CHAPTER 4

A FRAMEWORK FOR DYNAMIC SECURE AGGREGATION AND AUTHENTICATION (DSAA)

Hop-by-Hop mechanism needs data authentication to ensure integrity in the sensor network. However, Encrypted Data Aggregation (EDA) helps to provide data confidentiality, authentication and integrity in WSN. Data confidentiality in Wireless Sensor Network (WSN) faces major security challenges and issues due to insecure wireless channel. Packet loss and false data injection in sensor network are high due to inefficient mechanisms of key management techniques. In this thesis, an efficient hybrid key management technique is proposed by combining symmetric and asymmetric key agreement to ensure data confidentiality under cluster environment. This chapter presents a framework that ensures dynamic secure aggregation and authentication.

4.1 PROPOSED DYNAMIC SECURE AGGREGATION AND AUTHENTICATION (DSAA) FRAMEWORK

WSN consists of spatially distributed and autonomous sensors to monitor the physical and environmental conditions. It passes the sensed data to the base station through the network with the co-operation of other sensor nodes. It is often deployed in a hostile environment for transmitting confidential data without any human intervention. The proposed Dynamic Secure Aggregation and Authentication (DSAA) scheme provides data confidentiality, authentication and integrity of the sensor data and prevents

the clone attack in sensor network. It also eliminates data loss and false data injection by SCSHA. Sensor nodes are randomly deployed and formulated as a clustered architecture in order to increase the lifetime of the network. It consumes less power for providing additional services such as security, which operates on unrechargeable batteries. In this chapter, a secured clustering mechanism is proposed to provide security for wireless sensor network. Figure 4.1 shows the systematic implementation process of the proposed DSAA framework.

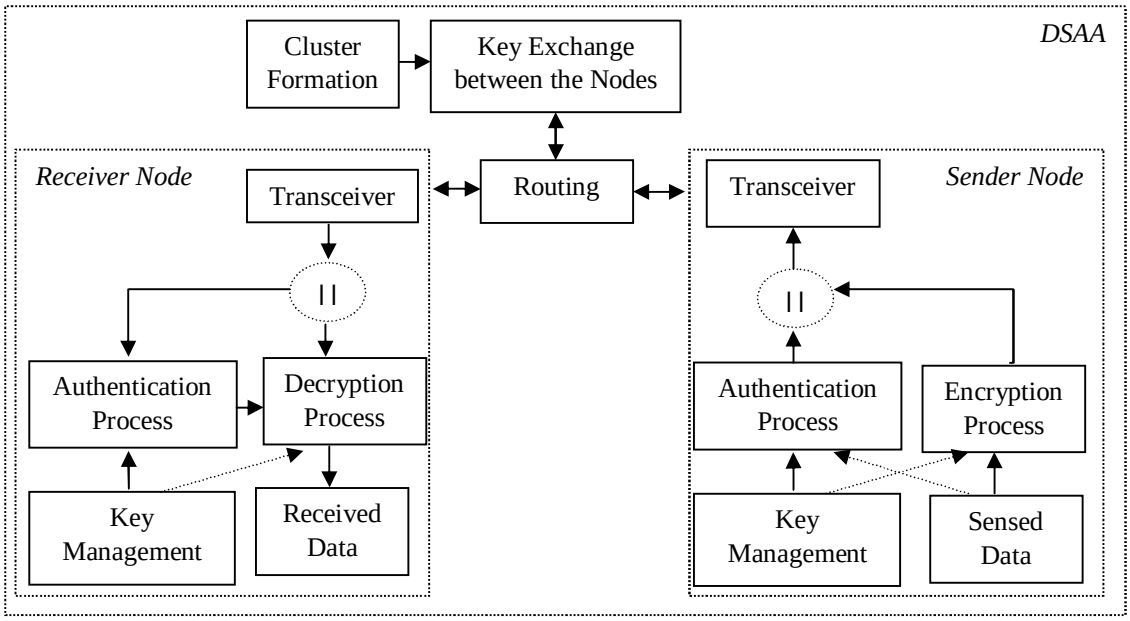

Figure 4.1 Flow diagram of the proposed Dynamic Secure Aggregation and Authentication (DSAA) framework

The proposed DSAA framework does the following process in the sender and the receiver side.

- Cluster formation
- Key generation and key exchange
- Claim Forward and Routing
- Message Authentication
- Data Encryption / Decryption process

The nodes are deployed randomly and clusters are formed. The cluster head and its members generate the shared secret key. When the sender node detects the event, the data is encrypted and authenticated using the shared secret key. This cipher data is transmitted via cluster head to reach the base station. Decryption process is enabled in every cluster head (receiver node) to authenticate the data in order to ensure the data integrity. Figure 4.2 shows the techniques and principles used in the proposed DSAA at node level.

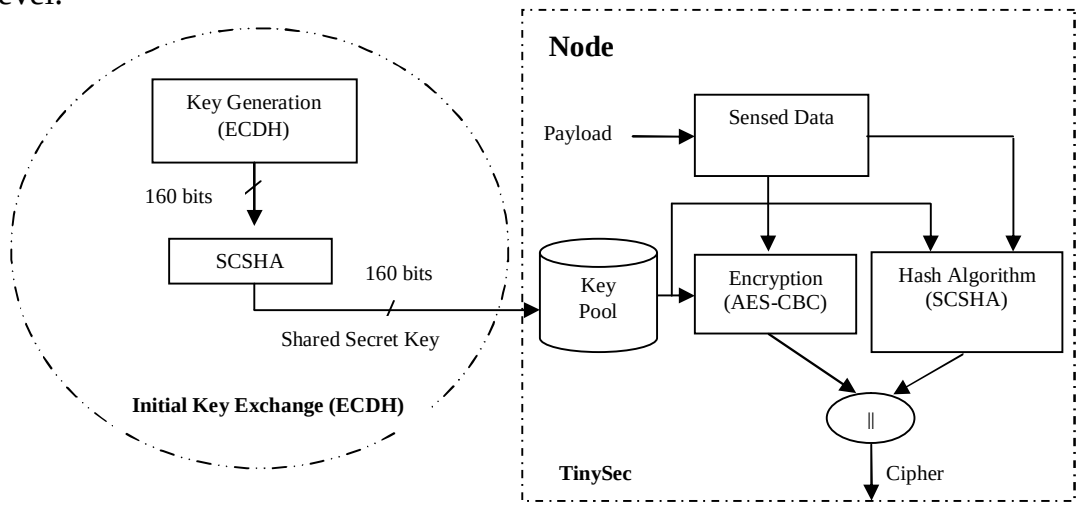

Figure 4.2 Techniques and principles of proposed DSAA

Existing Zigbee has Tiny Security (TinySec) module that provides security to the sensed data using Advanced Encryption Standard – Cipher Block Chaining (AES-CBC) and SHA-1 algorithm. The encrypted data and authenticated message digest are concatenated and considered as a cipher text, which is transmitted in the network to reach the Base Station (BS). Sensor Context Aware Routing (SCAR) algorithm (Lukosius 2006) exists in TinyOS handles the transfer of data.

This research work uses ECIES discussed in chapter 3 to secure the data. ECIES consists of ECDH, AES-CBC and proposed Scheduler based Combined SHA-1 (SCSHA) to ensure key generation, key exchange,

encryption, decryption and authentication. The detailed description and the working principle of the modules involved in the proposed DSAA framework are as follows:

4.1.1 Sensor Node Deployment and Cluster Formation

In most of the WSN applications, the nodes are deployed randomly in the hostile environment and the topology is self-organized in nature. However, it is necessary to monitor the behavior of the nodes to ensure the security. In this research work cluster based public key management technique is used for enforcing data security.

4.1.1.1 Node deployment

Several topologies are used for the deployment of wireless sensor network. Node deployment defines the number of sensor nodes to be deployed in the environment and it chooses either random or grid topology in the simulation scenario.

4.1.1.2 Cluster formation

Clusters are formed after the sensor node deployment. In general, a cluster head is selected based on the high computational capability of the sensor node group (Younis & Fahmy 2004). However, the proposed work elects a Cluster Head (CH) based on link quality and number of maximum member as a neighbor. The nodes within the communication range of cluster head are included as its member. Each node in the cluster communicates with the CH and the CH communicates with the base station via multihop. Nodes (cluster member) communicating with the cluster head is known as intra-cluster communication and each cluster head communicating with the base station is known as inter-cluster communication (Ya-nan et al 2013).

Table 4.1 shows the parameters defined to construct the environment. In the simulation, 50 nodes are considered.

Table 4.1 Parameters involved for constructing the environment

Parameters	Notations
No. of Nodes	n
Number of cluster head	m
i^{th} Node Private Key	K_i
i^{th} Node Public Key	T_i
Shared Secret Key	S_i
j^{th} Cluster Head Private Key	K_{CHj}
j^{th} Cluster Head Public Key	T_{CHj}

In the proposed work, routing is done using Sensor Context Aware Routing (SCAR) method (Mascolo & Musolesi 2006). The proposed work updates the routing for every 50 seconds.

4.1.1.3 Route update and cost estimation

Sensor Context Aware Routing (SCAR) uses probabilistic approach for movement and resource prediction, to forward the data in correct direction smartly. This technique is called as adaptive routing. The information to the sink nodes are forwarded by the neighbor nodes. MultiHopRouter method present in SCAR enables the data transfer between the nodes to reach the destination. The route is established based on the least number of hops over a reliable link by dynamic spanning tree. The reliable link is identified from the best link quality of a node when compared with the neighbor node. The spanning tree is refreshed every 10 seconds using Bcast and MultiHopRouting method (Lukosius 2006). Every node in the network sends a route request to

the root node of the spanning tree in order to route the data to the correct destination. The root node receives the route request and finds the hop count to reach the appropriate node. It reads the hop count and finds the traversal path to reach the root node (BS). From the received message, the address of the parent node and its depth information are identified and stored in each node. A route is selected by calculating the minimum number of hop count from the route table. The routing table is updated for every 50 seconds and routes are updated every 10 seconds in the proposed work. The sender node packet has control and data information for transferring the information from one node to another node. The control information packet format for routing is given in Table 4.2.

Table 4.2 Control information format (routing table)

Packet Size [in bytes]							
2	2	2	2	2	1	1	1
Source Address	Parent / Cluster Head Address	Missed Packets	Received Packets	Last Sequence Number	Hop Count	Receiving Estimation Factor (E_R)	Sending Estimation Factor (E_S)

When the neighbours are discovered, the sensor nodes start sending the sensed reading to the CH which in turn sends it to the base station. The destination address which was initially 0xffff during the time of broadcast messages now gets changed to the BS address. The format of the sensor packets contains source node address (2 Bytes), next hop node identification or cluster head address (2 Bytes), the number of missed packet by the node (2 Bytes), number of received packets from the appropriate node (2 Bytes), sequence number of the last received data from the node (2 Bytes), number of hops to reach the base station (1 Byte), two way link estimation values for each node namely receiving estimation factor (E_R) (1 Byte) and sending estimation factor (E_S) (1 Byte) to know the properties of a sensor node.

The receiving estimation factor 'E_R' of link quality is calculated using Equation (4.1):

$$E_R = \frac{2r+\left(\frac{B_{lq}*s}{s+f}\right)p}{8} \quad (4.1)$$

where, r → number of packets received by the node

B_{lq} → Beacon signal link quality

s,f → Success and failure rates respectively

p → power levels

The sending estimation factor 'E_S' of the link quality depends on the receiving estimation factor and is given in Equation (4.2).

$$E_s = \delta * E_R \quad (4.2)$$

where, δ → is the difference between received and sent packet (s-(s+f)).

Every node stores the network address of its parent in the tree and its depth. The link quality determines the choice of next hop neighbor. Figure 4.3 shows the link quality value ranging from 0 to 100. Green and red color indicates the best and worst link quality respectively. Other color (yellow) indicates the acceptable link quality.

The success and failure of link quality estimation depends on the difference between the number of packets received and missed packets of each node. The E_R and E_S will be '255' for a legitimate node under preferred condition and will be '0' under worst condition.

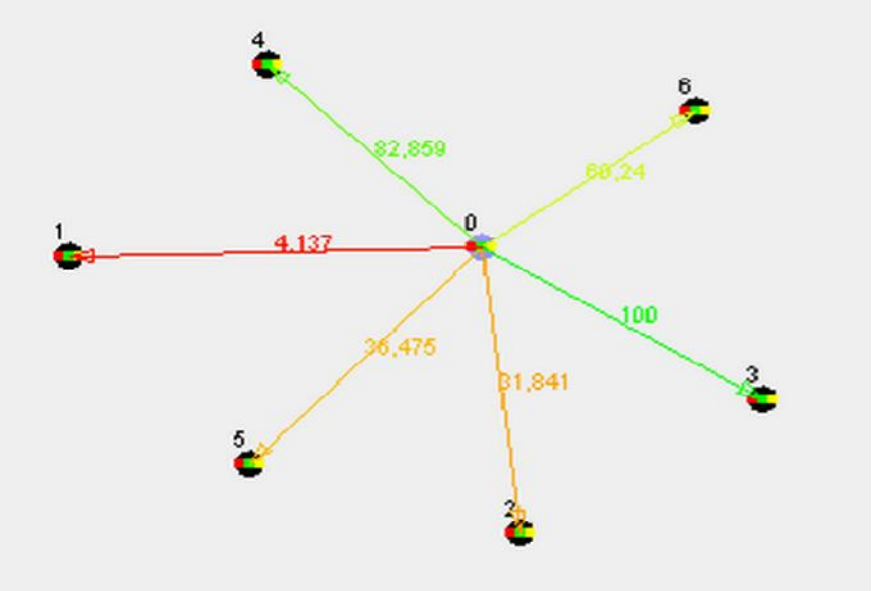

Figure 4.3 Diagram representing link quality values

The missed packets are identified if the packets are out of sequence. The routing table starts updating when the node sends the broadcast message. Initially, the hop count will be 255 which indicates that the destination node is not yet identified. The E_R and E_S is initially 0. Table 4.3 represents the format of a data packet.

Table 4.3 Data packet format

Packet Size [in bytes]									
Header				Payload					Trailer
2	1	1	1	3	X	1	1	2	1
Destination ID	Message Info	Group ID	Message Length	Source ID, loc	Data	Signal Strength	ACK	Time	CRC

ff ff	0a	7d	1a	01 00 28 00 01 00 d9 02 5f 03 7f 03 0c 00 54 02 46 02 21 03 96 03 00 03 00 00	01

Table 4.3 shows the format of data packet. The message information 0xff ff (2 bytes) denotes broadcast address, 0a (1 byte) denotes an active message, 7d (1 byte) denotes a group ID, 1a (1 byte) denotes message length, remaining (127 bytes) denotes the size of payload and (1 byte) Cyclic Redundant Check (CRC).

The destination address may be a broadcast address or a unicast address. If the address is ff, then it denotes that the destination address is a broadcast address and the packet is intended for set of users. If it is a unicast address, then the destination address of the packet consists of identity of a particular node. Active message field in the packet could be 05 or 0a. Here, active message '05' indicates that it belongs to the control message of the network, and '0a' indicates a data packet of the message. The data length field indicates the amount of the payload transmitted. The next process is key generation and key exchange to enforce security of the data packet.

4.2 KEY GENERATION AND KEY EXCHANGE PROCESS IN INTER AND INTRA CLUSTER HEAD

The ECDH discussed in Chapter 3 is used to generate and exchange a key value between inter and intra cluster heads. The key value is generated using Elliptic Curve with 160-bit elliptic curves coordinates. The private key is generated randomly from a finite field. The public key is generated by multiplying the random private key and 160-bit elliptic curve coordinates using the proposed DswNAF (scalar multiplication). A secret key is shared between the source and destination using Elliptic Curve Diffie Hellman algorithm (ECDH) to ensure the authentication.

Assume that an eavesdropper ($Node_E$) has the knowledge of $Node_A$'s public key (T_A) and $Node_B$'s public key (T_B). It computes the shared secret key $S = (k_E * T_B)$ using the private key (k_E). If the eavesdropper fails to generate the same shared secret key of source and destination i.e. $S = (k_B * T_A) \neq (k_E * T_B)$ then it is due to the different elliptic curve point chosen for source and destination pairs. Each node in the cluster generates and holds a public key, and a private key. Each sensor node in the cluster generates and exchanges a shared secret key with the cluster head. Therefore, each node in the cluster has private key, public key and a shared secret key. However, the CHs consists of (n+m+1) number of shared secret key and only one private key in its key pool. Similarly, Base Station (BS) has 'm' number of shared secret key and its private key. Here, 'n' and 'm' denotes the number of nodes and number of clusters heads respectively.

The sensor node broadcasts the encrypted public key of the node. The sensor node and cluster head communicate with each other in a secret manner by encrypting the messages using shared secret key.

4.3 ENCRYPTION / DECRYPTION

An event is observed by the sensor in an environment and forwarded via multihop communication to the base station. Along with the sensor data, the packet has location information and ID of the node. The location information of the sensor is authenticated in order to ensure the validation of exchanged location information of a sensor. The sensor data on the network is processed using ECIES algorithm (discussed in algorithm 3.6). To obtain the required key size as 128 in AES-CBC algorithm, the public key generated using LWECC algorithm is given to the Message Digest v5 (MD5) algorithm (AES 2001). Karlof et al (2004) performed encryption TinySec on sensor data using AES with the digest form of the shared secret key. The digest form of public key increases the security level of the public key rather

than the conventional key of AES encryption algorithm. The digest form of public key is given to AES 128-bit algorithm for encryption. This research work uses link-layer based security mechanism to guarantee the authenticity, confidentiality and integrity of the message.

4.3.1 Encryption of Sensed Data

The sensor data is encrypted using AES-CBC algorithm. AES is a symmetric encryption algorithm. The design of AES is based on the principle of substitution and permutation. It has fixed block size of 128-bit, 192-bit and 256-bit for plain text, cipher text and key size. In the proposed work, a combination of symmetric and asymmetric is used. This is because of the computational complexity of the symmetric-key encryption algorithm is lesser than the asymmetric-key encryption algorithm.

CBC mode is a better choice as it degrades more gracefully in the presence of repeated IVs. If we encrypt two plaintexts P, P0 with the same IV under CBC mode, then the cipher texts will leak the length (in blocks) of the longest shared prefix of P and P0. For instance, if the first block of P is different from the first block of P0 then the crypt analyst learns nothing apart from this fact. Consequently, CBC leaks only a small amount of information in the presence of repeated IVs, a significant improvement over a stream cipher (Karlof et al 2004). CBC mode is provably secure when IVs do not repeat (Verma et al 2011, Fourozon 2007). CBC mode is designed to be used with a random IV and CBC mode. The block diagram of CBC mode is given in Figure 4.4.

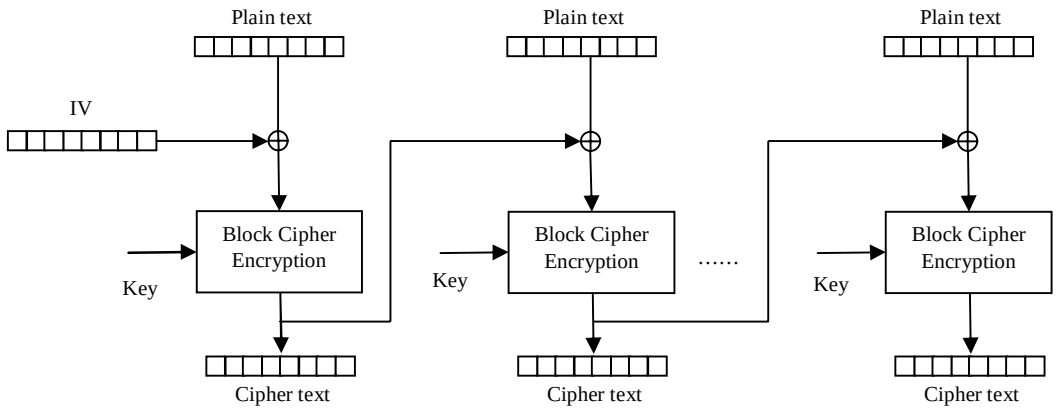

Figure 4.4 Block diagram of block cipher in CBC mode

Block diagram of CBC shows the dependency on the previous cipher text block or IV, which is effectively random and independent of the plaintext block. The obtained cipher is fed as an input to the next block etc. The final cipher text obtained is an effectively random string, which makes CBC better than other modes.

4.3.2 Authentication of Sensed Data

Secure Hash Algorithm (SHA) is used to authenticate the message using hash functions to ensure reliable communication. The SHA algorithms such as SHA-0, SHA-1, and SHA-2 produces message digest of varied length (Eastlake & Jones 2001, Forouzan 2007, Garg & Tiwari (2012)). SHA-1 is a set of cryptographic hash functions designed by the National Security Agency (NSA) and published by the NIST as a U.S. Federal Information Processing Standard (Forouzan 2007). Cost effective hardware is suggested to improve the performance of SHA algorithm (Chaves et al 2008).

4.3.2.1 Proposed Scheduler based Combined Secured Hash Algorithm (SCSHA) for WSN authentication

Hash functions are applied in several information security applications such as digital signatures, MAC and other forms of authentication. This research proposes a new method for generating digital signature based on Secured Hash Algorithm. This research work uses two SHA modules in parallel to provide highly secured, more efficient and higher throughput algorithm to compute Message Digest (MD) as shown in Figure 4.5.

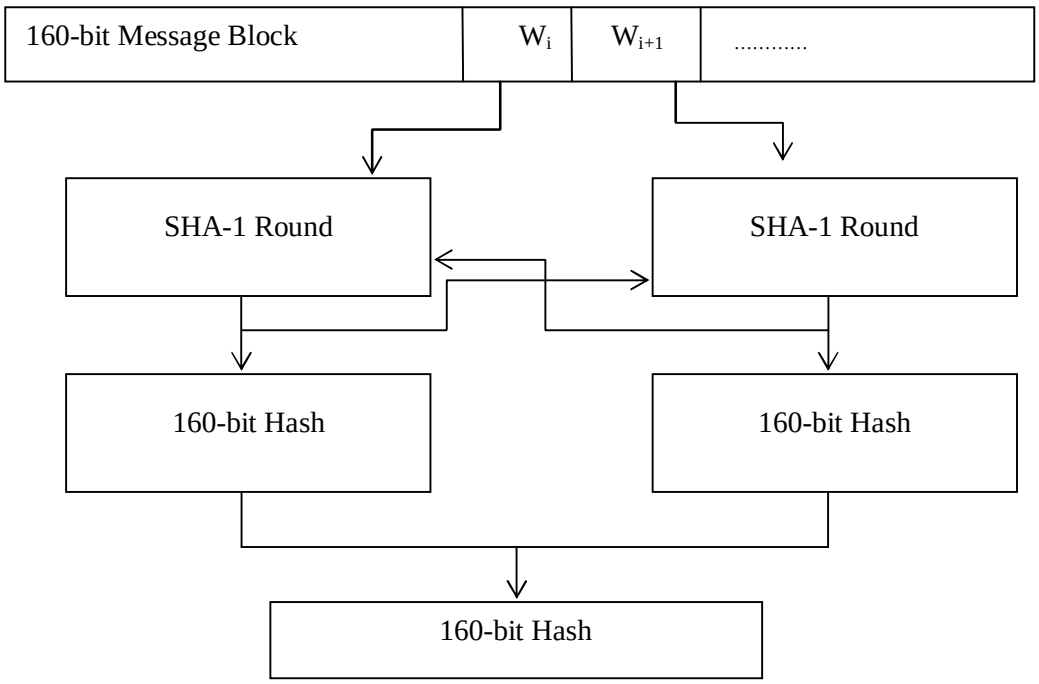

Figure 4.5 Proposed design for Combined SHA

The data is split into 128-bit each and the two SHA blocks are designed to operate simultaneously on alternate 32-bit words. The outputs of the SHA-1 rounds are cross connected to the other module to break the dependency of the data. The proposed work increases the probability of predicting the original data by permutation process in SHA-1 algorithm. It is highly difficult to get the original data as the hacker needs to perform 2^{352}

combinations to predict the original data. This kind of attack is referred as Birthday Paradox Attack (BPA). To avoid data dependency, this research proposes a message scheduler as shown in Figure 4.6. The proposed parallelization breaks the data dependency by increasing the permutations involved in the SHA-1 algorithm. Hence, the final message digest obtained will be more secured.

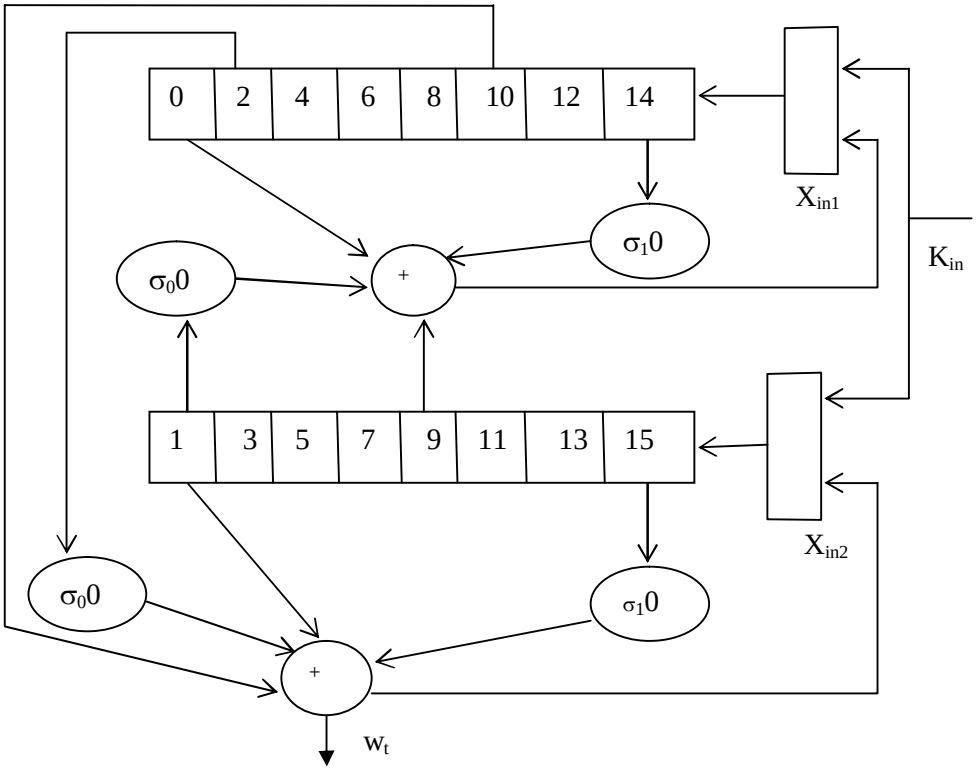

Figure 4.6 Message scheduler

To generate two words in a single clock cycle, 128 bits of message is given as input (K_{in}) to the scheduler block. This message scheduler uses two 8*64 bit shift registers instead of a single 16*64 bit shift register (Emam & Emami 2007). The 128-bit message input is then split into two 64-bit chunks (X_{in1} and X_{in2}) and fed to two register banks simultaneously.

$$w_t = \sigma_1^{512}(w_{t-2}) + w_{t-7} + \sigma_0^{512}(w_{t-15}) + w_{t-16} \quad (4.3)$$

Where $\sigma_0^{512}(x) = ROTR^1(x) \oplus ROTR^8(x) \oplus SHR^7(x)$,

$\sigma_1^{512}(x) = ROTR^{19}(x) \oplus ROTR^{61}(x) \oplus SHR^6(x)$,

ROTR – circular right shift (rotation),

SHR – left shift with padding zeros on the right,

+ - addition modulo, w_t is the word which derived from 512-bit message. The message scheduler generates two input words simultaneously to two SHA-1 modules to avoid data dependency. Moreover, the number of clocks used by the proposed work is lesser than the Combined Secure Hash Algorithm (CSHA) (Emam & Emami, 2007). In CSHA, 16 additional clock cycles are required to shift the data into a single 16*64 bit shift register.

DSAA scheme provides the security for the data; hence, the Hop-by-Hop (HbH) security increases the overhead for a node and the Encrypted Data Aggregation (EDA) reduces the number of decryption process in the node. Traditional encryption algorithms combined with efficient data aggregation allow encrypting messages hop-by-hop. This means that an aggregator node has to decrypt each received message, then aggregate the messages according to the corresponding aggregation function and finally encrypt the aggregation result before forwarding it. Hence, end-to-end encryption schemes are preferred for security.

4.3.3 Data Aggregation

The sensed values are encrypted using the shared secret key (S) generated by the key management algorithm and sent to the cluster head for aggregation. The values received by the cluster head are aggregated and forwarded it to the base station via multi-hop communication. In general,

$D_{S_{CH}}(E_{S_A} + E_{S_B} + E_{S_C} + \ldots E_{S_n})$ will not be equal to the sum of the plaintexts of $Node_A + Node_B + Node_C + \ldots Node_n$, where *D* and *E* denote the decryption and encryption process on each aggregator node respectively. This process is carried in intra cluster where the cluster head act as an aggregator node which does either addition or multiplication operation in order to get the cipher text. This is called as privacy homomorphism. Myletun et al (2006) investigates the use of additive homomorphic public-key encryption mechanism for WSN. It has the advantage of eliminating the need for intermediate aggregators to carry out decryption and encryption operations as well as storing sensitive data (ElGamal 1995, Ugus et al 2007). It also reduces the burden of BS.

4.3.3.1 Encryption

To send a message to another party whose public key is T_B, Encryption is done as follows:

Step 1: For a given plaintext $m \in [0, p-1]$, a random number $k \in [1, n-1]$ is selected, where *n* is the order of elliptic curve and *p* is the message length. $M = map(m)$.

Step 2: The aggregator node's public key is selected and shared secret key is generated $S_i = T_{S_i} \, mod \, p$ on the node.

Step 3: The message is encrypted by multiplying with $S_i \, mod \, p$, where S_i is the shared secret key of the i^{th} node.

Step 4: Send the public key and the enciphered message as a single block.

$$Cipher\ text\ (C) = E(m) = (r, s) = (kG, M + kY) \tag{4.4}$$

where r and s are random number and signature of the node respectively.

4.3.3.2 Decryption

Decryption is carried out by performing inverse of encryption.

$$Plain\ text\ (m) = D(C) = D(r, s) = -kr + S \quad (4.5)$$

$$m = r\ map(M) \quad (4.6)$$

where k is the private key and Y =kG is the public key generated using ECC parameters (ECDH Key exchange algorithm). The mapping function $map: m = mG$ with m $\in$ F_p, fulfills the required homomorphic property as given by Equation (4.7)

$$\begin{aligned} M_A + M_B + \ldots + M_n &= map\ (m_A + m_B + \ldots + m_n) \\ &= (m_A + m_B + \ldots + m_n)\ G \\ &= (m_A G + m_B G + \ldots + m_n G), \quad (4.7) \end{aligned}$$

where $m_A, m_B, \ldots\ m_n \epsilon\ F_p$.

The function map is necessary because addition over an Elliptic Curve is only possible with points on that curve and integers have to be mapped to corresponding points. This mapping function is not related to security, i.e. it neither increases nor decreases the security of the encryption scheme. Figure 4.7 shows the sequential diagram of the proposed DSAA scheme for WSN environment discussed before.

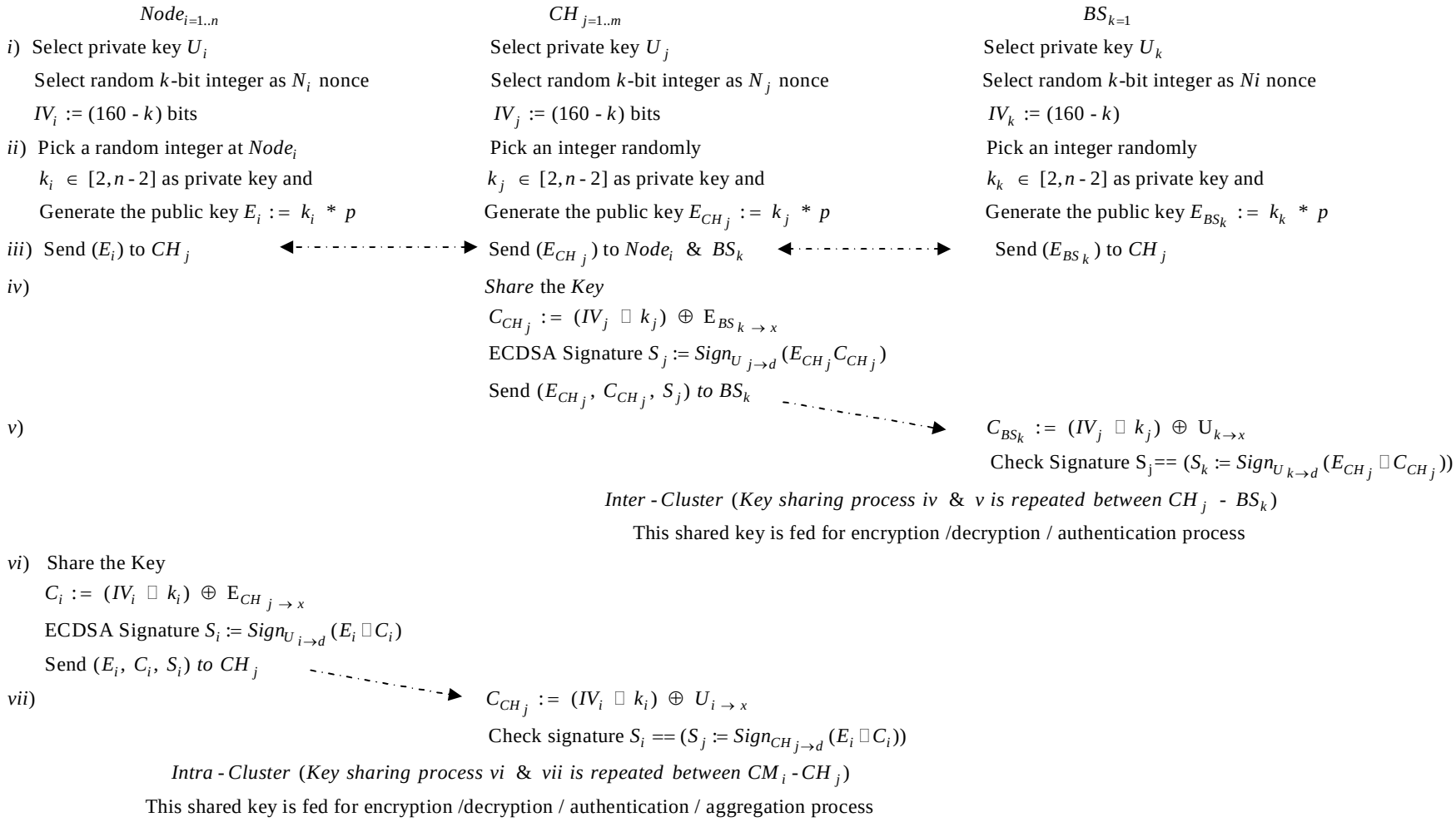

Figure 4.7 Sequential diagram of proposed DSAA framework

4.4 RESULT AND DISCUSSION

In the proposed research work, 50 nodes are randomly deployed and clusters are formulated.

4.4.1 Cluster Formation and Routing

The GUI of x and y axis is divided as 12.5mm each using grid formulation. Each node is associated with one of the cluster head based on the link quality using link estimation technique. Accordingly, each cluster has approximately ten cluster members and one cluster head. Here, node 0 is configured as a base station. This is simulated in Tiny Operating System Simulator (Levis et al, 2003, Eges-Lopez et al 2005) using TinyViz (TOSSIM-TinyViz). Figure 4.8 shows the simulation results of cluster formation in TinyViz with 50 nodes.

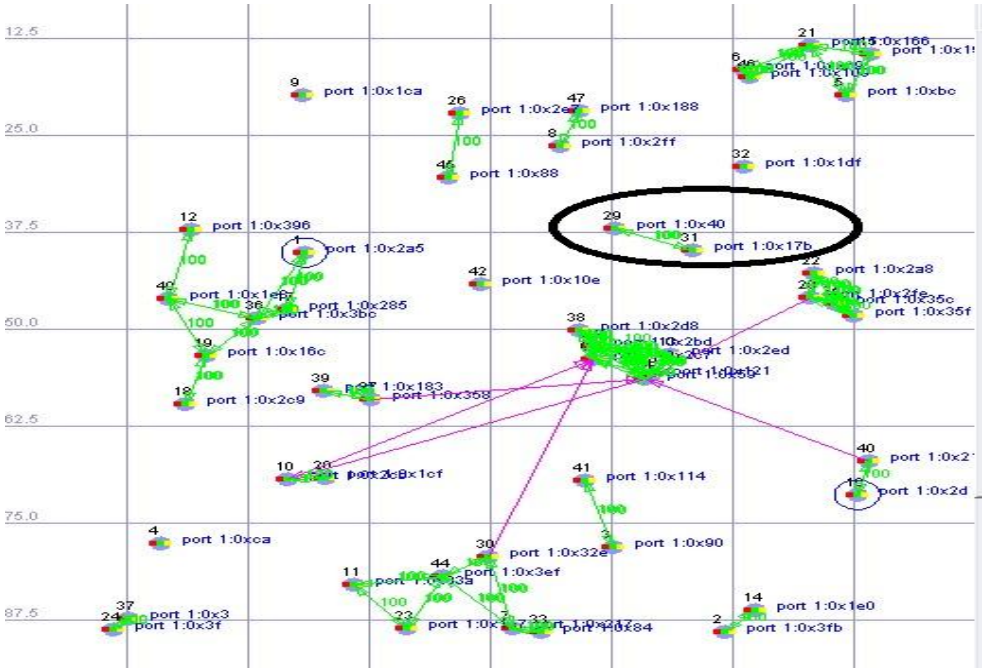

Figure 4.8 Simulation of cluster formation in TinyViz

The link in green color indicates that the link has the best link quality among other nodes, which are in between CH and CM. If the neighbors are identified, then the sensed values from the sensors are forwarded to the cluster heads. The cluster head forwards the sensed values to the base station. The pink color shows the data communication between the CH and the base station.

A packet to be sent to the base station consists of all header details along with the payload i.e., the sensed readings. For example, the reading of node 29 and 31 are 0x40 and 0x17B respectively circled as shown in Figure 4.8. In Figure 4.9 the packet contains payload (marked in red) along with its node id in hexadecimal value (marked in blue).

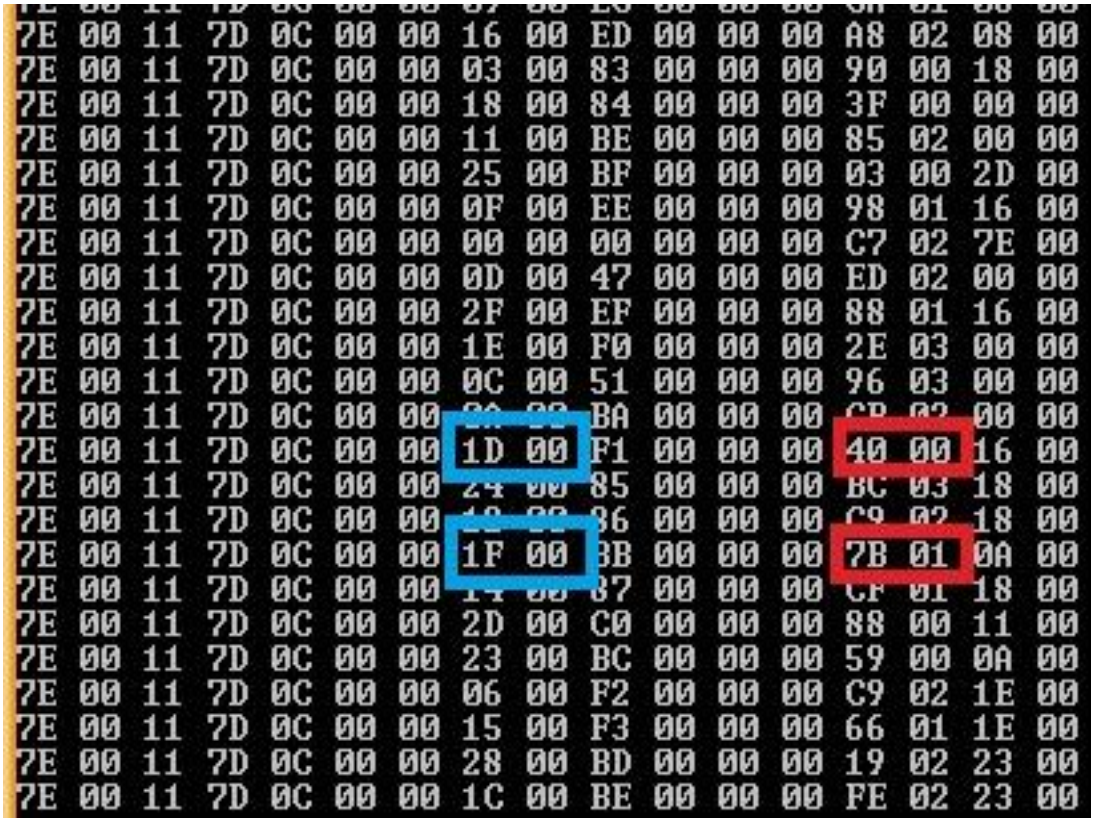

Figure 4.9 Listener port

Figure 4.10 shows the updated routing table for the 38th node and it gets updated each time the timer fires. It shows the node address (addr), cluster head address (cHead), number of missed packets (misd), number of received packets (rcvd), last sequence number (lstS), number of hops to reach the node (hop), receiving estimation factor (rEst) and sending estimation factor (sEst). The routing table is updated for every 50 seconds.

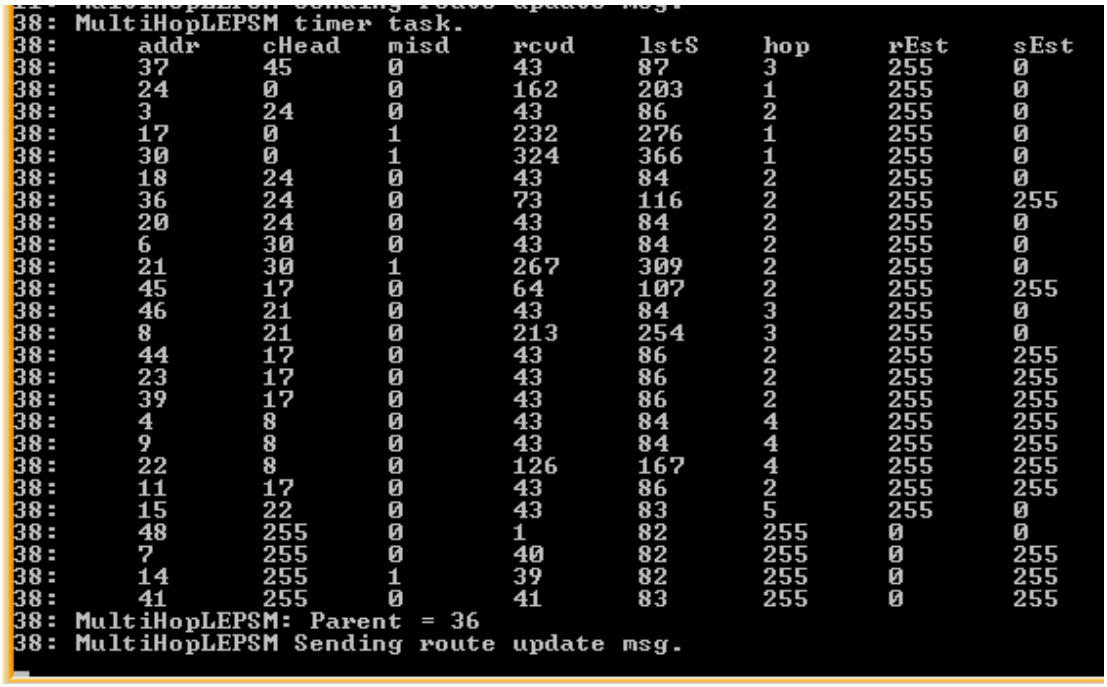

Figure 4.10 Routing table of 38th node

The forwarded packets are observed by enabling the sent radio packets option in the simulator. The destination address in the packets is changed into the corresponding node's destination address in the broadcast message, which is originally 0xffff at the time of broadcast message generation. The type of data is denoted by the message type id in the packet format. Figure 4.10 shows the packet transmission at every time interval. This shows the raw data that is received using SerialForwarder.java available in java package (TinyViz). The

result is in the form of big-endian format and is shown in Figure 4.11 in the user readable format.

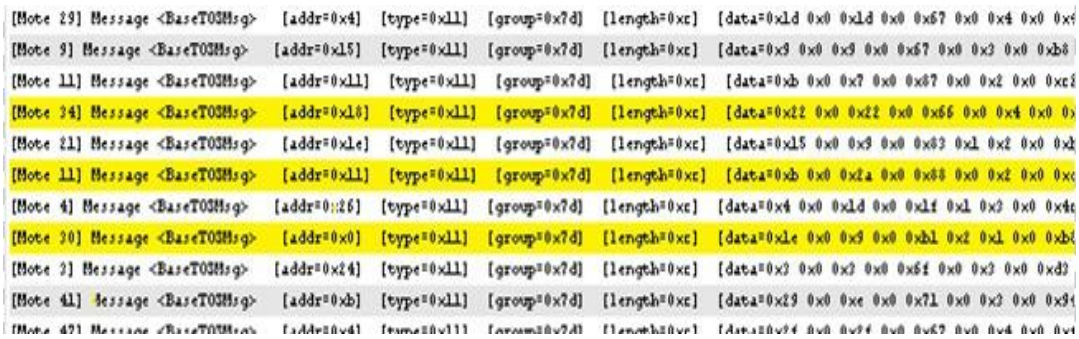

Figure 4.11 Packet transmission

4.4.2 Shared Secret Key Generation using ECDH

A shared secret key is generated between each node and cluster head using ECC and then it is exchanged by ECDH algorithm. Figure 4.12 shows the shared secret key generation between node 17 and 38 with the key size of 160 bits. Initially node 17 selects its private key (K_i) and produces the public key ($T_i(x, y)$) using scalar multiplication process. This generated public key is transmitted to node id 38 to obtain the shared secret key ($S_i(x, y)$) using ECDH algorithm as discussed in section 4.2

```
17: Generating shared secret,
17: with my private key:
02 b6 8e f3 00 9a db 50 4f 77 f6 f0 ba 9d 53 d7 f2 13 0c e4 d9
17: and  public key:
x:
02 59 fd 51 ce 64 99 3c 3d 05 27 0c 08 87 52 64 49 59 67 00 69
y:
07 69 3d 75 f0 15 96 b1 64 05 20 ff 76 24 9b 8e 8b 12 c6 af 1b
17: OUR SHARED SECRET:
x:
05 a1 06 39 49 62 fe a7 55 31 ed 4c 7f 3c 34 f3 d8 ae 52 02 8a
y:
00 ca cb 2b 68 f2 55 09 85 d3 5b c6 7d 1b 09 6c a8 80 85 80 39
```

Figure 4.12 ECDH shared secret key generation

4.4.3 Encryption and Decryption of Data

Encrypted messages are transmitted through the cluster heads to the base station. Figure 4.13 shows the simulated output after performing the encryption on the sender node. The encrypted packets are transmitted in the form of a specified format as shown in Table 4.3. In Figure 4.13, for $Node_0$ ff ff denotes a broadcast address, 0a denotes an active message ID, 7d denotes a group ID, 1a denotes message length, and remaining (128 bytes) denotes the size of payload. The destination address may be a broadcast address or a unicast address. If the address is ff, then it denotes the destination address is a broadcast address and a packet is intended for set of users. If it is a unicast address, then the destination address of the packet consists of identity of a particular node. Active message field in the packet could be 05 or 0a. Here, active message '05' indicates that it belongs to the beacon message of the network, and '0a' indicates a data packet of the message. The data length field indicates the length of the payload transmitted.

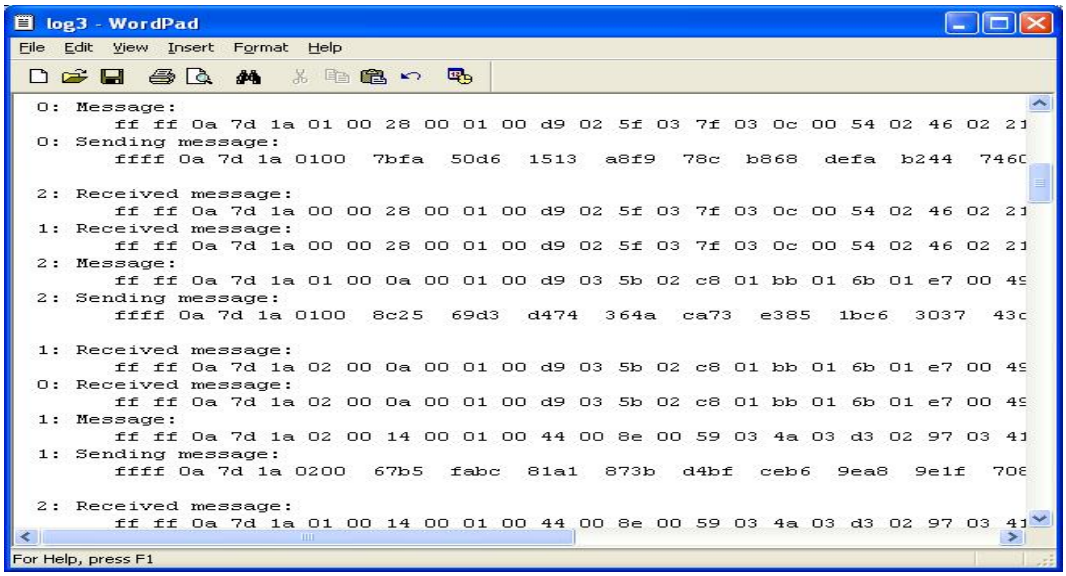

Figure 4.13 Simulated trace file of the sensor data after encryption process

The key generated using ECDH is passed to MD5 for generating the message digest of 128 bit key to be used in AES for encryption of the payload. Figure 4.14 shows the generation of message digest, encrypted data that is being forwarded to the base station, decryption of data in the base station. The sensed information at node 05 is shown in Figure 4.14. The time taken for both encryption and decryption of the sensed information are observed as 94705 ns and 79340 ns respectively.

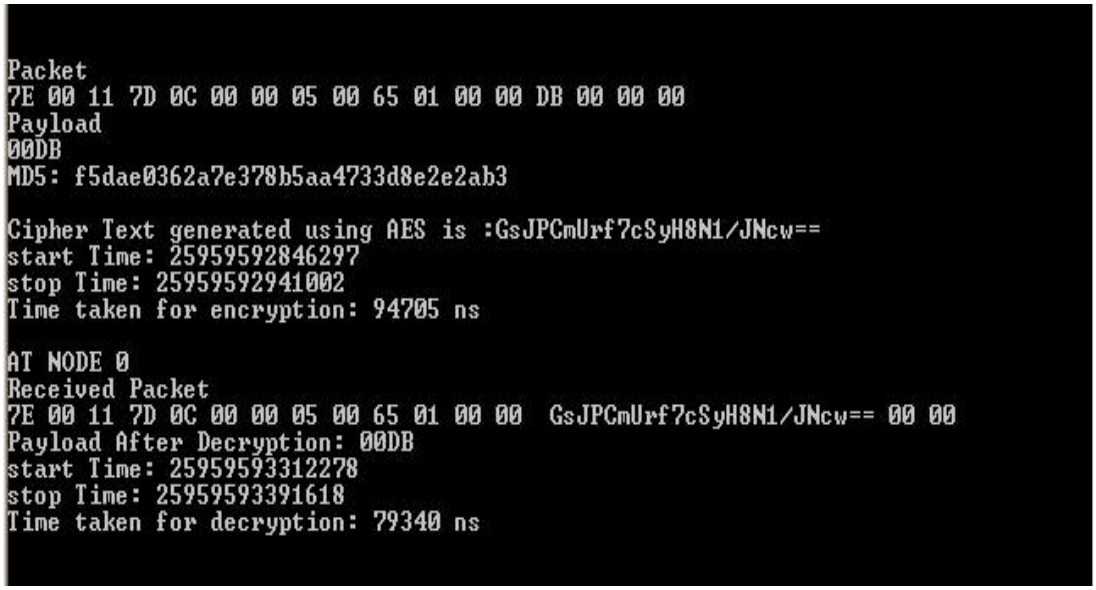

Figure 4.14 Encryption and decryption of data

4.4.4 Security Analysis of the Proposed Work

The performance of the proposed scheme is tested under a message modification attack to ensure the message integrity. The received message at the receiver is applied to the message modification attack for the detection of the security. In this proposed work, a received message at node 38 is validated with its hash value in order to test the security of the proposed scheme.

4.4.4.1 Storage requirement

In this research work, a combination of symmetric and asymmetric key is used. Though pairwise keys are maintained between the cluster head and nodes, the number of keys required for the proposed scheme is lesser than the existing symmetric key agreement scheme (Karlof et al 2004). Table 4.4 gives the storage requirement of various key agreement schemes, where n represents the number of nodes in the network environment.

Table 4.4 Comparison of key agreement scheme based on storage requirement

Algorithm	Number of keys to be stored
Symmetric Scheme	n(n-1)/2
Asymmetric Scheme	2n
Hybrid (proposed) Scheme	2n

4.4.4.2 Execution timing analysis

The execution time of various security algorithms in the proposed DSAA is measured for a single node. The computation time for key generation, key sharing and encryption are given in Table 4.5 for the proposed DSAA. The overall time taken to process the key generation is nearly 6 seconds. The key generation process takes place during the cluster formation. The packets are encrypted using AES-CBC mode and it takes ~17ms. In addition, to authenticate the message SCSHA takes ~338ns.

Table 4.5 Execution time for various security algorithms used in proposed work

Steps involved in DSAA	Time Taken (in Seconds)
ECDH Private key Initialization	0.9271658
public key at $Node_A$	0.90565334
public key2 at $Node_B$	0.9063373
Key_Agree at $Node_A$	1.0602349
Key_Agree and $Node_B$	2.1059017
Total Time Taken for ECDH	**5.90529304**
Time Taken for AES-CBC	**0.000174045**
Time Taken for SCSHA	**0.00000338**

4.4.4.3 Energy consumption

The energy consumed by the algorithms used in the proposed methods is measured. Table 4.6 gives the values of the energy consumed by the various security algorithms in the proposed DSAA.

Table 4.6 Energy consumption for various security algorithms in the proposed DSAA

Algorithm	Energy consumption
ECC Key Generation	0.8160 J
ECDH Key Exchange	0.0744 J
AES-CBC Encryption	0.384 mJ

With the proposed work, it is observed that the receiver complexity is increased in decrypting the information. Hence, the attackers need more processing time than the conventional. It concludes that the proposed work with

twisted Edward curve usage in TinyECC gives high performance than the existing centralized algorithm.

4.5 ANALYSIS ON PROPOSED SCHEDULER BASED COMBINED SECURED HASH ALGORITHM (SCSHA) FOR WSN AUTHENTICATION

SCSHA uses two cross-connected parallel-operated SHA modules, which executes on odd and even pairs of message. In order to increase the complexity of the hash function, message permutation is used. The use of 160-bit secret key increases the permutation complexity to a total complexity of $2^8! \times 2^{16}$. However, in order to store the key, it requires 16*8 bit memory, which is not huge for lightweight applications. The choice of using the permutation unit depends upon the level of security needed for the application.

4.5.1 Preimage Attacks

A preimage attack on a cryptographic hash is an attempt to find a message that has a specific hash value. There are two types of preimage attacks. First preimage attack, for a given hash h, it finds a message m such that Hash(m) = *h*. For a given fixed message m1, it finds a different message m2 such that hash(m2) = hash(m1) which is called as second preimage attack. The proposed SCSHA has the complexity of 2^{512}, for a n-bit ideal hash function and finding a first preimage or a second preimage has complexity of 2^n.

4.5.2 Birthday Attack

The “birthday attack” can find a pair of messages having the same hash value with a work factor of approximately $2^{\frac{n}{2}}$. Therefore, the complexity of

birthday attack for this design is 2^{512} that is high compared to basic SHA-1 algorithm. The combined SHA-1 model proposed in Emam & Emami (2007) contains two independent SHA modules operating alternatively on a message. Since this design uses two cross-connected SHA-1 modules operating simultaneously on two consecutive words of a message the hash function complexity is increased. The complexity of the hash function can be further increased by using the permutation unit suggested by Emam & Emami (2007). If a 160-bit secret key is used in permutation then computational complexity is increased, i.e. $2^{8!} \times 2^{512}$. However, in order to store the key, the proposed SCSHA algorithm requires 512*8 bit memory that is not huge for lightweight applications. So depending upon the application and the level of security, the permutation unit may be used or ignored. Table 4.7 shows the comparison of SHA with the proposed SCSHA.

Table 4.7 Comparisons of SHA, existing combined SHA and proposed SCSHA

Features	Classical SHA	Proposed SCSHA	Existing Combined SHA-512	Proposed SCSHA-512
Clock Cycles	80	40	80	40
Digest Length	160	160	512	1024
Time delay (ns)	677.719	338.85	814.629	314.931
Pre-image attack	2^{160}	2^{160}	2^{1024}	2^{1024}
Birthday attack	2^{80}	2^{160}	2^{256}	2^{512}

The net time delay to compute the message digest is reduced by 2 times due to improved parallelism in the proposed work. Security analysis shows that the design is more secure due to increased computational complexity. The

SCSHA is integrated with other public key algorithms to generate digital signature that is highly secured than the existing Combined SHA-512.

4.6 SECURITY ANALYSIS ON DSAA

The proposed DSAA scheme is simulated in AVISPA tool using HLSPL and validated under OFMC. A scenario in which a node, cluster head and base station are communicating with each other is considered as shown in Figure 4.15. A random value is generated as Initial Vector (IV) to ensure the data is afresh. This IV is used as a part of private key (Ka) and produces the public key. The pre-shared key is fused in all the nodes to prevent the participation of unauthorized nodes in the network. The key exchange between node to cluster head, cluster head to base station is performed using TinyECC-TE in DSAA as described in section 3.4.

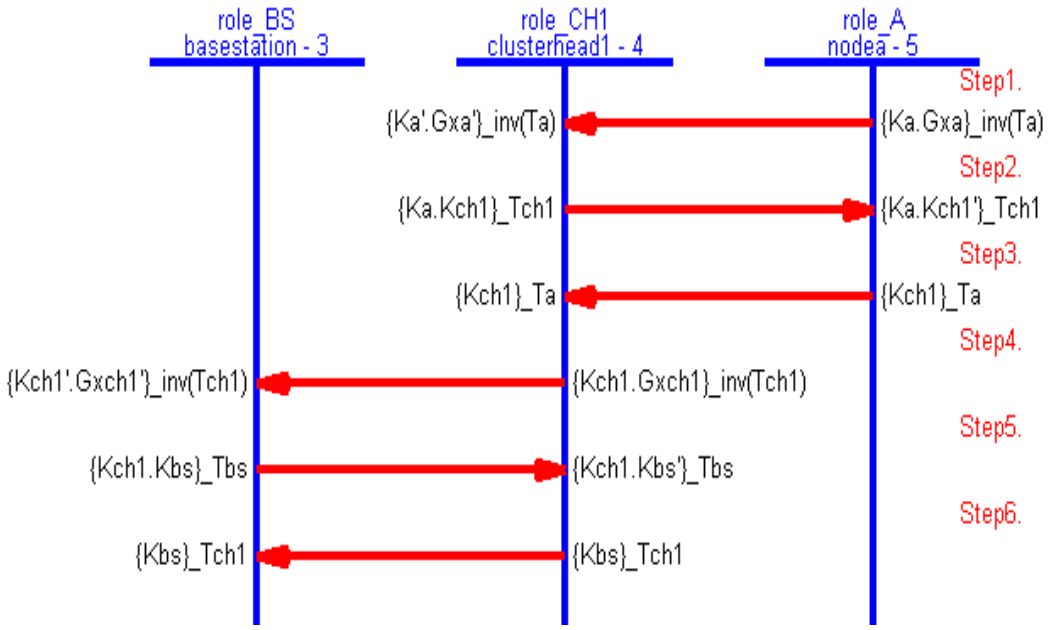

Figure 4.15 Proposed DSAA key exchange mechanism using TinyECC-TE between node to CH and CH to BS

4.6.1 MITM Attack

The proposed DSAA is validated by introducing an intruder in the network as shown in Figure 4.16. Assume that the intruder (I) knows user properties such as node ID (nodea), cluster head ID, base station ID, public key of the node (ta), public key of the cluster head (tch) and public key of the base station (tbs). The intruder knowledge is represented in the following format for MITM attack simulation scenario.

intruder_knowledge = {nodea, clusterhead, basestation, tch, tbs, ta}

The intruder scenario is similar to that of MITM is presented in section 3.4.1.

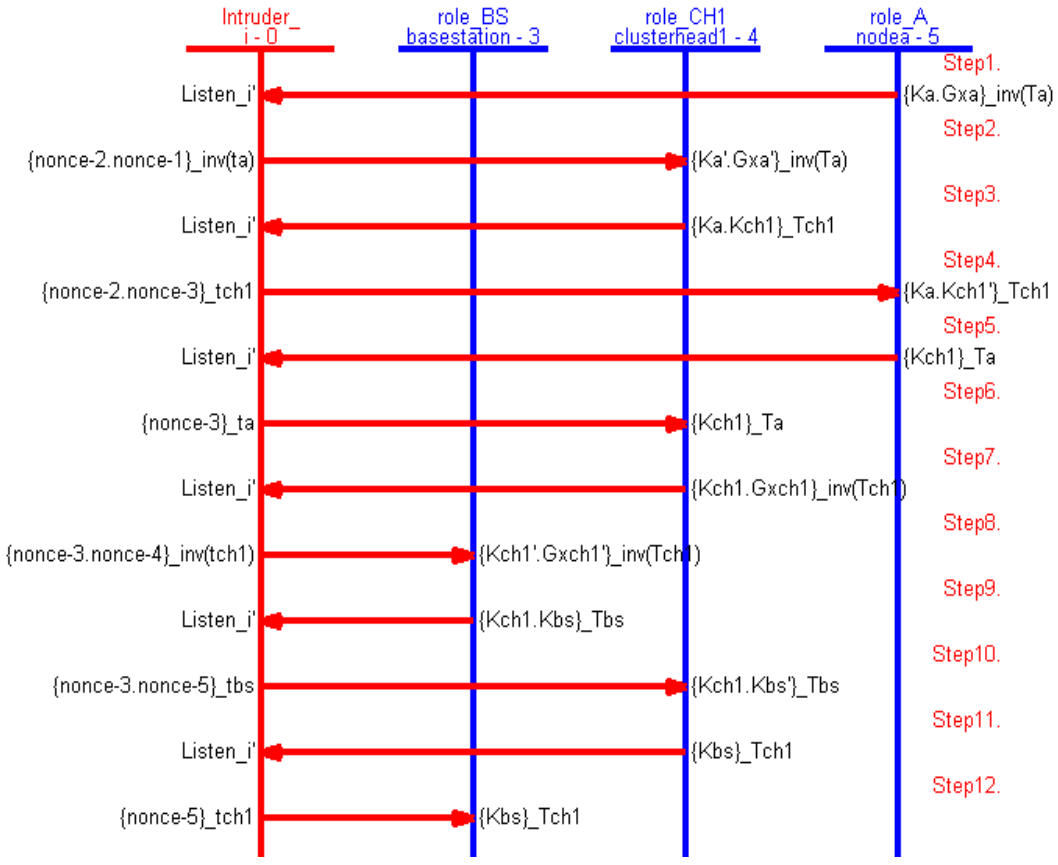

Figure 4.16 Simulation of proposed DSAA under MITM attack

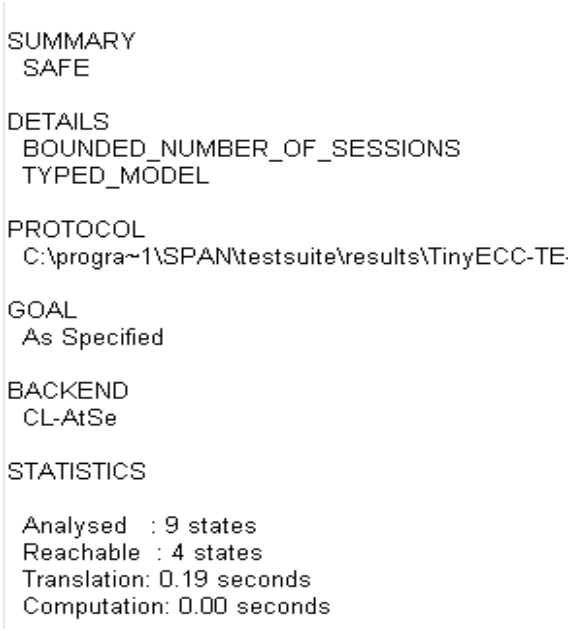

Figure 4.17 Evaluation of the proposed DSAA under MITM attack

Figure 4.17 presents the evaluation result of proposed DSAA under MITM attack and it can be seen that the network is in safe state. The protocol is validated on OFMC (CL-AtSe) model. The security analysis of the proposed DSAA scheme is carried out and the results are given in Table 4.8. In this analysis the number of nodes with a cluster head is increased upto 5. The parameters such as search time, visited nodes, depth are tabulated for the proposed DSAA scheme. Also the states analysed, reachable state by an intruder along with translation time, computation time are presented. It is found that the reachable state is always lesser than analysed state ensuring that the proposed scheme is in safe state. From the analysis the normal state and the intruder state of a node can be identified using equation 4.8.

$$C_p(\mathrm{N}) = \begin{cases} \leq \mathrm{H_{Cnt}}(\mathrm{V_n}), & \text{Normal state} \\ > \mathrm{e}^{\mathrm{H_{Cnt}}(\mathrm{V_n})}, & \text{Intruder state} \end{cases} \quad (4.8)$$

where Cp(N) - Communication Properties of a Network, H_{Cnt} - Total handshake count and V_n - Visited Node. From the equation 4.8, if $C_p(\mathrm{N})$ is greater than $\mathrm{e}^{\mathrm{H_{Cnt}}(\mathrm{V_n})}$ then the network is said to be in compromised state.

Table 4.8 Security Analysis under MITM attack

No. of Nodes	DSAA Scheme			Intruder (I)			
	Search Time (sec)	Visited Nodes	Depth (piles)	Analyzed (states)	Reachable (states)	Translation Time (Seconds)	Computation Time (Seconds)
1	0.01	6	5	7	4	0	0
2	0.06	20	10	37	22	0.03	0
3	0.12	55	15	265	176	0.13	0
4	0.32	108	15	2956	2118	0.22	0.16
5	0.75	208	15	46092	34452	0.56	2.73

4.6.2 Node Replication Attack

The proposed DSAA scheme is designed to detect clone node or node replication attack. In node replication, intruder can perform attack by replicating any node characteristics in the network as shown in Table 4.9. For example, assume that an intruder knows user properties for node (nodea), cluster head (clusterhead), base station (basestation). Public key of the node (ta), public key of the cluster head (tch), public key of the base station (tbs) and private key of node A (inv(ta)). Attacker replicates the malicious node A' with same property of node A. Here node A and A' are replicated and available in the network. This leads to a complex scenario where the attacker confuses the communication between the legitimate node A with CH or BS. The scenario of attack simulation with attacker knowledge is as follows:

intruder_knowledge = {nodea, clusterhead, basestation, tch, tbs, ta, inv(ta)}

Table 4.9 Security Analysis under node replication attack

No. of nodes	Node Replication Attack											
	Node A'				CH				BS			
	A	R	T	CT	A	R	T	CT	A	R	T	CT
1	0	0	0	0	1	1	0.01	0	3	2	0	0
2	47	20	0.01	0	4	4	0.01	0	0	0	0.01	0
3	212	90	0.05	0	4	4	0.05	0	0	0	0.08	0
4	1137	481	0.08	0.01	4	4	0.09	0	0	0	0.09	0
5	7190	3039	0.14	0.08	4	4	0.13	0	0	0	0.13	0
	Node B'				CH				BS			
	A	R	T	CT	A	R	T	CT	A	R	T	CT
1	-	-	-	-	1	1	0.01	0	3	2	0	0
2	50	25	0.01	0	4	4	0.01	0	0	0	0.01	0
3	265	124	0.05	0	4	4	0.05	0	0	0	0.08	0
4	1622	729	0.09	0.02	4	4	0.09	0	0	0	0.09	0
5	11507	5037	0.17	0.16	4	4	0.13	0	0	0	0.13	0

Table 4.9 illustrates the time taken for node replication. The analysis metrics namely analysed state (A), reachable state (R), translation time (T) and Computation time (CT) are used to measure the security of a node. From Table 4.9, it can be seen that the computation time and translation time are increased as the number of node is increased in the network. It also shows that if node A or B is replicated in a network, then the translation time for the node A or B is increased. This is due to the fact that the intruder node has to give more acknowledgements to act as a legitimate node. Hence it is possible to conclude that an intruder is present the network.

4.6.3 Impersonation Attack

An intruder impersonate as any of the node present in the network. On intercepting the transmission between two nodes, the intruder may know the public key of a node, cluster head and base station. From these known public key an intruder cannot gain any information, as security is based on hashing mechanism used. Finding the inverse is also very difficult for a known public key. This involves Elliptic Curve Discrete Logarithmic Problem (ECDLP); hence impersonation attack is difficult in DSAA.

4.6.4 Replay Attack

Replay attack is not possible in DSAA as each and every transaction uses the Initial Vector (IV) to ensure the freshness of a message. Hence an intruder needs to find the Initial Vector from the encrypted message, which is very difficult without finding the private key of a node.

4.7 SUMMARY

In this chapter, Key is generated using Twisted Edward based ECDH, confidentiality is ensured using AES over CBC mode and the authentication is achieved by proposed SCSHA. It concludes that the proposed work with Twisted Edward curve in TinyECC gives high performance in terms of space requirement, higher security level, lesser power consumption and lesser computation time than the existing TinyECC. The sensed data is aggregated using an Encrypted Data Aggregation scheme to reduce the overhead on base station. From the observation it found that SCSHA is more resilient for Birthday Paradox attack, whereas resilient remains the same for pre-image attack.