

Article

# A Polynomial Subset-Based Efficient Multi-Party Key Management System for Lightweight Device Networks

Zahid Mahmood <sup>1</sup>, Huansheng Ning <sup>1,\*</sup> and AtaUllah Ghafoor <sup>2</sup>

<sup>1</sup> School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China; b20140561@xs.ustb.edu.cn

<sup>2</sup> Department of Computer Science, National University of Modern Languages, Islamabad 44000, Pakistan; aullah@numl.edu.pk

\* Correspondence: ninghuansheng@ustb.edu.cn, Tel.: +86-136-0131-7155

Academic Editors: Yunchuan Sun, Zhipeng Cai and Antonio Jara

Received: 3 February 2017; Accepted: 18 March 2017; Published: 24 March 2017

**Abstract:** Wireless Sensor Networks (WSNs) consist of lightweight devices to measure sensitive data that are highly vulnerable to security attacks due to their constrained resources. In a similar manner, the internet-based lightweight devices used in the Internet of Things (IoT) are facing severe security and privacy issues because of the direct accessibility of devices due to their connection to the internet. Complex and resource-intensive security schemes are infeasible and reduce the network lifetime. In this regard, we have explored the polynomial distribution-based key establishment schemes and identified an issue that the resultant polynomial value is either storage intensive or infeasible when large values are multiplied. It becomes more costly when these polynomials are regenerated dynamically after each node join or leave operation and whenever key is refreshed. To reduce the computation, we have proposed an Efficient Key Management (EKM) scheme for multiparty communication-based scenarios. The proposed session key management protocol is established by applying a symmetric polynomial for group members, and the group head acts as a responsible node. The polynomial generation method uses security credentials and secure hash function. Symmetric cryptographic parameters are efficient in computation, communication, and the storage required. The security justification of the proposed scheme has been completed by using Rubin logic, which guarantees that the protocol attains mutual validation and session key agreement property strongly among the participating entities. Simulation scenarios are performed using NS 2.35 to validate the results for storage, communication, latency, energy, and polynomial calculation costs during authentication, session key generation, node migration, secure joining, and leaving phases. EKM is efficient regarding storage, computation, and communication overhead and can protect WSN-based IoT infrastructure.

**Keywords:** path key; symmetric; polynomial; hashing; proxy node; node storage

## 1. Introduction

A recent advancement in communication technology, Wireless Sensor Networks (WSNs) are widely used in several applications [1]. The science community has focused on the security of WSNs. Because of the resource-constrained environment of the WSNs, classic security mechanisms are not practical since they consume too much energy; hence researchers are proposing new lightweight security mechanisms for every possible security aspect of WSNs. WSNs consist of many small, low-cost, self-governing ends called sensor nodes with little ability to manipulate data [2] and with constrained computing, energy, and memory. MICA2 Motes use 8 bit, 16 MHz processors with 4 K bytes of

electronically erasable programmable read only memory (EEPROM) and 128 K bytes of programmable memory [3]. With the passage of time, the sizes of WSN are growing in clusters [4]. Traditional security mechanisms using public key cryptography cause significant overhead regarding computation and communication. Key management is mandatory and more challenging with limited resources in WSNs [5]. Many researchers have been recognized that grouped or dispersed heterogeneous sensor systems can sensibly perform with system effectiveness, operational execution, and enduring system lifetimes.

In a heterogeneously grouped approach, as described in Figure 1, the low end sensor (*L*-sensors) are resource-constrained devices with low power, short communication range, limited memory, and less computation power. On the other hand, *H*-sensors are equipped with tamper resistance and have enough resources, like high battery power, broad communication ranges, sufficient memory, and high computational capabilities. *L*-sensors are deliberately conveyed in a group, and every group is controlled by a group head (*H*-sensor). The *L*-sensors essentially sense environmental statistics and forward it to the *H*-sensors and the other way around. *H*-sensors can perform complex operations on the sensor information, and utilize longer radio and straightforwardly transfer it to the base-station. The base-station (BS) is a powerful hub, and it has abundant sources. The BS might be a remote server, and it might be connected with the external world utilizing the accelerated Internet.

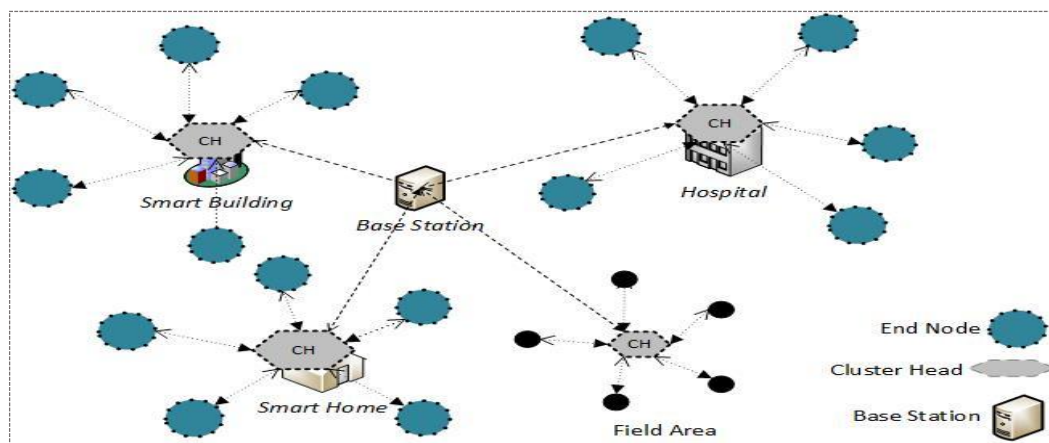


Figure 1. Heterogeneous WSN-based IoT architecture.

Various anonymous security protocols have been recommended for heterogeneous WSNs in recent past, and they display diverse sorts as well as levels of security protection at various cost. In this section, we discuss the existing cryptography techniques that addressed the issue of authentication in heterogeneous WSNs. Due to the limited resources of WSNs, a sound balance between security level and the associated energy utilization overhead is needed to mitigate the security threats. Symmetric parameters such as node ID, message authentication code, nonce number, and time stamps are the significant parameters for grouped key management techniques which are energy efficient and avoid the different type of attacks from malicious nodes and avoid node compromising attacks. Most security protocols developed for WSNs use symmetric encryption, due to ease of its implementation [6]. Besides this, single node authentication has become unable to meet the increasing communication demand, as the service demand increases with the passage time, multiparty computing is necessary for node authentication simultaneously and securely inter-cluster and intra-cluster where [7] participants collectively to set up a mutual session key to enable the multi-party computing and secure exchange of messages.

Multi-party computing for security credential calculations has been adopted using many-to-many nodes communication scenarios. Conventional node authentication involves computation at single sensor node for the entire security credential calculation, which is less secure as compared to

trustworthy multi-party involvement. These days, a new form of authentication, known as group authentication [8] is proposed which may be used to decide whether or not all users belong to the same group or not. The group authentication is very green for the reason that it can authenticate all members at one time. It can be used for pre-processing of node authentication to identify trustworthy contributors and non-contributors as well.

To communicate securely with the same group and among different groups, we employed a polynomial ( $P$ ) to achieve efficient intra-group key refreshment and generated a polynomial to create an inter-group key. For this reason, our proposed protocol could be highly effective in each authentication and key establishment because it may provide the most effective broadcast transmission. Moreover, for computations every sensor needs a polynomial assessment and key-hash feature and it is easier for performing encryption and decryption. In [9] a technique to optimize the energy consumption for computing polynomial expressions was proposed. In this context, it is, therefore, crucial to have efficient key revocation and renewal mechanisms in WSNs. The goal of this work is to propose secure and efficient protocols for key revocation and renewal.

This research proposes an Efficient Keying for Multi-party (EKM) scheme to calculate the polynomial values using lightweight XOR operations. It also includes pre-deployment, node joining, and key establishment protocols. Before the nodes' deployment in WSNs, each sensor node is pre-configured with a set of symmetric keys shared with all the other sensor nodes of the network to transmit IDs securely. After the network deployment, every sensor node identifies the specified symmetric keys used to communicate with the cluster head (CH). The cluster-head maintains all the symmetric keys shared with the sensor nodes belonging to its cluster. The main reason the use of these symmetric keys is to facilitate the multi-hop communication while transmitting secret data, particularly, the personal-proportion distribution and the exchanged facts among the cluster-heads and the base station (BS). The implementation of the polynomial that is applied for deriving an intra-group key can reduce the session key storage overhead at the member nodes and their CH. After the intra-group key is acquired, the member nodes self-generate the polynomial functions, which are necessary for growing an inter-organization key. This facilitates the reduction of the communication overhead on the CH. We have proposed a key generation and update mechanism for secure data sharing in identical clusters and among different cluster. In the session key generation mechanism for intra-clusters, the proposed technique significantly decreases the number of broadcast messages in the inter-cluster communication. The extra number of re-keying messages and communication overheads due to the proposed scheme has been conscientiously justified. Through the usage of the proposed mechanism, we have enhanced the WSN efficiency and network lifetime by reducing the number of exchanged messages to enhance the network lifetime during node mobility while reducing the coverage area of cluster heads and the need to migrate nodes to join the powerful coverage area of the CH. This research proposes a secure node migration protocol in which reliable handoff occurs and new connections are established between CHs and member nodes.

The organization of the rest of the paper is as follows: the system and security model for proposed protocol established in Section 2. Section 3 elaborates the related work to develop existing key establishment schemes. Section 4 provides a detailed discussion on a proposed EKM system for key distribution, node joining, and key establishment scenarios. The formal specification and security analysis of proposed technique is discussed in Section 5. Section 6 are about simulation parameters and scenarios to present results and illustrate the analytical discussion. Finally, Section 7 provides conclusions and future work.

## 2. System and Security Model

In this section, the network model, adversarial model, and security model are described, respectively. For multicast communications in WSNs, the network model used here is the heterogeneous and hierarchical topology depicted in Figure 2. We assume that the network consists of a sink and many groups, and each group has a powerful node which has abundant resources

(energy, computing, and storage) and many low ordinary nodes. The powerful node acts as the cluster head (CH) and is responsible for network security management, such as key issuing, updating, revocation, intrusion detection, etc. As an ordinary member, *L*-sensor is responsible for sensing data and authentication for secure communications.

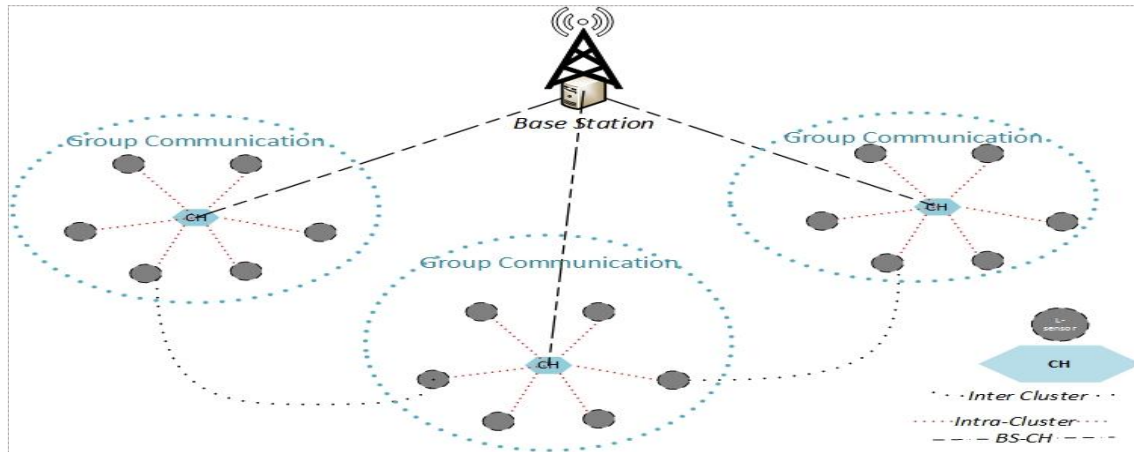


Figure 2. Network model of multi-party key establishment scenario.

When we consider that WSN may additionally encounter attacks from inside by compromised nodes or external attackers, so its lifetime can be classified into many phases. At the beginning of every session, all types of keys like *BS-CH*, *CH-L-Sensor*, and *L-Sensor-L-Sensor* for security functions will be re-issued. The length of every session could be modified dynamically with the security strength of the safety mechanism used or the regulations of group members like new node addition and migration. Session switching might be launched using cluster heads for group members and by the BS for cluster heads. The self-restoration key control schemes proposed in particular focus on the problems, updating and revocation manner of organization keys and pairwise session keys. Different styles of keys methods may be originated according to the devices for group keys and pairwise keys.

### 2.1. Threat Model

In this section, we describe the threat model adopted in the proposed scenario. We assume that there aren't any compromised nodes at network deployment time. Attacks are released a while after the nodes are deployed. With effective computing capability, the adversary can execute advanced fact evaluation to gather the important data. The adversary can be either inner or external and active or passive. Moreover, a single entity or a member of a coordinated group of entities can also perform malicious activities. An adversary can launch a physical attack by compromising a node and reading secret data to use it for future tracking. An insider adversary is a malicious node that may examine/alter all messages encrypted with its keys. Passive adversaries can spy on all communication channels in the community like idea channels, or unintended verbal exchange channels. Energetic adversaries can examine, tamper, and inject fraudulent packets over verbal exchange channels.

#### 2.1.1. Adversarial Model

Moreover, insider attackers and outside attackers can apportion the important thing, captured information, to interrupt the security collusively, in what is referred to as a collusive physical attack. For instance, the revoked nodes and the newly joined nodes can launch deceitful attacks on the session keys no longer belonging to them. We expect that the compromised or insider adversary behaving abnormally could be revoked from the network inside the following session with the assist of the intrusion detection mechanism, which is not addressed in this research.

Moreover, a comprehensive attacker can easily monitor the wireless data of WSNs and capture the wireless packets. The attacker will be able to gather plenty of useful sensor information from the captured packets like node IDs and node location and might adjust packet information even as a packet is in transit. Additionally important, an attacker can intentionally apprehend a node and capture all its cryptographic keys. As a result, node capturing physical attacks are very harmful to the network if excessive numbers of cryptographic keys remain stored on a node.

### 2.1.2. Potential Attacks

In general, attacks affecting secure key establishment are eavesdropping and node capture [10], referred to as passive adversaries and active adversaries. Passive adversaries gather some statistics via eavesdropping without any physical access to a network entry, and active adversaries encrypt information that is leaked during physical access to the node. The proposed key management technique is secure against eavesdropping and node capture, and we will explain the cause for this below. During cluster key derivation, the cluster head sends an expanded polynomial without any encryption. It is not easy to guess the intra-group key management from the polynomial, considering that it is hard to do polynomial factorization due to the fact there may be certainly a  $O(n(\log n))$  options to the polynomial enlargement problem [11,12] and the problem for polynomial factorization is NP-hard [13]. For example,  $P = (3x^3 + 15x^2 + 74x - 117)$  is not easy to factorize as  $P = (x - 5)|(x - 4)|(x - 6) + 3$  for the purpose, it's tough to guess 3. Even though the accelerated polynomial  $P$  is sent without any encryption, cluster head key credentials will no longer leak out. For this reason, it's far tougher to generate the coefficient of polynomial  $H(x)$  with the aid of an attacker due to the safety of these keys in clusters.

### 2.2. System Model

Based on the system and attack model, we particularly grasp the subsequent security goals: for the reason that  $L$ -sensors aren't truthful, initializing essential keys allows powerful security. To preserve the mutual authentication among the network devices (sensors, cluster-heads, and base-stations), devices have to perform mutual authentication and establish a dynamic session key before establishing the consultation, so that each the nodes/gadgets can confirm their legitimacy. In vital applications (place of birth security, healthcare, smart grids), nodes' identities are very critical, as a consequence, nodes' identities should be stored securely and privately [14–16]. To protect the Wi-Fi communication from unlawful eavesdropping and interception all the wireless messages should be secured (i.e., personal), so that a global adversary can't examine, reveal, intercept, or modify the Wi-Fi messages. During an adversary monitoring, the  $L$ -sensors are easily compromised, so then a key revocation mechanism can play a vital function to protect from any future misleading moves from a compromised node. Moreover, a node addition feature confers scalability to the network.

The proposed model for our key distribution scheme is designed for heterogeneous sensor networks. We assume a heterogeneous sensor network consists of high resource node ( $H$ -sensors) and their member nodes having fewer resources called  $L$ -sensors.  $H$ -sensors are nodes that are more powerful and have more computation, communication, energy supply and storage capability's sensors are ordinary sensor nodes with limited computation, communication, and energy supply and storage capability. We consider the hierarchical structure of the heterogeneous sensor network proposed in [17] in which high power sensor nodes work as cluster heads and low power sensor nodes work as their member nodes. This sensor clustering scheme enables local data processing, which reduces the communication load in the network to provide scalable solutions. To increase the lifetime of the deployed network the nodes are divided according to their workload. Initially, when a sensor network is deployed, there are large number of low power sensor nodes and a limited quantity of cluster heads. It is assumed that the  $H$ -sensor and  $L$ -sensors are deployed uniformly and randomly. It has been assumed that the  $CH$  is tamper resistant and cannot be compromised. The major issue with the polynomial calculation is that resulting multiplicative values become very large and slow for larger

values of polynomial factors. Due to constrained resources, when the number of  $L$ -sensors increases then the computation and communication cost to the cluster head also increases. The more  $L$ -sensors there are, the more memory required on the cluster head for restoring their IDs and multiplicative values, which causes memory overhead.

### 3. Related Work

Different issues with existing key management techniques are discussed in conjunction with the availability of limited resources of WSNs. Minimizing the resource consumption and maximizing security levels are two different points. The probabilistic scheme described in [18], where each node receives a group of keys smaller than the network size provides better connectivity and little memory overhead. It has key pre-distribution, path-key establishment, and shared key discovery phases that cause scalability problems. The larger network size for a given stage increases the computation and communication overheads. The secret key sharing is limited by the memory existing on the sensor nodes, since the chance that next node shares  $k/n$  give a key, for example, the memory overhead is  $O(n)$ . For fixed key chain size, the key connectivity achieved is lower than previously.

In [19] a dynamic generation of the polynomial (DGP) is proposed for heterogeneous WSNs scheme where nodes are pre-loaded with a master key  $K$  for secure communication during cluster key establishment. During the key management process, each node calculates  $MAC(ID)$  and then encrypts the concatenated message with the master key. Each  $L$ -sensor node ( $L_i$ ) generates ciphertext  $C_{Li} = E_{MK}(ID || MAC(ID))$  and transmits it towards a  $H$ -sensor as depicted in Figure 3. The  $H$ -sensor calculates the hash of  $L$ -sensor IDs and saves these values.  $H$ -Sensor generates a polynomial using Equation (1) where  $h(L_i)$  is hashed function for  $L$ -sensor  $L_i$  and  $i = 1, \dots, n$ .

$$(x - h(l_1))(x - h(l_2)) \dots (x - h(l_n))K_{chj} \quad (1)$$

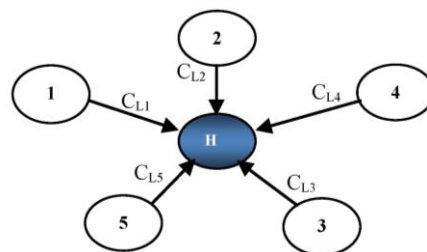


Figure 3. Polynomial computation at a  $H$ -sensor.

During a node joins or leave operation,  $H$ -Sensor revokes existing polynomial and broadcast the new polynomial to all  $L$ -Sensors by taking  $MAC(ID)$  of all  $L$ -sensors. It increases the lifetime of sensor networks by decreasing the key pool size. Our scheme further reduces the computational cost of this scheme.

In [20] where each node receives a unique ID, a private key, and a master key  $K$ . After deployment, sensor nodes broadcast their IDs encrypted with the global key. A loop is constructed by using a certain number of a node from set  $S$ . If sensor node  $A$  creates loop  $L$  with set  $S$ , then loop-key  $KL = Hash(ts || KA || IDA || \{ID\})$  is computed. The global key protects the distribution of the loop key inside the loop. A network-wide key can reduce communication, but capturing a single node exposes all the links in the network. Our approach prevents it from taking the credentials from member nodes.

In [21] every sensor node randomly picks up a unique secret key from the large key pool and produces a new secret key by applying one-way hash function. The  $H$ -sensor broadcasts a message in its communication range and then, the  $L$ -sensors discover their neighbors. It provides complete connectivity, resilience against sensor node capture and routing attack. Our scheme follows the same deployment pattern. Sajid recommended following a sequence of Key Pool  $\rightarrow$  Key Chain  $\rightarrow$  Key Ring [22]. It achieves an equal probability of key sharing among nodes and stores a small number of

key generations on the node. It reduces the memory overhead if the wrong parameter is fixed, but for maximum key sharing, it needs a larger pool size.

In [23] a secure symmetric key used for member nodes and public key cryptography for powerful nodes was proposed. Three scenario-based schemes reduce the memory and computational overhead along with high resilience against node capture attacks. A key management regime described in [24] used a keyed hash chain. It supports the establishment and renewal of five keys, and it also overcomes the drawback of a single key. A critical decision is to how determine the length of a key-chain. A large key-chain size increases the memory overhead and on the other hand, decreasing the chain size compromises the security strength.

A random key pre-distribution scheme where seeds are taken from the base station to drive a new key that is providing adequate security was described in [17]. It allows privately distributing a secret to a set of recipients with only one multicast communication [25]. A lightweight XOR-based re-keying scheme is illustrated in [26] that does not require message exchange from common nodes. In this system, *CH* transmits a Magic code and a nonce number in the encrypted message to all member nodes. Sensors take XOR of an existing key with a left shifted Magic word and get a new value that is further used to establish XOR with the nonce. Similarly, our scheme also supports key freshness on each node join or leave operation. A technique put forth in [27] enforces mutual authentication between a member and group key distributor while executing a re-keying process.

During the last decades, various applications have been benefitting from WSNs, such as environmental monitoring, battlefield intelligence, and reconnaissance, medical caring, intelligent home, industrial control [28]. However, in some harsh and critical environments, malicious attacks, including eavesdropping, Denial of Service (DOS) attacks, and packet tampering and infection, may disturb the network operations and degrade the system performance. Security issues should be considered carefully during the design of WSNs' protocols to defend against and tolerate attacks [29]. Cryptography is typically used to provide security features, such as confidentiality, authentication, integrity, and availability [30]. As the core role of security, key management is responsible for key distribution, updating, and revocation for sensor nodes, and establishes security associations between nodes [31]. As per scientific techniques, some related works explore polynomial [32,33] vector space secret sharing [34], bilinear pairings [35] and elliptic curve cryptography [36]. Polynomials can be used to improve the trade-off between security performances, computation complexity, and resource consumption better than other methods [37]. In this paper we concentrate on a polynomial-based strategy to achieve better system performance.

References [38,39] elaborate that radio-frequency identification (RFID) is a program distinguishing proven innovation and a promising innovation to be naturally incorporated into conventional remote sensor systems (WSNs) including a novel recognizable proof measurement into WSNs and forming RFID-based sensor frameworks. The lightweight protocol provides the security properties because basic logarithmic and intelligent operations without requiring costly cryptographic calculations are implemented in low-cost and resource-restricted wireless systems.

The scheme in [40] enforces mutual authentication between a member and the group key distributor while executing the rekeying process. It provides forward secrecy and backward secrecy properties, and resists replay attacks, impersonation attacks, group key disclosure attacks and malicious insider attacks. The proposed scheme can be used for rekeying purposes in a secure multicast scenario. The second one is an authentication mechanism to be used in environments in which a public-key infrastructure is not available. Forceful characteristics such as changing the key of a user in a chain of command influences the performance of an admittance organize scheme more than other dynamic properties such as adding or deleting a security class to the hierarchy [41]. To overcome forgery attacks, the authors focuses on the security of the data provenance specific to a wireless sensor network. The paper provides a scheme to detect forgeries in provenance data, detection of packet drop attacks and also identifies the attacker of packet drop attacks [42]. This scheme used a packet bloom filter data structure for encoding provenance data. Hash functions are used to encode the provenance data in

packet bloom filter. The use of a bloom filter data structure results in a lightweight scheme which is suitable for wireless sensor networks. Each packet consists of the sequence id of the packet, actual sensor data, and an n-bit bloom filter vector. To detect a forgery attack, the ID of the each node in the path is encoded into the bloom filter vector. It is assumed that base station knows the path of the received packet. When a packet is reached at the base station in the WSN, a fresh bloom filter is taken and encoded with all nodes in the path. If the generated bloom filter and bloom filter extracted from the received packet are the same, then there is no forgery attack, and otherwise there is a forgery attack. Actual sensor data is also secured along with provenance data with the scheme proposed in this paper. Experimental results proved the effectiveness and the lightweight nature of the scheme.

#### 4. Efficient Keying for Multi-Party Scheme (EKM)

In this phase, mutual authentication and session key establishment takes place between the participants. During cluster key establishment, an *H*-sensor or cluster head (*CH*) receives encrypted IDs of *L*-sensors from neighboring nodes. *CH* decrypts these IDs using the pre-distributed master key *K*. After receiving the IDs, a hash of these IDs is calculated and stored in memory as illustrated in Figure 3. *CH* selects random IDs from the stored hash list to compute the polynomial. Finally, the polynomial is concatenated with its MAC and encrypted using a master key to transmit towards all neighboring nodes. In our scheme, we considered the hierarchical structure of [20] for the deployment. It has been assumed that the *H*-sensor and *L*-sensors are deployed uniformly. It is also presumed that the *CH* is tamper resistant and cannot be compromised. A list of the notations for EKM is provided in Table 1.

**Table 1.** List of Notations for EKM.

Notation	Description
$P_{h_j}$	Polynomial Value
$ID_{Li}$	ID of member Node $L_i$
$h(ID_{Li})$	Hash of Node ID
$\gamma, n$	Cluster Size, Network Size
$N_{ID_{Li}}$	Array of Node IDs from $i = 1$ to $n$
$R$	Number of Random Hash Values
$CH, BS$	Cluster Head, Base Station
$K_{C-B}$	Key between <i>CH</i> and <i>BS</i>
$K_M$	Master Key
$K_{CL}$	Key between <i>CH</i> and $L_i$
$MAC(L_i)$	Message Authentication Code of $L_i$
$LIST_{L_i}$	List of ordinary sensors
$A_h(ID)$	Array of Hash values
$P_{L_i}$	Polynomial for $L_i$
$M_1-M_4$	Message 1 to 4
$C_1-C_3$	Cipher texts of $M_1$ to $M_3$
$H_{L_i}$	Hash calculated at $L_i$
$P_{L_i}, P_{CH_j}, P_{BS}$	Concatenated string at $L_i, CH_j$ and <i>BS</i> for transmission

Every common node has been preloaded with a hash function, master key, encryption and decryption functions along with a unique ID by the sink. The *CH* is also pre-loaded with similar credentials along with a polynomial calculation function. The *CH* and sink server have secure authentication mechanisms, including *ECC* which is hard to reverse. The *CH* forwards this request after making it secure to a sink that verifies the request and responds to the *CH* granting or denying this request. Figure 4 shows that the *CH* receives an encrypted message  $E_{KM}(ID_{L_i} || MAC(ID_{L_i}))$  containing the ID of  $L_i$  and  $MAC(ID_{L_i})$ . The *CH* selects the random IDs and hashing to calculate a polynomial using Equation (3). Finally, the *CH* transmits the encrypted message  $E_{KM}(Ph || MAC(Ph))$  to the *L*-sensors.



Initially, the polynomial  $P_h$  is calculated by using Equation (2) where  $x$  is a pre-defined significant value, and the hashes  $h$  are calculated for all member node identities from  $ID_{L_a}$  to  $ID_{L_{a+\gamma}}$  where  $\gamma$  be the cluster size. We further reduced the computation overhead by taking XOR of a randomly selected number of hash values instead of taking the XOR of all values. The CH generates the hashes of all IDs received from neighboring  $L$ -sensors using the initialize procedure.

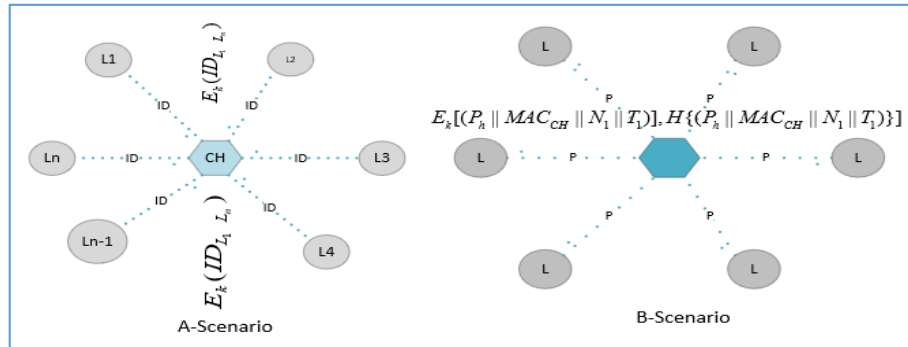


Figure 4. Intra-cluster key establishment.

After the hashing calculation, the CH makes a polynomial  $P_h$  by taking XOR of randomly chosen  $R$  hash values as shown in Equation (3), where  $a$ ,  $b$  and  $c$  are indices of randomly selected node IDs for  $R = 3$ :

$$P_h = (x - h_a(ID_{L_a}) \text{ XOR } (x - h_{a+1}(ID_{L_{a+1}})) \dots \text{ XOR } (x - h_{a+\gamma}(ID_{L_{a+\gamma}}))) \quad (2)$$

$$P_h = (x - h_a(ID_{L_a}) \text{ XOR } (x - h_b(ID_{L_b})) \text{ XOR } (x - h_c(ID_c))) \quad (3)$$

Moreover, the algorithmic steps of Equation (3) are illustrated in Figure 5. We recommend using  $R = 3$ , i.e., randomly taking three hash values and using XOR to generate the polynomial. It is enough to create confusion for the attacker in the selection of IDs out of  $N_{IDS}$ .

```

public:
Set  $N_{IDS}$  as Array //To store neighbor IDs
Set  $A_{h(ID)}$  as ArrayList //To store hash of IDs
Set  $P_h$  //To save Polynomial
procedure Initialize() {
  for  $i = 1$  to  $\text{sizeof}(N_{IDS}) - 1$ 
     $A_{h(ID)}[i] = \text{hash}(N_{IDS}[i])$ 
  end }
procedure generatePolynomial() {
Set  $R$  as Array //Random Node IDs <  $N_{IDS}$ 
 $P_h = x - A_{h(ID)}[R[0]]$ 
  for  $i = 1$  TO  $\text{sizeof}(R) - 1$ 
 $P_h = P_h \text{ XOR } (x - A_{h(ID)}[R[i]])$ 
end }

```

Figure 5. Polynomial calculation algorithm.

#### 4.1. Addition of New Node

After deployment of WSN, the addition of a new sensor node in the cluster may also occur. For new node addition, the base station plays a vital role in the authentication process and performs the setup

phase. Into the network setup model phase, the BS introduces the new sensor to the CH. The sensor node is pre-loaded with the master key ( $K$ ), a one-way hash function  $h(x)$  and a randomly selected node id  $ID_{L_i}$ . After that, the BS broadcasts an encrypted message containing nonce<sub>B</sub>,  $N_{ID_{L_i}}$  (list of  $L$ -sensor IDs) and  $MAC\_ADRS_{ID_{L_i}}$  to all cluster head nodes as illustrated in Protocol-1. CH replies with an encrypted message containing  $ID_{CH}$  and nonce<sub>B</sub>. The cluster head multicasts the session key, list of member nodes and MAC in encrypted form. This ensures the originality and reliability of the node joining process. The new node addition protocol and flow diagram are depicted in Figures 6 and 7, respectively, where before new node addition CH get an authentication from the BS. After successful authentication, CH runs Protocol-2 for verification using secret credentials.

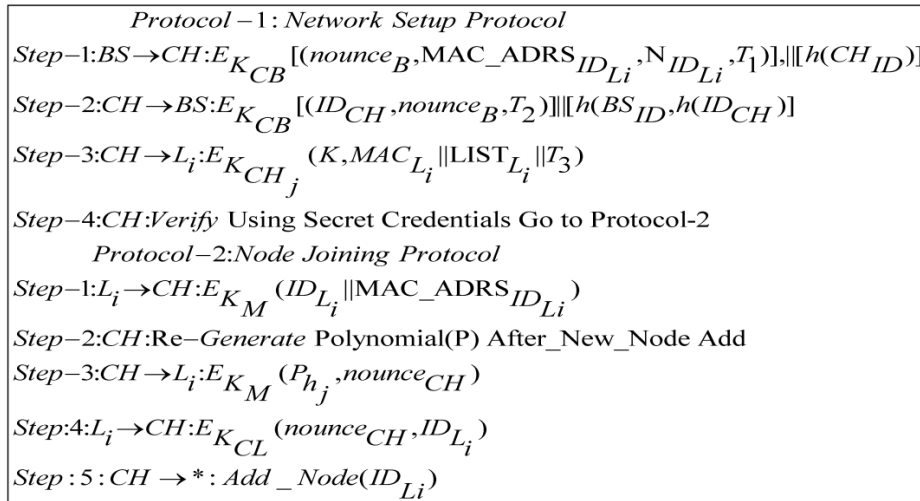


Figure 6. Network setup protocol.

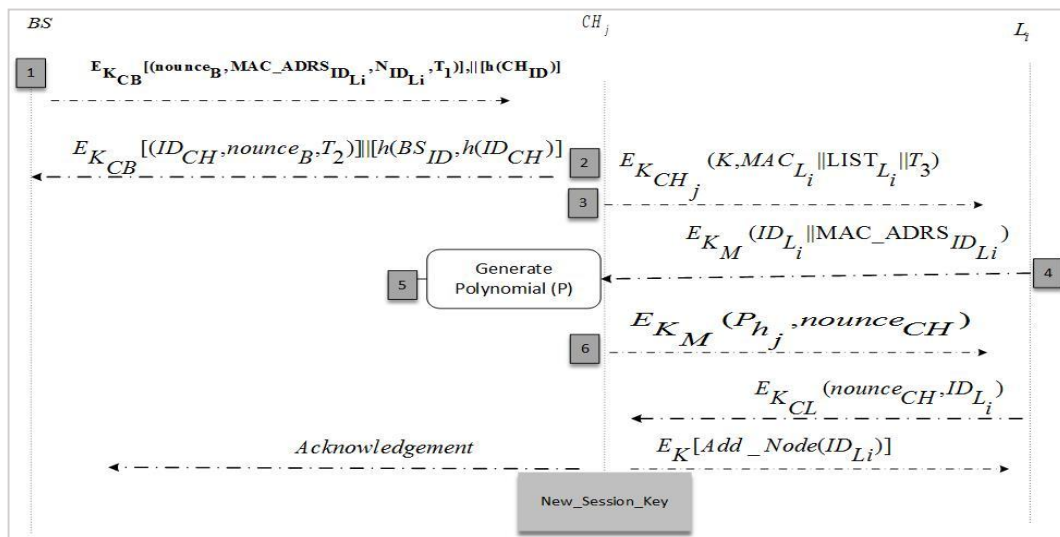


Figure 7. Node addition protocol flow diagram.

The new  $L$ -sensor is deployed randomly in the sensor network area. The new  $L$ -sensor encrypts its id and  $MAC\_ADRS_{ID_{L_i}}$  through its pre-loaded master key  $K_M$ . The cluster head, which already has a list of new sensors, verifies it by comparing the received MAC with its list. After verification, it replies with polynomials  $P_{h_j}$  encrypted using  $K_M$ . The new  $L$ -sensor decrypts the message and gets a polynomial to calculate a cluster key  $E_{K_{CL}}$  between CH and  $L_i$ . Finally, CH informs all members to add  $ID_{L_i}$  in member list as illustrated in Protocol 2. If a new node is a malicious node, then the CH sends an adversary

identification message to the base station. After the deployment in a heterogeneous network, new nodes are placed in such a position where they can communicate directly with the cluster head. Figure 6 shows the overall scenarios of Protocol-1 in steps 1 to 3 whereas Protocol-2 is in steps 4 to 7.

#### 4.2. Migration of Node

To increase the lifetime of a WSN network, we have proposed a secure node join protocol for moved nodes. The main purpose of this phase is to add displaced nodes to existing clusters by the CH sensing range in their locality. There are instances in which a node can belong to two clusters, inducing a loss of capability in the different one, which will resolve this issue; the cluster heads that do not have a specific sensor kind in their clusters invite the missing sensor to join or suggest another CH according to its sensing range. This achieves better security in an environment where nodes are mobile and leave/join the cluster frequently. We have maintained a binary search tree for all network nodes at the BS that is aware of each leave and join operation. Similarly, a Binary Search Tree (BST) is also maintained at the CH to keep the node list updated and maintain a record of leaving nodes. It also keeps a list of rejected nodes which do not have security credentials as per issued by BS. This guards against intruders joining the network during migration from other networks. We have supposed that an intrusion detection system is working to identify those nodes that were trustworthy but now are performing malicious activities due to some attack. Such nodes are also added to the rejected list. Due to this record keeping, the CH intimates in a timely way the nodes that want to send a message to a node that has left the network or been set as malicious to saves extra communication costs at low-powered nodes. During the node migration and addition scenario, the protocol in Figure 8 is described in a stepwise manner in following section. Figure 9 elucidates these protocol steps in a node wise interaction scenario.

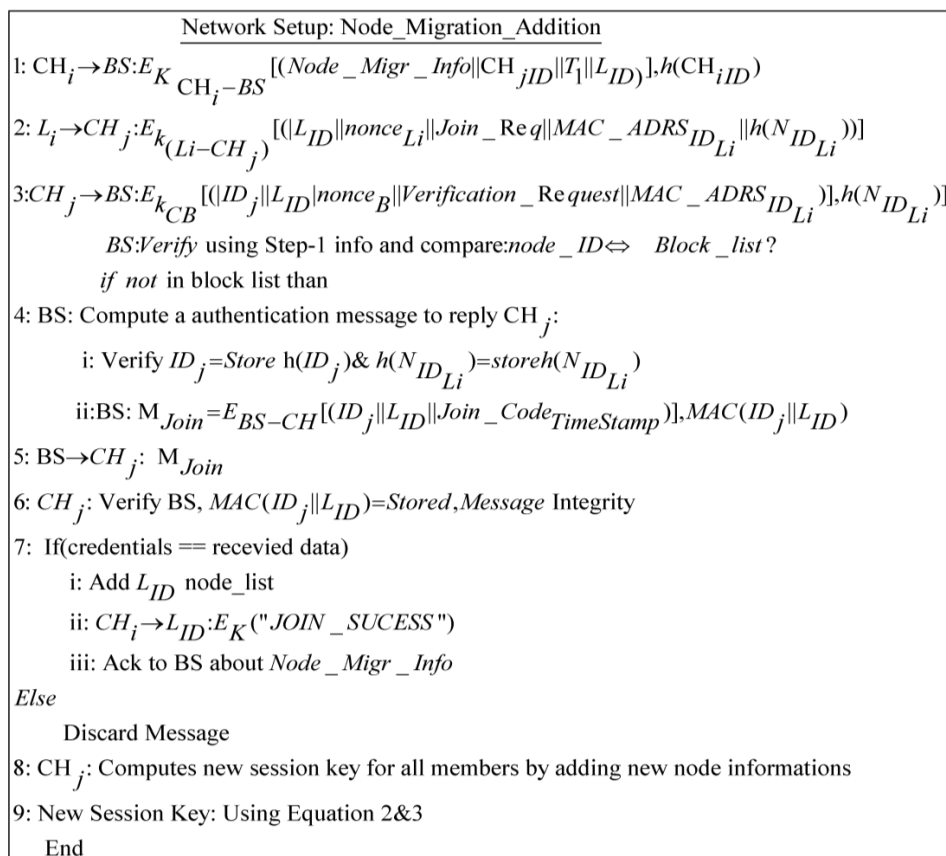


Figure 8. Node migrations and addition protocol.

- Steps 1–3 Before node migration,  $CH_i$  sends a node migration notation to BS with the valid ID of a sensor node that needs to migrate. A migrated node sends a joining request to join new cluster  $CH_j$  with a pre-shared key and transmits secret credentials for the joining process. Upon receiving a migrated node joining request, the  $CH_j$  get verification from the BS by sending a verification request. The BS verifies or rejects the node joining procedure by comparing said node’s identity in the block node database.
- Steps 4–5 After the migrated node joining request verification, the BS computes an authentication message for  $CH_j$  and transmits it in encrypted form using a shared key. The  $CH_j$  verifies the message credibility using secret parameters before adding the migrated node in its cluster. Carrying out verification,  $CH_j$  sends a join success message to the migrated node and an acknowledgment bit to the BS
- Steps 6–7 Upon the migrated node successfully joining,  $CH_j$  computes a fresh polynomial using Equations (2) and (3), and updates the cluster key for future communication and process aborts.

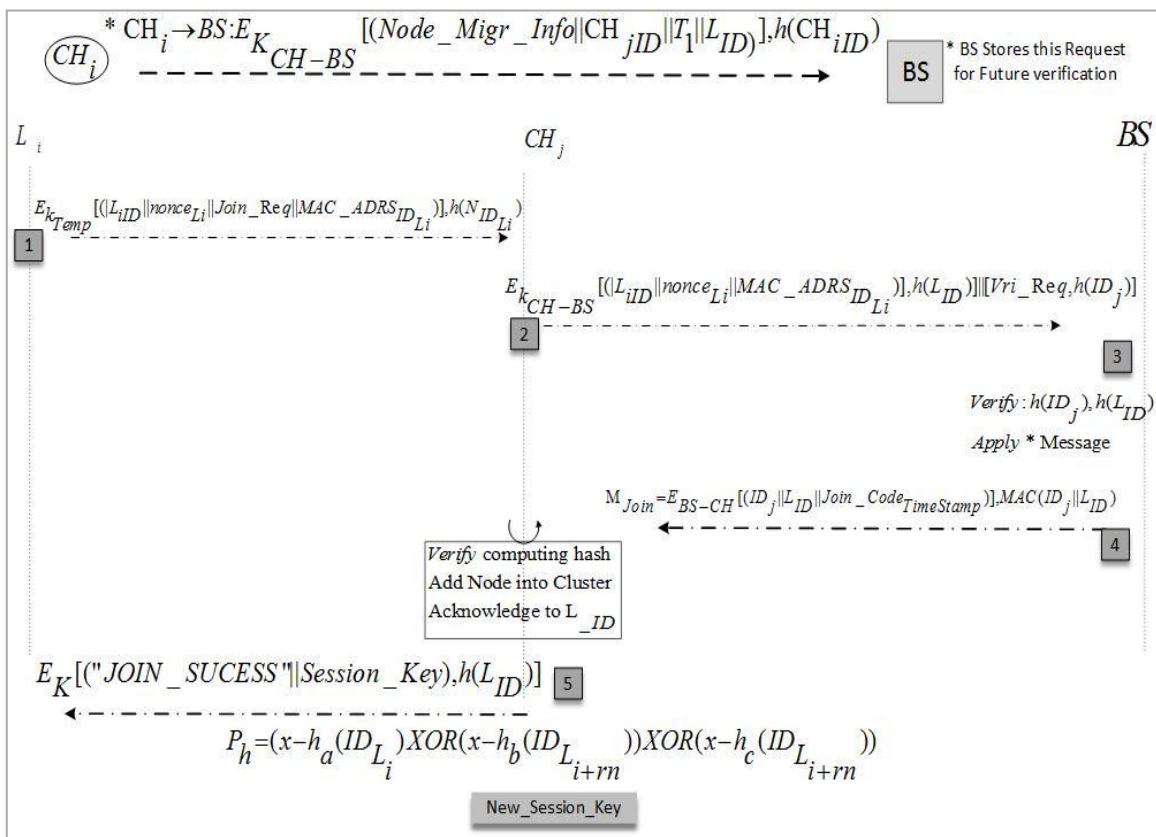


Figure 9. Node migration setup and new intra-group session keys.

### 5. EKM Formal Specification

We have carried out aspect modeling the use of Non-monotonic Cryptographic Protocol (NCP), as well referred to as Rubin logic [43] by analyzing and verify the EKM protocol as much as possible through formal specifications. It validates the protocol as approximating through standard requirements about cryptographic operations like node authentication, message integrity, message freshness, encryption, decryption, etc. It additionally helps to become aware of any deficiencies within the proposed protocol and possibilities regarding compromise attacks. It is close to the real implementation and flow of programming functions. In Rubin logic, the entities are allocated roles and a global set is additionally maintained where data about the protocol is maintained in sets then these also keep updated states on the users after every updateable operation. Global sets are available

in conformity with entire the part nodes, however, may stand labeled as secret, observer, rule and essential sets.

A local set is maintained at each entity or node and can be classified into possession POSS(), belief BEL(), seen and behavior sets BL(), the explicit specifications from [13] of these sets investigated. By applying Rubin logic on EKM, the local set involved is in Table 2, and then its verification and analysis part provides a detailed overview of all sets continue under the grouping of the local set. Local sets for all the participants including sender, receiver and actor nodes are independently managed. A possession set POSS (entity) holds all the constraints involved in message encryption, decryption and other procedures executed at a local RAM of each entity as described in the section below. A behavior list BL() holds the list of operations and input arguments that are performed in near to execution steps by entities  $L_i$ ,  $CH$  and  $BS$ .

**Table 2.** Local set for EKM.

1. Sender ( $L_i$ )	
POSS( $L_i$ ) = { $ID_{L_i}$ , $K_{L_i - CH_j}$ }	Receive( $CH_j$ , { $ID_{BS}$ , $C_3$ })
BEL( $L_i$ ) = {#( $ID_{L_i}$ ), #( $K_{L_i - CH_j}$ ) }	Split({ $ID_{BS}$ , $C_3$ })
BL( $L_i$ ) = Hash( $h(\cdot)$ ; $ID_{L_i}$ ) $\rightarrow H_{L_i}$	Decrypt({ $C_3$ }, $K_{CH_j - BS}$ ) to get
Concat( $ID_{L_i}$ , $nonce_{L_i}$ , $Join\_Req$ , $MAC\_ADRS_{L_i}$ , $H_{L_i}$ ) $\rightarrow P_{L_i}$	[ $ID_{BS}$ , $ID_{L_i}$ , $Join\_Code_T$ , $MAC(Concat(ID_{BS}, ID_{L_i}))$ ]
Encrypt( $\{P_{L_i}\}$ , $K_{L_i - CH_j}$ ) $\rightarrow C_1$	Verify $Join\_Code_T$ for $ID_{L_i}$
Send( $CH_j$ , { $ID_{L_i}$ , $C_1$ }) $\rightarrow M_1$	If false then abort
Update( $nonce_{L_i}$ , $H_{L_i}$ )	Add $ID_{L_i}$ in $node\_list$
Receive( $L_i$ , { $ID_{CH_j}$ , $M_4$ })	Concat( $ID_{CH_j}$ , 'Join_Success') $\rightarrow P_{CH_j}^{\sim}$
Split({ $ID_{CH_j}$ , $M_4$ })	Send( $BS$ , { $P_{CH_j}^{\sim}$ }) $\rightarrow M_4$
If message does not equal "Join_Success" then abort	<b>3. Base Station (BS)</b>
<b>2. Cluster Head (CH)</b>	POSS( $BS$ ) = { $ID_{BS}$ , $K_{BS - CH_i}$ , $K_{BS - CH_j}$ , $List\_Nodes$ }
POSS( $CH$ ) = { $ID_{CH_j}$ , $K_{L_i - CH_i}$ , $K$ , $K_{BS - CH_j}$ , $K_M$ }	BEL( $BS$ ) = {#( $ID_{BS}$ ), #( $K_{BS - CH_i}$ ), #( $K_{BS - CH_j}$ )}
BEL( $CH$ ) = {#( $ID_{CH_j}$ ), #( $K_{L_i - CH_i}$ ), #( $K_{BS - CH_j}$ ), #( $K_M$ )}	BL( $BS$ ) =
BL( $CH$ ) =	Receive( $BS$ , { $ID_{CH_j}$ , $C_2$ })
Receive( $CH_j$ , { $ID_{L_i}$ , $C_1$ })	Split({ $ID_{CH_j}$ , $C_2$ })
Split({ $ID_{L_i}$ , $C_1$ })	Decrypt({ $C_2$ }, $K_{CH_j - BS}$ ) to get [ $ID_{CH_j}$ , $ID_{L_i}$ , $nonce_{CH_j}$ ,
Decrypt({ $C_1$ }, $K_{CH_j - L_i}$ ) to get	$Ver\_Req$ , $MAC\_ADRS_{L_i}$ , $H_{L_i}$ ]
[ $ID_{L_i}$ , $nonce_{L_i}$ , $Join\_Req$ , $MAC\_ADRS_{L_i}$ , $H_{L_i}$ ]	Verify $L_i$ and check in $Block\_List$
Concat( $ID_{CH_j}$ , $ID_{L_i}$ , $nonce_{CH_j}$ , $Ver\_Req$ ,	If not verified then Abort
$MAC\_ADRS_{L_i}$ ) $\rightarrow P_{CH_j}$	Concat( $ID_{BS}$ , $ID_{L_i}$ , $Join\_Code_T$ ) $\rightarrow P_{BS}$
Encrypt( $\{P_{CH_j}\}$ , $H_{L_i}$ ), $K_{CH_j - BS}$ ) $\rightarrow C_2$	Encrypt( $\{P_{BS}$ , $MAC(Concat(ID_{BS}, ID_{L_i}))\}$ , $K_{CH_j - BS}$ ) $\rightarrow C_3$
Send( $BS$ , { $ID_{CH_j}$ , $C_2$ }) $\rightarrow M_2$	Send( $CH_j$ , { $ID_{BS}$ , $C_3$ }) $\rightarrow M_3$
Update( $M_{ID}$ )	Update( $M_{ID}$ )

The detailed analysis of the proposed scheme is presented in Table 2 by considering all communication actors. The sensor node  $L_i$  authenticates itself to its cluster head by sending a join request message containing  $Hash(h(\cdot); ID_{L_i})$ . Upon receiving this message,  $CH_i$  computes polynomial  $P_{L_i}$  and reply  $C_1$  in encrypted message  $M_1$  to  $L_i$ . Upon receiving  $L_i$  splits the received message and verifies it by getting  $CH$  and its information for joining a group. On the other hand,  $CH_i$  contains its own (ID), member node (IDs), master key, and encryption key to communicate with  $CH_i$  and member node  $L_i$  in its possession. In  $CH_i$ , we believe the received messages and participant information are reliable and trustworthy. Upon receiving credentials from member nodes,  $CH_i$  decrypts them by using the corresponding secret key and verifying the information after decrypting the ciphered text. Based on strong verification  $CH_i$  accepts or aborts the member node join, leave and migration requests as well as communicates with the BS. The third and the most important entity of this scenario is (BS), which has more computation, communication, and storage power.  $CH_i$  possess their own (ID),  $CH$ (IDs), secret keys and a list of member nodes with their (IDs) It believes it has secure communication mechanisms for ( $BS$ ) –  $CH_i$ , ( $CH_i$ ) –  $CH_i$  and strong, secure key generation protocols. It verifies,  $CH_i$  sensor nodes, and their authentication credentials and the validities of their identities.

### EKM Analysis and Verification

In this subsection, EKM is broken down for a node migration situation as discussed in Section 4.2. In this case, the key circulation is started by the base station that preloads the security credentials in  $L_i$  and then ( $L_i$ ) transmits a message  $M_1$  to the cluster heads  $CH_j$ . After receiving the message,  $CH_j$  gets the node ids of participant sensor node and calculates the shared session key. Upon the completion of the transmission operation, an update operation is performed to refresh the spectator list demonstrated as follows. Related parameters including messages, ciphers, nonce values, actual key, and hashes are saved in possession set at  $L_i$  during the key establishment phase as shown below. At the end of the migration, the forget operation removes the temporary values used in operations including  $nonce_{L_i}$ ,  $P_{L_i}$ ,  $H_{L_i}$ ,  $C_1$ ,  $M_1$ ,  $M_4$ .

$$POSS(L_i) = \{ID_{L_i}, K_{L_i-CH_j}, nonce_{L_i}, P_{L_i}, H_{L_i}, C_1, M_1, M_4\}$$

After receiving messages,  $CH_j$  decrypts them to retrieve a concatenated message that is further split to get the actual parameters. The possession set at  $CH_j$  contains the following parameters during execution of the abovementioned operations. The forget operation will result in removing the temporary values including  $P_{CH_j}$ ,  $nonce_{L_i}$ ,  $C_1$ ,  $C_2$ ,  $M_2$ ,  $M_3$ ,  $M_4$ ,  $H_{CH_j}$ ,  $P_{CH_j}$ ,  $nonce_{CH_j}$  nonce from the possession set:

$$POSS(CH_j) = \{ID_{CH_j}, K_{CH_j-L_i}, K_{CH_j-BS}, P_{CH_j}, nonce_{L_i}, C_1, C_2, M_2, C_3, M_4, H_{CH_j}, P_{CH_j}, nonce_{CH_j}, ID_{BS}, ID_{L_i}\}$$

The BS receives the message from  $CH_j$  and after decryption, it verifies the  $L_i$  and its security credentials. The possession set values at the BS contain temporary values as shown below. Forget operations are performed to remove the values  $P_{BS}$ ,  $C_2$ ,  $C_3$  and  $M_3$ :

$$POSS(BS) = \{ID_{BS}, K_{BS-CH_j}, P_{BS}, C_2, C_3, M_3, ID_{L_i}, Join-Code_T\}$$

The receiver node BS verifies the credentials sent by the  $CH_j$  and then returns a message containing  $Join-Code_T$  to  $CH_j$ . Finally, the possession set at  $L_i$  is refreshed after the completion of node migration from  $CH_i$  and successful joining at the  $CH_j$ .

## 6. Results and Analysis

The proposed scheme is simulated using NS2.35 using C on Fedora core-12. TCL is utilized for simulating the nodes in multiple clusters.  $H$ -sensors are separately configured for an initial energy of 10,000 Joules, and message transmission and receiving has a cost of 0.5809 J and 0.049 J, respectively. The transmission radii of the  $H$ -sensor and  $L$ -sensor are set at 300 m and 50 m, respectively. We deploy five clusters with varying cluster size from 10 to 100 in a region of  $1300 \times 1300$  m for measuring energy consumption, computational and storage overheads. The polynomial calculation algorithm is also implemented in Visual Studio 2013 using C#, as a programming language. PERL scripts are written to extract the resulting values from the trace file and plot the graph.

### 6.1. Computation Overheads

We have considered some basic operations  $v$  during long multiplication of two numbers  $x$ ,  $y$ . Computation cost is  $(2 \times v) \times v = 2 \times v^2$  by taking the same count of digits  $n$  for both numbers. Results of short multiplications are aligned under the respective numbers of  $\beta$ . To simplify, we put zeros in the empty positions. Figure 10a shows that a multiplication operation in DGP [18] takes 449.2 significant process shares and EKM consumes only 4.7 operation shares. The computational complexity to perform  $n$ -XOR operations is  $O(n)$  in the EKM. The computational complexity of adding

two binary numbers is  $O(n)$  whereas that of multiplying two numbers is  $O(n^2)$ . Doubling the number of bits in values  $x, y$ , results in a quadruple value as  $(2n)^2 = 4n^2$ .

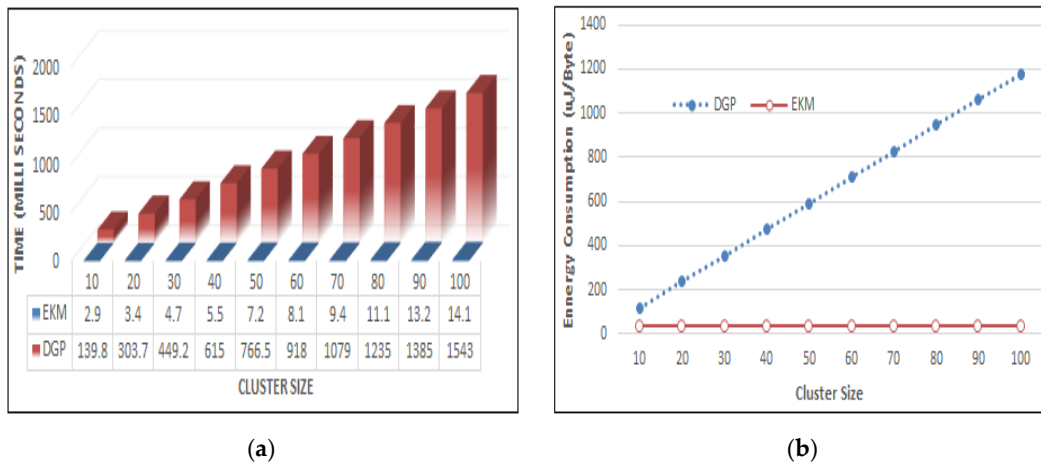


Figure 10. Polynomial computational cost in (a) and energy consumptions in (b).

To compute the hash value of a message using *SHA-1*, a sensor node consumed 5.9 µJ/Byte. For a 16-bit ID, it requires 11.8 µJ. For  $n = 20$  member nodes  $11.8 \times 20 = 236$  µJ energy is required just to calculate the hash of IDs. It is directly proportional to cluster size for DGP [18] as illustrated in Figure 10b. In the proposed EKM, only three hash values are randomly selected to calculate the actual key and hence it is independent of the cluster size. This greatly reduces the energy consumption without reducing the security strength.

### 6.2. Latency Time

Latency is the time consumed for a data packet to move from one cluster head in the sensor network to the  $L$ -sensors. Our result in Figure 11a indicates that the latency time is reduced considerably when it is compared with DGP [18]. EKM computed the cluster, key efficiency and reduced the delay on cluster head latency measured by finding the initiating protocol for session key and end time. In the proposed scenario the group head and member nodes participate during session key generation and symmetric parameters consuming less time in establishing a session key for group members. Figure 11a,b show the time delay with respect to group size and the proposed scheme shows less latency which is efficient.

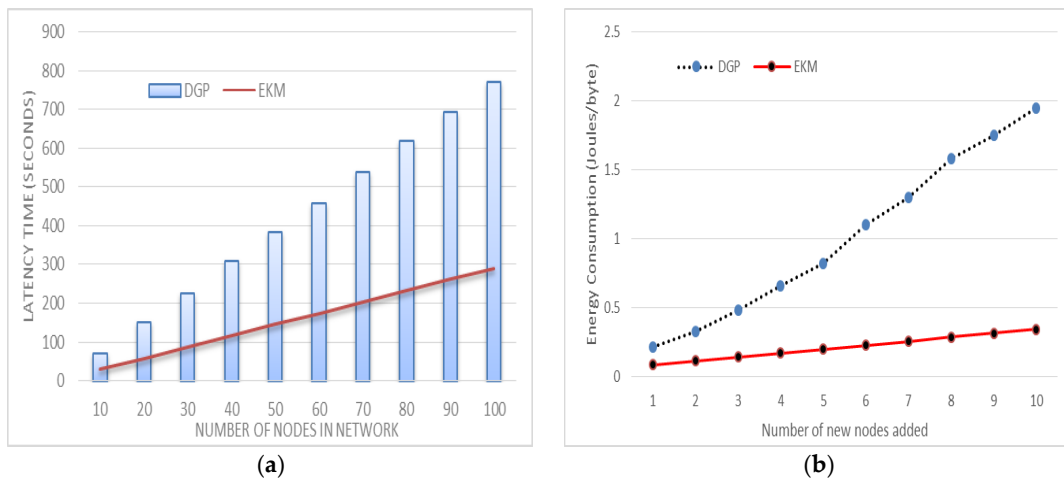


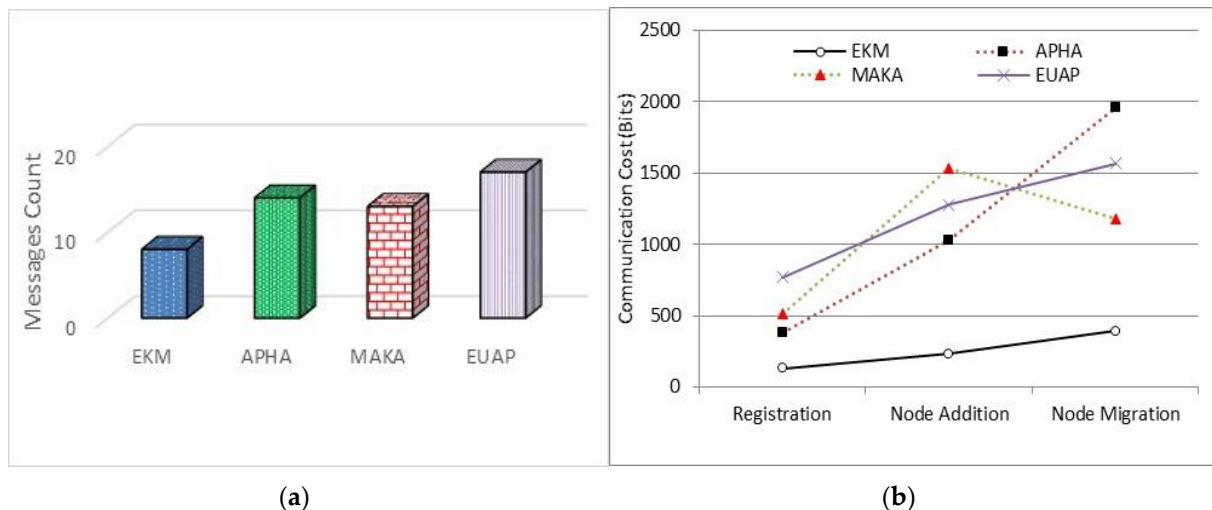
Figure 11. Latency comparisons in (a) communication energy in (b).

### 6.3. Communication Overhead

Communication cost is measured by taking the ratio of transmitting bytes to received bytes. EKM significantly reduces the number of messages transmitted during new node addition as compared to DGP [18], as depicted in Figure 11b. We have set the transmission cost  $T_c = 59.2 \mu\text{J}$  to transmit and the message receiving cost  $R_c = 28.6 \mu\text{J}$  per byte. The total energy consumed during transmission is  $m \times (T_c + R_c)$  where  $m$  is the number of messages. The lifetime of a sensor network is inversely proportional to its energy consumption. The proposed plan is a very computationally lightweight procedure outlined particularly for the client validation of WSNs in the IoT environment. The method guarantees high security while utilizing just straightforward hash and XOR calculations, along these lines, it requires more stockpiling than other related ideas. Since the novelty of the proposed plan is the “sensor-hub first-contact-approach”, together with its high security, the sensor hubs and the CH need to convey some additional capacity weight. In this subsection, we display the capacity utilization of our proposed plan and compare it with related ideas.

EKM refreshes keys during node migration, key freshness and node joining more than other similar ones, but it is still below the existing schemes concerning performance and efficiency. The analysis completed by assessing every scheme in detail and calculating the whole length of every variable the CH or the sensor node need to save into its memory, while efficiently executing one registration and authentication stage. Before the review, with the end goal of a targeted investigation, we firstly determined the length of each type of a variable, which could be used in the schemes, regardless of the lengths some authors proposed.

We have compared our proposed EKM scheme for efficiency of communication and number of of data exchanges in contrast to authentication protocol for healthcare applications (APHA) [44], mutual authentication and key agreement (MAKA) scheme [45] and enhanced user authentication protocol (EUAP) [46] that are recently proposed in IoT scenarios. We used some messages exchanged after deployment of the network and the results show that the proposed protocol is efficient for all possible scenarios. The communication and data used between all participating entities for secure node registration, migration and addition has been considered in Figure 12a,b.



**Figure 12.** Performance analysis on communication cost where (a) represents the message count and (b) represents the size of data exchanged.

### 6.4. Storage Overhead

The amount of memory required to store security parameters and key space is considered the storage cost. The total number of sensor nodes in a network are  $nT = nH + nL$  where  $nH$  and  $nL$  represent the total high power nodes and member nodes, respectively. DGP [18] requires multiplication



factors  $F = ID \times nL$  to calculate a polynomial. It stores  $(ID_B \times (\gamma - 1)) + (h(ID)_B \times (\gamma - 1))$ . bytes where  $ID_B$  represents the size of ID in bytes,  $\gamma$  is cluster size, and  $h(ID)_B$  is the size of the hash values in bytes. For  $\gamma = 30$ , it requires 30 factors to be multiplied for calculating just a one-time polynomial. Moreover, we have considered a value of  $ID_B = 16$ -bits and  $h(ID)_B = 160$  bits for a secure hash function. It could be much expensive as the cluster size increases. In proposing *EKM*, the suggested value of  $F = 3$  and requires 87% less factors as compared to DGP [18] with a cluster size of 30-nodes as illustrated in Figure 13.

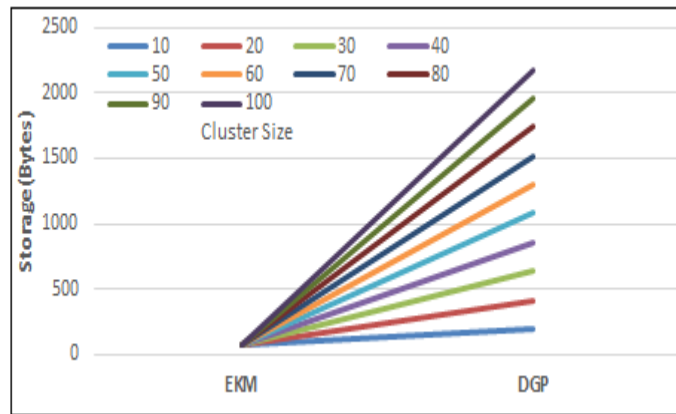


Figure 13. Storage overhead comparison.

6.5. Energy Consumption

To compute the hash value of a message using *SHA-1*, a sensor node consumed 5.9  $\mu$ J/Byte. For a 16-bit ID, it requires 11.8  $\mu$ J. For  $n = 20$  member nodes  $11.8 \times 20 = 236 \mu$ J energy is required just to calculate the hash of IDs. It is directly proportional to cluster size for DGP [18] as illustrated in Figure 14b. In the proposed *EKM*, only three hash values are randomly selected to calculate the actual key and hence it is independent of the cluster size. This greatly reduces the energy consumption without reducing the security strength.

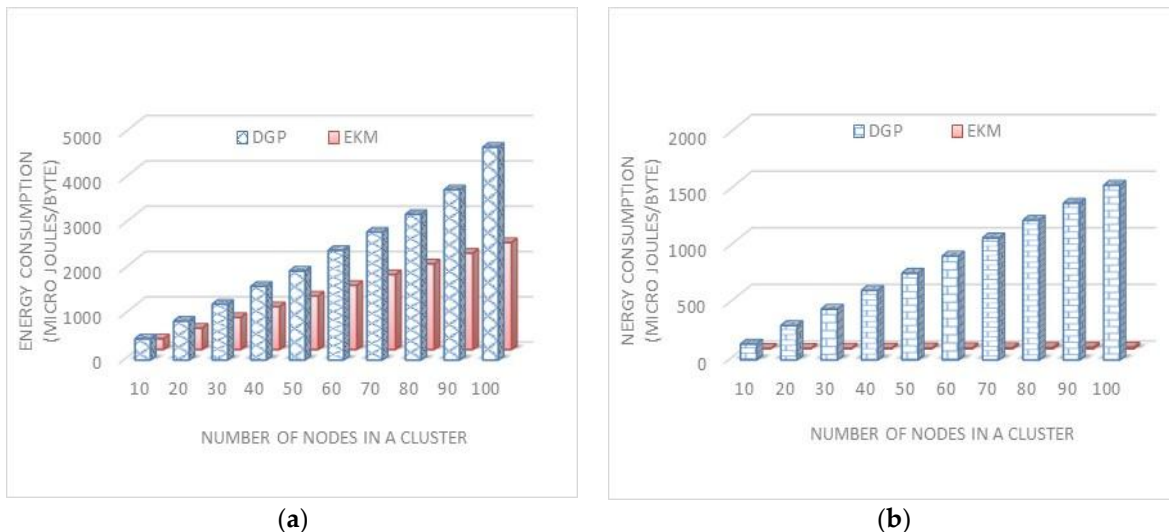


Figure 14. Energy for polynomial generation is presented in (a) and key generation energy in (b).

The amount of energy required to compute a polynomial by the cluster is linear and existing schemes have a multiple of  $n$ , where  $n$  is the number of nodes in a cluster. In the existing technique,

cluster head calculates the polynomial after receiving all the ids of the  $L$ -sensors in its communication range. To compute the secure hash of a sensor node consumes  $5.9 \mu\text{J}/\text{Byte}$  [6]. To compute the hash of  $n$   $L$ -sensors, the sensor node consumes the bulk of energy. We enhanced the existing technique for a more comprehensive decision about energy consumption for generating polynomials at cluster heads. To compute on a hash of an  $L$ -sensor id uses  $5.9 \mu\text{J}/\text{Byte}$ , so when we generated polynomials through the existing technique supposing the number of  $L$ -sensors is  $n$  then the consumption of energy is  $n \times 5.9$  which means it is directly proportional to the increment of  $L$ -sensors. The simulation of the proposed technique saves four-time more energy than existing techniques. A hash has 128-bit data to be processed which mean that  $16\text{-byte} \times 5.9 \mu\text{J}$  is consumed for one way hashing.

These results show that the generation of polynomials through multiplication is more energy consuming than the XOR. The number of sensor nodes varies from 10 to 100. The simulation results are shown in Figure 14a,b. We can find that the DGP [18] scheme consumes much more energy as compared to our proposed EKM scheme. This is due to the extensive energy consumption during execution of point multiplication, pairing, and mapping algorithms. Each node needs to run these algorithms many times according to the number of neighbors around the computing node.

## 7. Conclusions

The main objective of the proposed research was to enhance and optimize the polynomial generation during node joining in WSN clusters. In existing schemes, the concept of sequential multiplication was used to calculate the polynomial, which is expensive. This polynomial is recalculated every time a node joins or leaves the cluster, increasing communication, computation and storage overheads. EKM is based on generating polynomials using XOR of randomly selected values. It increases the lifetime of sensor networks and reduces communication cost and memory overheads. Multiplication operations require a complexity of  $O(n^2)$  whereas XOR requires  $O(n)$ . Moreover, the majority of schemes have opted for a centralized approach, where a central node is fully responsible for secure authentication. The proposed research adopted a hybrid approach which is adequate for WSNs as this shares the tasks of the group controller among group members. During our systematic and simulation results, we have exposed that our proposed scheme requires a constant number of hash values to calculate a polynomial whereas in existing schemes that number increases rapidly with cluster size. Our scheme requires 87% less factors for a cluster size of 30 nodes per cluster. In the future, we plan to implement EKM in the Ubiquitous to Internet of Thing (U2IoT) scenario.

**Acknowledgments:** This work was funded by the National Natural Science Foundation of China (61471035), and Fundamental Research Funds for the Central Universities (06105031, 06500010). In particular, it was supported by Cybermatics and Cyberspace International Science and Technology Cooperation Base.

**Author Contributions:** Zahid Mahmood and Ata Ullah Ghafoor conceived and designed the experiments; Zahid Mahmood performed the experiments; Huansheng Ning analyzed the data and overall proposed protocols, models, structure and flow of entire paper; Ata Ullah performed the formal modeling and verification of proposed scheme. Zahid Mahmood contributed reagents/materials/analysis tools and also wrote the paper along with literature.

**Conflicts of Interest:** The authors declare that there is no conflict of interest.

## References

1. Jawhar, I.; Mohamed, N.; Agrawal, D.P. Linear wireless sensor networks: Classification and applications. *J. Netw. Comput. Appl.* **2011**, *34*, 1671–1682. [[CrossRef](#)]
2. Yick, J.; Mukherjee, B.; Ghosal, D. Wireless sensor network survey. *Comput. Netw.* **2008**, *52*, 2292–2330. [[CrossRef](#)]
3. Karlof, C.; Sastry, N.; Wagner, D. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, Baltimore, MD, USA, 3–5 November 2004; pp. 162–175.
4. Abbasi, A.A.; Younis, M. A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.* **2007**, *30*, 2826–2841. [[CrossRef](#)]

5. Djenouri, D.; Khelladi, L.; Badache, N. A survey of security issues in mobile ad hoc networks. *IEEE Commun. Surv.* **2005**, *7*, 2–28. [[CrossRef](#)]
6. Venkatraman, K.; Daniel, J.V.; Murugaboopathi, G. Various attacks in wireless sensor network: Survey. *Int. J. Soft Comput. Eng.* **2013**, *3*, 208–211.
7. Diop, A.; Qi, Y.; Wang, Q. An Efficient and Secure Session Key Management Scheme for Cluster Based Wireless Sensors Networks. In Proceedings of the Joint International Conference on Pervasive Computing and the Networked World, Vina del Mar, Chile, 5–7 December 2013; pp. 33–44.
8. Harn, L.; Hsu, C.-F.; Ruan, O.; Zhang, M.-Y. Novel design of secure end-to-end routing protocol in wireless sensor networks. *IEEE Sens. J.* **2016**, *16*, 1779–1785. [[CrossRef](#)]
9. Hosangadi, A.; Kastner, R.; Fallah, F. Energy Efficient Hardware Synthesis of Polynomial Expressions. In Proceedings of the 18th IEEE International Conference on VLSI Design, Washington, DC, USA, 3–7 January 2005; pp. 653–658.
10. Saied, Y.B.; Olivereau, A.; Zeglache, D. Energy Efficiency in M2M Networks: A Cooperative Key Establishment System. In Proceedings of the 2011 3rd International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), Budapest, Hungary, 5–7 October 2011; pp. 1–8.
11. Sipser, M. Introduction to the theory of computation. *ACM SIGACT News.* **1996**, *27*, 27–29. [[CrossRef](#)]
12. Roche, D.S. Space-and Time-Efficient Polynomial Multiplication. In Proceedings of the International Symposium on Symbolic and Algebraic Computation, Seoul, Korea, 28–31 July 2009; pp. 295–302.
13. Gao, S.; Van Hoeij, M.; Kaltofen, E.; Shoup, V. The computational complexity of polynomial factorization. *Am. Inst. Math.* **2006**, *364*, 1–5.
14. Kumar, P.; Lee, H.-J. Security issues in healthcare applications using wireless medical sensor networks: A survey. *Sensors* **2011**, *12*, 55–91. [[CrossRef](#)] [[PubMed](#)]
15. Lopez, J.; Roman, R.; Alcaraz, C. Analysis of security threats, requirements, technologies and standards in wireless sensor networks. *Found. Secur. Anal. Des. V.* **2009**, *5705*, 289–338.
16. Misra, S.; Xue, G. Efficient anonymity schemes for clustered wireless sensor networks. *Int. J. Sens. Netw.* **2006**, *1*, 50–63. [[CrossRef](#)]
17. Banihashemian, S.; Bafghi, A.G. A New Key Management Scheme in Heterogeneous Wireless Sensor Networks. In Proceedings of the 12th International Conference on Advanced Communication Technology (ICACT), Gangwon-Do, Korea, 7–10 February 2010; pp. 141–146.
18. Traynor, P.; Cao, G.; La Porta, T. The Effects of Probabilistic Key Management on Secure Routing in Sensor Networks. In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), Las Vegas, NV, USA, 3–6 April 2006; pp. 659–664.
19. Li, M.; Long, J.; Yin, J.; Wu, Y.; Cheng, J. An Efficient Key Management Based on Dynamic Generation of Polynomials for Heterogeneous Sensor Networks. In Proceedings of the 2nd International Conference on Computer Engineering and Technology (ICCET), Chengdu, China, 16–18 April 2010.
20. Zeng, Y.; Zhao, B.; Su, J.; Yan, X.; Shao, Z. A Loop-Based Key Management Scheme for Wireless Sensor Networks. In Proceedings of the IFIP International Conference on Embedded and Ubiquitous Computing, Taipei, Taiwan, 17–20 December 2007; pp. 103–114.
21. Kausar, F.; Hussain, S.; Yang, L.T.; Masood, A. Scalable and efficient key management for heterogeneous sensor networks. *J. Supercomput.* **2008**, *45*, 44–65. [[CrossRef](#)]
22. Kausar, F.; Hussain, S.; Park, J.H.; Masood, A. Secure Group Communication with Self-Healing and Rekeying in Wireless Sensor Networks. In Proceedings of the International Conference on Mobile Ad-Hoc and Sensor Networks, Beijing, China, 12–14 December 2007; pp. 737–748.
23. Yang, Q.; Li, Q.; Li, S. An Efficient Key Management Scheme for Heterogeneous Sensor Networks. In Proceedings of the 4th International Conference on Wireless Communications, Networking and Mobile Computing, Dalian, China, 12–14 October 2008; pp. 1–4.
24. Tian, B.; Han, S.; Dillon, T.S. A Key Management Scheme for Heterogeneous Sensor Networks Using Keyed-Hash Chain. In Proceedings of the 5th International Conference on Mobile Ad-Hoc and Sensor Networks, Fujian, China, 14–16 December 2009; pp. 448–456.
25. Naranjo, J.A.M.; Antequera, N.; Casado, L.G.; López-Ramos, J.A. A suite of algorithms for key distribution and authentication in centralized secure multicast environments. *J. Comput. Appl. Math.* **2012**, *236*, 3042–3051. [[CrossRef](#)]

26. Ghafoor, A.; Sher, M.; Imran, M.; Saleem, K. A Lightweight Key Freshness Scheme for Wireless Sensor Networks. In Proceedings of the 12th International Conference on Information Technology-New Generations (ITNG), Las Vegas, NV, USA, 13–15 April 2015; pp. 169–173.
27. Wu, L.-C.; Hung, C.-H.; Kuo, W.-C. Group Key Management based on (2, 2) Secret Sharing. *TIIS* **2014**, *8*, 1144–1156.
28. Akyildiz, I.F.; Su, W.; Sankarasubramaniam, Y.; Cayirci, E. A survey on sensor networks. *IEEE Commun. Mag.* **2002**, *40*, 102–114. [[CrossRef](#)]
29. Chen, X.; Makki, K.; Yen, K.; Pissinou, N. Sensor network security: A survey. *IEEE Commun. Surv. Tutor.* **2009**, *11*. [[CrossRef](#)]
30. Roman, R.; Alcaraz, C.; Lopez, J. A survey of cryptographic primitives and implementations for hardware-constrained sensor network nodes. *Mob. Netw. Appl.* **2007**, *12*, 231–244. [[CrossRef](#)]
31. Zhang, J.; Varadharajan, V. Wireless sensor network key management survey and taxonomy. *J. Netw. Comput. Appl.* **2010**, *33*, 63–75. [[CrossRef](#)]
32. Staddon, J.; Miner, S.; Franklin, M.; Balfanz, D.; Malkin, M.; Dean, D. Self-Healing Key Distribution with Revocation. In Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, 12–15 May 2002; pp. 241–257.
33. Han, S.; Tian, B.; He, M.; Chang, E. Efficient threshold self-healing key distribution with sponsorship for infrastructureless wireless networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*. [[CrossRef](#)]
34. Gu, J.; Xue, Z. An Efficient Self-Healing Key Distribution with Resistance to the Collusion Attack for Wireless Sensor Networks. In Proceedings of the IEEE International Conference on Communications (ICC), Cape Town, South Africa, 23–27 May 2010; pp. 1–5.
35. Han, S.; Tian, B.; Zhang, Y.; Hu, J. An Efficient Self-Healing Key Distribution Scheme with Constant-Size Personal Keys for Wireless Sensor Networks. In Proceedings of the IEEE International Conference on Communications (ICC), Cape Town, South Africa, 23–27 May 2010; pp. 1–5.
36. Ye, N.; Zhu, Y.; Wang, R.-C.; Lin, Q.-M. An efficient authentication and access control scheme for perception layer of internet of things. *Source Appl. Math. Inf. Sci.* **2014**, *8*, 1617. [[CrossRef](#)]
37. Rams, T.; Pacyna, P. A survey of group key distribution schemes with self-healing property. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 820–842. [[CrossRef](#)]
38. Liu, H.; Ning, H. Zero-knowledge authentication protocol based on alternative mode in RFID systems. *IEEE Sens. J.* **2011**, *11*, 3235–3245. [[CrossRef](#)]
39. Ning, H.; Liu, H.; Mao, J.; Zhang, Y. Scalable and distributed key array authentication protocol in radio frequency identification-based sensor systems. *IET Commun.* **2011**, *5*, 1755–1768. [[CrossRef](#)]
40. Hsu, C.-F.; Harn, L.; Mu, Y.; Zhang, M.; Zhu, X. Computation-efficient key establishment in wireless group communications. *Wirel. Netw.* **2017**, *23*, 289–297. [[CrossRef](#)]
41. Odelu, V.; Das, A.K.; Goswami, A. A New Key Management Scheme for a User Hierarchy based on a Hybrid Cryptosystem. *Smart Comput. Rev.* **2013**, *3*, 42–54. [[CrossRef](#)]
42. Sultana, S.; Ghinita, G.; Bertino, E.; Shehab, M. A lightweight secure scheme for detecting provenance forgery and packet dropattacks in wireless sensor networks. *IEEE Trans. Dependable Secure Comput.* **2015**, *12*, 256–269. [[CrossRef](#)]
43. Rubin, A.D.; Honeyman, P. Nonmonotonic Cryptographic Protocols. In Proceedings of the Computer Security Foundations Workshop VII, (CSFW 7), Franconia, NH, USA, 14–16 June 1994; pp. 100–116.
44. Li, X.; Niu, J.; Kumari, S.; Liao, J.; Liang, W.; Khan, M.K. A new authentication protocol for healthcare applications using wireless medical sensor networks with user anonymity. *Secur. Commun. Netw.* **2016**, *9*, 2643–2655. [[CrossRef](#)]
45. Kumari, S.; Li, X.; Wu, F.; Das, A.K.; Arshad, H.; Khan, M.K. A user friendly mutual authentication and key agreement scheme for wireless sensor networks using chaotic maps. *Future Gener. Comput. Syst.* **2016**, *63*, 56–75. [[CrossRef](#)]
46. Choi, Y.; Lee, D.; Kim, J.; Jung, J.; Nam, J.; Won, D. Security enhanced user authentication protocol for wireless sensor networks using elliptic curves cryptography. *Sensors* **2014**, *14*, 10081–10106. [[CrossRef](#)] [[PubMed](#)]



© 2017. This work is licensed under  
<https://creativecommons.org/licenses/by/4.0/> (the “License”).  
Notwithstanding the ProQuest Terms and Conditions, you may use this  
content in accordance with the terms of the License.