EFFICIENT KEY GENERATION FOR DYNAMIC BLOM'S SCHEME

By

DIVYA HARIKA NAGABHYRAVA

Bachelor of Technology in Computer Science

Jawaharlal Nehru Technological University

Hyderabad, AP, India

2012

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 2014

EFFICIENT KEY GENERATION FOR DYNAMIC BLOM'S SCHEME

Thesis Approved:

Dr. Subhash Kak

Thesis Adviser

Dr. Christopher Crick

Dr. David Cline

Name: DIVYA HARIKA NAGABHYRAVA

Date of Degree: DECEMBER 2014

Title of Study: EFFICIENT KEY GENERATION FOR DYNAMIC BLOM’S SCHEME

Major Field: COMPUTER SCIENCE

Abstract:

The need for totally secure communication between the nodes in a network has led to many key distribution schemes. Blom’s scheme is a prominent key exchange protocol used for sensor and wireless networks, but its shortcomings include large computation overhead and memory cost. The main goal of our work is to find ways to overcome the shortcomings of the existing Blom Scheme and make it more efficient. In this thesis, we propose an efficient key management scheme by generating new public and private keys. We also focus on making Blom’s scheme, dynamic by randomly changing the secret key that is generated by the base station and also using mesh array for matrix multiplication for reducing the computation overhead.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

1.1 Sensor Networks and Key Management

Sensor networks are dense wireless networks of small low cost sensors, which collect and disseminate environmental data. Recent improvements electronic and computer technologies, have paved the way for the proliferation of wireless sensor networks (WSN). These sensors are used to collect environmental information and they have been considered for various purposes including security monitoring, target tracking and research activities in hazardous environments. Each sensor node is battery powered and equipped with integrated sensors, data processing capabilities, and short-range radio communications. Examples of sensor network projects include Smart Dust [1] and WINS [2]. To achieve security in wireless sensor networks, it is important to be able to encrypt and authenticate messages sent among sensor nodes. Key management[17] [18] is a very important security issue in wireless sensor networks. Authentication and confidentiality protocols require an agreed key between the nodes and security of the communication depends on the cryptographic schemes employed. Different Key management schemes have been developed for authentication and confidentiality purposes. Symmetric algorithm base key management protocols are use in wireless sensor networks [3] because key management protocols based on public keys are inefficient due to resource limitations and lack of security in the sensors.

The main challenges of the sensor nodes are as follows:

- Once deployed, sensor networks have no human intervention. Hence the nodes themselves are responsible for reconfiguration in case of any changes.
- In order to make optimal use of energy, communication should be minimized as much as possible because the sensor nodes are not connected to any energy source. There is only a finite source of energy, which must be optimally used for processing and communication.
- It is required that a sensor network system be adaptable to changing connectivity as well as changing environmental stimuli.

The examples of possible applications of sensor networks include:

- Sensors are attached to taxi cabs in a large metropolitan area to study the traffic conditions and plan routes effectively.
- Sensor networks are used in military to detect, locate or track enemy movements.
- Sensor networks can increase alertness to potential terrorist threats.

1.2 Security Requirements in Sensor Network

Sensor networks have various requirements [4]:

A. **Integrity**: Integrity refers to unmodified data. Data integrity ensures that the data received is not altered or modified by any adversary and is same as the data sent by the sender node. Integrity of data is achieved by using various cryptographic methods.

B. **Authentication**: It is important as the receiving node must be ensured that the received is from a trusted source and not from the adversary.

C. **Confidentiality**: The data being sent should be received only by the intended receiver and should not be exposed to any other node. Hence data travels in a highly secure and encrypted mode.

D. **Availability, reliability and resiliency**: It ensures proper connectivity of the nodes and that the data and information is available to access at all times whenever and wherever required by the authorized nodes. This ensures protection from attacks such as denial of service attacks etc. It also ensures that the data packet is delivered to its destination.

E. **Data freshness**: Data freshness ensures that the data and information being delivered is fresh and recent. This is one of the most important security aspects as the adversary can send old message and thereby the recently detected data is not communicated.

1.3 Sensor Network Constraints

In order to perform better in terms of accuracy, reliability and security there are some limitations that need to be overcome .Some of the limitations are:

A. **Energy**: Sensor network consists of tiny sensors which run on batteries. These batteries have limited power supply. Hence there comes the need to conserve energy and transmit data efficiently without consuming much energy.

B. **Computation**: Due to energy constraints, sensors are also limited by low computational capacity. Hence algorithms which require large computations must be avoided to incorporate in sensor networks.

C. **Communication**: Sensors are linked via wireless connection and therefore, the bandwidth is often limited. This also limits the transmission of data and information.

1.4 Main Objective of our scheme

In this thesis, we propose an efficient key management scheme. The main contributions of this paper are as follows:

- Reduce the computation and memory overhead.
- Use of mesh array for matrix multiplication [5].
- Making the Blom's scheme dynamic in order to ensure secure communication between the nodes.

CHAPTER II

REVIEW OF LITERATURE

2.1 Related Work

Key management protocols can be based on either symmetric or asymmetric management functions. But due to the scarcity of the resources, protocols based on public keys are inefficient. Hence, symmetric algorithm based key management schemes are favored in WSNs [6]. Key management in sensor network includes:

1) **Key set up**: It is a process of generating keys by a central authority or by individual nodes.

2) **Key distribution**: It refers to the distribution of keys among the sensor nodes in case key is set up by a central authority.

3) **Key revocation**: It is the process of removing the key from nodes after the data has been transmitted or after a fixed interval of time.

Different approaches to address the problem of key management can be summarized as below:

- The ***online key management system*** proposed in [7] is one of the simple and easy ways of solving this key management problem. But the drawback with this method is that it carries high overhead.
- ***Pre-distribution of keys*** among the sensors can result in low cost key establishment in wireless sensor networks. But such schemes fail in handling security or efficiency problems.
- ***Probabilistic key pre-distribution*** method was proposed by Eschenauer and Gligor [8] to establish pair-wise keys between neighboring nodes. In this method simple shared-key discovery protocol is used for key distribution, revocation and node re-keying. That is each node is preloaded with a key subset from a global key pool in such a way that any two neighboring nodes can share at least one common key with a certain probability. But the drawback of this method is that the keys of normal nodes can be known when some nodes are compromised by adversaries.
- Chan, Perrig, and Song [9] proposed the ***q-composite random key pre-distribution scheme***, in which they modified E-G scheme by only increasing the number of keys that two random nodes share from at least 1 to at least q. Their scheme achieves good security under small scale attacks while increased vulnerability in large scale node to compromise attacks.
- Du et al [10] proposed the ***multiple-space key pre-distribution scheme*** where each key is replaced by a special key space and many more people came with certain modifications to the existing scheme. All the above methods mentioned so far make use of random node deployment model where each sensor node has

direct pair-wise keys shared with only portion of the neighbors, and depends on the multi hop or the path which is established in order for the nodes to communicate with long distance nodes.

- Our scheme builds a dynamic Blom's key distribution scheme to ensure secure communication between the nodes in the network. We also use the Mesh array for Matrix multiplication in order to make the scheme computation efficient.

2.2 Graph Matrices

Instead of using the Vandermonde matrix [11], the following matrices can be used as a pubic matrix in the Blom's scheme. The main advantage of the following matrices is that they reduce the cost of saving the columns in the memory of sensor because any node can easily generate these matrices of known size.

- **Hadamard matrix**: A non–binary Hadamard matrix is used as the public matrix to reduce the computation and memory overhead in Blom's scheme [13]. A Hadamard Matrix is a square matrix with values 1s and -1s. It reduces the complexity of calculating values of all the elements corresponding to the columns in Vandermonde matrix.

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

Figure-1: Hadamard Matrix

- **Adjacency Matrix**: To reduce the computation and memory overhead in Blom's scheme, instead of using Vandermonde matrix, an Adjacency Matrix is used as a public matrix [14]. An Adjacency Matrix is a square matrix with 1s and -0s, it reduces of complexity of calculating values for all the elements corresponding to the columns in Vandermonde matrix. This Adjacency matrix is formed in such a way that all nodes that are neighbors of a particular node are filled with 1s and remaining with q-1(since public matrix cannot contain 0s).

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Figure-2: Adjacency Matrix

- **Incidence Matrix:** The incidence matrix of a graph gives the (0, 1) matrix which has a row for each vertex and column for each edge, and (v,e) = 1. For the following graph the incidence and adjacency matrices are,

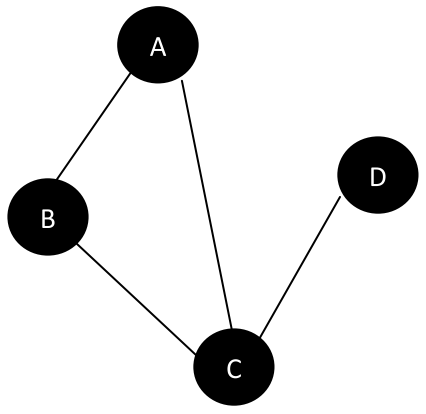


Figure-3: Example Graph

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure-4: Incidence Matrix for example graph

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure-5: Adjacency Matrix for example graph

2.3 Theorem for Generating Symmetric Matrix

In Blom's scheme, the Central authority or the base station randomly generates the secret key which is a symmetric matrix. Here, we obtain the secret symmetric matrix using the following steps:

FOR A SQUARE MATRIX OF EVEN ORDER:

Step-1: Consider a square matrix,

$$X = \begin{bmatrix} a11 & a12 & a13 & a14 \\ a21 & a22 & a23 & a24 \\ a31 & a32 & a33 & a34 \\ a41 & a42 & a43 & a44 \end{bmatrix}$$

Step-2: Now, we take transpose of the matrix X.

$$X^T = \begin{bmatrix} a11 & a21 & a31 & a41 \\ a12 & a22 & a32 & a42 \\ a13 & a23 & a33 & a43 \\ a14 & a24 & a34 & a44 \end{bmatrix}$$

Step-3: Multiply X and X^T.

XX^T=

$$\begin{bmatrix} a11a11 + a12a12 + a13a13 + a14a14 & a11a21 + a12a22 + a13a23 + a14a24 & a11a31 + a12a32 + a13a33 + a14a34 & a11a41 + a12a42 + a13a43 + a14a44 \\ a21a11 + a22a12 + a23a13 + a24a14 & a21a21 + a22a22 + a23a23 + a24a24 & a21a31 + a22a32 + a23a33 + a24a34 & a21a41 + a22a42 + a23a43 + a24a44 \\ a31a11 + a32a12 + a33a13 + a34a14 & a31a21 + a32a22 + a33a23 + a34a24 & a31a31 + a32a32 + a33a33 + a34a34 & a31a41 + a32a42 + a33a43 + a34a44 \\ a41a11 + a42a12 + a43a13 + a34a14 & a41a21 + a42a22 + a43a23 + a44a24 & a41a31 + a42a32 + a43a33 + a44a34 & a41a41 + a42a42 + a43a43 + a44a44 \end{bmatrix}$$

This is a symmetric matrix which can be used as a Secret matrix by the Central Authority. Similarly, the secret matrix using non-square matrix and square matrix of odd order can be obtained.

FOR DYNAMICALLY UPDATING THE SECRET MATRIX:

In the above symmetric matrix XX^T, let us assume the following:

a= $a11a11 + a12a12 + a13a13 + a14a14$

b= $a11a21 + a12a22 + a13a23 + a14a24$

c= $a11a31 + a12a32 + a13a33 + a14a34$

d= $a11a41 + a12a42 + a13a43 + a14a44$

e= $a21a21 + a22a22 + a23a23 + a24a24$

f= $a21a31 + a22a32 + a23a33 + a24a34$

g= $a21a41 + a22a42 + a23a43 + a24a44$

h= $a31a31 + a32a32 + a33a33 + a34a34$

i= $a31a41 + a32a42 + a33a43 + a34a44$

j= $a41a41 + a42a42 + a43a43 + a44a44$

So, the matrix XX^T can be written as,

$$\begin{bmatrix} a & b & c & d \\ b & e & f & g \\ c & f & h & i \\ d & g & i & j \end{bmatrix}$$

Now, the secret symmetric matrix can be updated as follows:

$$\begin{bmatrix} a & b & c & d \\ b & e & f & g \\ c & f & h & i \\ d & g & i & j \end{bmatrix} \begin{bmatrix} d & g & i & j \\ c & f & h & i \\ b & e & f & g \\ a & b & c & d \end{bmatrix} \text{ or } \begin{bmatrix} a & b & c & d \\ b & e & f & g \\ c & f & h & i \\ d & g & i & j \end{bmatrix} + \begin{bmatrix} a' & b' & c' & d' \\ b' & e' & f' & g' \\ c' & f' & h' & i' \\ d' & g' & i' & j' \end{bmatrix}$$

Which is again a symmetric matrix that can be used as a secret matrix.

2.4 Mesh Array for matrix multiplication

The mesh array of matrix multiplication [5][15] was introduced by Kak in 1988 to speed up the computation for multiplying two n×n matrices using distributed computing nodes. In contrast to the standard array that requires 3n-2 steps to complete its computation, the mesh array requires only 2n-1 steps. The speedup of the mesh array is a consequence of the fact that no zeros are padded in its inputs.

In the modified Blom's scheme, the mesh array for matrix multiplication can be used by the central authority or the base station for generating the private matrix and by the user pair in order to obtain the shared key by multiplying the public and private key.

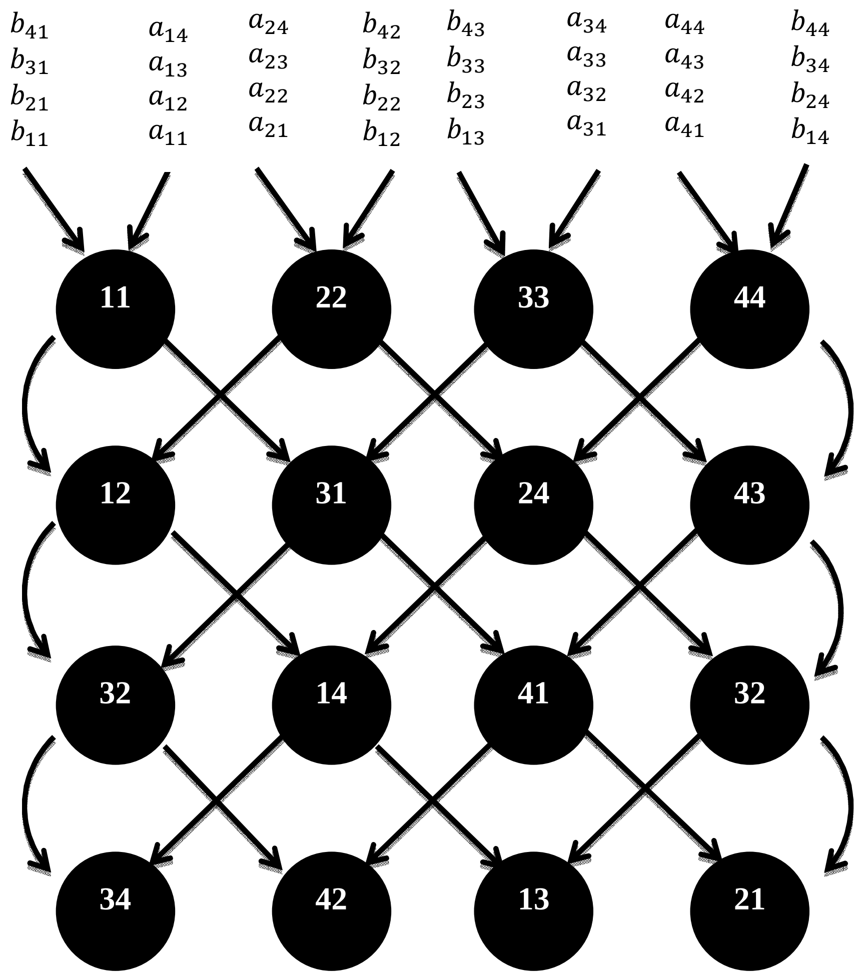
Consider two 4*4 matrices:

$$A=\begin{bmatrix} a11 & a12 & a13 & a14 \\ a21 & a22 & a23 & a24 \\ a31 & a32 & a33 & a34 \\ a41 & a42 & a43 & a44 \end{bmatrix}$$

And

$$B=\begin{bmatrix} b11 & b12 & b13 & b14 \\ b21 & b22 & b23 & b24 \\ b31 & b32 & b33 & b34 \\ b41 & b42 & b43 & b44 \end{bmatrix}$$

The mesh architecture for multiplying the above two matrices (C=AB) is given by:



b_{41} b_{31} b_{21} b_{11}
a_{14} a_{13} a_{12} a_{11}
a_{24} a_{23} a_{22} a_{21}
b_{42} b_{32} b_{22} b_{12}
b_{43} b_{33} b_{23} b_{13}
a_{34} a_{33} a_{32} a_{31}
a_{44} a_{43} a_{42} a_{41}
b_{44} b_{34} b_{24} b_{14}
11
22
33
44
12
31
24
43
32
14
41
32
34
42
13
21

Figure-6: Mesh Array for Matrix Multiplication

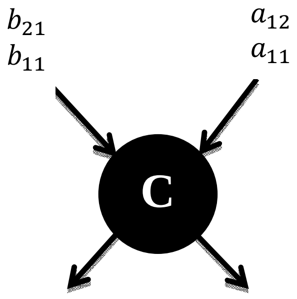


Figure-7: Computing node that generates c= $a_{11}b_{11}+a_{12}b_{21}$

Properties of Mesh Array for Matrix Multiplication:

- For a matrix with odd order, the rows 2 to (n+1)/2 are mirror image to rows (n+3)/2 to n.

  Example: For a 5*5 matrix:

$$\begin{bmatrix} 11 & 22 & 33 & 44 & 55 \\ 12 & 31 & 24 & 53 & 45 \\ 32 & 14 & 51 & 25 & 43 \\ 34 & 52 & 15 & 41 & 23 \\ 54 & 35 & 42 & 13 & 21 \end{bmatrix}$$

- For a matrix with even order, the rows 2 to n/2 are mirror reversed image to rows n/2+2 to n, and the middle row (n/2+1) has self-symmetry.

  Example: For a 6*6 matrix:

$$\begin{bmatrix} 11 & 22 & 33 & 44 & 55 & 66 \\ 12 & 31 & 24 & 53 & 46 & 65 \\ 32 & 14 & 51 & 26 & 63 & 45 \\ 34 & 52 & 16 & 61 & 25 & 43 \\ 54 & 36 & 15 & 15 & 41 & 23 \\ 56 & 64 & 42 & 42 & 13 & 21 \end{bmatrix}$$

2.4 Handshake Protocol:

Cryptography is the study of techniques for the secure communication between two parties even in the presence of the third party. The handshake is the process which is used for the process of authentication. It ensures that a secure channel is established between the two communicating parties before the data transfer takes place. It verifies that information transmitted during the session is not being monitored or diverted to a malicious third party. The handshake process can further be explained by the following figure:

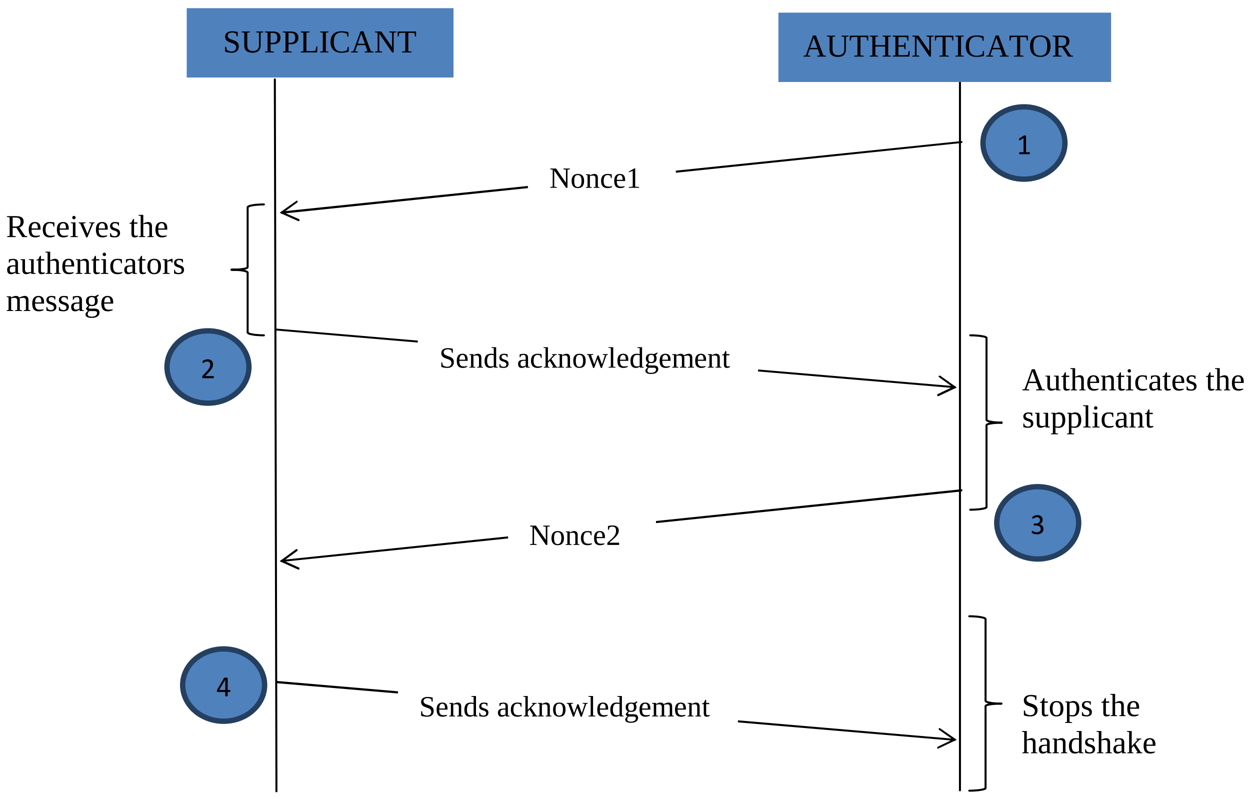


Figure:8– The Handshake process

- The handshake process is started by the authenticationg party. In step-1, the authenticator sends a nonce message to the supplicant.
- In step-2, on receiving the nonce from the authenticator the supplicant acknowledges to the authenticator by sending a message.
- In step-3, the authenticator verifies the information and authenticates the supplicant and informs it to the supplicant by sending another nonce message say nonce2.
- In step-4, on receiving nonce2, the supplicant sends an acknowledgement message to authenticator.
- Finally, on receiving the acknowledgement the authenticator stops the handshake.

CHAPTER III

THE BLOM SCHEME

Blom’s scheme [12] is a symmetric threshold key exchange cryptography protocol. It allows any pair of users in the system to find a unique shared key for secure communication.

- In this scheme, a network with N users and a collusion of less than t+1 users cannot reveal the keys which are held by other users. Thus, the security of the network depends on the chosen value of t, which is called Blom’s secure parameter ($t \ll N$).
- Larger value of t leads to greater resilience but a very high t value increases the amount of memory required to store key information.
- *Generation of Public matrix:*

  Initially, a central authority or base station first constructs a $(t + 1)\times N$ matrix P over a finite field GF(q), where N is the size of the network and q is the prime number. P is known to all users and it can be constructed using Vandermonde matrix. It can be shown that any t+1 columns of P are linearly independent when n_i, $i=1, 2,\ldots N$ are all distinct.

$$P = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ n1 & n2 & n3 & \dots & n_N \\ n1^2 & n2^2 & n3^2 & \dots & {n_N}^2 \\ \dots\dots & \dots\dots & \dots\dots & \dots\dots & \dots\dots \\ {n_1}^t & {n_2}^t & {n_3}^t & \dots & {n_N}^t \end{bmatrix}$$

- *Generation of Secret Key (Private matrix):*

  The central authority or the base station selects a random (t + 1) × (t + 1) symmetric matrix S over GF (q), where S is secret and only known by the central authority.

  An N × (t + 1) matrix $A = (S.\ P)^T$ is computed which is needed for generating the shared key..

  K =A.P

  $= (S.\ P)^T.\ P$

  $= P^T.\ S^T.P$

  $= P^T.S.P$

  $= (A.P)^T$

  $= K^T$

- *Generation of shared key by the user pair:*

  User pair (i, j) will use Kij, the element in row i and column j in K, as the shared key. Because Kij is calculated by the i-th row of A and the j-th column of P, the central authority assigns the i-th row of A matrix and the i-th column of P matrix to each user i, for 1, 2….. N. Therefore, when user i and user j need to establish a shared key between them, they first exchange their columns of P, and then they can compute Kij and Kji, respectively, using their private rows of A. It has been proved in [10] that the above scheme is t-secure if any t + 1 columns of G are linearly independent.

- The t-secure parameter guarantees that no compromise of up to t nodes has any information about Kij or Kji.

$$\begin{array}{ccc} A=(S.P)^T & P & (S.P)^T\,P \\ \begin{bmatrix} & & & \\ i_1 & i_2 & \cdot\ \cdot & i_{t+1} \\ & & & \\ j_1 & j_2 & \cdot\ \cdot & j_{t+1} \\ & & & \end{bmatrix} & \begin{bmatrix} i_1 & j_1 \\ i_2 & j_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ i_{t+1} & j_{t+1} \end{bmatrix} & \begin{bmatrix} & K_{ij} \\ & \\ K_{ji} & \end{bmatrix} \end{array}$$

Figure-9 Generating keys in Blom’s Scheme

CHAPTER IV

THE PROPOSED SCHEME

In original Blom's scheme all the computations that are involved in generating the keys are based on a Vandermonde matrix which is a public matrix (P) and known to even the adversaries.The security of the network depends on the secure parameter "t". However, for large values of t, number of rows in the matrix increases and which in turn corresponds to a greater value in the columns because the column values increase in a geometric series.

The proposed method makes use of the original Blom's scheme .In the Blom's scheme for any two nodes to generate a shared key, each node should store its private key and the public key of the other node. Since every sensor node is provided with limited memory and energy, it will be difficult to store both the row and column in the sensor memory for a large network. So, the goal is to reduce the memory and computation overhead. To achieve this in Blom's scheme, instead of using Vandermonde matrix we propose the use any random matrix as the public matrix. The overall computation cost can also be reduced because the cost of generating a random matrix is less as compared to the cost of generating a Vandermonde matrix.Also, a random prime number q is chosen.

Now, similar to Blom's scheme the operation which are to be performed to generate the keys will depend on the prime number, i.e., the number which depends on the desired key length.

Following are the steps involved in calculating the shared key using the modified Blom's Scheme:

- *Generation of Public matrix:*

  Initially, any random matrix of order (t+1) * (t+1) is chosen as the public matrix.

- *Generation of Secret matrix (symmetric matrix):*

  The central authority calculates a $(t + 1) \times (t + 1)$ symmetric matrix S, where S is secret and only known by the central authority.

- *Generation of Private Matrix for obtaining the shared key:*

  A $N \times (t + 1)$ matrix $A = (S. P)^T$ is computed which is used for generating the shared key. The base station stores each row of the matrix A in the node memory with corresponding index. This is shown in Figure.

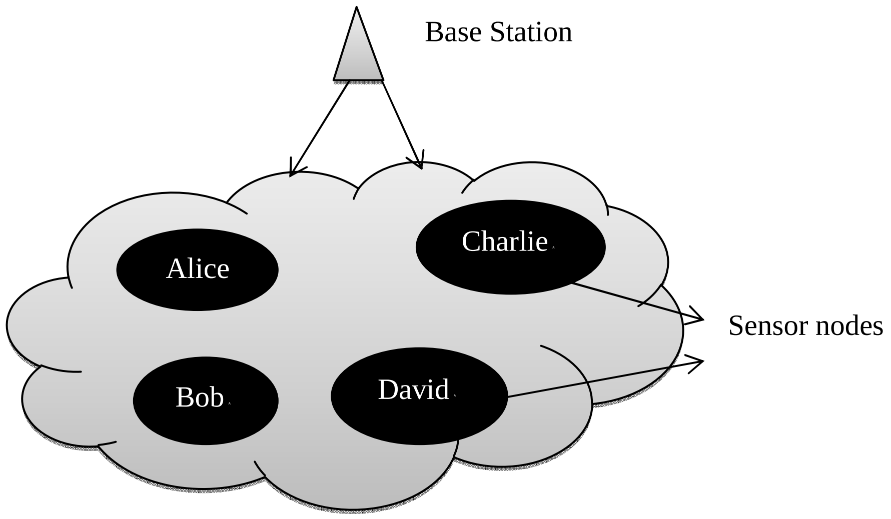


Figure-10: Sensor Nodes and Base Station in a Networks

- *Assigning Unique ID:* The Central Authority assigns unique id's to all the nodes in the network, which is public to the entire network before the public and private matrices are calculated. Once the node receives its unique id, it responds to the CA with an acknowledge type message or a reply. A question here arises, what if there is already an intruder in the network before assigning id's or how to distinguish a new node that enters a network as a trusted node or a malicious node. For this reason, we assume that every node is authenticated in some other network before it enters this network. The following handshake takes place before the node enters any network.

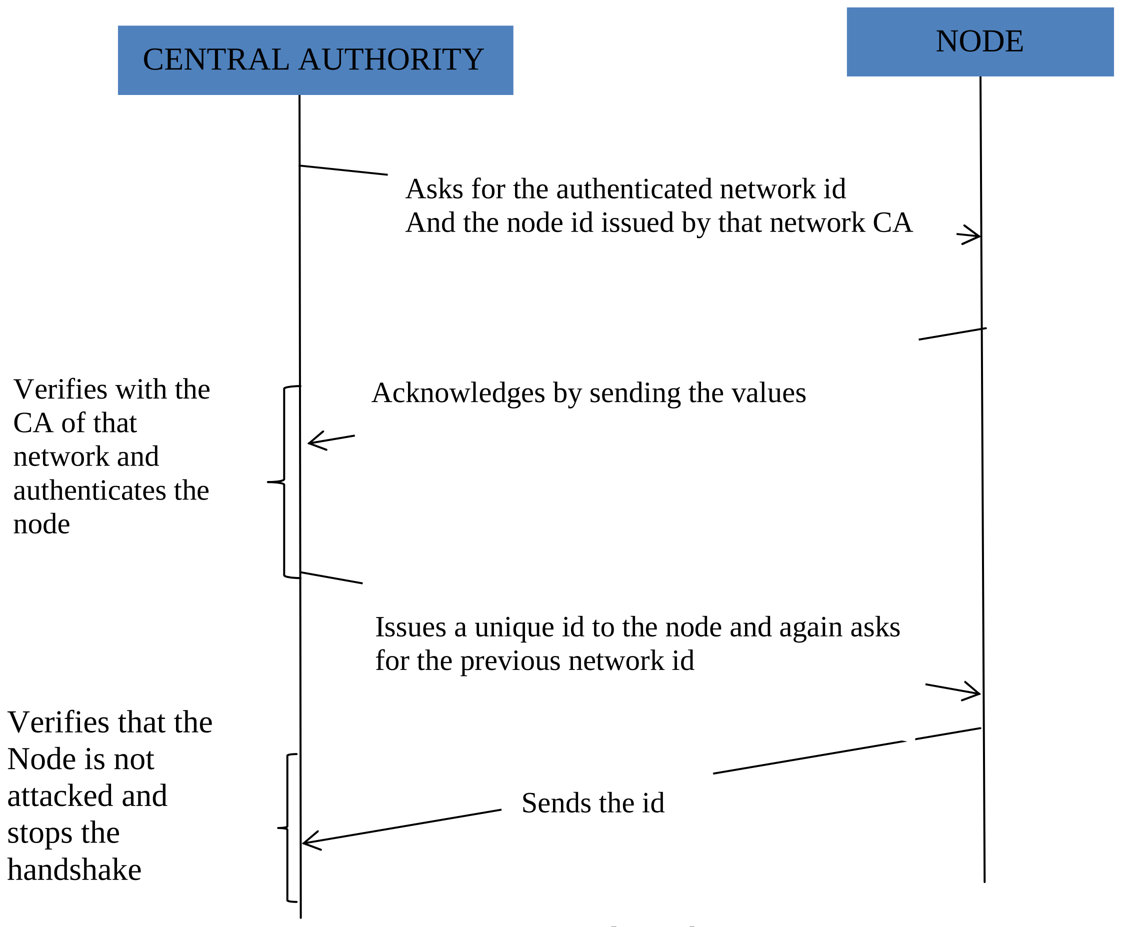

CENTRAL AUTHORITY
NODE
Asks for the authenticated network id
And the node id issued by that network CA
Verifies with the CA of that network and authenticates the node
Acknowledges by sending the values
Issues a unique id to the node and again asks for the previous network id
Verifies that the Node is not attacked and stops the handshake
Sends the id

Figure-11: New Node Authentication

Now, if suppose Alice and Bob want to communicate with eachother, they require the private key from the central authority to calculate the shared key. In order to obtain the key, the following handshake takes place.

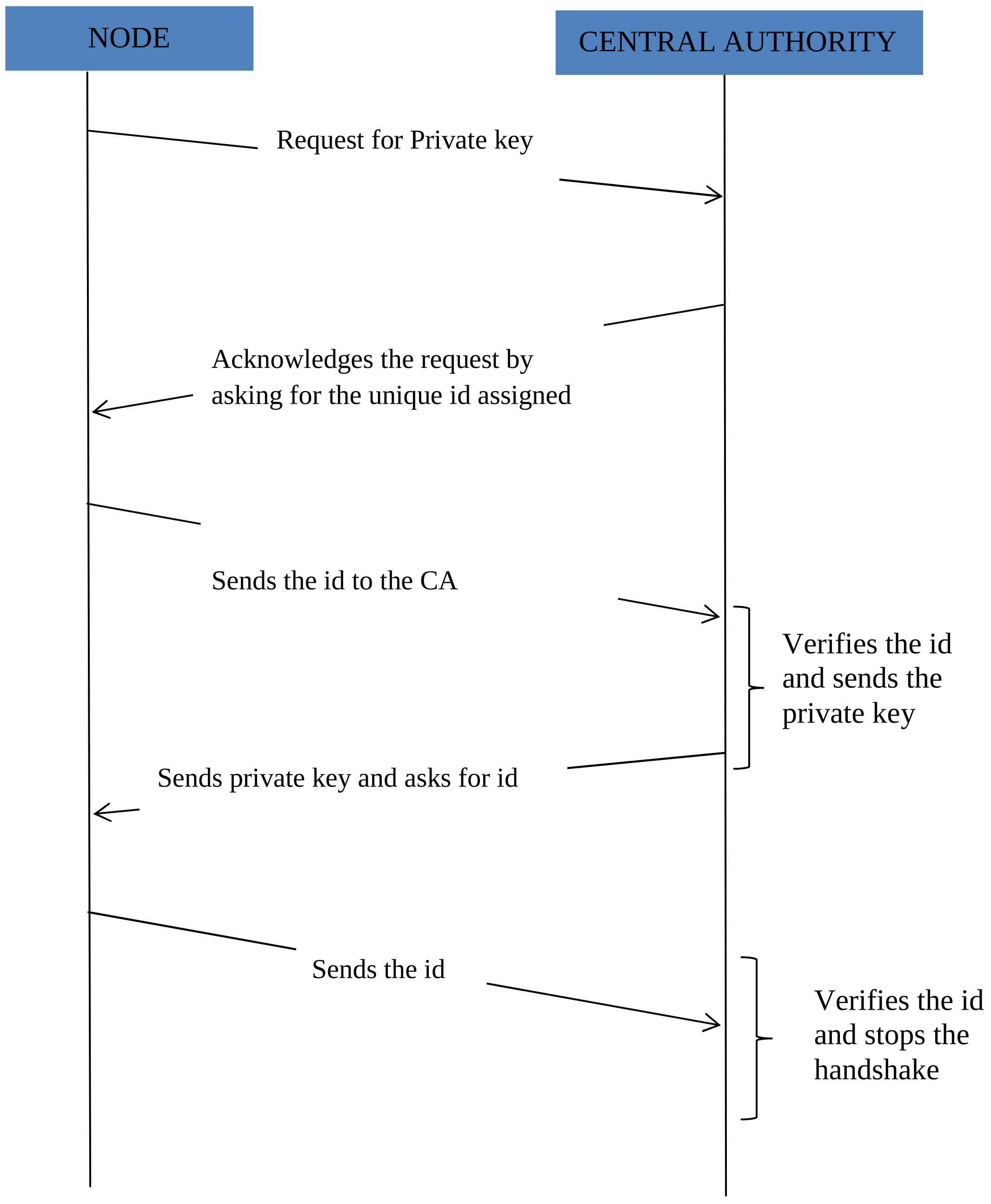

NODE
CENTRAL AUTHORITY
Request for Private key
Acknowledges the request by asking for the unique id assigned
Sends the id to the CA
Verifies the id and sends the private key
Sends private key and asks for id
Sends the id
Verifies the id and stops the handshake

Figure-12: Private Key Distribution

- The CA maintains the following intrusion table for the malicious nodes.

ID (malicious nodes)	INTRUSION COUNT
• MN1 • MN2	• 1 • 2

Table-1: Intrusion Table

- In any of the above steps, if the Central Authority detects any node to be malicious, it then updates the unique id value of that node as MN1, increments the intrusion count by 1, blocks all the communication in that particular path and multicasts it to all the nodes in the network.
- *Generation of shared key:*

  Finally user pair (Alice, Bob) can compute the shared key by multiplying the secret row of matrix A stored in the node with column of the public matrix corresponding to the node index with which it wants to communicate. The key generation between any two nodes is shown in the following Figure.

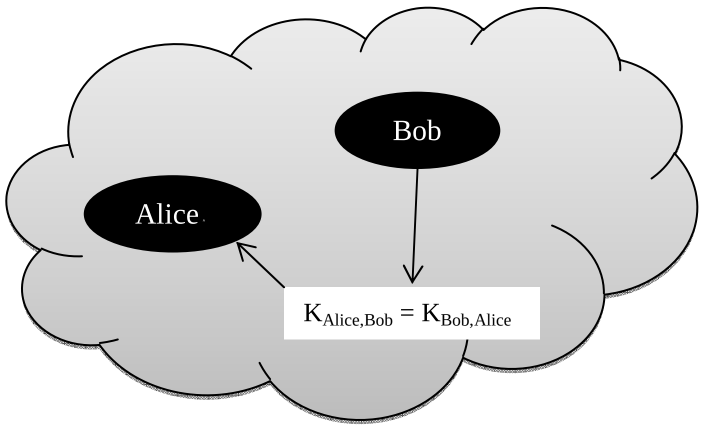


Figure-13 : Shared Key Generation

- The CA does the following handshake at regular time intervals in order to detect the malicious nodes.

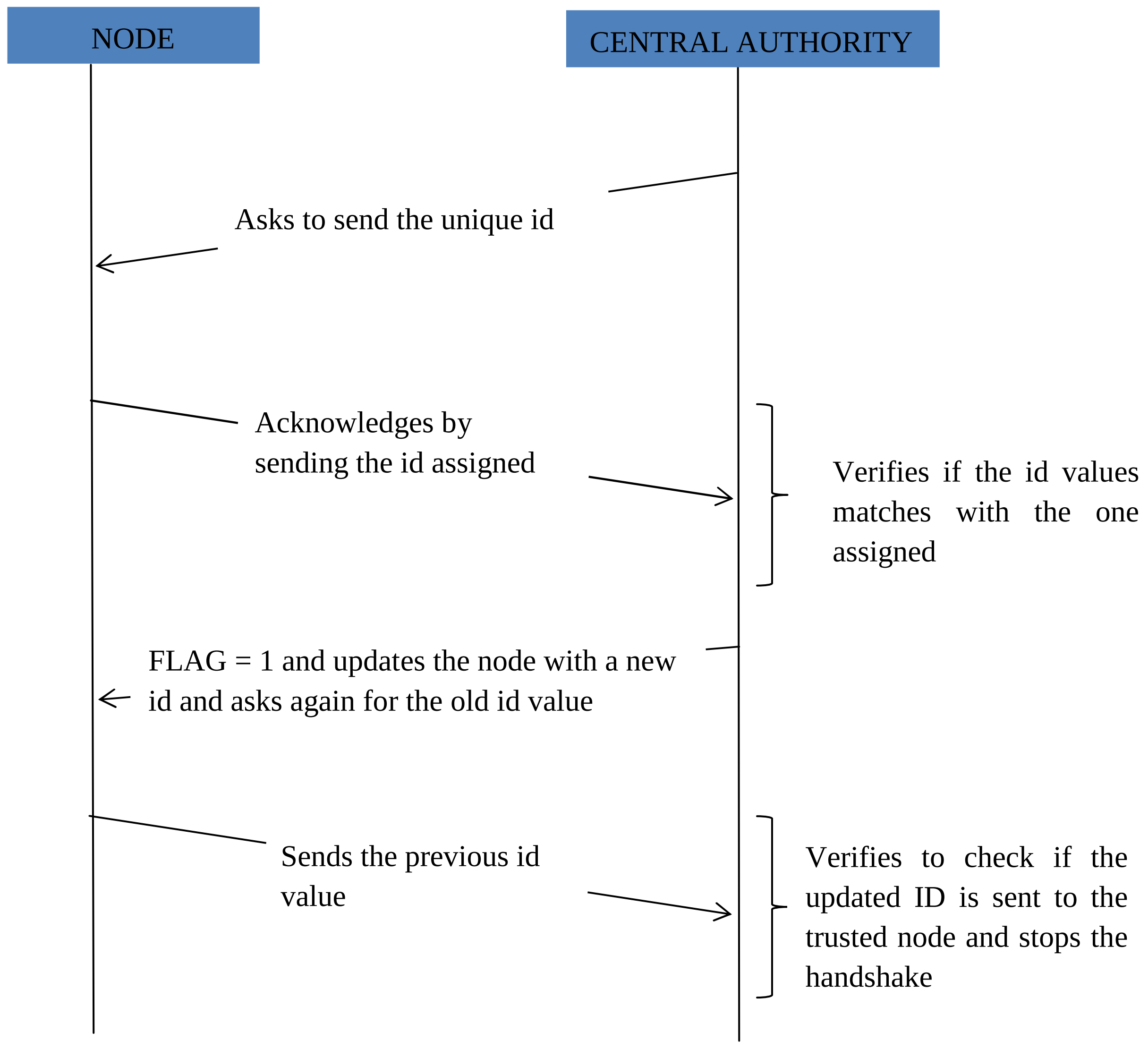

NODE
CENTRAL AUTHORITY
Asks to send the unique id
Acknowledges by sending the id assigned
Verifies if the id values matches with the one assigned
FLAG = 1 and updates the node with a new id and asks again for the old id value
Sends the previous id value
Verifies to check if the updated ID is sent to the trusted node and stops the handshake

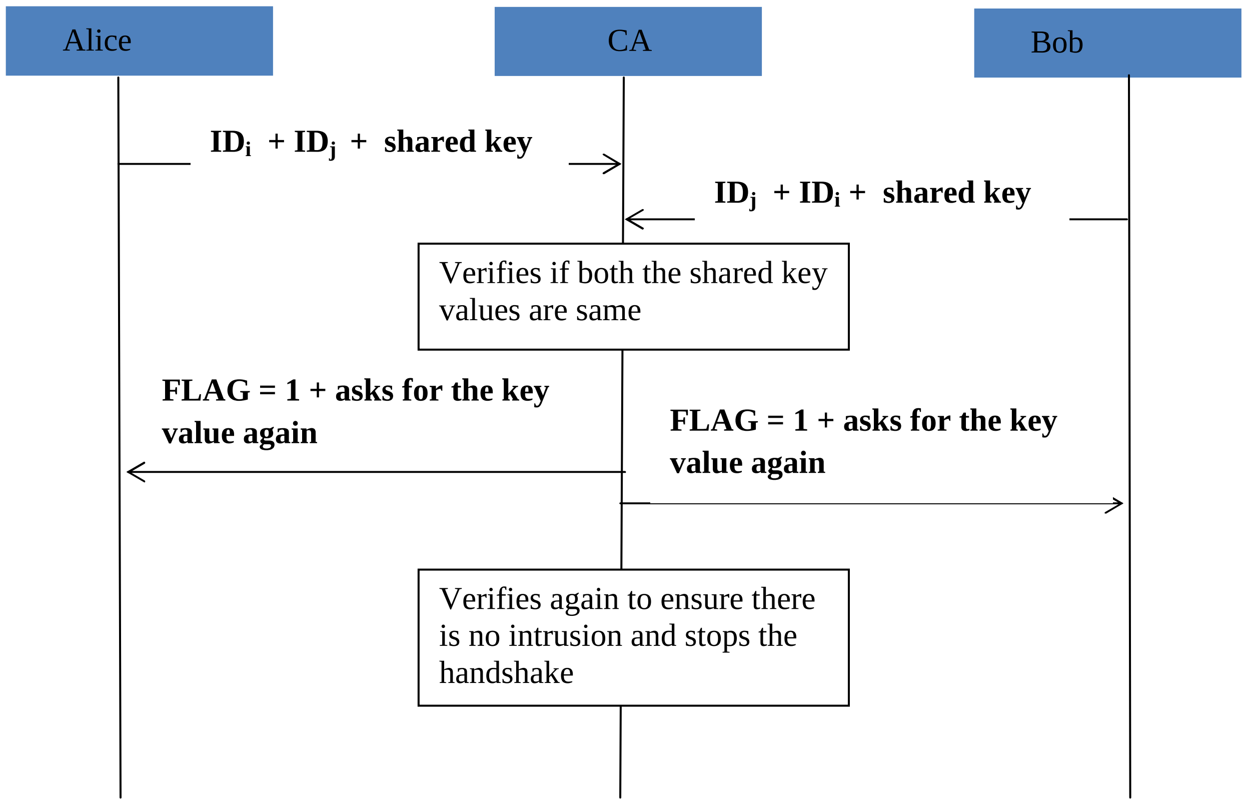
Figure- 14 : Malicious Node Detection

- *Dynamic Secret Matrix:*

  The CA dynamically updates the secret matrix in each of the following cases:

  - For every increment of the intrusion count in the intrusion table.
  - Suppose, if node i wants to communicate with node j (both i and j being trusted parties and not intruders). Once, the CA gives the private key to both the nodes, it updates the secret matrix for the next request from the nodes.

- Once the shared key has been established between the nodes, before the nodes start communicating with each other, the following handshake takes place between the nodes and the Central Authority in order to authenticate each other. Let us consider that Alice wants to communicate with Bob:



Alice
CA
Bob
$ID_i + ID_j +$ shared key
$ID_j + ID_i +$ shared key
Verifies if both the shared key values are same
FLAG = 1 + asks for the key value again
FLAG = 1 + asks for the key value again
Verifies again to ensure there is no intrusion and stops the handshake

Figure- 15: Node Authentication

Detection of Compromised Nodes:

A compromised node is a trusted node that has been taken over by the attacker.The individual nodes of a wireless sensor network (WSN) could be easily compromised by the adversary due to the constraints such as limited battery lifetime, memory space and computing capability. It is critical to detect and isolate the compromised nodes in order to avoid being misled by the falsified information injected by the adversary through compromised nodes. The following protocol can be used to detect the compromised nodes. There are five types of messages used in this protocol:

- HELLO
- AYT – Are You There
- IMF – I am Fine
- IMC – I am Captured
- Captured

Every node in the network sends a HELLO message packet to its neighbouring nodes which contains the particular unique node id assigned by the Central Authority. On receiving the HELLO message, the node is expected to send a corresponding HELLO message as an acknowledgement. Every node should run a monitoring routine which keeps a counter that records the number of consecutively missed HELLO messages from that particular neighbor. When the counter overflows the threshold of three, a probing AYT message is sent to the neighbor. The protocol demand that all nodes must respond to an AYT request. If a positive response, i.e. IMF message, is not received from that node after a fixed timeout, the guardian node broadcasts a Captured message and reports the node to be malicious to the Central Authority.

4.3 Implementation

The example below shows the working of our modified Blom's Scheme that uses a random matrix as public key and generates a private key. Let us consider a network with 5 nodes and the following parameters:

- Let C be the Central Authority or Base Station.
- Secure parameter t = 4, which says if more than 3 nodes in the network are compromised, it is not possible to find the keys of other users.
- Prime number q= 53.

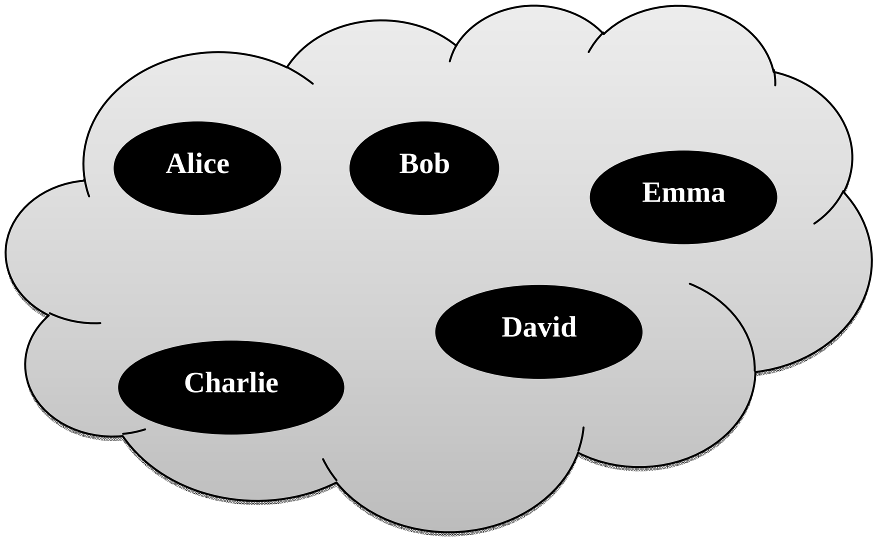


Figure-16: Node Structure Example

- As the public matrix (P) should be of the order (t+1) * (t+1), P can be taken as any random (4+1) * (4+1) matrix.

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 1 & 2 & 1 \\ 2 & 4 & 3 & 0 & 1 \\ 1 & 1 & 0 & 9 & 1 \\ 6 & 3 & 0 & 2 & 1 \end{bmatrix}$$

- The secret symmetric matrix can be obtained as follows:

  Let us take a random matrix,

$$A=\begin{bmatrix}3 & 4 & 2 & 3 & 2\\ 1 & 4 & 2 & 9 & 1\\ 6 & 2 & 4 & 1 & 0\\ 3 & 2 & 1 & 0 & 1\\ 2 & 1 & 1 & 0 & 1\end{bmatrix} \qquad B=\begin{bmatrix}3 & 1 & 6 & 3 & 2\\ 4 & 4 & 2 & 2 & 1\\ 2 & 2 & 4 & 1 & 1\\ 3 & 9 & 1 & 0 & 0\\ 2 & 1 & 0 & 1 & 1\end{bmatrix}$$

  We obtain the secret matrix S by taking the product of two matrices A and B,

$$S=\begin{bmatrix}42 & 52 & 37 & 21 & 14\\ 52 & 103 & 31 & 14 & 9\\ 37 & 31 & 57 & 26 & 18\\ 21 & 14 & 26 & 15 & 10\\ 14 & 9 & 18 & 10 & 7\end{bmatrix}$$

- Now, we calculate matrix A using,

  $A = (S.P)^T \bmod q$

$$(S.P)^T=\begin{bmatrix}377 & 491 & 378 & 190 & 129\\ 347 & 372 & 413 & 205 & 140\\ 289 & 352 & 313 & 155 & 105\\ 489 & 558 & 480 & 267 & 178\\ 334 & 417 & 317 & 170 & 114\end{bmatrix}$$

$$A=(S.P)^T \bmod 53=\begin{bmatrix}6 & 14 & 7 & 31 & 23\\ 29 & 1 & 42 & 46 & 34\\ 24 & 34 & 48 & 49 & 52\\ 12 & 28 & 3 & 2 & 19\\ 16 & 46 & 52 & 11 & 8\end{bmatrix}$$

  Once A is calculated, each sensor node memory is filled with unique row chosen from A with corresponding index. These are the private keys for the nodes.

- Now, for the key generation, we need the public and private keys of the nodes. Suppose, Bob and Charlie wants to communicate with each other. In order to generate the shared secret key, Bob should multiply the private key given by the Central Authority from A with the public column of Charlie in P.

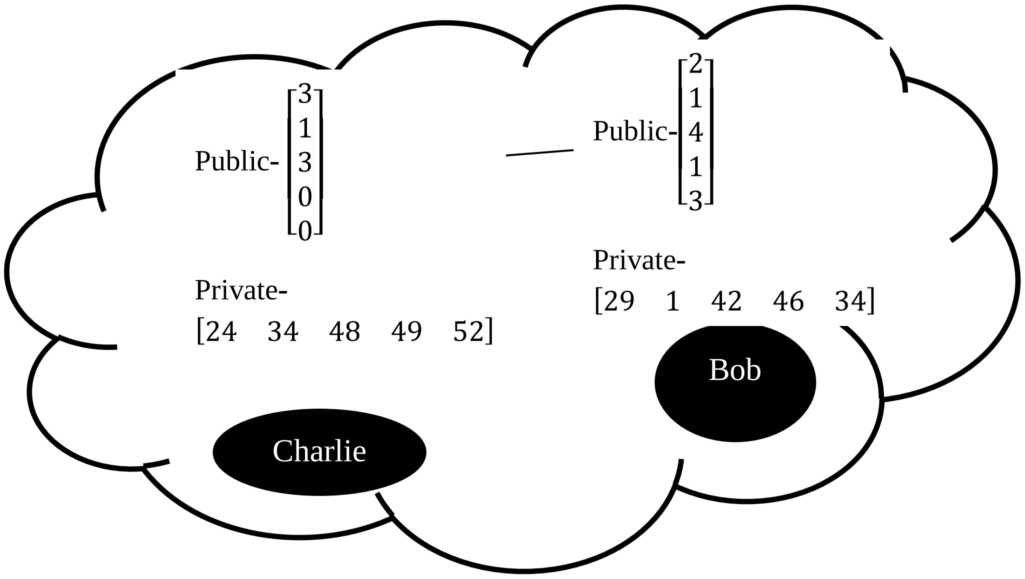

Public- $\begin{bmatrix}3\\1\\3\\0\\0\end{bmatrix}$
Private- $[24 \quad 34 \quad 48 \quad 49 \quad 52]$
Public- $\begin{bmatrix}2\\1\\4\\1\\3\end{bmatrix}$
Private- $[29 \quad 1 \quad 42 \quad 46 \quad 34]$
Bob
Charlie

Figure-17 : Public and Private keys

$$K_{\text{Bob, Charlie}} = A_{\text{Bob}}.\ P_{\text{Charlie}} = [29 \quad 1 \quad 42 \quad 46 \quad 34]\begin{bmatrix}3\\1\\3\\0\\0\end{bmatrix} = 214 \bmod 53 = 2$$

$$K_{\text{Charlie, Bob}} = A_{\text{Charlie}}.\ P_{\text{Bob}} = [24 \quad 34 \quad 48 \quad 49 \quad 52]\begin{bmatrix}2\\1\\4\\1\\3\end{bmatrix} = 479 \bmod 53 = 2$$

As seen above, both the nodes generate a common key and further communication can be done using the shared key generated.

Let us assume the following:

Alice – 1

Bob – 2

Charlie – 3

David – 4

Emma - 5

The table below shows the Public and Private keys for different nodes:

NODE PAIR	**PUBLIC KEY**	**PRIVATE KEY**	**SHARED KEY**
$K_{1,2}$	$\begin{bmatrix}2\\1\\4\\1\\3\end{bmatrix}$	$\begin{bmatrix}6 & 14 & 7 & 31 & 23\end{bmatrix}$	48
$K_{2,1}$	$\begin{bmatrix}1\\3\\2\\1\\6\end{bmatrix}$	$\begin{bmatrix}29 & 1 & 42 & 46 & 34\end{bmatrix}$	48
$K_{1,3}$	$\begin{bmatrix}3\\1\\3\\0\\0\end{bmatrix}$	$\begin{bmatrix}6 & 14 & 7 & 31 & 23\end{bmatrix}$	0
$K_{3,1}$	$\begin{bmatrix}1\\3\\2\\1\\6\end{bmatrix}$	$\begin{bmatrix}24 & 34 & 48 & 49 & 52\end{bmatrix}$	0
$K_{1,4}$	$\begin{bmatrix}4\\2\\0\\9\\2\end{bmatrix}$	$\begin{bmatrix}6 & 14 & 7 & 31 & 23\end{bmatrix}$	6

$K_{4,1}$	$\begin{bmatrix}1\\3\\2\\1\\6\end{bmatrix}$	$[12\quad 28\quad 3\quad 2\quad 19]$	6
$K_{1,5}$	$\begin{bmatrix}5\\1\\1\\1\\1\end{bmatrix}$	$[6\quad 14\quad 7\quad 31\quad 23]$	52
$K_{5,1}$	$\begin{bmatrix}1\\3\\2\\1\\6\end{bmatrix}$	$[16\quad 46\quad 52\quad 11\quad 8]$	52
$K_{2,3}$	$\begin{bmatrix}3\\1\\3\\0\\0\end{bmatrix}$	$[29\quad 1\quad 42\quad 46\quad 34]$	2
$K_{3,2}$	$\begin{bmatrix}2\\1\\4\\1\\3\end{bmatrix}$	$[24\quad 34\quad 48\quad 49\quad 52]$	2
$K_{2,4}$	$\begin{bmatrix}4\\2\\0\\9\\2\end{bmatrix}$	$[29\quad 1\quad 42\quad 46\quad 34]$	17
$K_{4,2}$	$\begin{bmatrix}2\\1\\4\\1\\3\end{bmatrix}$	$[12\quad 28\quad 3\quad 2\quad 19]$	17
$K_{2,5}$	$\begin{bmatrix}5\\1\\1\\1\\1\end{bmatrix}$	$[29\quad 1\quad 42\quad 46\quad 34]$	3

$K_{5,2}$	$\begin{bmatrix}2\\1\\4\\1\\3\end{bmatrix}$	$\begin{bmatrix}16 & 46 & 52 & 11 & 8\end{bmatrix}$	3
$K_{3,4}$	$\begin{bmatrix}4\\2\\0\\9\\2\end{bmatrix}$	$\begin{bmatrix}24 & 34 & 48 & 49 & 52\end{bmatrix}$	20
$K_{4,3}$	$\begin{bmatrix}3\\1\\3\\0\\0\end{bmatrix}$	$\begin{bmatrix}12 & 28 & 3 & 2 & 19\end{bmatrix}$	20
$K_{3,5}$	$\begin{bmatrix}5\\1\\1\\1\\1\end{bmatrix}$	$\begin{bmatrix}24 & 34 & 48 & 49 & 52\end{bmatrix}$	38
$K_{5,3}$	$\begin{bmatrix}3\\1\\3\\0\\0\end{bmatrix}$	$\begin{bmatrix}16 & 46 & 52 & 11 & 8\end{bmatrix}$	38
$K_{4,5}$	$\begin{bmatrix}5\\1\\1\\1\\1\end{bmatrix}$	$\begin{bmatrix}12 & 28 & 3 & 2 & 19\end{bmatrix}$	6
$K_{5,4}$	$\begin{bmatrix}4\\2\\0\\9\\2\end{bmatrix}$	$\begin{bmatrix}16 & 46 & 52 & 11 & 8\end{bmatrix}$	6

Table-2: Key Generation for all nodes

Now, the CA can anytime update the secret matrix S, to protect the network from malicious users.

$$S' = \begin{bmatrix} 42 & 52 & 37 & 21 & 14 \\ 52 & 103 & 31 & 14 & 9 \\ 37 & 31 & 57 & 26 & 18 \\ 21 & 14 & 26 & 15 & 10 \\ 14 & 9 & 18 & 10 & 7 \end{bmatrix} + \begin{bmatrix} 10 & 20 & 21 & 22 & 13 \\ 20 & 30 & 12 & 13 & 14 \\ 21 & 12 & 50 & 15 & 16 \\ 22 & 13 & 15 & 70 & 17 \\ 13 & 14 & 16 & 17 & 90 \end{bmatrix}$$

$$S' = \begin{bmatrix} 52 & 72 & 58 & 43 & 27 \\ 72 & 133 & 43 & 27 & 23 \\ 58 & 43 & 107 & 41 & 34 \\ 43 & 27 & 41 & 85 & 27 \\ 27 & 23 & 34 & 27 & 97 \end{bmatrix}$$

The Private matrix A can be calculated as,

$A = (S.P)^T \bmod q$

$$A = \begin{bmatrix} 589 & 722 & 646 & 453 & 773 \\ 532 & 545 & 730 & 443 & 531 \\ 402 & 478 & 538 & 279 & 206 \\ 793 & 843 & 755 & 1045 & 591 \\ 460 & 586 & 515 & 395 & 316 \end{bmatrix} \text{ Mod } 53$$

$$A = \begin{bmatrix} 6 & 33 & 10 & 29 & 31 \\ 2 & 15 & 41 & 19 & 1 \\ 31 & 1 & 8 & 14 & 47 \\ 51 & 48 & 13 & 38 & 8 \\ 36 & 3 & 38 & 14 & 51 \end{bmatrix}$$

Now, suppose Bob and Alice wants to communicate with each other.

$$K_{Alice,\ Bob} = A_{Alice}.\ P_{Bob} \quad = [6 \quad 33 \quad 10 \quad 29 \quad 31] \begin{bmatrix} 2 \\ 1 \\ 4 \\ 1 \\ 3 \end{bmatrix}$$

= 207 mod 53

= 48

$K_{Bob,\ Alice} = A_{Bob} .\ P_{Alice}$

$$= \begin{bmatrix} 2 & 15 & 41 & 19 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \\ 1 \\ 6 \end{bmatrix}$$

$= 154 \bmod 53$

$= 48$

Both the nodes generate a common key and further communication can be done using the shared key generated.

Hence, by randomly updating the secret matrix, the Central Authority makes sure that the malicious nodes attack on the network is reduced.

CHAPTER V

SIMULATION AND RESULTS

The main drawbacks of Blom’s scheme are memory overhead and computation overhead. This paper analyzes the ways to overcome these drawbacks. The modified Blom’s scheme proposes the use of random matrix as the public matrix instead of the Vandermonde matrix. This reduces the computational complexity corresponding to the columns in Vandermonde matrix.In this paper, we have performed an analysis between the original Blom’s scheme and the proposed Blom’s scheme with respect to computational effort. Different simulations have been made to a node size of 2,4,6, 8,10,20,30,40,50,100 and 200. Also, the Blom’s scheme is made dynamic by changing the values of the secret matrix in order to protect the network from the malicious nodes. Different handshakes have been introduced so that the Central Authority can monitor the network at regular intervals.

The results of the simulation show that the computational complexity of the modified Blom’s scheme is less when compared to that of the original Blom’s scheme.

The time complexity of obtaining different matrices like the Hadamard matrix, Adjacency matrix and the complexity of obtaining the random matrix is calculated and the following results have been obtained:

For a given number n, where n is the number of nodes in the network, the time taken to generate a random matrix of order n has the plot shown below. The random matrix can be generated by using the below equation.

r = ceil((k-l).*rand(n,n) + l);

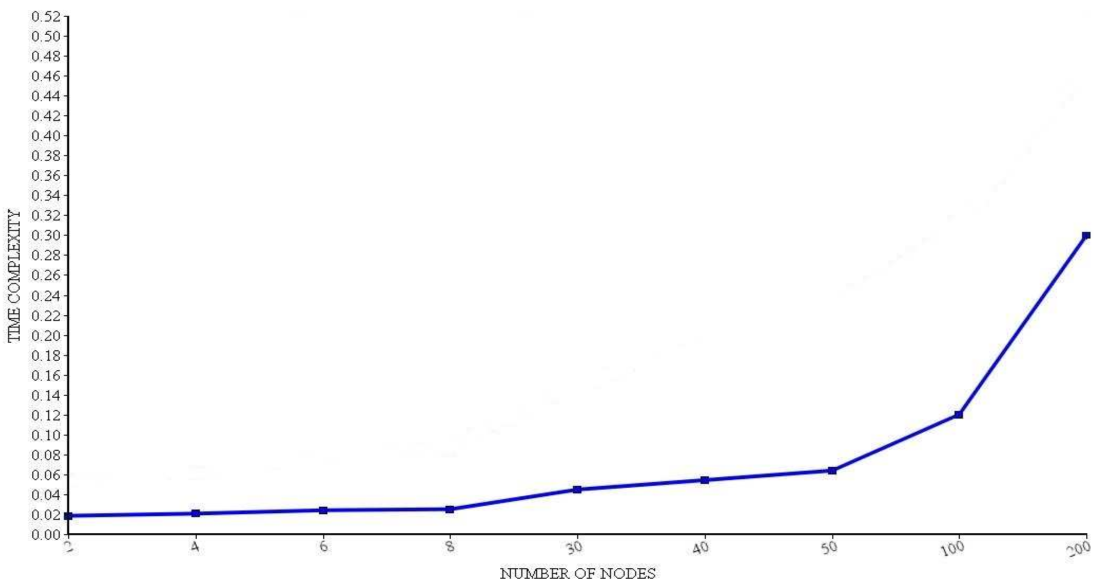


Figure:18 - Time Complexity for obtaining Random Matrix

The time taken to generate adjacency matrix for graphs with different node sizes 2,4,6,8,30,40,50,100 and 200 is shown below:

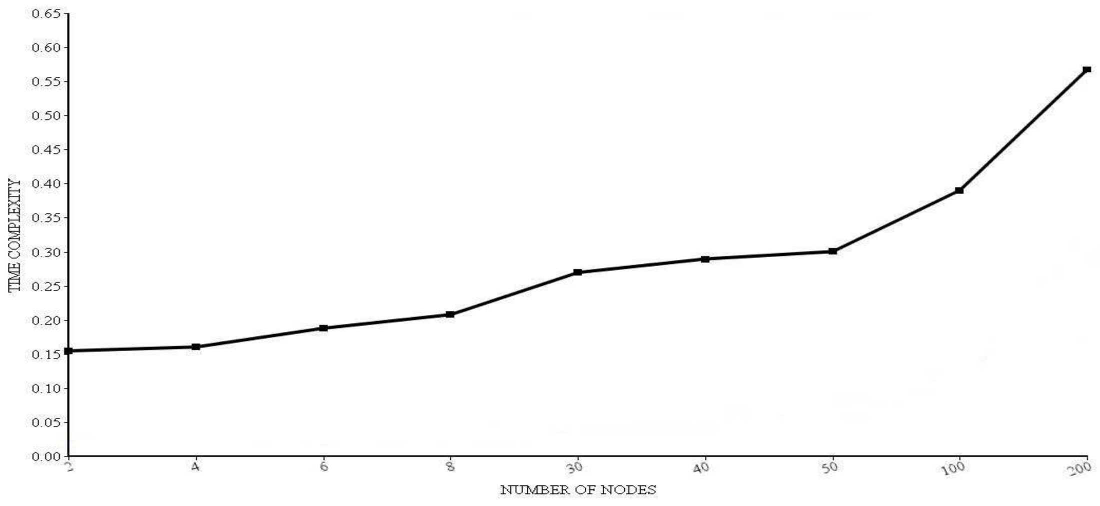


Figure:19- Time Complexity for obtaining Adjacency Matrix

For different values of n, where n is the order of the matrix, the time varying time taken to generate a hadamard matrix is shown below.

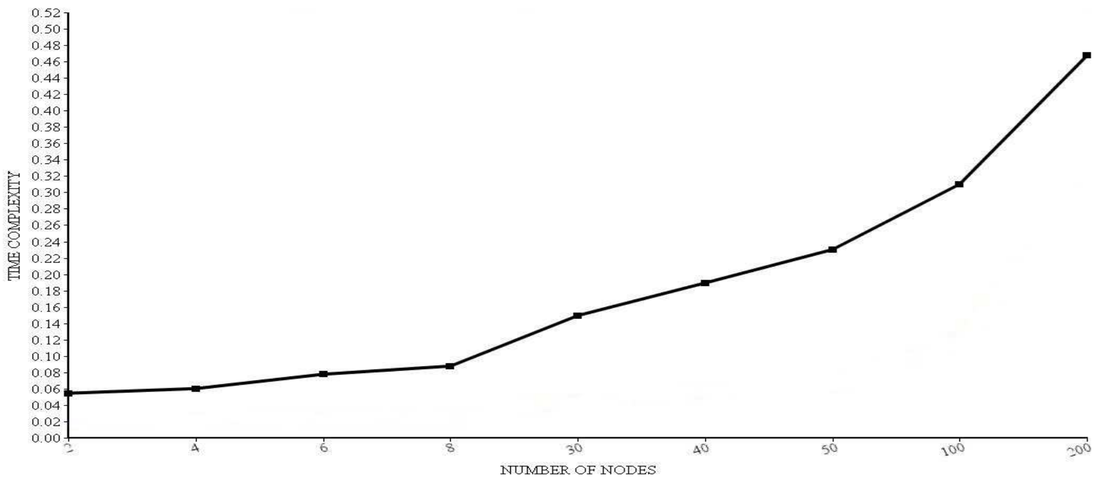


Figure:20 - Time complexity for obtaining Hadamard Matrix

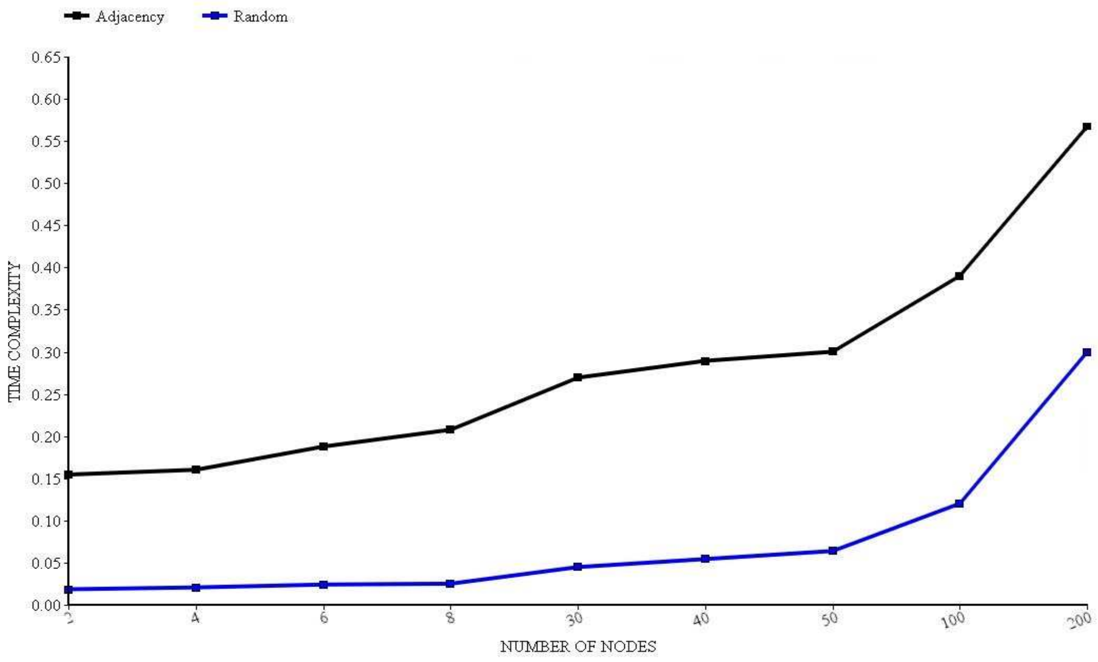


Figure: 21- Comparision of Time complexity for Adjacency and a Random Matrix

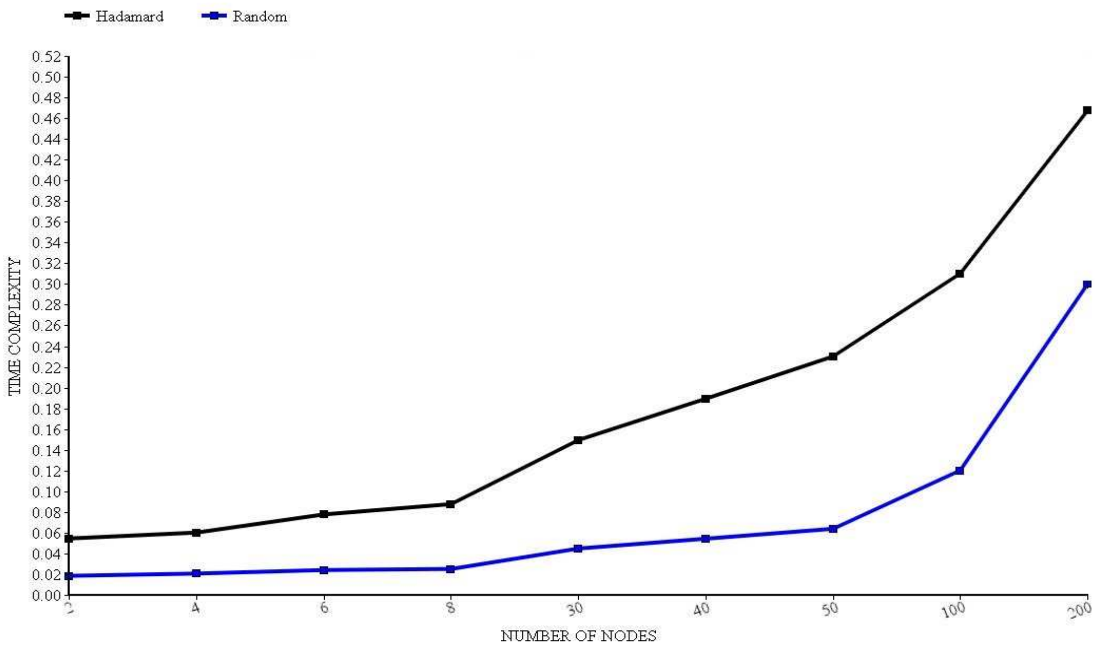


Figure:22 - Comparision of Time complexity for Hadamard and a Random Matrix

Also, the matrix multiplication complexity has been compared between the original scheme and the proposed scheme. The original scheme requires (3n-2) steps to complete the matrix multiplication where as the proposed scheme completes the computation in (2n-1) steps. Both the computations are compared and the following results have been obtained.

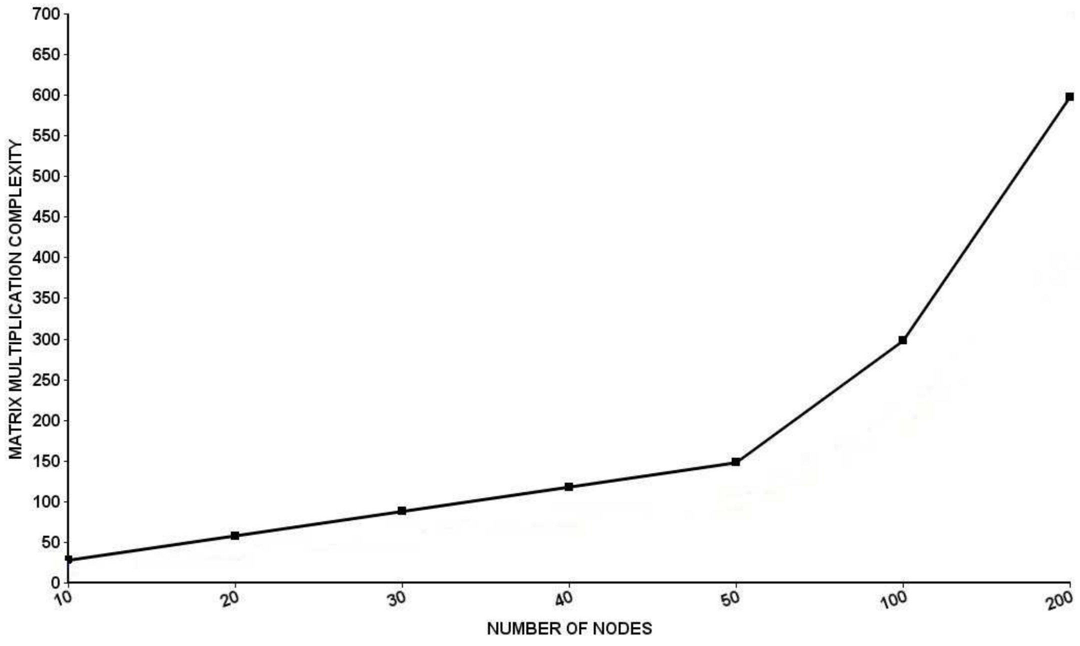


Figure- 23 :Matrix Multiplication complexity for original scheme

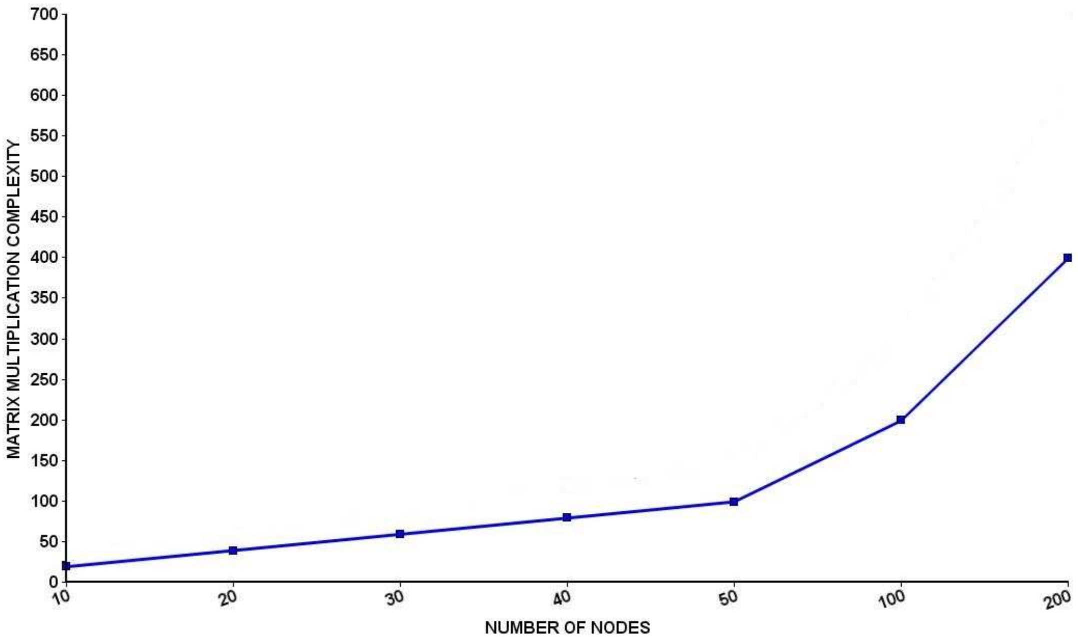


Figure-24: Matrix Multiplication complexity for proposed scheme

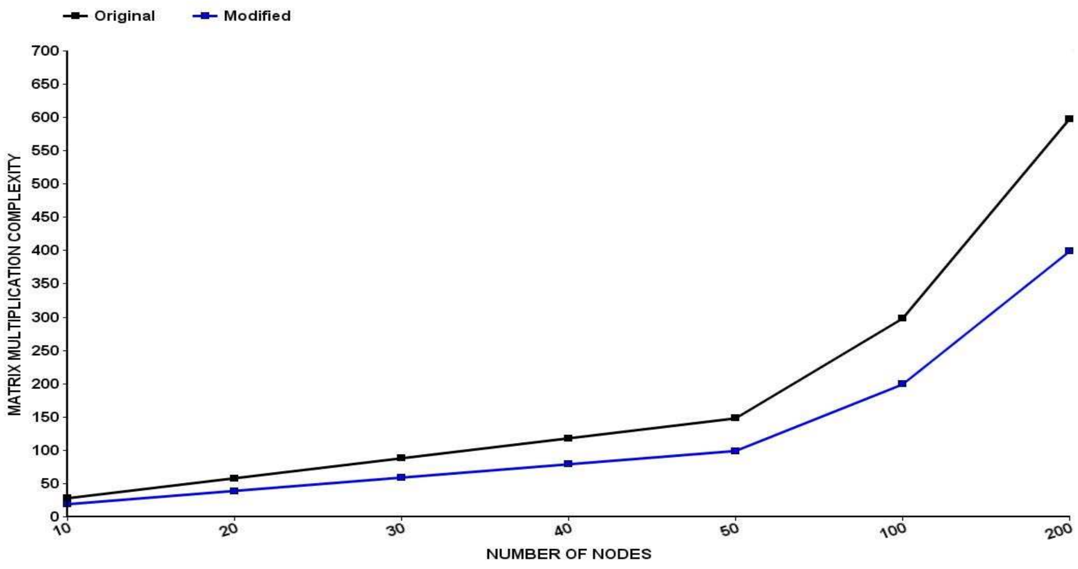


Figure-25: Comparision of Matrix multiplication complexity for original and proposed schemes

Hence, the computational complexity can further be reduced by using any random public matrix instead of the Vandermonde matrix as in the original scheme and the efficiency of the original scheme can be improved by making the scheme dynamic. The Vandermonde matrix can be generated by using:

A = vander(v); fliplr(A); ,where v is the order of the matrix.

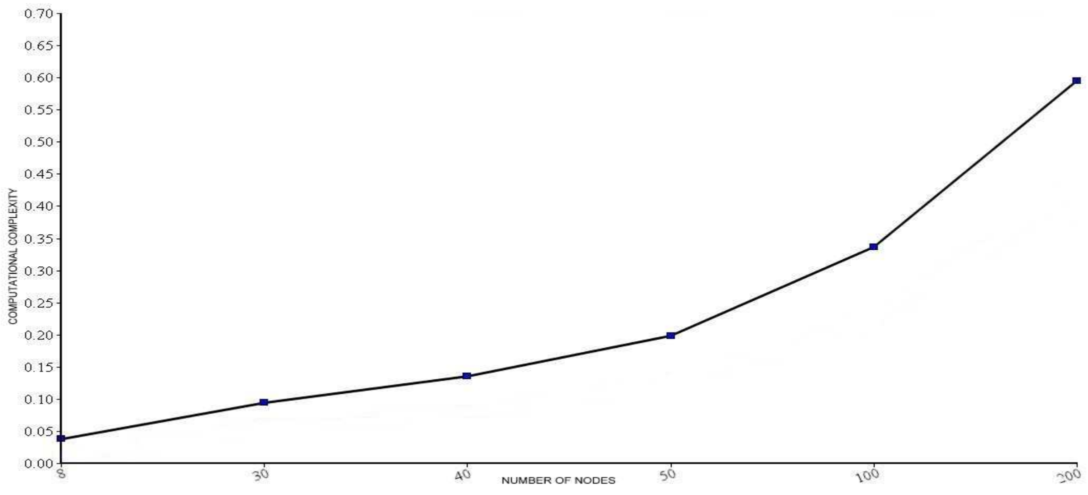


Figure-26: Computational Effort for Original Blom's Scheme

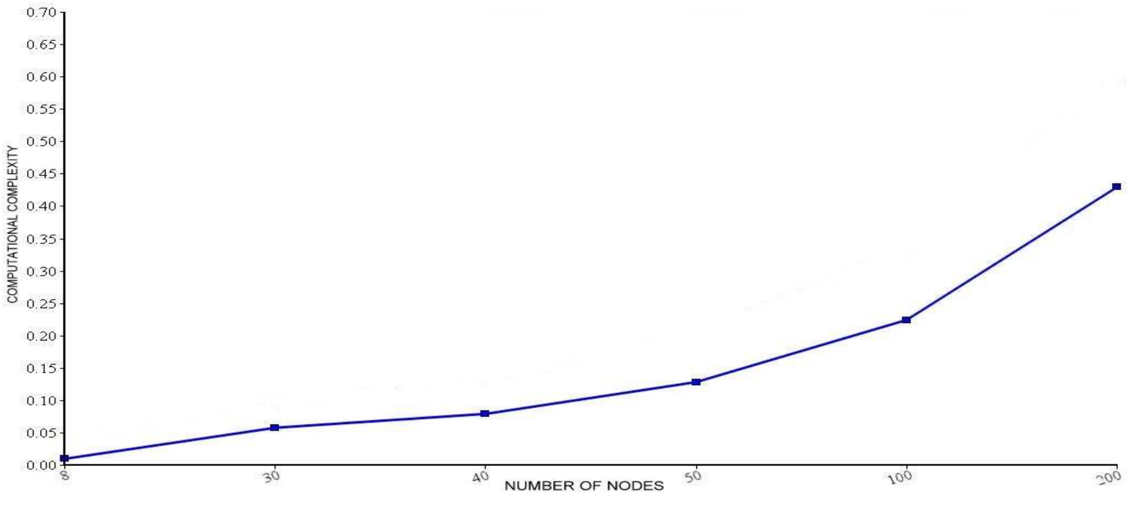


Figure-27: Computational Effort for proposed scheme

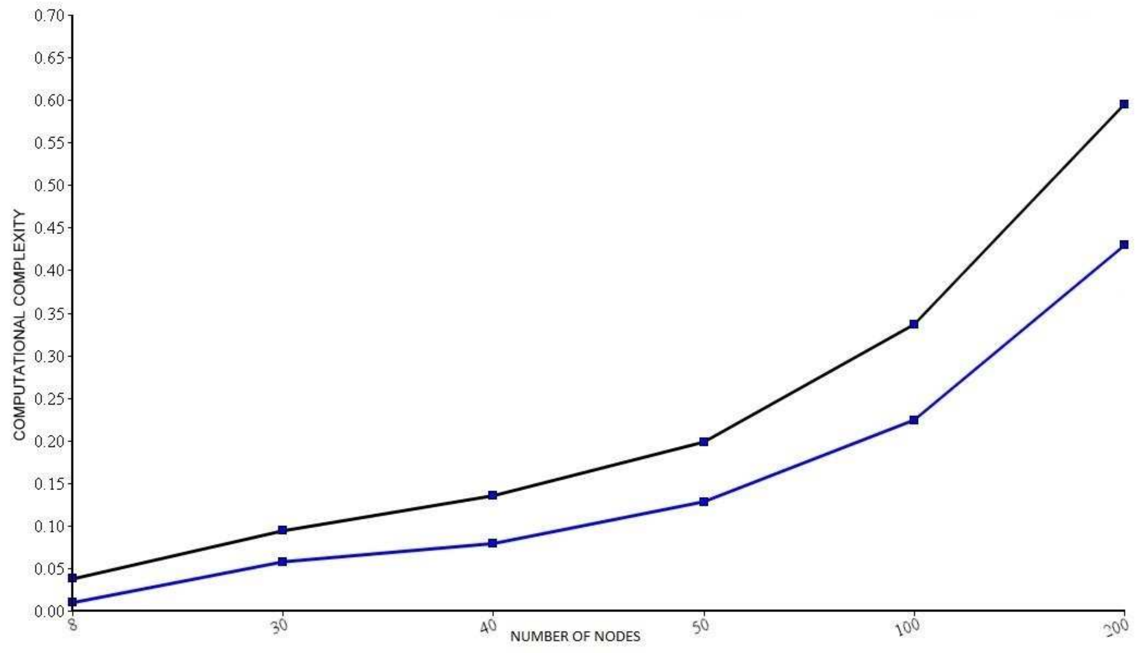


Figure-28: Comparison of the Computational Effort for the original and proposed scheme

Hence, from the above results, it can be observed that the computational effort required by the proposed scheme is less as compared to that of the original scheme. Moreover, this difference can be more as the number of sensor nodes in the network increases.

CHAPTER VI

CONCLUSION

Key Management is an important issue in wireless networks. For any network, in order to ensure secure data transmission between the nodes, Authentication and confidentiality play a major role. Blom’s scheme is a symmetric key management scheme that allows users to find a shared key for secure communication. Its drawbacks include computational and memory overhead.

The proposed scheme overcomes the drawbacks of the Blom’s scheme using Random matrix and provides different ways to protect the network from malicious nodes. Different Handshake protocols have been introduced for the purpose of authentication and confidentiality. A more generic and dynamic algorithm from a scalability point of view is developed.

Also, simulations are carried out for different node sizes and is analyzed from the computational complexity point of view. We have shown how the protocols proposed for dynamic changing of keys can be run in an efficient manner.

REFERENCES

[1] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: Mobile networking for smart dust. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 483–492, 1999.

[2] Wireless Integrated Network Sensors, University of California, Available: http://www.janet.ucla.edu/WINS.

[3] G. J. Pottie and W. J. Kaiser. Wireless Integrated Network Sensors. Communications of the ACM, 43(5):51-58, 2000.

[4] E. Stavrou, A. Pitsillides, G. Christoforos Hadjicostis, " Security in future mobile sensor networks-Issues and challenges", Department of Electrical and Computer Engineering, University of Cyprus.

[5] S. Kak, A two-layered mesh array for matrix multiplication. Parallel Computing 6, 383-385, 1988.

[6] G.J. Pottie, W. J. Kaiser, "Wireless Integrated Network Sensors", in communications of the ACM, 2000, vol 43(5) pages 51-58.

[7] Z. Yu and Y. Guan, "A key management scheme using deployment knowledge for wireless sensor networks," *IEEE Trans. on Parallel and Distributed Systems* 19:1411-1425, 2008.

[8] L. Eschenauer and V. Gligor, "A key management scheme for distributed sensor networks," in ACM CCS2002, Washington D.C., 2002.

[9] W. Du, J. Deng, Y S. Han, P. Varshney, J. Katz, and A. Khalili. A pairwise key pre distribution scheme for wireless sensor networks. In *ACM Transactions on Information and System Security (TISSEC)*, 2005.

[10] D. Liu, P. Ning, and R. Li. Establishing pairwise keys in distributed sensor networks. In*ACM Trans. Inf Syst. Secur*., 8:41-77, Oct 2005.

[11] W. H. Press, B.P. Flannery, S. A. Teukolsky and W. T. Vetterling, "Vandermonde Matrices and Toeplitz Matrices." 2.8 in Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2nd ed. Cambridge, England: Cambridge University Press, pp. 82-89, 1992.

[12] R. Blom, "An optimal class of symmetric key generation systems," in Proc. of EUROCRYPT '84, pages 335-338, 1985.

[13] Rohith Singh Reddy, "Key management in wireless sensor networks using a modified Blom scheme", arXiv: 1103.5712.

[14] Suraj Sukumar, "Computational Analysis of Modified Blom's Scheme",
arXiv: 1303.7457.

[15] S. Kak, Multilayered array computing. Information Sciences 45, 347-365, 1988.
A.Parakh and S. Kak.

[16] A.Parakh and S. Kak, Efficient key management in sensor networks. Proceedings IEEE GLOBECOM workshops (GC workshops). pp. 1539–1544, 2010.

[17] A.Parakh and S. Kak, Matrix based key agreement algorithms for sensor networks. Proceedings IEEE 5th International Conference on Advanced Networks and Telecommunication Systems (ANTS), pp. 1–3, 2011.

[18] A. Parakh and S. Kak, Space efficient secret sharing for implicit data security. Information Sciences, vol. 181, pp. 335-341, 2011.

VITA

DIVYA HARIKA NAGABHYRAVA

Candidate for the Degree of

Master of Science

Thesis: EFFICIENT KEY GENERATION FOR DYNAMIC BLOM'S SCHEME

Major Field: Computer Science

Biographical:

Education:

- Master of Science in Computer Science at Oklahoma State University, Stillwater,Oklahoma, December, 2014.

- Bachelor of Technology in Computer Science and Engineering at Jawaharlal Nehru Technological University, Hyderabad, Andhra Pradesh/India, 2012.

Experience:

Graduate Teaching Assistant at Computer Science Department, OSU from August 2013 to May 2014