Certificates Shared Verification Key Management for SurvSec Security Architecture

Mohamed Helmy Megahed
School of Information Technology and Engineering
University of Ottawa
Ottawa, Canada
mmega080@uottawa.ca

Dimitrios Makrakis
School of Information Technology and Engineering
University of Ottawa
Ottawa, Canada
dimitris@site.uottawa.ca

Hisham Dahshan
Communications Department
Military Technical College
Egyptian Armed Forces
Cairo, Egypt
hishamdahshan@yahoo.com

***Abstract*—SurvSec is a novel security architecture for reliable network recovery from base station BS failure of surveillance wireless sensor network (WSN) in hostile environment. Key management is the fundamental security mechanism in WSN which is needed for secure localization, secure clustering, secure data aggregation, secure authenticated broadcasting and secure routing. In this paper, a novel hybrid and dynamic key management scheme was proposed. This new scheme established secret keys between sensor nodes for SurvSec security architecture with high security level, high performance and low setup time. Hybrid key management provides high security level in the hostile environment however previous work assumed heterogeneous network utilizes high end sensor nodes (HSNs) with high power for high computations of certificates verification. This assumption provides attackers the best chance to destroy the network by targeting the HSNs. Also, HSN is connected to large number of nodes and there is no backup node for it. In addition, if the attackers target HSNs, then the connectivity and scalability will be affected where these nodes are points of failure. Moreover, previous work did not explain how to revoke compromised HSN. Furthermore, increasing the number of HSNs will increase the network deployment cost. Finally, if HSN is destroyed, nodes cannot have rekeying or addition of new nodes or revocation of compromised nodes. This paper proposed a hybrid scheme with homogenous network that uses some sensor nodes named as security managers (SMs) with a proposed novel mechanism called certificates shared verification to verify the certificates of group of nodes with distributed computations to overcome the absence of HSNs. This paper presents analytical evaluation and extensive simulation. The simulation results showed that at the cost of increasing communication overhead, the certificates shared verification mechanism was developed. Also, simulation results showed that the proposed scheme has lower computation overhead at SM side and lower setup time than HSN model. Both schemes have the same storage overhead.**

Keywords-Key Management; ***Dynamic***; ***Hybrid***; ***Certificate Shared Verification***.

I. INTRODUCTION

Researchers have investigated WSNs key management schemes and divided them into three categories. The category based on encryption has three classes which are: (i) symmetric key based key management [1-8]; (ii) asymmetric key based key management [9, 10] and (iii) hybrid key management [11-13]. The category based on location produced location based key management [14-16]. The category based on dynamicity and it has two classes which are static key management and dynamic key management [17, 18].

Hybrid key management combines the advantages of symmetric key and public key and it is the best solution for the hostile environment. Previous researches for hybrid key management [11-13] suggested using heterogeneous network with HSNs and low end sensor nodes (LSNs), where HSNs are used to perform high power calculations such as certificate verification, exponentiation, elliptic curve scalar multiplications and additions and modular multiplications.

HSNs are the best targets for the attackers to destroy the network where HSN is connected to large number of nodes. Also, HSN verifies certificates one by one within its connected nodes, which takes large time. Our scheme uses security managers to process a certificate shared verification process in distributed manner, with lower time for the same number of nodes, as shown in section 6. Moreover, HSN scheme does not provide backup for HSN where our scheme provides backup for SMs. Furthermore, no node can revoke HSN if it is compromised but in our scheme BKSM can revoke compromised SM. Besides, connectivity and scalability is affected by a compromised HSN, while our scheme provides BKSM to maintain high connectivity and scalability if SM is compromised. Destroying HSN in the middle of a branch results in cut communications in the branch. Also, each node underneath HSN needs three certificates verification for beacon nodes which is high cost for large number of nodes while our scheme assumes beacon nodes certificates verification once for the whole cluster. Finally, if HSN is destroyed, nodes cannot have rekeying or addition of new nodes or revocation of compromised nodes.

The proposed key management scheme has four types of nodes, which are SM, BKSM, initiator node and sensor nodes. The key management scheme assumes seven phases which are: key predistribution, key establishment, secure localization, secure clustering, rekeying, keys revocation and addition of new nodes. The protocol has four algorithms. The first algorithm is used for certificates verification and keys distribution. The second algorithm is used for initiator nodes to initiate key management process. The third algorithm is used for secure localization. The fourth algorithm is used for secure clustering. Sensor nodes near the BS are the first layer SMs. SMs are located every two layers. First, SMs near the BS verify the certificate of BS and the BS verifies the certificates of the first layer SMs

 ISBN: 978-1-61208-207-3

then they share a symmetric link keys. Second, first layer SMs determine their locations from their neighbour beacon nodes after receiving the neighbour beacon nodes certificates and then send them to BS for verification. Third, SMs broadcast their certificates to their neighbour nodes underneath and these nodes verify the certificate of SMs. Fourth, neighbour nodes underneath SMs broadcast their certificates to SMs which in turn send these certificates to BS for verification then SMs and neighbour nodes underneath share a symmetric keys. Fifth, neighbour nodes underneath SMs determine their locations from their neighbour beacon nodes after receiving the neighbour beacon nodes certificates and then send them to SMs then to BS for verification. Sixth, SMs and their neighbour nodes underneath form secure clustering then SMs select BKSMs according to maximum connectivity between BKSM and sensor nodes in the cluster. Finally, lower layer SMs send certificates of their neighbour nodes underneath and beacon nodes to higher layer SMs for verification.

Our scheme proposed to deploy an initiator node every predefined number of nodes to start the process of key management in distributed manner and to finish it in controlled efficient time where these nodes are SMs. These nodes collect the certificates of their underneath nodes for verification and execute our proposed second algorithm. Finally, every initiator node communicates with its higher layer node and its upper layer SM.

In this paper, we proposed a new hybrid and dynamic key management in homogenous network that uses a novel idea of certificates shared verification to avoid using HSN and our scheme has BKSM for every cluster to replace the SM if it is compromised.

The proposed scheme provided secure clustering algorithm to choose backup security managers (BKSMs). In addition, the proposed scheme can revoke the compromised SM by the BKSM. Moreover, BKSM will maintain the network scalability and connectivity if the SM is compromised. Furthermore, the proposed scheme provided secure localization algorithm with certificates shared verification to lower computation overheads and to verify certificates of beacon nodes once for the whole cluster. The proposed dynamic key management uses certificates shared verification to reduce computations overheads and setup time for rekeying and addition of new nodes. The proposed scheme used initiator nodes every predefined number of nodes to start key management process for its underneath nodes to overcome absence of HSN. The proposed scheme can distribute link keys in time lower than HSN model.

A. Contributions

1- We designed a homogenous network that utilizes SMs, BKSMs and initiators to implement the distributed security concept instead of using HSNs which is the best target for the attackers.
2- We designed the certificates shared verification mechanism to distribute the high power computations of certificates verification among sensor nodes in the cluster.
3- We designed an integrated key management scheme that combines hybrid key management; and dynamic key management to resist attacks in the hostile environment.
4- We designed a secure localization algorithm that employs the certificates shared verification scheme with low computation overhead through verifying beacon nodes certificates only once for the cluster where previous scheme assumes that each sensor node verifies certificates of three beacon nodes.
5- We designed a secure clustering algorithm that chooses BKSM to replace and revoke the SM if it is compromised. Also, BKSM will maintain high connectivity and high scalability if SM is compromised.
6- We designed the network with low setup time, and low cost compared to network with HSNs. The computation overhead at SM is lower than that at HSN.
7- We designed our key management to be dynamic to provide rekeying, revocation of compromised sensor nodes and addition of new nodes using certificates shared verification.

B. Outline of the Paper

Section 2 presents the related work. Section 3 describes the network assumptions and threat model. Section 4 describes the proposed hybrid and dynamic key management scheme along with certificates shared verification. Section 5 presents security analysis of the proposed scheme. Section 6 presents the performance analysis. Section 7 presents simulation results. Section 8 presents comparison with previous works. Finally, Section 9 concludes the paper.

II. RELATED WORK

In this section, we present related work to our proposed scheme.

A. SurvSec Security Architecture

Surveillance Security (SurvSec) is a new designed security architecture for reliable network recovery from single BS failure of surveillance WSN with single BS [19]. SurvSec relies on a set of sensor nodes serve as SMs for management and storage of the security related data of all sensor nodes. SurvSec has three components: (1) Sensor nodes serve as SMs, (2) Data Storage System, (3) Data Recovery System.

SurvSec is used for securing surveillance WSN during the time between the BS failure and the new mobile BS deployment which is the perfect time for attackers to compromise many legitimate nodes then destroy the security of the whole network. Also, SurvSec describes how the new BS will verify the trustworthiness of the deployed WSN otherwise a new WSN must be deployed.

B. WSN Key Management Schemes

1. Static versus Dynamic Key Management

Static key management schemes assume that once administrative keys are predeployed in the nodes, they will not be changed. Most static schemes use the overlapping of

administrative keys to determine the eligibility of neighbouring nodes to generate a direct pair-wise communication key.

Basically, dynamic key management schemes change administrative keys periodically, or on demand or on detection of node capture. The major challenge in dynamic keying is to design a secure yet efficient rekeying mechanism. A proposed solution to this problem is using exclusion-based systems (EBSs); a combinatorial formulation of the group key management problem developed in [17, 18]. A drawback of the basic EBS-based solution is that a small number of nodes may collude and collectively reveal all the network keys.

2. Key Management based on Encryption Key

Symmetric key based key management schemes are widely used because these schemes consume less computation time and power than other schemes, which are suitable for the limited resource characteristics. Based on the key distribution, key discovery and key establishment in the schemes, we can divide these schemes into eight categories: entity based key management schemes [1], pairwise key pre-distribution schemes [2], pure probabilistic-based schemes [3], polynomial-based key pre-distribution schemes [4], matrix- based key pre-distribution schemes [5, 25], tree-based key pre-distribution schemes [6], combinatorial design-based key pre-distribution schemes [7] and exclusion basis systems EBS-based key pre-distribution schemes [8].

Public key based key management schemes have many advantages such as low communications overhead, low storage overhead, high scalability. It can provide simpler solution with much stronger security strength. Public key based schemes have been categorized into three types: RSA-based asymmetric encryption system, ECC-based asymmetric encryption system and ID-based key agreement schemes. Several research groups have successfully implemented the public-key in WSNs [9, 10]. Asymmetric key based key management requires higher computations and energy cost than symmetric key based key management.

Hybrid key establishment schemes are proposed by several research groups [11-13]. The motivation is the needs for high security level and to exploit the difference among the BS, the HSNs and the sensors, and place the cryptographic burden on the BS or the HSNs. Sensors have limited computational power and energy resources. On the other hand, the BS and HSNs have much more computational power and other resources. Previous hybrid key establishment schemes reduce the high computational cost on the sensors by placing them on the HSN side and assume certificates verification for large number of nodes at HSN. Huang et al. [11] proposed a hybrid authenticated key establishment scheme, which is based on a combination of elliptic curve cryptography (ECC) and symmetric-key operations. The hybrid key establishment protocol reduces the high cost elliptic curve random point scalar multiplications at the sensor side and replaces them with low cost and efficient symmetric-key based operations.

3. Key Management based on Location

Recently researchers have suggested utilizing the location of sensor nodes [14- 16] after node deployment to improve the security of key management. Location based key management protocols are very efficient methods in terms of key connectivity and storage overhead. Location-aware key management is resilient against node capture attacks in large-scale sensor networks.

III. NETWORK MODEL&THREAT MODEL

A. Network Model

We consider a hierarchical WSN consisting of a BS, sensor nodes which are grouped into clusters and beacon sensors equipped with GPS called beacons. Each node has a unique ID, unique location and unique certificate. The assumptions of model are as follows:

1- We assume sensor nodes are static and some nodes continuously store the detected security threats and all other security data related to nodes where these nodes are SMs.
2- Nodes near the BS have the public key of the BS.

B. Threat Model

We consider an adversary that tries to uncover the keys of the network through capturing some nodes.

IV. PROPOSED SCHEME

The proposed scheme has seven phases which are key predistribution phase, key establishment phase, secure localization phase, secure clustering phase, key revocation phase, rekeying phase and add new node phase. The proposed scheme has four types of sensors: SMs, BKSMs initiators and sensor nodes.

A. Key Predistribution Phase

The key predistribution phase consists of acquiring the sensors certificate from the certificate authority CA. ECC is used in this protocol to perform security functions on sensors with limited computing resources. The protocol uses the elliptic curve explicit certificate scheme instead of X.509 because of the resulting low storage overhead, low communication overhead, which is a dominant factor for low bit transmission channels in WSN.

The certificate generation processes for any sensor node U is performed offline before it joins the network.

1- An elliptic curve E defined over $GF(p)$ (p is the characteristic of the base field) with suitable coefficients and a base point P of large order n is selected and made public to all users.
2- CA selects a random integer q_{CA} as its static private key, and computes the public key $Q_{CA} = q_{CA} X P$.
3- To obtain a certificate and private-public key pair, the sensor U randomly selects a key pair (q_U, Q_U) where $Q_U = q_U X P$ and sends Q_U and q_U to CA.
4- CA verifies U's identity and private-public key pair.
5- The implicit certificate for U is the concatenation of CA's public key Q_{CA}, the device identity ID_U, the U public key Q_U and the certification expiration date t_U , i.e., the

certificate is (Q_{CA}, ID_U, Q_U, t_U) signed by the CA private key using Elliptic Curve Digital Signature Algorithm ECDSA.

B. Key Establishment Phase

Certificates Verification &Keys Distribution

Power of the signature verification for ECDSA is about 1000 times more than the power of the signature transmission [20]. Each node in HSN model performs four times certificate verification for three beacon nodes and for HSN certificates. With the same number of certificates verification at each node, we developed our proposed certificates shared verification scheme. Each node in our scheme verifies four certificates only with the cost of increasing the communication overhead with four messages for every node. These verifications are: first verification for SM certificate, two verifications for two nodes underneath that node, and one verification for beacon node certificate. We assume that there are nodes named as security managers SMs and these nodes are located every two layers. We assume that there are nodes named as initiators every predefined number of nodes such as 30, 20 or 10 nodes to start the operation of key management process.

We explain our scheme in the form of two algorithms.

Algorithm 1: Certificates Verification and Keys Distribution

1: BS → n : {BS (Q_{CA}, ID_{BS}, Q_{BS}, t_{BS}) }

The BS broadcasts its certificate to nodes near BS at layer n and the nodes verify certificate of BS. These nodes are SMs.

2: n → BS : {n (Q_{CA}, ID_U, Q_U, t_U) }

The nodes near the BS at layer n broadcast their certificates to the BS and the BS verifies the certificates of these nodes.

3: n : selects (k), calculates (d_U), encrypts (d_U)

Each node near BS at layer n selects a k-bit random number c_U of 160 bits to produce its link key contribution with the BS.

Each node at n calculates the value of $d_U = H(c_U \| ID_U)$ where H is a cryptographic hash function. Each node at n encrypts d_U with BS public key Q_{BS}. To encrypt and send a message d_U to BS, each node at n chooses a random positive integer x and produces the ciphertext C_m consisting of the pair of points which are: $C_m = (x\,P,\ d_U + x\,Q_{BS})$.

4: n → BS : { C_m }

Each node near BS at layer n sends its encrypted link key contribution with the BS which is C_m.

5: BS : decrypts (C_m), selects (k), calculate (d_{BS}), encrypts (d_{BS})

BS decrypts C_m for every node at n. BS multiplies first point in the pair by BS's private key and subtracts result from second point: $d_U + x\,Q_V - q_V\,(x\,P) = d_U + x\,(q_V\,P) - q_V\,(x\,P) = d_U$.

BS selects a k-bit random number c_{BS} of 160 bits for each node near BS to produce its link key contribution with nodes near BS. BS calculates the value of $d_{BS} = H(c_{BS} \| ID_{BS})$ for every node near BS where H is a cryptographic hash function.

BS encrypts d_{BS} for every node near BS using symmetric key encryption under key d_U, generating value $y = E_{du}$ ($ID_{BS} \| d_{BS}$).

6: BS → n : { y }, {hash {K}}

BS sends y, the encrypted link key contribution of BS, to every node near BS. BS generates the link key with every node near the BS at n by calculating $K = H\,(d_u \| ID_U \| d_{BS} \| ID_{BS})$ then $H(K)$ where H is a cryptographic hash function. BS sends $H(K)$ of every node at n to its participant to achieve correctness.

7: n : decrypts (y), calculates (K)

Every node at n decrypts the received message y using symmetric key encryption under key d_U to obtain the value d_{BS}.

Every node at n generates the link key with BS by calculating $K = \mathrm{H}(d_u \| ID_U \| d_{BS} \| ID_{BS})$.

8: n → BS : {z}

Every node at n calculates $z = \mathrm{H}(K)$ and sends z to BS. BS checks if $z = H(K)$. If yes, the link key is established correctly. Otherwise, the protocol is terminated.

9: n → n-1 : {n (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Each SM at layer n broadcasts its certificate to nodes at layer n-1 and nodes at n-1 verify the certificate of its SM. Each node at layer n-1 verifies SM certificate.

10: n-1 → n : {n-1 (Q_{CA}, ID_U, Q_U, t_U) }

Each node at layer n-1 sends its certificate to its SM at layer n

10: n → BS : all certificates {n-1 (Q_{CA}, ID_U, Q_U, t_U) }

Every SM at layer n sends the certificates of its nodes at layer n-1 to BS for verification because SM will lose high power and consume large time for verifying certificates of at least four nodes connected to it.

11: BS → n : {valid certificates or invalid certificates}

BS sends to each SM an encrypted message indicating that its certificates from layer n-1 are valid or not.

Then SMs at layer n executes steps from 3 to 8 to share symmetric link keys with nodes at layer n-1.

12: n-1 → n-2 : {n-1 (Q_{CA}, ID_U, Q_U, t_V) }

Every node at layer n-1 sends its certificate to its neighbour node at layer n-2 and the node at layer n-2 verifies the certificate of node at layer n-1. Nodes at layer n-2 are SMs.

13: n-2 → n-1, n : {n-2 (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Every node at layer n-2 sends its certificate to its connected node at layer n-1 then to the SM at layer n. The node at layer n-1 verifies the certificate of node at layer n-2 and node at layer n-2 verifies certificate of

node at layer n-1.

14: n-2 → n, n-1 : {share link keys }

SM at layer n-2 executes steps from 3 to 8 to share symmetric link keys with node at layer n-1 and SM at layer n.

15: n-2 → n-3 : {n-2 (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Every node at layer n-2 which is a SM broadcasts its certificate to nodes at layer n-3 and nodes at n-3 verify the certificate of its SM.

16: n-3 → n-2 : {n-3 (Q_{CA}, ID_U, Q_U, t_U) }

Each node at layer n-3 sends its certificate to its connected SM at layer n-2.

17: n-2 → n : all certificates {n-3 (Q_{CA}, ID_U, Q_U, t_U) }

Every SM at layer n-2 sends the certificates of its nodes at layer n-3 to its SM at layer n for verification.

18: n → n-1 : all certificates {n-3 (Q_{CA}, ID_U, Q_U, t_U) }

SM at layer n sends the certificates of nodes at layer n-3 to its downstream nodes at layer n-1 for verification.

18: n-1 → n : {valid certificates or invalid certificates}

Every node at layer n-1 sends to its SM indicating that the checked certificate from layer n-3 is valid or not.

19: n → n-2 : {valid certificates or invalid certificates}

SM at layer n sends to the SM at layer n-2 indicating that the checked certificates from layer n-3 are valid or not. Then SMs at layer n-2 executes steps from 3 to 8 to share symmetric link keys with nodes at layer n-3. **Finally**, lower layer SMs send certificates of their neighbour nodes underneath to higher layer SMs for verification.

Discussion

The bottleneck of algorithm 1 is the number of SMs near the BS because if the number of these nodes increases, this will reduce setup time for the nodes underneath the SMs. Therefore, if the number of SMs near BS is more than three, SMs near BS execute algorithm 2.

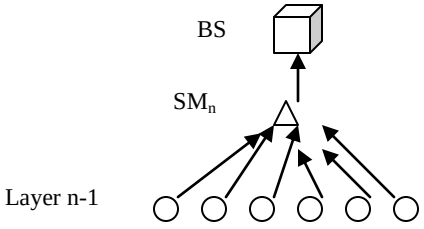

Fig. 1.a Certificates Verification for layer n-1

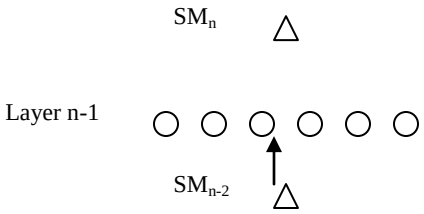

Fig. 1.b Certificates Verification for layer n-2

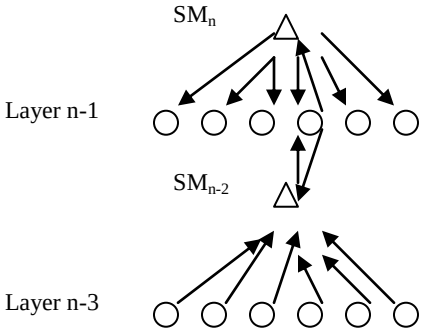

Fig. 1.c Certificates Verification for layer n-3

Fig. 1 shows certificates shared verification process in three layers using the first algorithm. Initiator nodes start the process of key management in distributed manner where these nodes are predetermined every number of nodes such as 30, 20 or 10 nodes. Initiator nodes work as HSN to control the setup time for the key management.

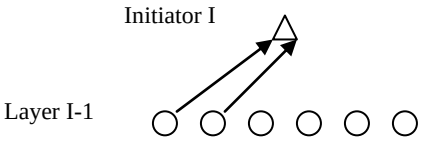

Fig. 2.a Certificates Verification using Initiator for 2 nodes

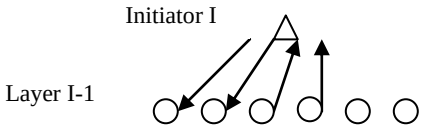

Fig. 2.b Certificates Verification using Initiator for 2 nodes

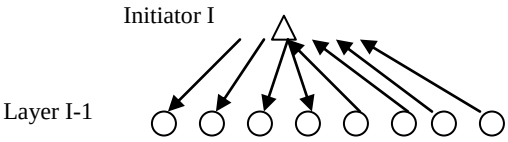

Fig. 2.c Certificates Verification using Initiator for 4 nodes

Fig. 2 shows certificates shared verification process for one layer using algorithm 2. SM verifies certificates of first two nodes then it sends the certificates of the second two nodes to the first two nodes then it sends certificates of other four nodes to the verified four nodes. Algorithm 2 is efficient in terms of distribution of power consumption among sensor

 ISBN: 978-1-61208-207-3

nodes in the cluster and it can be used with all SMs in their clusters. Algorithm 1 provides high speed for certificates verification but its drawback is that the cluster nodes between an initiator and its upper layer SM are not involved in the process of certificates verification. Therefore, there is a trade off between high speed certificates verification using algorithm 1 and distributed power consumption using algorithm 2.

Algorithm 2: Initiator nodes to start key management process

1: I → n : { I (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Each initiator node broadcasts its certificate to its underneath nodes at layer n to verify it. The nodes at layer n verify the certificate of the initiator.

2: n → I : { n (Q_{CA}, ID_U, Q_U, t_U) }

The initiator node receives the certificates of its underneath nodes for verification. We assume there are n nodes underneath the initiator node. First, the initiator node verifies the certificate of the first two nodes.

3: I → $n_{1,2}$: { share link keys }

The initiator node shares link keys with node 1 and node 2 as steps from 3 to 8 in algorithm 1.

4: I → $n_{1,2}$: { $n_{3,4}$ (Q_{CA}, ID_U, Q_U, t_U) }

The initiator node sends to node 1 and node 2 underneath the certificates of node 3 and node 4 for verification.

5: $n_{1,2}$ → I : { valid certificates or invalid certificates }

Node 1 and node 2 send to the initiator node two messages indicating that certificates of nodes 3 and 4 are valid or not.

6: I → $n_{3,4}$: { share link keys }

The initiator node shares link keys with node 3 and node 4 as steps from 3 to 8 in algorithm 1.

7: I → $n_{1,2,3,4}$: { $n_{5,6,7,8}$ (Q_{CA}, ID_U, Q_U, t_U) }

The initiator node sends to node 1, node 2, node 3 and node 4 underneath the certificates of node 5, node 6, node 7 and node 8 for verification and nodes 1, 2, 3, 4 respond with valid certificate or not.

8: I → $n_{5,6,7,8}$: { share link keys }

The initiator node shares link keys with node 5, node 6, node 7 and node 8 as steps from 3 to 8 in algorithm 1. **Finally**, the process of the initiator continues to verify all of its underneath nodes then its underneath nodes use algorithm 1 to share link keys with their underneath nodes and so on.

1. Certificates shared verification between SM near BS and BS needs two messages but it needs four messages between SM at lower layer and SM at upper layer.
2. Each SM establishes a link key with its nodes underneath in ten messages but SM near BS establishes a link key with its nodes underneath in eight messages.
3. After the SMs and the sensor nodes establish link keys, they determine their locations using our proposed secure localization scheme with certificates shared verification.

C. Secure Localization Phase

A number of secure localization algorithms [21] have been reported. Different researchers have different strategies to categorize them. These strategies can be divided into direct and indirect localization, centralized localization and distributed localization, range-based localization and range-free localization, absolute localization and relative localization. We propose to get the location information from the followings approach:

The indirect approaches of localization were introduced to overcome some of the drawbacks of the GPS-based direct localization techniques while retaining some of its advantages. In this approach, a small subset of nodes in the network, called the beacon nodes, are equipped with GPS receivers to compute their location. Beacon nodes send beams of signals providing their location to all nodes in their vicinity. Using the transmitted signal containing location information, nodes compute their location. Each node needs three beacon nodes to locate its position.

Our proposed scheme depends on SM and certificates shared verification for secure localization. We assume that each cluster has three beacon nodes. Sensor nodes in the cluster send the beacon nodes certificates to SM then SM sends these certificates to its upper layer SM for verification to insure one verification time for beacon nodes certificates for the whole cluster. The upper layer SM sends these certificates to its underneath nodes for verification. Verification power is 1000 times more than communication power.

Algorithm 3: Secure Localization

1: $Beacons_{1,2,3}$ → SM_n : {$Beacons_{1,2,3}$ (Q_{CA}, ID_B, Q_B, t_B) }

The beacon nodes near BS broadcast their certificates and locations to SMs near BS. We need three beacon nodes to locate the position.

2: SM_n → BS : { $Beacons_{1,2,3}$ (Q_{CA}, ID_B, Q_B, t_B) }

The SMs near BS at layer n send the certificates of the beacon nodes to BS for verification.

3: BS → SM_n : {valid certificates of $Beacons_{1,2,3}$ }

BS sends to SMs at layer n that beacon nodes certificates are valid.

4: SM_n → $Beacons_{1,2,3}$: { $Key_{1,2,3}$ }

Every SM at layer n shares a link key with the three beacon nodes in four steps.

5: SM_n : calculates (x, y) position

Every SM at layer n calculates its position.

6: $Beacons_{1,2,3}$ → n-1 : {$Beacons_{1,2,3}$ (Q_{CA}, ID_B, Q_B, t_B) }

The beacon nodes near BS broadcast their certificates and locations to nodes at layer n-1.

7: n-1 → SM_n : { $Beacons_{1,2,3}$ (Q_{CA}, ID_B, Q_B, t_B) }

 ISBN: 978-1-61208-207-3

The nodes at layer n-1send the certificates of beacon nodes to SMs at layer n for verification. If the beacon nodes certificates are previously verified, it is ok but if there are new beacon nodes certificates, then SMs at layer n send the new beacon nodes certificate to BS for verification.

8: $SM_n \rightarrow$ n-1 : { $Key_{1,2,3}$ }

Every SM at layer n sends its link keys with the beacon nodes to its connected nodes at layer n-1.

9: n-1 : calculates (*x*, *y*) position

Every node at layer n-1 calculates its position.

10: $Beacons_{4,5,6} \rightarrow SM_{n-2}$:{$Beacons_{4,5,6}$ (Q_{CA}, ID_B, Q_B, t_B) }

The beacon nodes near SMs at layer n-2 broadcast their certificates and locations to SMs at layer n-2.

11: $SM_{n-2} \rightarrow SM_n$: { $Beacons_{4,5,6}$ (Q_{CA}, ID_B, Q_B, t_B) }

The SMs at layer n-2 send the certificates of the beacon nodes to SMs at layer n for verification.

12: $SM_n \rightarrow$ n-1 : { $Beacons_{4,5,6}$ (Q_{CA}, ID_B, Q_B, t_B) }

The SMs at layer n send the certificates of the beacon nodes to nodes at layer n-1 for verification.

13: n-1 $\rightarrow SM_n$: { valid certificates of $Beacons_{4,5,6}$ }

The nodes at layer n-1 send to SMs at layer n that beacon nodes certificates are valid.

14: $SM_n \rightarrow SM_{n-2}$: { valid certificates of $Beacons_{4,5,6}$ }

The SMs at layer n send to SMs at layer n-2 that beacon nodes certificates are valid.

15: $SM_{n-2} \rightarrow Beacons_{4,5,6}$: { $Key_{4,5,6}$ }

Every SM at layer n-2 shares a link key with the three beacon nodes in four steps.

16: SM_{n-2} : calculates (*x*, *y*) position

Every SM at layer n-2 calculates its position.

17: $Beacons_{4,5,6} \rightarrow$ n-3 : {$Beacons_{4,5,6}$ (Q_{CA}, ID_B, Q_B, t_B) }

The beacon nodes near nodes at layer n-3 broadcast their certificates and locations to nodes at layer n-3.

18: n-3 $\rightarrow SM_{n-2}$: { $Beacons_{4,5,6}$ (Q_{CA}, ID_B, Q_B, t_B) }

The nodes at layer n-3 send the certificates of beacon nodes to SMs at layer n-2 for verification. If the beacon nodes certificates are previously verified, it is ok but if there are new beacon nodes certificates, then SMs at layer n-2 send the new beacon nodes certificate to SMs at layer n for verification.

19: $SM_{n-2} \rightarrow$ n-3 : { $Key_{4,5,6}$ }

Every SM at layer n-2 sends its link keys with the beacon nodes to its connected nodes at layer n-3.

20: n-3 : calculates (*x*, *y*) position

Every node at layer n-3 calculates its position. **Finally**, lower layer SMs send certificates of beacon nodes to higher layer SMs for verification.

1. Certificates shared verification for beacon nodes certificates between SM at lower layer and SM at higher layer will reduce setup time and reduce computations complexity at the cost of increasing only four messages.
2. Certificates verification for beacon nodes is done only one time at the SM not multiple times at each node underneath the SM to reduce computations complexity.
3. Sensor nodes underneath SM will use the shared keys between the SM and the beacon nodes which will reduce the setup time, computations and storage overhead.
4. After the SMs and the sensor nodes determine their locations, they form secure clustering.

D. Secure Clustering Phase

SMs can form secure clustering [22] with their nodes underneath and SM can choose BKSM to replace it if the SM is compromised.

Algorithm 4: Secure Clustering

1: BS $\rightarrow$ n : {req SM_msg }

BS sends to nodes near BS at layer n that these nodes are SMs using its shared symmetric key with these nodes.

2: $SM_n \rightarrow$ n-1 : { adv cluster_msg }

Every SM at layer n sends an encrypted advertise message to nodes at layer n-1 to form a cluster.

3: n-1 $\rightarrow SM_n$: { join cluster_msg }

Every node at layer n-1 sends an encrypted message to its SM at layer n to join the cluster.

4: $SM_n \rightarrow$ n-1 : {choose BKSM }

The SM at layer n chooses BKSM according to maximum connectivity between the BKSM and the nodes in the cluster where BKSM must be connected to all nodes in the cluster.

5: $BKSM_n \rightarrow$ n-1 : { BKSM (Q_{CA}, ID_{BKSM}, Q_{BKSM}, t_{BKSM}) }

The BKSM sends its certificate to the nodes at layer n-1 where SM at layer n verifies this certificate. Also, the BKSM sends its certificate to its upper layer node to establish a link key with it to reroute data if SM is compromised.

6: n-1 $\rightarrow$ n-2 : { req SM_msg }

The nodes at layer n-1 send to nodes at layer n-2 an encrypted message that these nodes are SMs.

7: $SM_{n-2} \rightarrow$ n-3 : { adv cluster_msg }

Every SM at layer n-2 sends an encrypted advertise message to nodes at layer n-3 to form a cluster.

8: n-3 $\rightarrow SM_{n-2}$: { join cluster_msg }

Every node at layer n-3 sends an encrypted message to its SM at layer n-2 to join the cluster.

9: $SM_{n-2} \rightarrow$ n-3 : {choose BKSM }

The SM at layer n-2 chooses BKSM according to maximum connectivity between the BKSM and the nodes in the cluster where BKSM must be connected to all nodes in the cluster.

10: $BKSM_{n-2} \rightarrow$ n-3 : { BKSM (Q_{CA}, ID_{BKSM}, Q_{BKSM}, t_{BKSM}) }

The BKSM sends its certificate to the nodes at layer n-3

where SM at layer n-2 verifies this certificate. Also, the BKSM sends its certificate to its upper layer node to establish a link key with it to reroute data if SM is compromised. **Finally**, the steps of forming the secure clustering are performed until the last layer of SM.

1. Our proposed secure clustering scheme assumes hybrid key management protocol to achieve high security level.
2. Our proposed scheme chooses BKSM to solve the problem of compromised SM and to sign the message of revoked SM.
3. Our scheme achieves secure clustering in four messages.

E. Key Revocation Phase

The first component of our dynamic based key management scheme is the keys revocation of the compromised sensor nodes. SurvSec security architecture has a compromised nodes detection algorithm at the first stage to be able to detect compromised nodes but it is not discussed in this paper.

When a sensor node is compromised by an adversary, all the session keys used by this sensor node will be revoked. The SM will broadcast a revocation message containing the identification of the compromised node to all the nodes underneath. A digital signature is computed over the message by utilizing Elliptic Curve Digital Signature Algorithm ECDSA at [23] with SMs private key. When a node receives the revocation message, it checks the message by verifying the digital signature. This prevents an adversary from sending a fake revocation message. If SM is compromised, it is revoked by the BKSM.

F. Rekeying Phase

The second component of our dynamic based key management scheme is rekeying after compromised nodes detection or rekeying can be done periodically. Rekeying is used when the SM is compromised. The BKSM will share a link key with its upper layer SM then the BKSM will use our novel scheme of certificates shared verification with its upper layer SM to verify the certificates of the cluster nodes. Finally, BKSM will share link keys with its lower SM and its nodes in the cluster.

G. Add New Node Phase

When a new node joins the network, it tries to find its nearest SM by broadcasting a Hello message contains the new node certificate.

To support the addition of new nodes, the SM verifies the certificate of the new nodes using our novel scheme of certificates shared verification.

V. SECURITY ANALYSIS

Security analysis of our protocol focuses on resilience to node compromise attack, and collusion attack.

A. Compromised Node Attack

1- If an attacker compromises one ordinary node, therefore, the number of insecure link is $P_{insec} = 1 / N$ where N is the number of nodes at the network. For n compromised ordinary nodes, number of insecure links is $P_{insec} = n / N$.

2- If the attacker compromises one SM, therefore, the number of insecure links is $P_{insec} = (n_s + 3) / N$ where n_s is the number of nodes in the cluster of the SM and 3 represents the links with upper SM, lower SM and SM upper node. For n compromised SMs, the number of insecure links is $P_{insec} = n (n_s + 3) / N$.

3- Suppose that in a network of N nodes, there are m SMs and BKSMs. The probability to compromise one SM or one BKSM is P(com) = 2m / N, so the probability of at least k nodes from the SMs and BKSMs are captured is:

$$p(k) = 1 - \sum_{i=0}^{k} \binom{n}{i} p(com)^i (1 - p(com))^{n-i} \quad (1)$$

The probability that all SMs and BKSMs are captured is:

$$p(k) = 1 - \sum_{i=0}^{2m} \binom{n}{i} p(com)^i (1 - p(com))^{n-i} \quad (2)$$

4- Our proposed key management assumes compromised node detection at the first stage and compromised nodes revocation. Therefore, SM will revoke the ordinary compromised node and the BKSM will revoke the SM to eliminate the insecure links.

B. Collusion Attack

Two nodes can collude when they share their keys with each other. Our designed protocol is resistant to collusion attack because each sensor node communicates only with a SM therefore; compromised nodes cannot discover themselves.

VI. PERFORMANCE ANALYSIS

The performance analysis is measured in Computation complexity, communication complexity and storage complexity. We assume that the network is secure during setup time which depends on number of initiators.

A. Computation Complexity

Our proposed hybrid key management scheme using certificates shared verification has much lower computations overhead at SM side rather than computations at HSN in heterogeneous network. For algorithm 1, our scheme assumes each sensor node in each cluster verifies four certificates for the keys distribution and localization which are the certificate of its SM, two certificate from its underneath nodes and one beacon node certificate. SM verifies one certificate which is its upper node. For algorithm 2, our scheme assumes each sensor node in each cluster verifies at most four certificates for the keys distribution and localization which are the certificate of its initiator, two certificates from the nodes of its cluster and one beacon node certificate. Initiator node verifies three certificates which are two certificates from its underneath nodes and one certificate for its upper node.

Each sensor node and SM performs three times hash to generate one link key. The sensor node encrypts its part of the link key with the SM's public key using ECC 160 bits scalar multiplication and addition. Also, the SM decrypts the received message from the sensor node with its private key. The SM encrypts its part of the link key using symmetric key under the key from the sensor node. The sensor node decrypts the message from the SM using symmetric key. Our

 ISBN: 978-1-61208-207-3

scheme has less computation overhead at SM than the scheme uses HSNs at HSN.

In our scheme:

Each node performs at most 4 verifications and shares key with SM or initiator for keys distribution and localization.

SM or initiator performs at most 3 verifications and shares keys with n nodes for keys distribution and localization where n nodes are ranged from 4 to 8 nodes in the cluster.

In HSN scheme:

Each node performs 4 verifications and shares key with HSN for keys distribution and localization.

HSN performs n verifications and shares keys with n nodes where n nodes are ranged from 10 to 30 nodes underneath HSN.

Our scheme has lower computations than HSN scheme.

B. Communication Complexity

Communication complexity is the number and size of packets sent and received by a sensor node. In our protocol, the number of messages sent and received to establish a key between one sensor node and a SM is ten messages and we need six messages to establish link key between lower layer SM and upper layer SM. Device ID is 64 bits, expiration time is 64 bits, random number is 160 bits and L the sensor location is 64 bits. The certificate is 56 bytes from 20 bytes CA public key, 8 bytes node ID, 20 bytes node public key and 8 bytes validity time. Our scheme has higher communication overhead than HSN model with 4 messages to establish link key for every node.

In our scheme:

For algorithm 1:

Communication overhead = 6 N_{SM} + 10 m N_{SM},, N_{SM} is number of SMs and m is the number of nodes underneath SM within its cluster.

For algorithm 2:

Communication overhead = I (12 + 8 (m – 2)) + 6 I, I is the number of initiator nodes, m is the number of nodes underneath initiator. We have 2 nodes needs 12 messages and other nodes in the cluster need 8 messages and 6 represents the communication between the initiator and its upper node.

For algorithm 1 and 2: Total communication overhead is C_{com}.

$C_{com} = N_{SM} (6 + 10 m) + I (2 + 8 m)$.

We found that communication overhead for algorithm 2 is lower than communication overhead for algorithm 1.

In HSN scheme:

For one HSN every 30 nodes: communication overhead is C_{com}.

$C_{com} = N_{HSN} (6 + 6 n_0 + 8 n_1 + 10 n_2 + 12 n_3)$. Where N_{HSN} is the number of HSNs, n_0 is the number of first layer nodes underneath HSN, n_1 is the number of second layer nodes underneath HSN, n_2 is the number of third layer nodes underneath HSN, n_4 is the number of fourth layer nodes underneath HSN and 6 represents the communication between the HSN and its upper node.

For one HSN every 20 nodes: communication overhead is C_{com}.

$C_{com} = N_{HSN} (6 + 6 n_0 + 8 n_1 + 10 n_2)$.

For one HSN every 10 nodes: communication overhead is C_{com}.

$C_{com} = N_{HSN} (6 + 6 n_0)$.

Our model has lower communication overhead than HSN model for one HSN every 30 but our model has higher communication overhead than HSN model for one HSN every 20 or 10 nodes.

C. Storage Complexity

Storage complexity is the amount of memory units required to store security credentials. Each sensor node stores its public key, private key, BKSM public key and the link key shared with the SM. The SM stores all of the shared keys with each sensor node underneath plus its public, private key, link key with upper SM, link key with the lower SM and link key with its upper node. Our scheme has the the same storage overhead as HSN scheme.

In our scheme:

Total SMs storage overhead = (N_S+5) N_{SM}, N_S is the number of nodes underneath SM and N_{SM} is the number of security managers.

Sensor nodes storage overhead = 3 N_S.

In HSN scheme:

Total HSNs storage overhead = (N_S+5) N_{HSN}.

Sensor nodes storage overhead = 3 N_S.

D. Setup Time

We assume that verification using ECDSA takes about 4 sec [24], share link key takes about 1 sec [11] and certificate transmission takes about 0.1 sec [11]. The transmission time is dominant factor but on the other hand, the bottleneck will be the certificate verification operation time. Setup time is equal to verification time plus communication time plus share link key time.

In our scheme:

For algorithm 1 the setup time is T.

T = 4S + n X 1S + (5 n + 2) X 0.1S, verification is done in parallel where upper layer SM sends to its underneath nodes the certificates of the nodes underneath its lower layer SM which is n nodes. Therefore, we need one verification time and n times to share link keys and (5n+2) messages to send all certificates to the verifiers and have the result.

For algorithm 2 the setup time is T.

T = m X 4S + n X 1S + (12 + 8 (n - 2)) X 0.1S, verification is done m times, share link keys is done n times and we need number of messages equal to (12 + 8 (n-2)).

 ISBN: 978-1-61208-207-3

Setup time for algorithm 1 is lower than setup time for algorithm2.

In HSN scheme:

Setup time = n X 4S + n X 1S + 6n X 0.1S, where n is the number of nodes underneath the HSN and 6n is the number of messages between nodes and HSN.

Our proposed scheme with algorithm 1 has much lower setup time than HSN model where we perform parallel verification but HSN model performs sequential verification.

Our proposed scheme with algorithm 2 has lower setup time than HSN model where we perform parallel verifications but HSN model performs sequential verification. Our proposed model combines both algorithm 1 and algorithm 2.

E. Scalability

In our scheme:

BKSM will replace the SM if it is compromised and this insures high scalability to extend the network.

In HSN scheme:

If a HSN is compromised in a branch, the scalability of the branch cannot be achieved because there is no backup HSN.

F. Connectivity

In our scheme:

BKSM will replace the SM if it is compromised and this insures high connectivity with its underneath nodes.

In HSN scheme:

If a HSN is compromised in a branch, the connectivity for the nodes underneath the HSN cannot be achieved because there is no backup HSN.

VII. SIMULATION RESULTS

In this section, we evaluate the communication overhead, the computations overhead and the network setup time under different number of nodes *N* for our proposed model and HSN model.

We built our proposed model and HSN model and we implemented a simulator in MATLAB that can scale to thousand of nodes. In this simulator, sensors can send and receive data from each other's. The simulation verifies the correctness and the feasibility of our security architecture. It is our future work to implement SurvSec in some sensor network testbeds with all its ingredients. Our simulation scenarios include *N* nodes distributed randomly. We choose *N* 1000, 2000 and 3000 sensor nodes.

In the simulations, these parameters are given as follows:

1- The number of sensor nodes *N* is varied from 1000, 2000 and 3000 sensor nodes.
2- The simulation is done for HSN or initiators every 30 nodes, 20 nodes and 10 nodes.

The communication overhead for security manager to exchange a key with a node is according to algorithm 1 or algorithm 2 or both as shown in section 6.

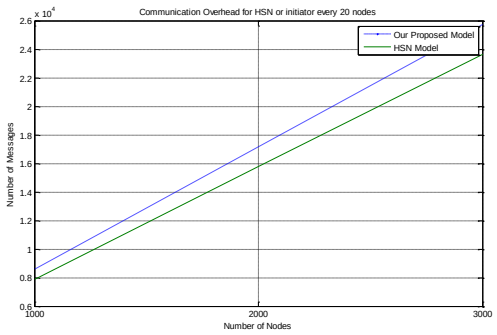

Fig. 3 Communication overhead every HSN or Initiator every 10 nodes

Fig. 3 shows the communication overhead for HSN model and our proposed model for one HSN node every 10 nodes and one initiator every 10 nodes. Our proposed model has higher communication overhead than HSN model with 20%.

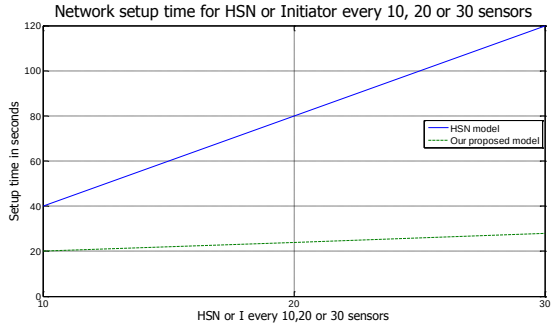

Fig. 4 Network Time Setup for HSN or Initiator every 30 nodes, 20 nodes, and 10 nodes

Fig. 4 shows the network setup time for HSN model and our proposed model for one HSN node or one initiator every 30 nodes, 20 nodes and 10 nodes. Our proposed model has at least half the network setup time than HSN model.

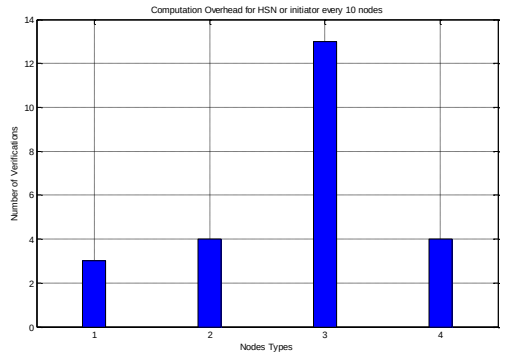

Fig. 5 Computation Overhead of Certificates Verifications for HSN or Initiator every 10 nodes

Fig. 5 shows the computation overhead for certificates verifications for HSN model and our proposed model for one HSN node or one initiator every 10 nodes. Number 1 at x-axis is the number of certificates verification at the SM which is 3 verifications for key establishment and secure localization. Number 2 at x-axis is the number of certificates verification at every node in our proposed model which is 4

 ISBN: 978-1-61208-207-3

verifications for key establishment and secure localization. Number 3 at x-axis is the number of certificates verification at the HSN which is 13 verifications for key establishment and secure localization. Number 4 at x-axis is the number of certificates verification at every node in HSN model which is 4 verifications for key establishment and secure localization. Our proposed model has lower computation overhead than HSN model. Our scheme has one quarter lower certificates verifications overhead then HSN model at SM side and one half lower certificates verification overhead in total. Finally, for HSN or Initiators every 10 nodes we increase communication overhead by 20% and we decrease the computation overhead to one half where power of certificates verification using ECDSA is 1000 times more than power of communication.

VIII. COMPARISON with others WORKS

Now, we compare between our proposed model and HSN model.

TABLE 1, COMPARISON between OUR MODEL and HSN MODEL.

	Property	**HSN Model [11-13]**	**Our Model**
1	Computation overhead for key establishment and secure localization	N verification at HSN and 4 verifications at node	3 verifications at SM and 4 verifications at node
2	Storage overhead	3 keys at node (n+5) at HSN	3 keys at node (n+5) at SM
3	Communication overhead for key establishment	6 or 8 or 10 or 12 messages for each node according to HSN every 30 or 20 or 10 nodes	8 messages for algorithm 2 or 10 messages for algorithm 1 for each node
4	Communication overhead for secure localization	No	3 messages from each node to SM and one verification message from SM to each node plus 6 messages for one time verification
5	Computation overhead for secure localization	3n verifications for the cluster	3 verifications for the whole cluster
6	Setup time	n verifications time	parallel verifications executes in 1/n time of HSN model for algorithm1 and n/2 time of HSN model for algorithm 2
7	Scalability	Affected by compromised HSN	High
8	Connectivity	Affected by compromised HSN	High
9	Backup node	No	BKSM
10	Secure localization	High cost at each node for 3 verifications	Low cost for 3 verifications for the whole cluster
11	Rekeying	High cost at HSN	Low cost at SM
12	Addition of new nodes	High cost at HSN	Low cost at SM
13	Probability of insecure links	High with compromised HSN	Low after compromised SM revocation
14	Effect of compromised nodes	No	Affect certificates shared verification
15	Nodes revocation	Cannot revoke HSN	BKSM revokes SM
16	Cost	High	Low

Our proposed scheme distributes certificate verification at nodes underneath SM rather than verifies certificates at SM. Also, our scheme verifies beacon nodes certificates once for the whole cluster. Our scheme has higher connectivity and scalability than HSN model. Our scheme can revoke compromised SM through BKSM and has lower network cost than HSN scheme. Our scheme has lower network setup time than HSN scheme and it has same storage overhead. Our scheme has lower computations overhead than HSN scheme.

IX. CONCLUSION

In this paper, we proposed a novel hybrid and dynamic key management scheme for WSNs which utilizes a novel scheme of certificates shared verification to verify the certificates of nodes in distributed computations and eliminate the usage of high end sensor nodes which are the best targets for the attackers. Our scheme is based on some sensor nodes called security managers which are chosen every two layers. We proposed a secure localization scheme with low computation overhead. Also, we proposed a secure clustering algorithm to choose backup security manager for every cluster to replace and revoke the security managers if it is compromised. Our proposed scheme can distribute link

 ISBN: 978-1-61208-207-3

keys in lower setup time than the model uses high end sensor nodes. Our proposed scheme has higher communication overhead, lower computation overhead at the security manager node and lower energy cost at the security manager node than scheme uses high end sensor node. Both schemes have the same storage overhead. Our proposed scheme has low cost than model used high end sensor nodes. Our proposed scheme connectivity and scalability are not affected if the security manager is compromised.

REFERENCES

[1] Perrig A., Szewczyk R., Wen V., Cullar D., and Tygar J.D. "SPINS: security protocols for sensor networks". In: Proceedings of the 7th annual ACM/IEEE international conference on mobile computing and networking, July 2001, pp. 189–199.

[2] Chan H., and Perrig A., "Random key predistribution schemes for sensor networks". In: Proceedings of the 2003 IEEE symposium on security and privacy, May 2003, pp. 197–213.

[3] L. Eschenauer and V. Gligor, "A Key Management Scheme for Distributed Sensor Networks," Proc. 9th ACM Conf. Comp. and Commun. Sec., Nov. 2002, pp. 41-47.

[4] Liu D., and Ning P. "Establishing pairwise keys in distributed sensor networks". In: Proceedings of 10th ACM conference on computer and communications security (CCS03). July 2003. pp. 41–47.

[5] Yu Z., and Guan Y. A "Robust group-based key management scheme for wireless sensor networks". In: Proceedings of IEEE wireless communications and networking conference (WCNC 2005), New Orleans, LA USA. IEEE Press; July 2005. pp. 13–17.

[6] Lee J., and Stinson D.R. "Deterministic key predistribution schemes for distributed sensor networks". In: Proceedings of ACM symposium on applied computing 2004, Lecture notes in computer science, vol. 3357, 2005, Waterloo, Canada, August 2004. pp. 294–307.

[7] Camtepe S.A., and Yener B. "Combinatorial design of key distribution mechanisms for wireless sensor networks". IEEE/ACM Transactions on Networking (TON) 2007;15(2)pp:346–358.

[8] M. Eltoweissy , "Combinatorial Optimization of Key Management in Group Communications," J. Network and Sys. Mgmt., Special Issue on Network Security, Mars. 2004, pp 33-50.

[9] N. Gura, A. Patel, A.Wander, H. Eberle, and S.C. Shantz. "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs". In CHES, Cambridge, MA, August 2004.

[10] Gaubatz G., Kaps J.P., and Sunar B. "Public key cryptography in sensor networks". In: 1st European workshop on security in ad-hoc and sensor networks (ESAS 2004), Mars 2004.

[11] Qiang Huang, Johnas Cukier, Hisashi Kobayashi, Bede Liu and Jinyun Zhang, "Fast authenticated key establishment protocols for self-organizing sensor networks", Proceedings of the 2nd ACM international conference on WSNs and applications, pp 141-150.

[12] Gang Han, Rui Zhou, and Hua Yang, "A SPT-Routing Key Management Scheme for Heterogeneous Wireless Sensor Networks Based on ECC", International Conference on Uncertainty Reasoning and Knowledge Engineering, IEEE 2011, August 2011, pp. 243-247.

[13] Rui Zhou and Hua Yang, "A Hybrid Key Management Scheme for Heterogeneous Wireless Sensor Networks Based on ECC and Trivariate Symmetric Polynomial", International Conference on Uncertainty Reasoning and Knowledge Engineering, IEEE 2011, August 2011, pp. 251-255.

[14] Cungang Yang, Celia Li, and Jie Xiao, "Location-based design for secure and efficient WSNs", Elsevier 2008, pp. 3119–3129.

[15] Katerina Simonova, Alan C. H., Ling, X., and Sean Wang, "Location-aware Key Predistribution Scheme for Wide Area Wireless Sensor Networks", SASN'06, ACM 2006, pp. 157-168.

[16] Yanchao Zhang, Wei Liu, Wenjing Lou and Yuguang Fang, "Securing Sensor Networks with Location-Based Keys, IEEE 2005, Mars 2005, pp. 1909-1914.

[17] Eltoweissy M, and Mukkamala R. "Dynamic key management in sensor networks". IEEE Comm. Magazine 2006;April: pp. 122–130.

[18] M. Eltoweissy, "Group Key Management Scheme for Large-Scale WSNs" Ad Hoc Networks, 2005, pp.796-802.

[19] Mohamed Megahed, and Dimitrios Makrakis, "SurvSec: A New Security Architecture for Reliable Network Recovery from Base Station Failure of Surveillance WSN", 2nd International Conference on Ambient Systems, Networks and Technologies, ANT 2011, August 2011, pp. 141-148.

[20] Krzysztof Piotrowski, Peter Langendoerfer and Steffen Peter, "How Public Key Cryptography Influences Wireless Sensor Node Lifetime", Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks, ACM 2006, pp. 169-176.

[21] Qi Mi, John A. Stankovic, and Radu Stoleru, "Practical and secure localization and key distribution for wireless sensor networks", Journal of Adhoc Networks, Elsevier 2012, August, pp. 946-961.

[22] Leonardo B. Oliveira, Hao C. Wong, M. Bern, Ricardo Dahab, and A. A. F. Loureiro, "SecLEACH – A Random Key Distribution Solution for Securing Clustered Sensor Networks", Proceedings of the Fifth IEEE International Symposium on Network Computing and Applications NCA 06, IEEE 2006, pp. 145-154.

[23] Chunguang Ma, Guining Geng, Huiqiang Wang, and Guang Yang, "Location-aware and secret share based dynamic key management scheme for WSN", Networks Security, Wireless Communications and Trusted Computing Conference, IEEE 2009, April, pp. 770-773.

[24] Erik Dahmen and Christoph Krau, "Short Hash-Based Signatures for Wireless Sensor Networks", 8th International Conference on Cryptology and Network Security, ACM 2009, pp. 463-476.

[25] Di Pietro, L. V. Mancini, and A. Mei, "Efficient and resilient key discovery based on pseudo-random key pre-deployment", IEEE Workshop on Wireless, Mobile, and Ad Hoc Networks, April 2004, pp. 2132-2140.

 ISBN: 978-1-61208-207-3