WSN. 4) Robustness. The proposed protocol can resist node capture attacks, node cloning attacks, wormhole attacks and energy consumption attacks.

4. Architecture and Layers of the Proposed Protocol

The integrated proposed protocol combines three different mechanisms for authentication and key management. First, pre-distribution scheme to initially establish the main secret keys into each communicating party. Second, the trusted third party to generate the independent session keys. Which are obtained from the pre-uploaded main keys to encrypt the exchanged messages. Third, asymmetric algorithm for performing authentication and key management. This protocol is based on a hierarchical clustering architecture that consists of three main layers: the top layer is the trusted third party or the base station (BS), the middle layer is composed of the cluster heads (CHs), and the lower layer has a number of sensor nodes which are organized into clusters called cluster members (CMs). Each layer has a certain type of communicating party with different allocated resources. Therefore, these layers are varied capabilities in terms of computational processing, size of memory storage, battery, and the range of radio communication. The overload will be fairly distributed among these layers depending on their allocated resources and abilities, in order to extend the lifetime of the WSN. Furthermore, these layers are cooperating together to perform the required cryptographic operations, to achieve the fundamental security demands of the WSN. Functionally, there are three main phases to perform the required cryptographic operations for authentication and key management: pre-distribution phase, registration phase, and authentication and key establishment phase.

The proposed architecture is composed of three layers as shown in Figure 4, where the base station (BS) constitutes the first layer. It controls the cluster heads (CHs) and all cluster members (CMs) in the cluster which constitute the second layer and the third layer respectively. The network model of this proposed hierarchical scheme is mainly consisted with three types of nodes which are randomly distributed in the network. Nodes are categorized into three categories as: Base station (BS), cluster head (CH), and cluster member (CM).

Cluster member (CM): They are inexpensive and limited-capability. Each sensor has limited battery power, memory size, data processing capability and short radio transmission range. Sensor only communicates with it is corresponded cluster head (CH) directly; no communication between sensors exists in this model. After deployment, sensor nodes keep stationary during the network operation period.

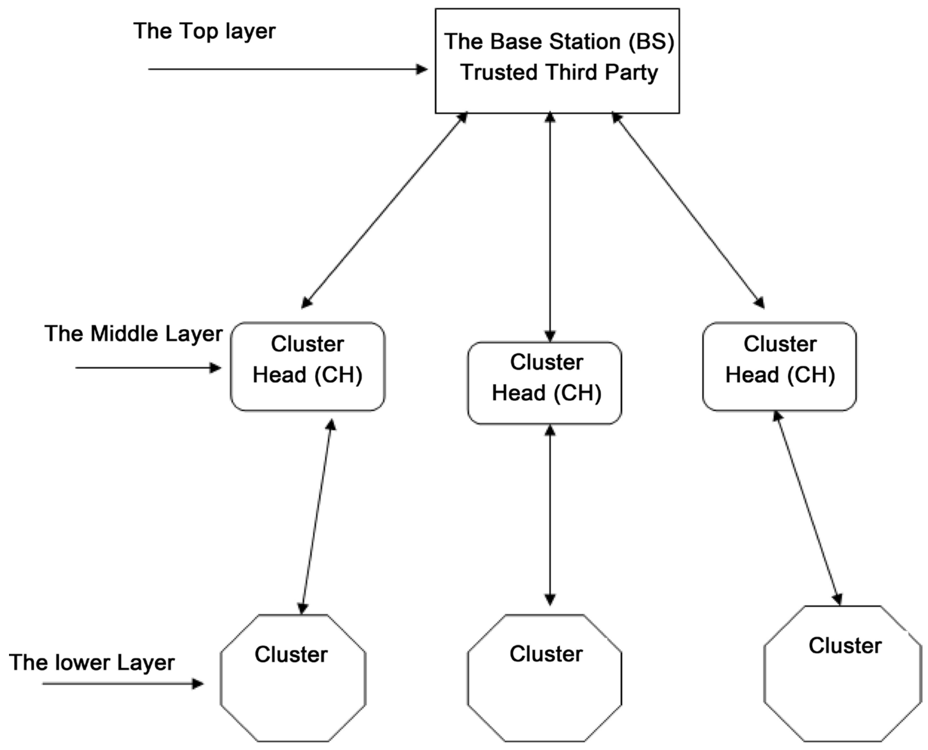

Figure 4. The architecture and layers of the proposed protocol.

Cluster head (CH): They have considerably more resources than sensors. Equipped with high power batteries, large memory storages, powerful antenna and data processing capacities, cluster heads can execute relatively complicated numerical operations and has much longer radio transmission range than sensor nodes. Cluster heads can communicate with each other directly and relay data between their cluster members and the base station.

Base station (BS): It is the most powerful node in the WSN, it has virtually unlimited computational and communication power, unlimited memory storage capacity, and very large radio transmission range which can reach all the nodes in a network. BS can be located either in the center or at a corner of the network based on the application. In this network model, a large number of CMs are randomly distributed in an area. BS is located in a well-protected place and takes charge of the whole network's operation.

As mentioned before, the authentication and key management operations for this protocol are going through three phases:

1) Pre-distribution phase: firstly, each single CM will be uploaded with an ID and two keys: public key and private key, and the public key of it is CH. Secondly, each CH will be uploaded with set of it is cluster member's IDs and three categories of keys: it is own private key and public key, the BS's public key, and a set of public keys for it is related cluster members. Thirdly, the base station will be uploaded with it is own private and public keys, all other public and for all cluster heads and cluster members, and the IDs for all cluster heads and cluster members. Additionally, few extra IDs will be uploaded to the base station in case of deploying new nodes so the new deployed node should have an ID that is matched one of the unallocated IDs in the BS's database. All these keys will be saved before the WSN gets deployed.

2) Clustering and registration phase: since the nodes are going to randomly deploy in the wanted environment, so this phase will help the nodes to be organized into clusters by determining it is related cluster head, through exchanging it is encrypted ID with it is private key to the available cluster heads, in order to find out it is related cluster head and introduce itself. When a cluster head receives the initial message from a node that contains it is encrypted ID, the cluster head will decrypt this message by using the corresponding public key of this cluster member. After that, it will match between this ID and the list of IDs inside it is memory, if it finds the same ID, it will send confirmation message back to this node, in order to inform this node that it is one of this cluster head members. If the cluster head cannot find this ID within its list of IDs, it will broadcast the initial message for this node to other cluster heads with a counter equals 1 and so on. If any cluster head could find the same ID within it is list, it will send the confirmation message to that CM. But the counter reaches the value that equals the total number of CHs that means all CHs have searched for this ID through their lists of IDs and there is no matching. In this case, this CM will be considered as unauthorized node. And a report will be sent to the BS by CHs as shown in Figure 5. By the end of this phase, all nodes will know their associated cluster heads and nodes will be organized into clusters. Consequently, cluster heads and nodes will be ready for the third phase.

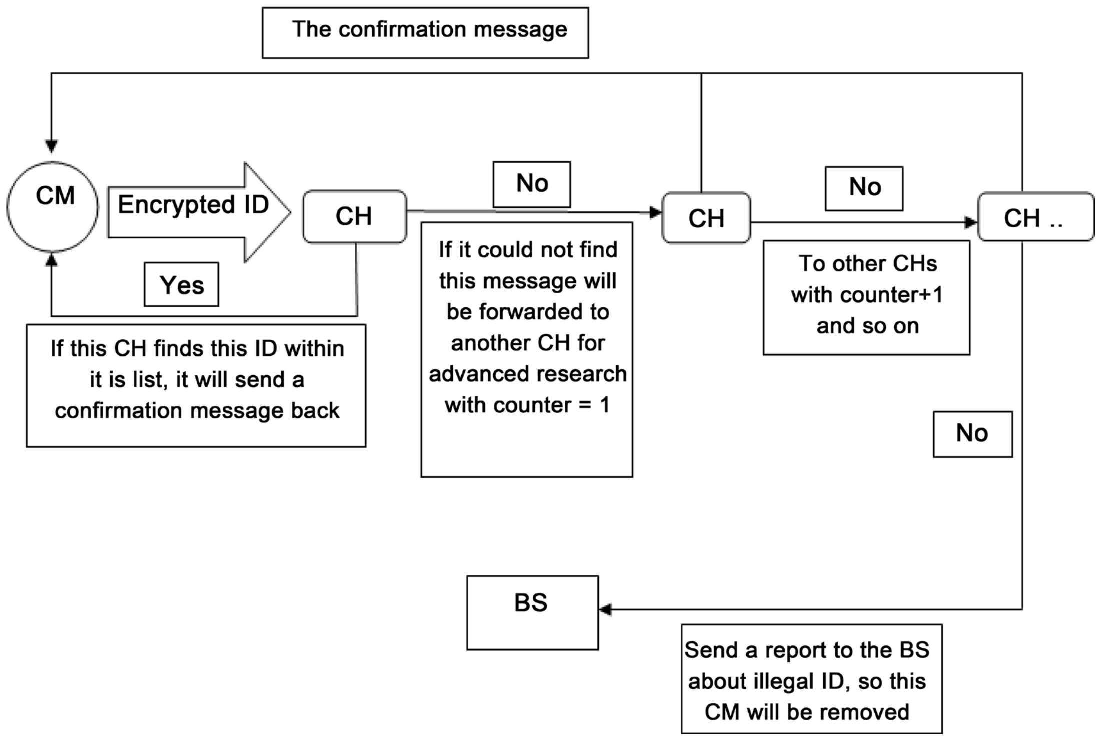
The confirmation message
CM
Encrypted ID
CH
Yes
If this CH finds this ID within it is list, it will send a confirmation message back
No
CH
If it could not find this message will be forwarded to another CH for advanced research with counter = 1
No
CH ..
To other CHs with counter+1 and so on
No
BS
Send a report to the BS about illegal ID, so this CM will be removed

Figure 5. The registration part of the second phase.

3) Authentication and key establishment phase: where both CHs and it is CMs can authenticate each other, and maintain the confidentiality and integrity for exchanged messages, by encrypting them with the generated session keys. These session keys will be generated by the base station, which is considered as a trusted third party.

BS applies a particular asymmetric algorithm on the public keys of CH and CM and the forwarded random values X and Y, in order to generate the session key between them. Assume that a CM wants to communicate with its CH, firstly a new shared session key should be produced between them, which they can use it to encrypt the exchanged messages between them. In this case, the CM will send it is ID, the random value X, and the current timestamp. All of them will be encrypted with it is private key. This message is considered as the first part. Then the cluster head will use the corresponding public key of this CM to decrypt the message, and it will check the validity of both ID and the timestamp. After that the cluster head will produce the second part by adding it is ID, it current timestamp, and the random value Y, and then it will encrypt them by using it is private key. The CH will concatenate these two parts together and encrypt them by using the BS's public key and send it to the BS. The BS will decrypt this message with it is private key. And it will use the public key for the CM to decrypt the first part, and the CH's public key to decrypt the second part. In order to extract: ID of the cluster head, ID of the cluster member, X, and Y. these values will be considered as parameters. Then the BS will apply the asymmetric algorithm on these parameters and supplied public keys to establish a new session key to be used to encrypt the exchanged messages between both CH and CM as shown in Figure 6. After generating the

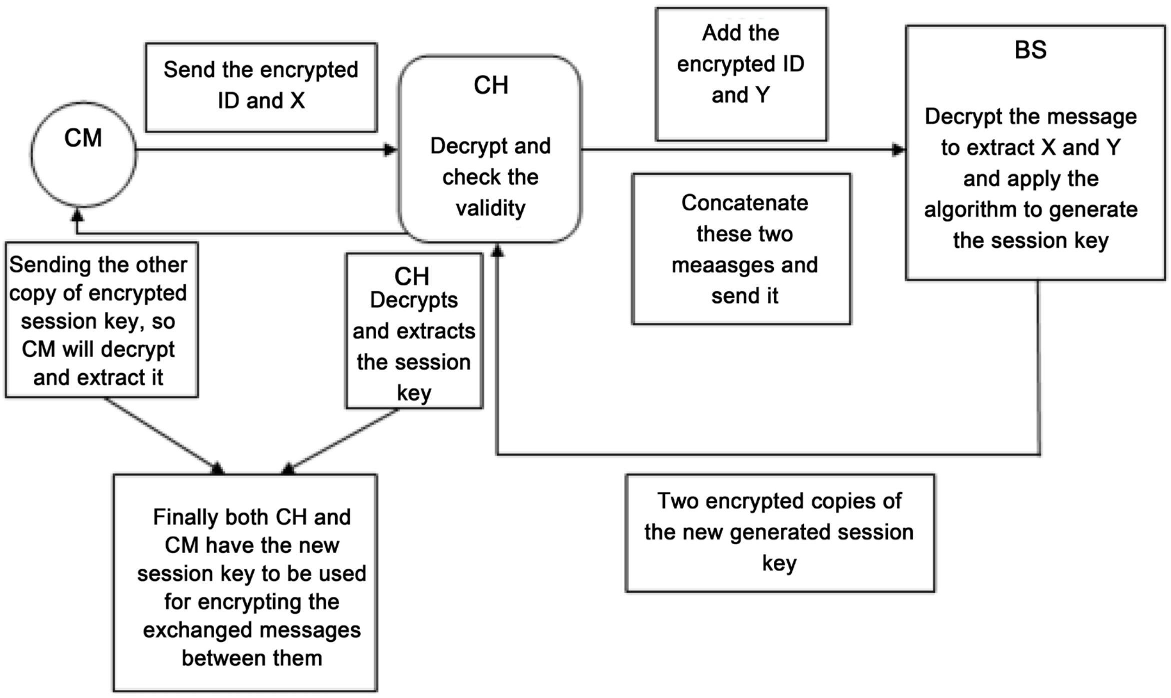
CM
Send the encrypted ID and X
CH
Decrypt and check the validity
Add the encrypted ID and Y
Concatenate these two meaasges and send it
BS
Decrypt the message to extract X and Y and apply the algorithm to generate the session key
Sending the other copy of encrypted session key, so CM will decrypt and extract it
CH Decrypts and extracts the session key
Finally both CH and CM have the new session key to be used for encrypting the exchanged messages between them
Two encrypted copies of the new generated session key

Figure 6. The session key generation.

required session key, BS will produce two copies of this key, the first copy will be encrypted with the CH's public key, and the second copy will be encrypted by the CM's public key, since BS has a list of all public keys. These two copies will be concatenated together with it is current timestamp. The whole message will be encrypted with the BS's private key and sent to the CH. The CH will use the public key of the BS to decrypt the message. And then it will use it is private key to decrypt the first part and extract the first copy of the session key. After that CH will forward the second copy to the CM with it is current timestamp, which uses its private key to decrypt the message and extract the second copy of the session key. Consequently, both CH and CM have the new session key, which can be used to encrypt the exchanged messages between them. Once any communicating party is received a message, it will check the timestamp validation for security purposes.

5. The Adopted Encryption Algorithm in the Proposed Protocol

The proposed protocol can be described as new independent session key generation based protocol. It takes the advantage of asymmetric cryptographic scheme and pre-distribution mechanism to achieve both key management and authentication. Based on hierarchical clustered model and trusted third party. This section discus the transactions between the involved communicating parties at each layer and their roles, the related keys and their pre-distribution, the three main phases and action during each one, and the adopted asymmetric algorithm for session key generation and authentication of communicating parties. First of all, this section will use the following notations to describe security protocols and cryptographic operations as shown in Table 1.

Table 1. Notations and descriptions of adopted algorithm.

(a)

Notations	Descriptions
CM	Cluster Member
CH	Cluster Head
BS	Base Station
SK_{CM}	The Private Key of Cluster Member
PK_{CM}	The Public Key of Cluster Member
ID_{CM}	The Identity of Cluster Member
SK_{CH}	The Private Key of Cluster Head
PK_{CH}	The Public Key of Cluster Head
ID_{CH}	The Identity of Cluster Head
SK_{BS}	The Private Key of Base Station
PK_{BS}	The Public Key of Base Station
X	The Random Value Generated by Cluster Member (where $X < q$)
Y	The Random Value Generated by Cluster Head (where $Y < q$)
Mod	Modulo
q	Large Prime Integer generated by the Base Station
α	Primitive Root Modulo q
$\|$	Concatenating
$\{M\}ESK_{CM}$	A Message Encrypted by Cluster Member's Private Key
$\{M\}EPK_{CM}$	A Message Encrypted by Cluster Member's Public Key
$\{M\}ESK_{CH}$	A Message Encrypted by Cluster Head's Private Key
$\{M\}EPK_{CH}$	A Message Encrypted by Cluster Head's Public Key
$\{M\}ESK_{BS}$	A Message Encrypted by Base Station's Private Key
$\{M\}EPK_{BS}$	A Message Encrypted by Base Station's Public Key

(b)

Notations	Descriptions
$\{M\}DSK_{CM}$	A Message Decrypted by Cluster Member's Private Key
$\{M\}DPK_{CM}$	A Message Decrypted by Cluster Member's Public Key
$\{M\}DSK_{CH}$	A Message Decrypted by Cluster Head's Private Key
$\{M\}DPK_{CH}$	A Message Decrypted by Cluster Head's Public Key
$\{M\}DSK_{BS}$	A Message Decrypted by Base Station's Private Key
$\{M\}DPK_{BS}$	A Message Decrypted by Base Station's Public Key
SIK_{CM}	The New Generated Session Key for Cluster Member
SIK_{CH}	The New Generated Session Key for Cluster Head

During the session key generation, the applying of adopted algorithm is going through two manners: offline and online. The offline mode includes all performed procedures before the deployment of the nodes into wanted area. So this manner is applied during the first phase or pre-distribution phase. There are two main actions that are happened under this manner: firstly, the generating of both PKCM and PKCH to be compatible and to maintain the consistency, since they will be used in the key derivation of the session key. Secondly, pre-uploading of all required keys and IDs into each communicating party among the three main layers as shown in Table 1(b).

There are main steps in order to generate the both compatible public keys PK_{CM} and PK_{CH} during this manner:

1) Selecting the two prime values of q and α.

2) Selecting two initial values of X and Y which are given to CM and CH respectively.

3) Computing both PK_{CM} and PK_{CH} by using these two calculations where:

$$PK_{CM} = \alpha^{x} \bmod q \tag{1}$$

$$PK_{CH} = \alpha^{y} \bmod q \tag{2}$$

Then PK_{CM} will be uploaded into CM, and PK_{CH} will be uploaded into corresponding CH.

For example:

1) Let q = 353 and $\alpha = 3$.

2) Assume that X = 97 and Y = 233.

3) Based on aforementioned assumptions:

$$PK_{CM} = 3^{97} \bmod 353 = 40 \tag{3}$$

$$PK_{CH} = 3^{233} \bmod 353 = 248 \tag{4}$$

Figure 7 shows the main procedures of the adopted algorithm during the offline manner which are: the generating of PK_{CM} and PK_{CH} by the BS, the pre-uploaded keys, IDs, and initial random values at each involved communicating party. After finalizing the all procedures of the offline mode, all communicating parties will be ready to be deployed into wanted area. And the second online mode of the adopting algorithm will be started after the deployment of the communicating parties.

After the deploying of the communicating parties into the wanted environment, the second phase of clustering and registration will be started. In which the deployed CMs are organized into number of clusters with one gateway that is a CH for each. Consequently, the network based on hierarchical clustered architecture will be built. Within this architecture, there are three types of communication: CM to CH, CH to BS, and BS to CH. Therefore CMs are not able to directly communicate either between each other or with BS, except through their corresponding CH. In order to achieve the goals of this phase, three main actions between CHs and CMs must be done as following:

1) An initial message is sent by CMs for the nearest CH. In order to introduce itself, check the validity of it is ID, and know it is related CH. This initial message contains the CM's encrypted ID with it is private key: {IDCM}ESKCM.

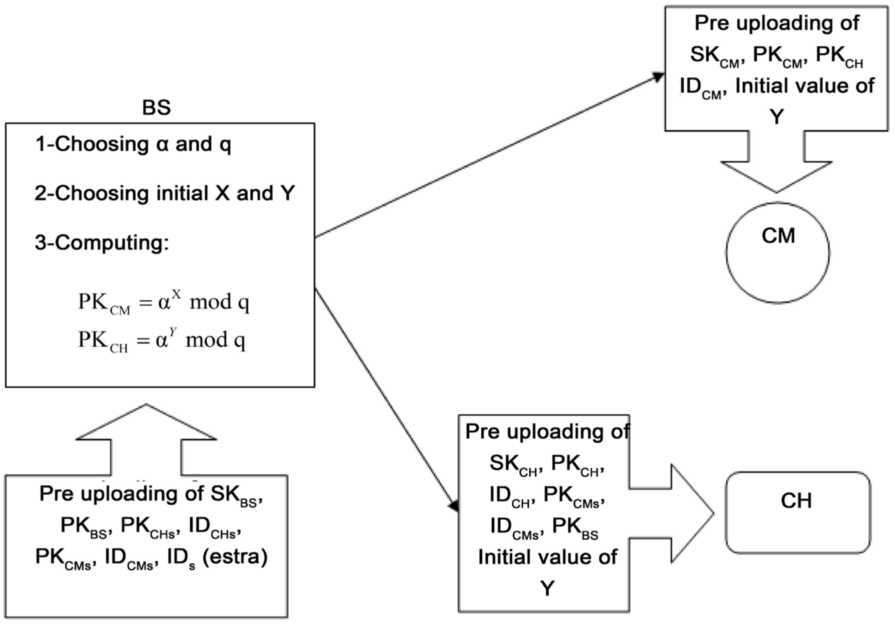
BS
1-Choosing α and q
2-Choosing initial X and Y
3-Computing:
$PK_{CM} = \alpha^X \bmod q$
$PK_{CH} = \alpha^Y \bmod q$
Pre uploading of SK_{CM}, PK_{CM}, PK_{CH} ID_{CM}, Initial value of Y
CM
Pre uploading of SK_{BS}, PK_{BS}, PK_{CHs}, ID_{CHs}, PK_{CMs}, ID_{CMs}, ID_s (estra)
Pre uploading of SK_{CH}, PK_{CH}, ID_{CH}, PK_{CMs}, ID_{CMs}, PK_{BS} Initial value of Y
CH

Figure 7. The main procedures of the offline mode of the adopted algorithm.

2) The CHs will receive the initial messages, but only the related CH can decrypt this message with using the corresponding public key of this CM (PKCM), by performing this cryptographic operation: {{IDCM}ESKCM }DPKCM. Then CH can extract the IDCM, compare it with it list of IDCMs to find out the matching, and check it is validity to make sure that it is an authorized CM. The main goal of this step is to enhance the authentication by verifying the IDs of CMs. This would not be done without the saving lists of IDCMs and PKCMs into each related CH.

3) After the verifying that this IDCM is valid and this CM is an authorized or authenticated node, a confirmation message will be sent back from the CH to this CM. In order to inform this CM that it is one of these cluster members under the supervision of this CH.

Secondly, the online mode will be started after the communicating parties being deployed, and organized into clusters. The online mode represents the third phase in which the session key is generated. All communicating parties will be involved to achieve the goals of this phase by authenticating each other, enhancing the confidentiality and integrity of exchanged data, and checking the timestamp validation. There are some procedures are done between the three layers, in order to negotiate and generate the new session key as following:

Firstly, CM sends a message for it is IDCM, the random value X, and T1 to it is related CH. They will be encrypted with it is private key SKCM as: {IDCM, X, T1} ESKCM. This message is considered as a request from this CM that it needs to establish a new session key.

Secondly, the CH receives this message at T2, and decrypted with the corresponding public key (PKCM) by performing: {{IDCM, X, T1}ESKCM} DPKCM, in order to check the validity of IDCM and the timestamp ($T1 - T2 <= \Delta T$). Then CH will add

IDCH, the random value Y, and T3 encrypted with it is private key (SKCH) as: {IDCH, Y, T3}ESKCH. The CH will concatenate the first part that has been sent by CM ({IDCM, X}ESKCM), and the second part that CH is added ({IDCH, Y}ESKCH) together with T3 as: ({IDCM, X}ESKCM)‖({IDCH, Y}ESKCH)‖(T3), then this concatenation will be encrypted by the BS public key(PKBS) as:
{({IDCM, X}ESKCM)‖({IDCH, Y}ESKCH)‖(T3)} EPKBS, and then send it to the BS.

Thirdly, the BS will receive the message at T4, Then, it will use it is private key (SKBS) to decrypt this message as:
{{({IDCM, X}ESKCM)‖({IDCH, Y}ESKCH)}EPKBS)‖(T3)} DSKBS to extract: {({IDCM, X}ESKCM)‖({IDCH, Y}ESKCH‖(T3)}, and check the timestamp validation
(T3 – T4 <= ΔT). Then the BS will use the corresponding public key of the sending CM (PKCM) to decrypt the first part ({IDCM, X}ESKCM) and extract X as:
{IDCM, X}ESKCM} DPKCM. Also, it will do the same for the second part
({IDCH, Y}ESKCH}) by using the corresponding public key of CH (PKCH) by performing this operation: {{IDCH, Y}ESKCH}} DPKCH, in order to obtain Y. After that, BS will have X, Y, IDCM, IDCH, PKCM, and PKCH which are required to generate the new session key and send it to CM and CH.

Fourthly, the BS is ready to calculate the new session key (SIK) by using the obtained values. In order to calculate both the shared session key at CM side SIKCM and the session key at CH side (SIKCH) which must be the same (where SIKCM = SIKCH), the BS will use X, Y, PKCM, PKCH, and q. By performing these calculations:

$$SIK_{CM} = \left(PK_{CH}\right)^{X} \bmod q \tag{5}$$

$$SIK_{CH} = \left(PK_{CM}\right)^{Y} \bmod q \tag{6}$$

where $SIK_{CM} = SIK_{CH}$ since they are shared session keys. Based on the previous example where: q = 353, X = 97 and Y = 233, $PK_{CM} = 40$, and $PK_{CH} = 248$. Therefore:

$$SIK_{CM} = 248^{97} \bmod 353 = 160 \tag{7}$$

$$SIK_{CH} = 40^{233} \bmod 353 = 160 \tag{8}$$

So both CM and CH can use this shared session key to encrypt the exchanged messages between them.

Fifthly, the BS will prepare to copies of the session key (SIK). The first copy will be encrypted using the public key of CM (PKCM) as: {SIKCM}EPKCM, and the second copy will be encrypted by CH's public key (PKCH) as: {SIKCH}EPKCH. Then BS will concatenate these two copies together into one message with T5 as
(({SIKCM}EPKCM)‖({SIKCH}EPKCH)‖(T5)). This concatenation will be encrypted by the BS's private key (SKBS) by performing:
{({SIKCM}EPKCM)‖({SIKCH}EPKCH)‖(T5))} ESKBS. After that, this message will be sent to CH.

Sixthly, CH receives this message at T6, and use the BS's public key (PKBS) for decryption as: {({SIKCM}EPKCM)‖({SIKCH}EPKCH)‖(T5)}ESKBS)}DSKBS, in order to obtain ({SIKCM}EPKCM)‖({SIKCH}EPKCH)‖(T5). Then CH will check the timestamp validation (T5 – T6 <= ΔT). CH will use it is private key (SKCH) to decrypt the second part {{SIKCH}EPKCH}}DSKCH to extract it is copy of the new session key (SIKCH). While as T7 will be add to the first part ({SIKCM}EPKCM)‖(T7). And both of them will be encrypted by CH's private key (SKCH) by {{SIKCM}EPKCM)‖(T7))}ESKCH, and then send it to the CM.

Seventhly, CM receives the message at T8, and it performs to two decryptions: the first one is performed by using the CH's public key (PKCH) to obtain {SIKCM}EPKCM and (T8) by this decrypting: {{SIKCM}EPKCM)‖(T8)}ESKCH}DPKCH. Then CM checks the timestamp validation (T7 – T8 <= ΔT). The second one is performed by using it is private key (SKCM) by performing {{SIKCM}EPKCM}DSKCM, in order to obtain it is copy of the new shared session key (SIKCM). Then, CM will delete it is private key (SKCM) for security purposes.

Finally, both CH and CM have the shared (SIK), and they can use it to encrypt the exchanged messages between them. Therefore, if CM wants to send a message to the CH, this message must be encrypted with the shared session key as: {M‖(TM)}ESIKCM

So CH can decrypt it by using it is shared session key (SIKCH) by applying this operation on the same message: {{M‖(TM)}ESIKCM}DSIKCH. This can be done since ESIKCM and SIKCH are the same at each side. In the other direction, if CH wants to send a message to CM, CH will perform: {M‖(TM)}ESIKCH, and CM will be able to decrypt it by {{M‖(TM)}ESIKCH }DSIKCM. However, if the shared session key (SIK) between these CM and CH gets compromised by an attacker, a new shared session key will be generated by BS.

6. The Security Analysis of the Proposed Protocol

This section represents a comprehensive evaluation of the proposed protocol, by discussing three main aspects. Firstly, the contributions that the proposed protocol is added, by taking the advantageous of combining three different mechanisms for authentication and key management. It makes use of the strong point of these mechanisms and overcomes their limitations. Secondly, evaluating the performance of the proposed protocol based on specific norms which are used to compare the proposed security protocols for WSN. Mainly, the performance of this protocol reflects on the efficient energy consumption. Thirdly, the security which is considered as the most important concern in this field. The proposed protocol provides a high level of security, since it is meeting the fundamental requirements of a secure WSN, and guaranteeing the immunity against different security attacks. Moreover, it supports three security services: encryption, authentication, and key management.

Computation overhead: In WSNs scenario, it is highly desirable for a security protocol to have low computational overhead on resource constrained sensor nodes. Many specialists argued that using asymmetric cryptographic as an authentication mechanism in WSN is not practical. Because of it is related encryption algorithms have expensive computations, so it is not compatible with the limited resources of WSN. This leads to a high power consuming and shortens the expected lifetime of the deployed nodes. Since, theses algorithms need a lot of computations in order to finalize the generating of the required keys.

The proposed protocol could refute this argument by maintaining both: firstly, the pre-distribution mechanism during the offline mode, where the required keys are pre-uploaded before the deployment, in order to avoid the performing of the computations of key generation. Secondly, the arrangement of the network based on layered hierarchical model with involving a trusted third party. As mentioned before, the architecture of the proposed protocol has three layers with different allocated resources and capabilities. Therefore, the top layer has the powerful trusted third party (BS) with higher resources compared with other communicating parties existing into other two layers. The BS is mainly responsible about the key generation. The proposed protocol fairly distributes the computational burdens among these layer based on their allocated resources and capabilities. So the layer that has higher allocated resources will perform the higher computational operations. The proposed protocol does not require CMs to compute any expensive pairing function. Moreover, it imposes very light computational and communication. This leads to efficient power consumption and extend the expected lifetime of the WSN. Thus, key establishment protocols using public-key cryptography can no longer be considered prohibitively expensive in terms of energy consumption.

Key storage overhead: Assume that there is N number of deployed nodes within a WSN where asymmetric cryptographic scheme is adopted. Therefore, each node must store into it is memory storage (N − 1) number of public keys, in order to be able to communicate with other nodes and perform the required cryptographic operations. However, the sensor node has a limited storage capacity or small memory. So it is impractical to save this huge amount of keys inside every single node. Furthermore, this will reduce the scalability of the WSN, because by increasing the number of deployed nodes(N), the number of stored public keys at each node will be increased as well, so there is no way to add new nodes. In other words, it is key storage overhead is still sub-linearly with the network size.

In order to avoid this problem, the proposed protocol does not maintain node to node communication. Since it is hierarchical clustering based protocol, three kinds of communications are supported: CM to CH, CH to CH, and CH to BS. Therefore, CMs inside any cluster can only communicate with their corresponding CH, which is considered as the intermediate node or the gateway between these CMs and the BS. In the proposed protocol, CM only needs to initially store three keys (PKCM, PKCH, and SKcm) in its memory no matter how many nodes in the network. These keys will pre-uploaded during the offline mode or pre-distribution phase. Furthermore, after the generating of the session key (SIKCM), CM will delete it is private key (SKCM). That means only two keys will be stored into it is memory (PKCM, SIKCM), which is extremely memory efficient for the large-scale WSN. In this way, each CM has to save only two keys instead of saving number of public keys equals the number of deployed nodes. Since it only needs to have exchanged messages with it is related CH. This will save the storage capacity for each node. Additionally, this enhances the scalability of

WSN, because the deployed CM stores a fixed number of keys regardless about the number of deployed nodes. Also, it offers the ability to deploy new nodes without overloading their memories.

Re-negotiate the key: In the previous proposed protocols which are adopted symmetric cryptographic scheme, all nodes have the same shared master key that is never disclosed. They will use this key to authenticate other nodes and encrypt the exchanged messages. However, if an attacker could compromise one node and extract it is master key by conducting node capture attack. Consequently, all nodes and the exchanged messages between them will be compromised since they are using the same shared key. So these protocols do not show the protection against node capture attack. Furthermore, these protocols do not guarantee the resiliency of the network, which is used to evaluate how much fraction of the communication between non-captured nodes will be compromised when a certain number of sensors are captured by the adversary.

The proposed protocol can overcome these limitations by generating independent new session keys for each new link between the communicated parties (CMs and CHs). In this case, even if a CM has been compromised by an attacker the other CMs will not be affected, since each CM has a different independent session key for encryption and authentication. Also, at any point the BS is able to generate a new session key, if the previous one gets compromised. This operation is called as rekeying. That means the proposed protocol can protect the CMs against capture node attack, and improves the resilience of WSN. When the sensor network has been working for a while, nodes might run out of session keys. It is insecure to reuse the same key for data transmission and will be easily compromised. Therefore nodes in sensor network need to renegotiate new session keys.

Network resiliency: key pre-distribution is one of key agreement schemes, where key information is distributed to all sensor nodes prior to deployment. There exist a number of key pre-distribution schemes. A naive solution is to let all the nodes carry a master secret key. Any pair of nodes can use this global master secret key to achieve key agreement and obtain a new pair wise key. This scheme does not exhibit desirable network resilience. If one node is compromised, the security of the entire sensor network will be compromised. One of the key pre-distribution schemes is to let each node to carry $N - 1$ (assuming N is the total number of sensors). Therefore, each pair of nods shares the same key, which is known only these two nodes and they can use it to encrypt the exchanged messages between them. The resilience of this scheme is perfect; because of compromising one node does not affect the security of communications among other nodes. However, this scheme is impractical to be applied into WSN with limited storage capacity, especially with great number of deployed nodes since each node must save $N - 1$ number of keys. Moreover, adding a new node to a pre-existing sensor network is difficult because the existing nodes do not have the new deployed nodes' key. In spite of that the proposed protocol is using the pre-distribution key scheme, but it does not have this limitation. It is adopting asymmetric cryptography so each node will be pre-loaded with two keys (Private and Public), which are used to generate a new session key. In this case the number of stored keys at each node is fixed, and the number of stored keys will not be increased by increasing the number of nodes. Actually, the proposed protocol provides three security requirements: firstly, the known session key security, this property emphasizes that if an adversary obtains a session key, the session keys of the coming sessions remain secure. Secondly, the forward secrecy, this property notifies that by revealing the long term private keys of the two nodes (perfect forward secrecy) or one of these nodes (weak forward secrecy), the adversary cannot obtain the previous session keys. Strong security is a kind of forward secrecy which states that if the short term private keys of the two nodes, or one node's long term private key and the short term private key of the other are revealed, the previous session keys cannot be computed by the adversary. Thirdly, the key compromise impersonation (KCI), let A and B to be two nodes. If the adversary has the long term private key of A, it can obviously forge A. KCI states that the adversary cannot forge B by obtaining the long term private key of A.

The proposed protocol provides the ability of deploying new nodes, in order to extend the coverage area of the WSN and enhance the scalability, or replacing the damaged one. Therefore, the BS is uploaded with extra unallocated IDs for new deployed nodes. The three main phases will be applied before the new node starts working. On the other hand, the proposed protocol can remove nodes if they being compromised or unauthorized. By inserting it is ID into the list of block IDs and informing the related CH. Also, the BS will cancel the adopted session key of this node. Consequently, this node is no longer belonged to this cluster.

Scalability: The overheads of computations and communications in the proposed protocol do not increase with the increasing of network size. Furthermore, each sensor node stores two keys in its memory no matter how many nodes are deployed. New sensor nodes can be added to the WSN easily at any time. Moreover, CHs and BS have relatively large memory size, sufficient power, data processing and transmission capacity, so the distance between them can be extended. Therefore, the proposed protocol supports large scale deployment of WSNs. Theoretically, this protocol can be applied for any size of WSN, since a suitable algorithm and hierarchical architecture are properly selected. And it is considered as maximum supported network size.

7. Conclusion

This paper presented combined security protocol combines three mechanisms based on clustering layered architecture for authentication and key management. This integrated protocol takes the benefits of asymmetric encryption, trusted third party, and pre-distribution. In order to generate shared independent session keys, and achieve the balance between the energy consumption and a higher level of security, the security analysis shows that the proposed protocol can extend the expected lifetime of a WSN, by reducing the number of exchanged messages for key negotiation, and saving a fixed number of secret keys regardless of the number of deployed nodes. This will enhance the scalability and resiliency. Also, it considers the limited resources of the communicating party by distributing the overload depending on each layer's allocated resources. Furthermore, it enhances the fundamental security requirements of the WSN such as confidentiality, integrity, and authentication. Moreover, it provides the immunity against different kinds of security attacks comparing with other security protocols. The calculations show that the proposed protocol has lower energy consumption, lower computation time, less key storage and lower communication overheads. Therefore, one of system benefits is enhancing the energy consumption. Saving energy means decreasing number of communications and computations, and this improves the speed of the network.

References

[1] Porambage, P., Schmitt, C., Kumar, P., Gurtov, A. and Ylianttila, M. (2014) PAuthKey: A Pervasive Authentication Protocol and Key Establishment Scheme for Wireless Sensor Networks in Distributed IoT Applications. *International Journal of Distributed Sensor Networks*, **10**, Article ID: 357430. http://dx.doi.org/10.1155/2014/357430

[2] Yasmin, R., Ritter, E. and Wang, G. (2011) A Pairing-Free ID-Based One-Pass Authenticated Key Establishment Protocol for Wireless Sensor Networks. *The 5th International Conference on Sensor Technologies and Applications*, 21-27 August 2011, French Riviera, 340-347.

[3] Singh, U.R., Singh, Kh.M. and Roy, S. (2015) Energy Efficient Key Management Analysis Using AVL Trees in Wireless Sensor Network. *International Journal of Engineering Science Invention*, **4**, 59-70.

[4] Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V. and Culler, D.E. (2002) SPINS: Security Protocols for Sensor Networks. *Wireless Networks*, **8**, 521-534. http://dx.doi.org/10.1023/A:1016598314198

[5] Lai, B., Kim, S. and Verbauwhede, I. (2010) Scalable Session Key Construction Protocol for Wireless Sensor Networks. *IEEE Workshop on Large Scale Real Time and Embedded Systems (LARTES)*, December 2010, 7.

[6] Zhou, L. and Haas, Z.J. (1999) Securing Ad Hoc Networks. *IEEE Network*, **13**, 24-30. http://dx.doi.org/10.1109/65.806983

[7] Mansour, I., Chalhoub, G. and Lafourcade, P. (2014) Evaluation of Secure Multi-Hop Node Authentication and Key Establishment Mechanisms for Wireless Sensor Networks. *Journal of Sensor and Actuator Networks*, **3**, 224-244. http://dx.doi.org/10.3390/jsan3030224

[8] Lai, B.C.C., Hwang, D.D., Kim, S.P. and Verbauwhede, I. (2004) Reducing Radio Energy Consumption of Key Management Protocols for Wireless Sensor Networks. *Proceedings of the 2004 International Symposium on Low Power Electronics and Design*, San Francisco, 8-10 August 2016, 351-356. http://dx.doi.org/10.1145/1013235.1013320

[9] Jemai, A., Mastouri, A. and Eleuch, H. (2011) Study of Key Pre-Distribution Schemes in Wireless Sensor Networks: Case of BROSK (Use of WSNet). *Applied Mathematics & Information Sciences*, **5**, 655-667.

[10] Hu, F., Ziobro, J., Tillett, J. and Sharma, N.K. (2013) Secure Wireless Sensor Networks: Problems and Solutions. Rochester Institute of Technology, Rochester, New York.

[11] Delgado-Mohatar, O., Fúster-Sabater, A. and Sierra, J.M. (2015) A Light-Weight Authentication Scheme for Wireless Sensor Networks. *Ad Hoc Networks*, **9**, 727-735. http://dx.doi.org/10.1016/j.adhoc.2010.08.020