VLKM: Virtual Location-Based Key Management Scheme for Wireless Sensor Networks

Rohit Vaid
CSE Department
M. M. Engineering College,
M. M. University,
Mullana, Ambala, Haryana, India-133207
rohit_vaid1@rediffmail.com

Vijay Katiyar (Member, IEEE)
CSE Department
M. M. Engineering College,
M. M. University,
Mullana, Ambala, Haryana, India-133207
dean.acadaffairs@mmumullana.org

Abstract— **Designing an energy efficient key management scheme to secure Wireless Sensor Networks is a challenging task because sensor nodes in the network are resource constrained. If an initial key is used in the network lifetime, a key stolen by an unauthorized node will results in data compromised that is generated in the network. So a re-keying is necessary after a specified number of rounds to avoid the side effect of stale key in the network. In clustering environment, a number of keys are needed for every sensor. If the role of sensor is cluster head, then one key is required to collect data from all cluster members. This key is shared between the cluster head sensor and all the sensors which are members of that cluster. The different key is required to transfer the aggregated data to base station this key is only shared between the sensor node which is cluster head and the base station. But if the role of cluster head is changed from one sensor to different sensor randomly, a new key will be require that is share between sensor node and new cluster head and also a new key will be required that is shared between this new cluster head and the base station. If the scheme is followed then re-keying after every round is a bottleneck of the network as more than one re-keying is require for every sensor. In this paper, we have presented a new virtual location based key management scheme (VLKM). This scheme used virtual location to generate a round key for every sensor. Simulation results show that proposed scheme performs better than other comparable schemes in the literature without increasing the communication overheads.**

Keywords- Backward Secracy, KMS, Security, key distribution, Forward Secracy, Clustring, rekeying, WSNs

I. INTRODUCTION TO WSNs

Wireless sensor network (WSN) consists of large number of battery-operated sensor nodes. These sensors are very small in size. They have also a built-in processor that is used for the computing functions. In case of wireless sensor network, communication among the sensors is done using wireless transceivers. So every sensor is equipped with a built in antenna that will help them in communication to other sensors in their limited communication range. Each sensor consist of four subsystems: Power Supply subsystem, sensing subsystems, processing sub systems and communication Subsystems. So with the help of these subsystems, sensors are able to sense the environment, compute simple tasks and exchange data among each other. But all the sensors are resource constrained in terms of memory, energy, processing power and communication bandwidth. Every subsystem uses energy for their working. Once the battery is drained, sensor nodes are useless. The situation of network disconnection is also arises if battery is drained in few of the nodes. So energy consumption by a node is a critical aspect, in order to increase the lifetime of the network. In most of the cases it is very difficult to recharge or replace the battery. Thus it is necessary that a protocol in WSN must be energy efficient. Sensor nodes are usually deployed in harsh or hostile environments such as battlefield, environmental monitoring or disaster area where they are operated without any attendance. Thus unattended operation makes the secure data aggregation even harder.

A. Key Management Requirements in WSNs

The basic goal of key management scheme in WSNs is to protect the information communicated over the network from attacks and misbehavior. The key management requirements in WSNs include:

1) Memory Storage (MS): A sensor node is usually resource constrained in terms of memory. Therefore, it is important that the amount of memory needed by a key management scheme is minimum. In general, the simpler the protocol is, the more memory space should be made available for storing the security credentials.

2) Communication Overhead (CO): In most key management scheme (KMS), the nodes must exchange control messages with their group members through communication channels in order to establish a group key. Some protocols may require the exchange of a small amount of control messages to share a group key wehile other protocols may need to undergo a complex negotiation among them.

3) Resource overhead (RO): Sensor nodes are resource constrained in terms of processing and storage. So it is important to determine the amount of resources (such as time and storage) necessary to execute the encryption algorithm to implement the security in the existing protocol. Fortunately, there are many KMS that are not very computationally efficient.

4) Key Security (KS): It should be necessary that the key distribution or key updation process must be secure by itself. So the protocols do not need to exchange sensitive information (e.g. Key, node ID, etc) for this purpose.

5) Forward secrecy: This ensures that a compromised current secrets or keys should not be able to compromise any secret or key used in future.

6) Backward secrecy: This ensures that a compromised current secrets or key should not be able to compromise any earlier secret or key.

B. *Type of Key Management Schemes*

There are two fundamental key management schemes that are used in WSNs, i.e. static and dynamic. In static key management scheme, the sensors have a fixed number of keys loaded prior to network deployment but in dynamic key management schemes, the key is redistributed periodically or on demand as needed by the network. One significant disadvantage of this scheme is that it will increase the communication overhead due to keys redistribution to each and every node in the network. But proposed scheme calculates dynamic key by running a key generation function periodically or on demand as needed by the network. This function generates a dynamic key for each sensor with the help of its virtual location. There are many reasons for key refreshment that includes: updating keys after a key revocation has occurred, refreshing key such that it does not become stale or changing keys due to dynamic changes in the topology.

Since a sensor node will be either forwarding the aggregated data by collecting it from all cluster members if it is a cluster head or injecting its own data to the cluster head if it is not a cluster head. In case if it is a cluster head, it will needed two different keys, one that is shared between this sensor and the base station only and second key that is shared between this sensor and all the cluster members of the same cluster. But in other case when it is not a cluster head then it needs a single key that is shared between this sensor and the cluster head.

II. Literature Review

Recently, many schemes were proposed to secure the communication in WSNs. In this section, we present a brief overview of the related works that are used to enhance the security in wireless sensor networks.

In [2] the author analyzed basic issues related to security in WSNs. Two new kind of attacks (Forward and backward secrecy) identified in [2], are very serious threats in any type of dynamic key management scheme. So these attacks must be considered when design a key management scheme.

In [5] the author introduces an energy-efficient Virtual Energy-Based Encryption and Keying (VEBEK) scheme for WSNs. The proposed scheme reduces the number of transmissions needed for rekeying to avoid stale keys. The key used in the encryption process dynamically changes as a function of the residual virtual energy of the sensor. Thus, a one-time dynamic key is employed for one packet only and different keys are used for the successive packets of the stream. The intermediate nodes along the path to the sink are able to verify the authenticity and integrity of the incoming packets using a predicted value of the key generated by the sender's virtual energy, thus requiring no need for specific rekeying messages. But synchronization is a big issue in this scheme. Once virtual energy is unsynchronized, the packets will never decrypt. The result of not decrypting the packet is that it is completely impossible to differentiate between authenticated and malicious nodes.

In [6] author addresses pair-wise and triple key establishment problems in wireless sensor networks (WSN). The scheme presented in [6] is highly resilient against node capture attacks and is applicable for mobile sensor networks while preserving low storage, computation and communication requirements. Author proposed a novel concept of triple key distribution, in which three nodes share common keys to secure forwarding, detecting malicious nodes and key management in clustered sensor networks. The scheme is based on polynomial and a combinatorial approach (using trades) for triple key distribution.

In [7] the author presented an efficient key distribution scheme. The proposed scheme is useful to secure data-centric routing protocols in Wireless Sensor Networks. The proposed scheme bootstraps secure key distribution with a centralized process which gives a multi-level hierarchical organization to WSNs. The scheme permits to use local key distribution process to establish Group Key and Pair-wise Key. These two types of keys are useful to secure respectively data request diffusion and data forwarding through multi-hop routing paths. The scheme is very suitable in dynamic topology.

In [8] the author propose a lightweight implementation of public key infrastructure known as cluster based public infrastructure (CBPKI), CBPKI is based on the security and the authenticity of the base station for executing a set of handshakes intended to establish session keys between the base station and sensors over the network used for ensuring data confidentiality and integrity.

III. SYSTEM MODEL

In this section, we present virtual location based dynamic key management scheme (VLKM). We first give an overview of the technique and then present the details of the protocol.

A. *Division of Network into Clusters*

Proposed scheme uses the concept of physical clustering, i.e. every cluster is identified by its physical boundary. The boundary of each cluster is similar in size. Number of clusters into horizontal and vertical directions is decided in the setup phase. All the clusters are equal in size. Each cluster is indentified by its cluster ID.

There are two type of clustering scheme that is used in wireless sensor networks, i.e. static and dynamic. In static clustering scheme, clusters are fixed and there is no update into size and member of the cluster after formation. But in dynamic clustering scheme, the size and members of every cluster are changed in every phase of cluster formation. The members of every cluster are increases or decreased in every cluster formation, thus the size of cluster grows or shrink accordingly. Proposed model uses the concept of static clustering. Any model is used for clustering. But once the cluster is decided for any sensor, it is fixed and permanent for every round throughout the network lifetime. In proposed model, the cluster for any sensor is decided by its physical location in the network as shown in Figure 1.

B. *Random Locations*

All sensors in the network are deployed randomly as shown in Figure 1. Every sensor is provided with a virtual location. Virtual location is a random X and Y coordinates. This virtual location is used to generate a key that is used in the network. This location is given to every sensor within its cluster boundary. Similarly every cluster has also provided a random cluster virtual location. This virtual location is also a random location within cluster boundary. This location is also saved in the memory of all the cluster members of the same cluster. Virtual location of all the clusters is shown in Figure 1.

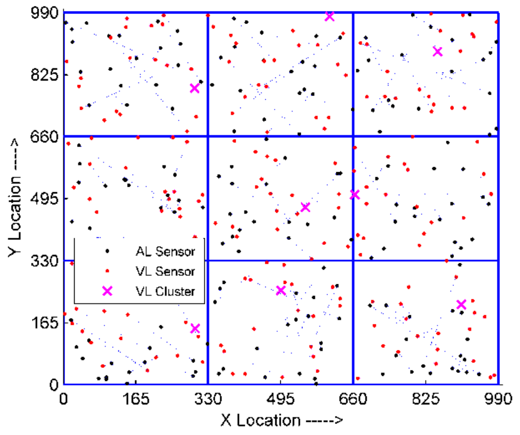

Figure 1. Random Sensor Deployment and Virtual locationing

C. *Virtual Origins*

A virtual origin is used for the virtual locations, i.e. a random virtual origin is provided to the network and all clusters will map their virtual locations on this virtual origin of the network. Similarly virtual origin is also provided to every cluster and this origin is any location within the cluster boundary as shown in Figure 2.

This model is used in the networks where more than one key is required for any type of sensor. One key that is shared between sensor node and base station and other key is shared between all the group members, i.e. member of the same cluster. The types of keys used in wireless sensor networks are given below:-

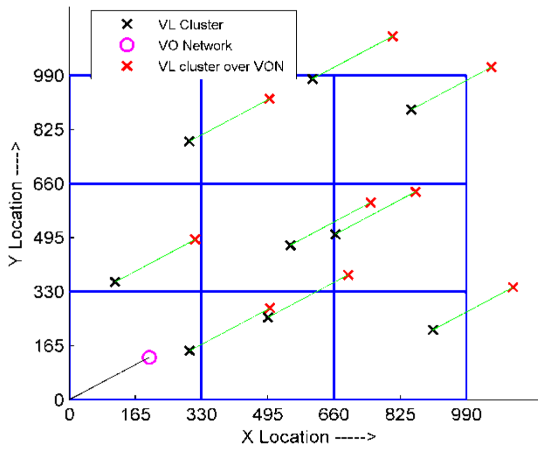

Figure 2. Cluster Virtual Location Mapping

1) Cluster key (KC_j): This key is shared between all the sensors of jth cluster. Where the value of j is between one and number of clusters (NOC) in the network, i.e. $1 \geq j \geq$ NOC. This cluster key is shared between more than two sensors and is only known to all the members of jth cluster and also to the base station. If a normal sensor node 'A' which is not a cluster head, wants to transfer the sensed data to the cluster head sensor 'B' (head of the same cluster) then 'A' will use its cluster key to transfer the data to 'B'.

a) Every cluster has an initial virtual location, virtual boundary, and virtual speed of movement, virtual angle of movement and virtual direction of movement. These parameters are only known to all the members of that cluster.

b) The network is provided a virtual origin and all the clusters map their virtual locations on this virtual origin as shown in Figure 2. The coordinates of this virtual origin is not (0, 0). For example if virtual location of any cluster is (5, 2) and the coordinates of virtual origin is (3, 2) then the virtual location of this cluster after mapping is (8, 4).

c) All the sensor of same cluster will move virtually in same direction, with same speed and angle of movement.

d) Current location of any cluster is updated, every time when the members of that cluster are moved. This current location is used to generate a key.

e) All the sensors will calculate current virtual location for current round. This loction contains 'X' and 'Y' coordinates (decimal value upto two digit decimal place). This current location is same for all the members of same cluster.

f) All the sensors will apply one way hash function on current virtual location to produce the cluster key used in current round.

g) Whenever all the sensors of a cluster are elected as a cluster head, all sensors will update their virtual location thus current virtual location is updated that results in updation of cluster key.

h) To change the cluster key for any clusters, only virtual location of that cluster is updated.

i) Virtual origin of the network is updated when all the sensors of entire network are elected as a cluster head.

2) Sensor key (K_i): This key is shared only between the sensor node 'i' and base station. This key is different for all the sensors in the network. Where the value of i is between one and number of sensors (N) in the network, i.e. $1 \geq i \geq$ N. If some sensor node 'A' is elected as a cluster head then 'A' will use its sensor key to transfer the information to the base station.

a) Every sensor has an initial virtual location, boundary, speed, angle and direction of movement, only known to the sensor and the base station only. This location and other parameters are different and independent than the parameters of a cluster.

b) All the sensors will map their virtual locations on virtual origin of the cluster as shown in Figure 3.

c) Every sensor will move virtually in its virtual boundary with a specified direction, speed and angle of movement.

d) Sensors will calculate its current virtual location for current round. This loction contains 'X' and 'Y' coordinates (decimal value upto two decimal places). This current location is different and independent for all the members in a network.

e) Sensors will apply one way hash function on its initial virtual location and current virtual location to produce the sensor key used in current round.

f) Whenever all the sensors of the cluster are elected as a cluster head, all sensors will update their virtual location thus current virtual location is updated that results in updation of sensor key.

g) To change the sensor key for a particular sensor, only virtual location of that sensor is updated.

h) Virtual origin of the network is updated when all the sensors of entire network are elected as a cluster head.

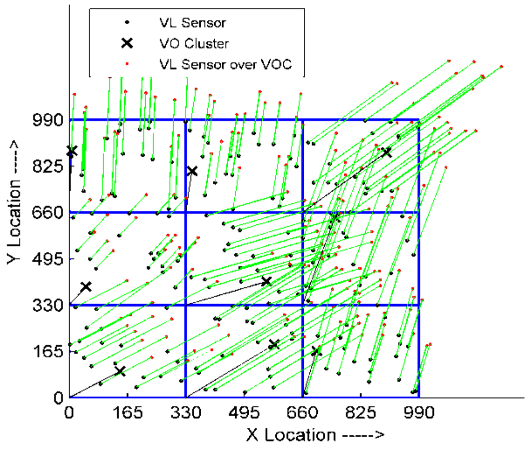

Figure 3. Sensor Virtual Location Mapping

D. *Virtual Movements*

Every sensor has a unique virtual boundary. This virtual boundary is a surrounding area around the sensor where the senor is free to move with a specific angle in a particular direction with a constant speed. When sensor hits to its virtual boundary by moving virtually, its direction of movement is changed accordingly, i.e. from Bottom Right (BR) to Right Top (RT) similarly from RT to Top Left (TL) and from TL to Left Bottom (LB) and from LB to BR. Whenever any sensor moves, its current virtual location is changed accordingly as shown in Figure 4.

All the Sensors initially mapped its virtual locations to the virtual origin, i.e. to generate the sensor key it will map its virtual location over virtual origin of the cluster but if it generates a cluster key then it will map its virtual cluster location over virtual origin of the network. With this movement speed, angle and virtual boundary, the sensor can generate any number of keys. There is no need to update the keys manually. The key is dynamic, that is updated every time when the sensor changes its virtual location by its virtual movement. When all the sensors of a cluster are elected as a cluster head, there is a need to update the cluster key. So all the sensor of that cluster updates there virtual location by moving themselves to a new location with a given virtual angle, speed and boundary. This will generate a new cluster key for next phase, till all the sensors in a cluster are not elected as a cluster head once again. Similarly when there is a need to update the sensor key which is shared between a particular sensor and the base station, the sensor updates its virtual location by moving itself to a new location with a given virtual angle, speed and boundary. This will generate a new sensor key for next phase. The sensor will generate dynamic key by applying a one way hash function on its initial virtual location and current virtual locations. To generate a next dynamic key, it will change its current virtual location by moving itself to a new virtual location again with same angle and direction with a constant speed.

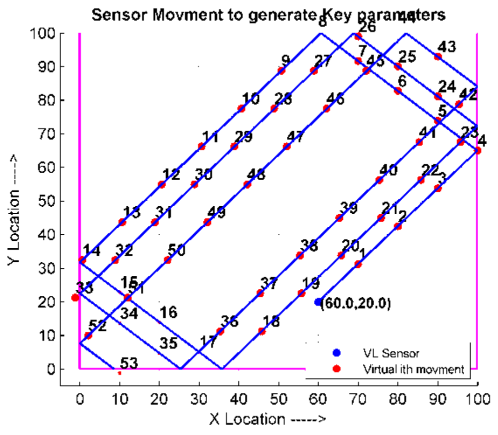

Figure 4. Sensor movement within virtual boundary

Its initial virtual location, angle of movement speed of movement and virtual boundary are already known to base station so base station easily calculates dynamic key of all the sensors of network. Normal sensors will transmit the sensed data to the cluster head sensor with cluster key. Similarly, cluster head will receive data from normal sensors with the help of a cluster key. Cluster head will transmit the aggregate data to the base station so it will use its sensor key (shared between cluster head and base station only). As base station will receive data from cluster heads only so it will calculate keys only for cluster head sensors. To balance the power consumption between all the nodes in the cluster it is necessary to rotate the role of cluster head. This time base station needs a different key that is shared between base station and new cluster head.

If all the sensors in a network has same size virtual boundary, the virtual locations of two different sensors is different irrespective of their boundary because their boundary surrounds to their virtual location which is different for each sensor and only known to that sensor. Otherwise there are several other methods to produce this difference, i.e. with different size of virtual boundary, with different angle of virtual movement or with different speed of virtual movement etc. *Compute Virtual Location (CVL)* is an algorithm that specifies how the sensor is moving into its virtual boundary. The abbreviations used in this algorithm are shown in Table 1.

Table 1: CVL Notations

Notation	Description
IVL	Iinitial virtual location (IVX, IVY) mapped onto network virtual origin
Cr	Current round
VL_{Cr-1}	Virtual location (VX_{Cr-1}, VY_{Cr-1}) in previous round
VL_{Cr}	Virtual location (VX, VY) for current round
VB	Virtual Boundary
VA	Virtual angle of movment in degree
VS	Virtual speed (distance covered by a sensor in moving a single round)
VD	Current virtual direction of movment, i.e left to right (LR), right to top (RT), top to left (TL) or from left to bottom (LB).

Algorithm 1: Compute Current Virtual Location (VX, VY) for current round (Cr).

Input: Virtual Location in previous round VL_{Cr-1}, Virtual Boundary (VB), Virtual angle of movement in degree (VA), Virtual speed of movement (VS) and current virtual direction of movement (VD);
Output: Virtual location (VL_{cr});
Procedure: CVDL (VL_{Cr-1}; VB;; VA; VS; VD)

```
1.  Begin
2.  Set previous location as a start location
    a.  X1=VX_{Cr-1};
    b.  y1=VY_{Cr-1};
        i.   opposite= VS*sin(ang);
        ii.  adjacent= VS*cos(ang);
3.  Switch(VD)
    a.  case(BR)then
        i.   if((x1+adjacent)>Vmax_x) then
             a)  VX=Vmax_x
             b)  Direction=RT
        ii.  elseif((y1+ opposite)>Vmax_y) then
             a)  VY=Vmax_y
             b)  Direction=TR
        iii. Else
             a)  VX=x1+adjacent
             b)  VY=y1+opposite
    Endcase (3.a)
    b.  case(RT)then
        i.   if((x1-adjacent)<Vmin_x) then
             a)  VX=Vmin_x
             b)  Direction=RL
        ii.  elseif((y1+ opposite)>Vmax_y) then
             a)  VY=Vmax_y;
             b)  Direction=TL
        iii. Else
             a)  VX=x1-adjacent
             b)  VY=y1+opposite
    Endcase (3.b)
    c.  case(TL)then
        i.   if((x1-adjacent)<Vmin_x) then
             a)  VX=Vmin_x
             b)  Direction=LB
        ii.  elseif((y1- opposite) < Vmin_y) then
             a)  VY=Vmin_y;
             b)  Direction=BR
        iii. Else
             a)  VX=x1 - adjacent
             b)  VY=y1 - opposite
    Endcase (3.c)
    d.  case(LR)then
        i.   if((x1 + adjacent) > Vmax_x) then
             a)  VX=Vmax_x
             b)  Direction=BR
        ii.  elseif((y1- opposite) < Vmin_y) then
             a)  VY=Vmin_y;
             b)  Direction=RT
        iii. Else
             a)  VX=x1+ adjacent
             b)  VY=y1 - opposite
    Endcase (3.d)
4.  End
```

E. *Key Generation*

The virtual dynamic Keying module will generate dynamic key by applying one way hash function on initial and current virtual locations.

The location function produces a current virtual location that is calculated when this function runs or when the sensor changes its virtual location by moving virtually within virtual boundary to calculate the key that is used in the encryption process. This module takes old virtual location (VL) that is used in the previous round, Virtual direction of movement (VD) that is used or updated in the previous round, Virtual boundary (VB) that is virtual surrounding rectangular area around the sensor and virtual angle of movement. This module calculates current virtual location (CVL). This location is used by the keying module to generate the dynamic key used in the encryption process.

1) Cluster key (CK_j) is calculated by applying a one way hash function on cluster initial virtual location ($CIVL_i$) and current virtual location (CVL_i).
So Cluster key (KC_i) =f_H ($CIVL_i$, CVL_i)
Where CIVL is the cluster initial virtual X and Y locations and CVL_i is cluster current virtual X and Y locations. Both these X and Y location are decimal numbers upto two digits. Figure 5 shows different virtual locations of different clusters with same boundary size, same angle of movement and same direction of movement.

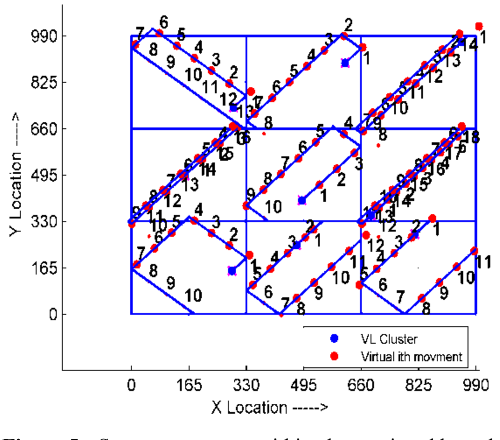

Figure 5. Sensor movement within cluster virtual boundary

2) Sensor key (K_i) is calculated by applying a one way hash function on sensor initial virtual location (IVL_k) and current virtual location (VL_k).
Sensor key (KS_k) = f_k (IVL_k, VL_k)

Where IVL is the sensor initial virtual X and Y locations and VL is sensor current virtual X and Y locations. Both X and Y locations are represented by a decimal number upto two digit. Figure 6 shows different virtual locations of different sensors in same cluster with same boundary size, same angle of movement and same direction of movement.

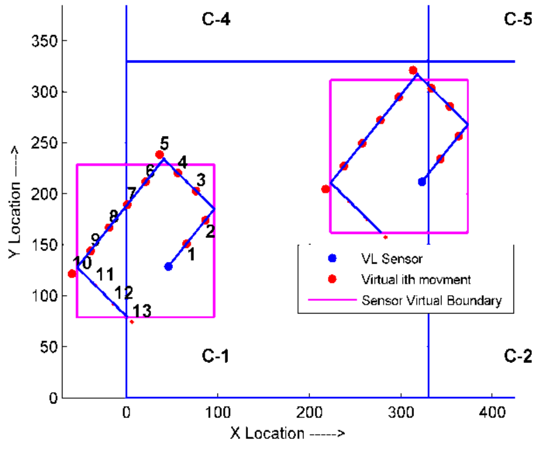

Figure 6. Two different Sensors moving in same clusters

Whenever any sensor is elected as a cluster head, all the sensor will send data to cluster head by encrypting the data with the help of a cluster key and when cluster head will receive the data from any sensor then it will decrypt the data with the same key and calculate aggregate value by mixing the data of all the sensors and send this aggregated data to base station by encrypting this data with the help of a sensor key that is only share between the sensor node and the base station. Compute Dynamic Key (CDK) is an algorithm that specifies how the sensor will generate a key used for current round with the help of initial and current virtual locations. The abbreviations used in this algorithm are shown in Table 1.

Table 2: CDK Notations

Notations	Description
IVL	Initial Virtual Location, i.e. IVX, IVY
CVL	Current virtual location, i.e. CVX, CVY
FIX	Folding Addition of Integer parts (three digits) in ‘X’ coordinates of initial and current virtual locations.
FIY	Folding Addition of Integer parts (three digits) in ‘Y’ coordinates of initial and current virtual locations.
FIZ=FIX+FIY	Folding Addition of Folded Integers parts in initial and current virtual locations.
FDX	Folding Addition of decimal parts (two digits) for ‘X’ locations in initial and ith Virtual movement
FDY	Folding Addition of decimal parts (two digits) for ‘Y’ locations in initial and ith Virtual movement
FDZ=FDX+FDY	Folding Addition of Folded decimal parts in initial and current virtual location.
KEY	Key of sensor node in current round (cr)

F. *Virtual Location-Based Keying Module*

The virtual location-based keying module (VLKM) is one of the primary contributions of this paper. It is method that produces a dynamic key that is then used by the crypto module to encrypt the packet sense by a node.

After deployment, all the sensor nodes traverse virtually thus change its virtual location. The current value of the virtual location (CVX, CVY), along with the initial virtual location after origin mapping (IVX, IVY) is used as the input to the key generation function, F. the keying module uses the concept of Folding Addition (FA) instead of using the simple addition. Folding addition is a one way function which is non-inversible, i.e. input will produce an output but the output will never produce an input. The method of folding addition is given in algorithm 2.

In folding addition method, if integer two numbers (three digits in each number) are added and the result is more then three digit then LSB is added after extraction to MSB in the result. The process is repeated till the result is not a number that has three or less number of digits as given in example:

A=995, B=994, $C = A \hat{+} B$ where ‘ $\hat{+}$ ’ is a folding addition, i.e. C=995+994=1989

Algorithm 2: Folding Addition

```
While (C>999)// Folding Addition
    a.  LSB=mod(C,10)
        ii.  C=floor(C/10)
        iii. C=C+(LSB*100)
End while
C=909
```

So it is completely impossible to generate two input values A=995 and B=994 from output value C=909.

The process of key generation is initiated when sensor changes its current virtual location, after data is sensed; thus, no separate mechanism is needed to update or refresh the key. Moreover, the dynamic nature of the keys makes it difficult for an intruder to intercept any packets to break the security parameters. As described earlier, in the proposed scheme all X and Y locations are in three digit numbers upto two digit decimal place. The hierarchy can be increased upto any label. The detail of algorithm is; Separate integer parts from initial (IVX, IVY) and current virtual locations (CVX, CVY). Then separate decimal parts from both initial (IVX, IVY) and current virtual locations (CVX, CVY). After that add both integer parts of X locations by folding method, i.e. if we add two three digit numbers and result is in four digit then extract *least significant bit (*LSB) from the result and add this LSB in *most significant bit (*MSB) and repeat the process till the result is not in number that has three or less digits. Similarly add both integer parts of Y locations using folding method. Then add both the results using folding method. In the similar manner, add both decimal parts of X Locations using folding method, both decimal parts of Y Locations using folding method. Then add these two results separately. Concatenate both the result to prepare the dynamic key. The details of the algorithm are given in Algorithm 3.

Algorithm 3: Compute Dynamic Key for current round by initial virtual location (IVX, IVY) and current virtual movement (CVX, CVY)).
Input: Initial virtual location (IVX, IVY); Current virtual location (CVX, CVY));
Output: KEY$_{cr}$;
Procedure: ComputeDynamicKey (IVX, IVY, CVX, CVY)

```
1.  Begin
2.  Seprate integer parts from initial (IVX, IVY) and current
    virtual locations (CVX, CVY)
        a.  IX1=floor(IVX)
    b.  CX1=floor(CVX)
    c.  IY1=floor(IVY)
    d.  CY1=floor(CVY)
3.  Seprate Decimal parts from initial (IVX, IVY) and current
    virtual locations (CVX, CVY)
    a.  IX2=10^2*(IVX-IX1)
    b.  CX2=10^2*(CVX-CX1)
    c.  IY2=10^2*(IVY-IY1)
    d.  CY2=10^2*(CVY-CY1)
4.  Add both integer parts of X Locations in folding method
    a.  I1=IX1+CX1
    b.  While (I1>999)
          i.   LSB=mod(I1,10)
          ii.  I1=floor(I1/10)
          iii. I1=I1+LSB*100
        End While 4.b
5.  Add both integer parts of Y Locations in folding method
    a.  I2=IY1+CY1
    b.  While (I2>999)
          i.   LSB=mod(I2,10)
          ii.  I2=floor(I2/10)
          iii. I2=I2+LSB*100
        End While 5.b
6.  Add both integer part of folded X and Y locations using
    folding method
    a.  I3=I1+I2
    b.  While (I3>999)
          i.   LSB=mod(I3,10)
          ii.  I3=floor(I3/10)
          iii. I3=I3+LSB*100
        End While 6.b
7.  Add both Decimal parts of X Locations using folding method
    a.  D1=IX2+CX2
    b.  While (D1>99)
          i.   LSB=mod(D3,10)
          ii.  D1=floor(D1/10)
          iii. D1=D1+LSB*10
        End While 7.b
8.  Add both Decimal parts of Y Locations using folding method
    a.  D2=IY2+CY2
    b.  While (D2>99)
          i.   LSB=mod(D2,10)
          ii.  D2=floor(D2/10)
          iii. D2=D2+LSB*10
        End While 8.b
9.  Add both Decimal part of folded X and Y locations using
    folding method
    a.  D3=D1+D2
    b.  While (D3>99)
          i.   LSB=mod(D3,10)
          ii.  D3=floor(D3/10)
          iii. D3=D3+LSB*10
        End While 9.b
10. KEY=str2num(strcat(num2str(D3),num2str(I3)))
11. Return KEY
12. End
```

G. *Example:*

The sensor is moving into its virtual boundary. Virtual boundary of sensor is represented with bottom left X (minx), bottom left Y (miny), Top right X (maxx) and Top Right Y (maxy). The initial virtual location of sensor is (VX, VY). And this location is mapped over virtual origin (VOX=15, VOY=5). The initial direction of movement is BR. The virtual speed of movement (VS) is 20 (meter). And finally the virtual angle (VA) of movement is 40 degree. Network parameters to produce different keys for different virtual movement are given below:-

Virtual Boundary: minx=900, maxx=1000, miny=900, maxy=1000
Virtual Origin: NVOX=15, NVOY=5
Initial Virtual location: VX= 927.00, VY=927.00
Virtual location over virtual origin (IVL): VXN=942.00, VYN=932.00
Virtual Direction: VD='BR'
Virtual Angle of movement: VA= $40°$
Virtual speed of movement: VS=20 meter
Current Virtual location (CVL): CVX= 957.32, CVY= 944.86
FIX=588, FIY=287, FIZ=875, FDX=32, FDY=86, FDZ=91
KEY=91875
KEY in binary format: 10110011011100011

IV. RESULT AND DISCUSSIONS

A. *Key Duplication*

In every round a new dynamic key is provided to each and every sensor in the network. Key duplication is the process when a key is similar between two or more sensors in same round. In previous schemes, dynamic key is provided to each and every sensor in every round from a fixed key pool. To check the behavior of network, the proposed scheme is compared with the traditional scheme. There are total 54 sensors in the network. The network is assumed to run for 50 rounds. So to provide a new key in each round the size of fix keypool is enhanced from a keypool of single key to a keypool with length 100 keys. A random key is chosen from the key pool and compared with the proposed scheme. Simulation results in Figure 7 shows that key duplication is 100% in case of keypool with length 1, i.e. in every round the key of every sensor is matched with all other sensors in the network. When the length of keypool is increased from 1 to 100, key duplication is decreased from 100% to 20%. In proposed key management scheme, the key duplication % is between 0 to 1%.

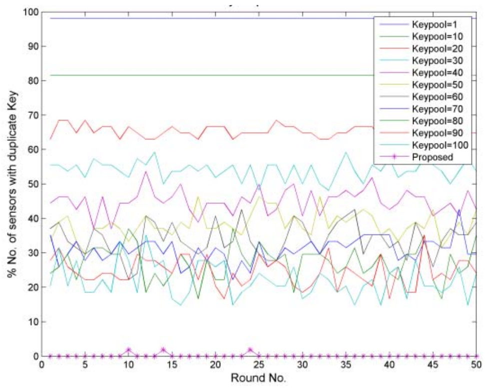

Figure 7. Key Duplication

In Figure 8, when different parameters (virtual movement, virtual angle of movement and virtual boundary) to calculate the key of any sensor are matched. Figure 8-A shows key duplication when random movement and random boundary (RMRB) is provided to each and every sensor in the network. Figure 8-B shows key duplication when same movement but random boundary (SMRB) is provided to every sensor. Figure 8-C shows key duplication when random movement but same boundary (RMSB) is provided to every sensor. Figure 8-D shows the behavior of network when same movement and same boundary (SMSB) is provided to every sensor in the network.

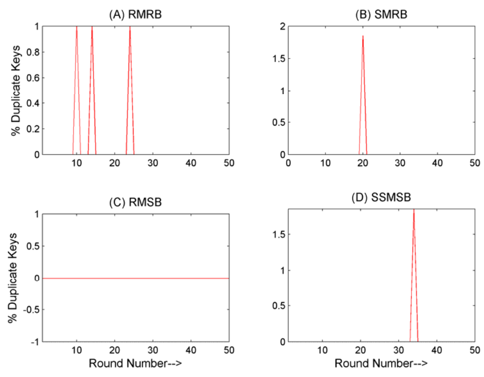

Figure 8. key duplication for different key parameters

B. *Key Uniqueness*

Total number of unique keys that are used by any sensor for all rounds is known as key uniqueness. Simulation results in Figure 9 shows that key uniqueness is 1% in case of keypool with length 1, i.e. in every round the same key is provided to every sensor in the network. When the length of keypool is increased from 1 to 100, key uniqueness is increased from 1% to 85%. But in proposed key management scheme, key uniqueness is 100%.

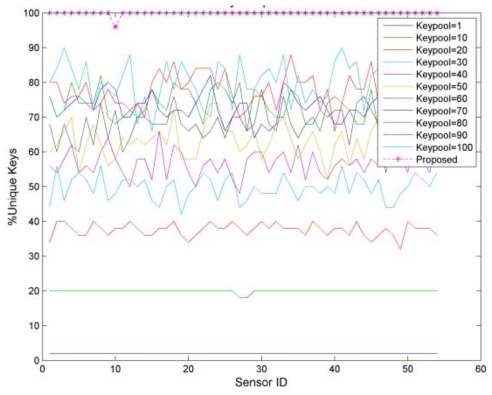

Figure 9. Keys uniqueness

In Figure 10, when different parameters (virtual movement, virtual angle of movement and virtual boundary) to calculate the key of any sensor are changed. Figure 10-A shows key uniqueness when random movement and random boundary (RMRB) is provided to each and every sensor in the network. Figure 10-B shows key uniqueness when same movement but random boundary (SMRB) is provided to every sensor. Figure 10-C shows key uniqueness when random movement but same boundary (RMSB) is provided to every sensor. Figure 10-D shows the behavior of network when same movement and same boundary (SMSB) is provided to every sensor in the network.

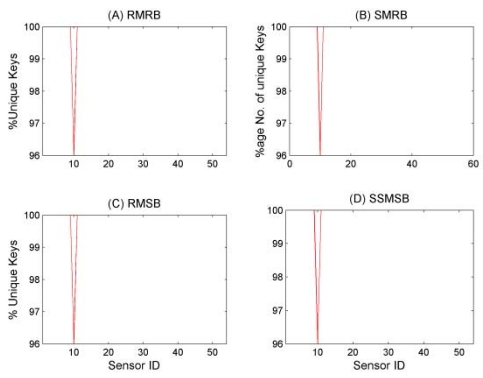

Figure 10. key uniqueness for different key parameters

C. *Key Chain Duplication*

Key chain duplication is the process when a sensor has the common key in consecutive two or more rounds. Simulation results in Figure 11 shows that key chain duplication is 100% in case of keypool with length 1, i.e. in every two consecutive rounds the same key is provided to every sensor in the network. When the length of keypool is increased from 1 to 15, key chain duplication is decreased from 100% to 4%. But in proposed key management scheme, key chain duplication is .1%.

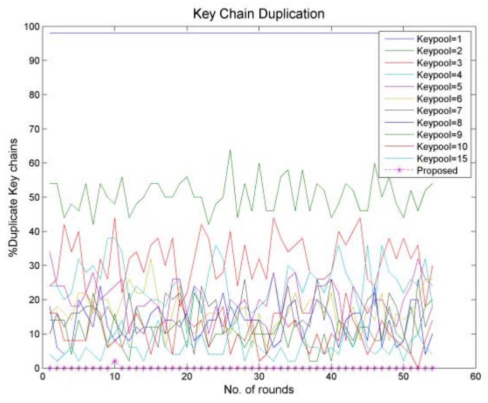

Figure 11. Number of two or more keys match in series between any two consecutive rounds

In Figure 12, when different parameters (virtual movement, virtual angle of movement and virtual boundary) to calculate the key of any sensor are changed. Figure 12-A shows key chain duplication when random movement and random boundary (RMRB) is provided to each and every sensor in the network. Figure 12-B shows key chain duplication when same movement but random boundary (SMRB) is provided to every sensor. Figure 12-C shows key chain duplication when random movement but same boundary (RMSB) is provided to every sensor. Figure 12-D shows the behavior of network when same movement and same boundary (SMSB) is provided to every sensor in the network.

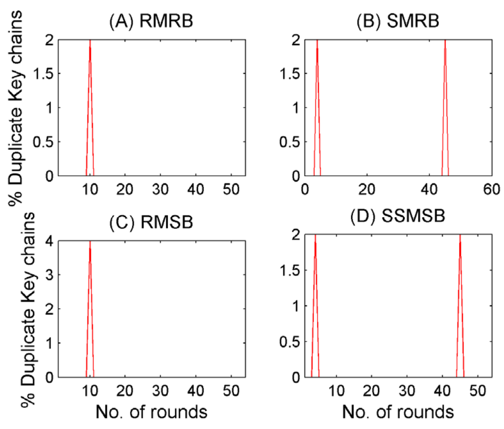

Figure 12. key chain duplication for different key parameters

V. Performance Evaluation

1. *Forward secrecy:* This secrecy ensures that a compromised key of current round should not be able to compromise any secret or key in future.
2. *Backward secrecy:* This secrecy ensures that a compromised key of current round should not be able to compromise any earlier keys or secrets of previous rounds.

The effect of compromising several key generation parameters is given below:-

a) Key compromise: Suppose that a key of current round is compromised. Keying module in VLKM uses the concept of Folding Addition (FA) which is already a one way function. So it is completely impossible to generate a virtual location with the help of a key.

b) Current virtual location compromise: Suppose the virtual location of current round is compromised. To generate the next virtual location, several other parameters are necessary, i.e. virtual boundary, virtual angle of movement and virtual speed of movement which is never being communicated over the network. So with the help of only current virtual location, it is impossible to generate virtual locations for next round or previous round.

c) Virtual location compromise: Suppose the virtual location that is used in keying module is compromised. Now to generate the key, several other parameters are also necessary, i.e. initial virtual location that is mapped onto virtual origin of cluster (if sensor key is generated) or virtual origin of network (if cluster key is generated). So again with the help of a virtual location, it is impossible to generate a key.

d) All Key parameters of a sensor are compromised: Suppose all the key generation parameters used to generate a key are compromised. Now the parameters of one sensor are different than other sensor in the network and similarly the parameters of one cluster are different than other cluster. So parameters of a one sensor will not estimate the parameters of other sensor or cluster. The key of that sensor is also compromised till a new virtual location is not allocated to that sensor. Once a new virtual location is allocated to that sensor or the virtual origin is updated, compromised key parameters are changed, thus old parameters are useless.

VI. Conclusions

In this paper, we proposed an energy-efficient Virtual location based key management (VLKM) scheme for WSNs that significantly reduces the number of transmissions needed for rekeying. This scheme is very suitable for clustering environment where more than one key is required for every sensor for the functioning of the network. Simulation results show that the proposed scheme performs better in terms of forward and backward secrecy without increasing the communication overheads. For the sake of clarity, we describe the protocol in a two-dimensional plane. However, our approach can be applied to higher-dimensional spaces as well.

References

[1] Rohit Vaid, Vijay Kumar, “Pairing based Encoding Schemes (PBES) for Secure Wireless Sensor Networks”, in International Journal of Computer Applications, Volume 70, No.17, pp. 43-49, ISSN (Online): (0975 – 8887), May 2013.

[2] Rohit Vaid, Vijay Kumar, “Security Issues and Remedies in Wireless Sensor Networks- A Survey”, in International Journal of Computer Applications, Volume 79, No. 4, pp. 31-39, ISSN (Online): 0975 – 8887, October 2013.

[3] Chan, Haowen, Adrian Perrig, and Dawn Song. "Random key predistribution schemes for sensor networks." In Security and Privacy, 2003. Proceedings. 2003 Symposium on, pp. 197-213. IEEE, 2003.

[4] Eschenauer, Laurent and Virgil D. Gligor. “A key-management scheme for distributed sensor networks”, in Proceedings of the 9th ACM conference on Computer and communications security, pp. 41-47. ACM, 2002.

[5] Arif Selcuk Uluagac, Raheem A. Beyah, Yingshu Li and John A. Copeland, “VEBEK: Virtual Energy-Based Encryption and Keying for Wireless Sensor Networks” in IEEE Transactions on Mobile Computing, Vol. 9, No. 7, July 2010.

[6] Ruj, Sushmita, Amiya Nayak, and Ivan Stojmenovic, “Pairwise and triple key distribution in wireless sensor networks with applications”, in Computers, IEEE Transactions, pp. 2224-2237, Vol. 62, No. 11, November 2013.

[7] Guermazi, Abderrahmen and Mohamed Abid, “An Efficient Key Distribution Scheme to Secure Data-Centric Routing Protocols in Hierarchical Wireless Sensor Networks”, in 2nd International Conference on Ambient Systems, Networks and Technologies (ANT) Procedia Computer Science pp. 208-215, Vol. 5, 2011.

[8] Kadri, Benamar, Djilalli Moussaoui, Mohammed Feham, and Abdellah Mhammed. "An Efficient Key Management Scheme for Hierarchical Wireless Sensor Networks." Wireless Sensor Network, pp. 155-161, Vol. 4, No. 6, June 2012.