Key Pre-distribution and Key Revocation in Wireless Sensor Networks

Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Technology

in

Computer Science and Engineering

(Specialization: Information Security)

by

Subhankar Chattopadhyay

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela, Orissa, 769 008, India

May 2011

Key Pre-distribution and Key Revocation in Wireless Sensor Networks

Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Technology

in

Computer Science and Engineering

(Specialization: Information Security)

by

Subhankar Chattopadhyay

(Roll- 209CS2084)

Supervisor

Dr. Ashok Kumar Turuk

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela, Orissa, 769 008, India

May 2011

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India.

Certificate

This is to certify that the work in the thesis entitled ***Key Pre-distribution and Key Revocation in Wireless Sensor Networks*** by ***Subhankar Chattopadhyay*** is a record of an original research work carried out by him under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of Master of Technology in Computer Science & Engineering with specialization in Information Security from the department of Computer Science and Engineering, National Institute of Technology Rourkela. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

NIT Rourkela
May, 2011

Dr. Ashok Kumar Turuk
Associate Professor, CSE Department
NIT Rourkela, Orissa

Acknowledgment

It gives me immense pleasure to see my thesis complete. I would like to thank all those who helped me to make it possible. I am very much lucky that I had Dr. Ashok Kumar Turuk as my guide. At the time during my thesis work he was the head of the department. In spite of his busy schedule, he always had time for me. His valuable comments and suggestions were encouraging. He was the one who showed me the path from the beginning to the end.

I am also thankful to Dr. Majhi, Dr. S. K. Rath, Dr. S. K. Jena, Dr. D. P. Mohapatra, Dr. R. Baliarsingh, and Dr. P. M. Khilar for giving encouragement and sharing their knowledge during my thesis work.

I express my gratitude to Dr. Sushmita Ruj and Dr. Bimal Roy for their valuable suggestions at important times.

I am really thankful to my all my lab mates and friends. They made my life beautiful and helped me every-time when I was in some trouble.

I must acknowledge the academic resources that I have got from NIT Rourkela. I would like to thank administrative and technical staff members of the Department who have been kind enough to help us in their respective roles.

Last, but not the least, I would like to dedicate this thesis to my family, for their love, patience, and understanding.

Subhankar Chattopadhyay

Email - subho.atg@gmail.com

Abstract

Sensor networks are composed of resource constrained tiny sensor devices. They have less computational power and memory. Communication in sensor network is done in multi-hop, and for secure communication, neighboring sensor nodes must possess a secret common key among them. Symmetric and public key cryptography require more processing and memory space. Hence, they are not suitable for sensor network. Key pre-distribution is a widely accepted mechanism for key distribution in sensor network.

In this thesis we proposed a deterministic key pre-distribution scheme using BCH codes. We mapped the BCH code to key identifier and the keys corresponding to each key identifier are installed into the sensor nodes before deployment. We compared our proposed scheme with existing one and found that it has a better resiliency. Our proposed scheme is scalable and requires the same or less number of keys for a given number of nodes than the existing well known schemes. We have also proposed an efficient key revocation technique using a novel distributed voting mechanism in which neighboring nodes of a sensor can vote against it if they suspect the node to be a compromised one. In the proposed key revocation scheme compromised nodes as well as the compromised keys are completely removed from the network.

Contents

Certificate **ii**

Acknowledgement **iii**

Abstract **iv**

List of Figures **vii**

List of Tables **viii**

1 Introduction **2**

1.1 Sensor Network 3

1.2 Key Management in Wireless Sensor Networks 4

1.3 Design Theory 6

1.4 Motivation 8

1.5 Thesis Organization 8

2 Background **11**

2.1 Background of Key Pre-distribution Schemes 11

2.1.1 Basic Schemes 11

2.1.2 Random Pairwise Scheme 13

2.1.3 Grid-based Key Pre-distribution Schemes 14

2.1.4 Group Based Key Pre-distribution 15

2.1.5 Key Pre-distribution Using Combinatorial Structures 16

2.1.6 Key Pre-distribution Using Deployment Knowledge 20

2.2 Background of Key Revocation Schemes 21

3 Key Pre-distribution Using BCH Code **24**

3.1 Terms and Definitions 25

3.2 Key Pre-distribution Using BCH Code 26
3.3 Shared Key Discovery and Path-Key Establishment Phase 30
3.4 Scalability of the Scheme 31
3.5 Result and Comparison with Other Schemes 33
3.6 Conclusion 34

4 Proposed Key Revocation Scheme **36**

4.1 Problems with Chan *et. al.* [1, 2] Key Revocation Mechanism 36
4.2 Network Model and Assumptions 37
4.3 Proposed Scheme 40
4.3.1 Scheme I 40
4.3.2 Scheme II 42
4.4 Analysis of the Proposed Scheme 43
4.5 Conclusion 45

5 Conclusion and Future work **47**

Bibliography **48**

List of Figures

4.1 Division of network into basic and non-basic region 38

List of Tables

2.1 Various generalized quadrangles used by Camtepe and Yener and their different parameters 16
2.2 Connection Probability and Resiliency(fail(1)) for different value of t In Camtepe and Yener scheme 17

3.1 Mapping of parameters between key pre-distribution and their corresponding parameters from BCH codes 28
3.2 Conjugate sets and their corresponding minimal polynomials 29
3.3 Node ID along with its corresponding node polynomial, code polynomial and codeword for *Sixteen* number of nodes 30
3.4 Key Identifiers for Node ID 1 30
3.5 Key Identifiers of all the *Sixteen* nodes considered in the example . 31
3.6 Symbolic Code Representation and Code polynomial for all the newly added Nodes 32
3.7 Key Identifiers of all the newly added *Sixteen* number of nodes in the network 33
3.8 Comparison of proposed key pre-distribution scheme with Ruj and Roy (R R) scheme and Camtepe and Yener (C Y) scheme. Number of nodes in the network is N, keys per node is k, number of compromised node is s and resiliency is *Fail(s)*[*Fail(s)* is the probability of affected links due to the compromise of s nodes]. 34

4.1 Comparison between key revocation Scheme I and Scheme II 46

Chapter 1

Introduction

Sensor Network
Key Management in Sensor Network
Design Theory
Motivation
Thesis Organization

Chapter 1

Introduction

Cryptography has been used from a very long time by human being for secure communication. Today, when the electronic communication is growing rapidly, need for secure communication is also growing at a brisk speed. Hence cryptography is now of immense importance. Cryptography is a set of mathematical operations and algorithms by which we can preserve the secrecy of a data while transmitted through insecure channel. The four main goals of a cryptographic algorithm are

1. *Confidentiality* : Hiding data from the unauthorized users.

2. *Data Integrity* : Protection of data from alteration.

3. *Authentication* : Verification of sender.

4. *Non Repudiation* : Prevention of malicious users from hiding their inactivity.

Key is the most important component for most of the Cryptographic algorithms. Keys are generally numbers randomly selected from a large set of numbers. Management of these keys are very important in cryptography. Management of keys includes the following:

1. *Key Generation* : It is the process in which a pool of keys are generated. Mainly it is done in off-line mode by a trusted authority.

2. *Key Establishment* : It is the most important phase of key management process. Key establishment is the process by which right keys for right users can be determined and key rings for each users are sent to them accordingly.

Key establishment can be done in many ways. *Trusted Authority* can help in sending the keys to each user through a secure channel. But this mechanism is a costly one and does not suit for sensor networks. So, in sensor network we use *Key Pre-distribution* in which key rings are installed in the nodes before deployment of network in off-line mode.

3. *Key Updation* : It is the process by which we can update the keys of all the users after a certain time interval.

4. *Key Revocation* : This process is the deletion of compromised keys.

Based on the use of key, cryptographic algorithms can be divided into two types.

1. *Symmetric Key Cryptography* : Same key is used for encryption and decryption module.

2. *Asymmetric Key Cryptography* : Different keys are used for encryption and decryption.

Asymmetric cryptography needs huge computational and communicational costs. So, for those areas where resources are constrained, symmetric cryptography is used. Sensor network security is one such area.

1.1 Sensor Network

Recently a lot of research is going on in the area of sensor network. Sensor network is composed of large number of tiny resource constrained sensor nodes with no fixed network topology. It has a wide range of applications in military as well as in civilian services. Some of the applications of sensor networks has been listed below.

1. Sensor nodes are deployed in a battlefield to detect enemy intrusion.

2. They are also used to measure various environmental variables such as temperature, heat, sound, pressure, magnetic and seismic fields etc. of a region.

3. Sensor network has several use in industry such as in machine health monitoring, waste water monitoring etc.
4. It is used for traffic monitoring also.
5. Detection of bio-chemical or any explosive material is also possible with this.
6. They are used for security in public places.
7. Sensor networks are used in parking zone to help parking cars.

1.2 Key Management in Wireless Sensor Networks

Because of the various application it has in different fields, the data which are transmitted needs to be kept secret. For example in military applications all the data transmitted through the network are critical and secure communication is needed for them. So cryptographic key management is a challenging task in wireless sensor networks. But sensor networks have some characteristics which make it difficult to communicate securely. Some of those characteristics are listed below:

1. Generally sensor networks consist of large number of sensor nodes which makes it difficult to secure each and every nodes. Sensor nodes are very inexpensive tiny devices and most of the time they are kept unattended. That makes them a victim of physical attack.
2. Sensor nodes are constrained in resources which makes difficult to implement complex cryptographic algorithms. We have discussed previously that because of constrained resources it is difficult to implement public key cryptography in sensor networks.
3. Wireless nature of the networks makes it easier to eavesdrop.
4. There is no definite network topology in sensor network. Because of that it is difficult to implement any protocols.

Because of these challenges, key establishment is a very challenging task in sensor network. Key establishment via a trusted center through secure channel is difficult to implement because of it is too costly. So, we generally use key pre-distribution as a procedure to establish keys in case of sensor networks. Key pre-distribution is a mechanism in which keys for each node are chosen from a large key pool. The main goals of a good key pre-distribution algorithm are

1. Key connectivity: If two sensor nodes share some common key as well as they are in communication range of each other then they can communicate with each other. The probability that any two nodes can communicate with each other must be high. Connectivity is defined as $P_c = \frac{L}{\frac{N(N-1)}{2}}$ where L is the number of links in the network and N is the number of nodes in the network.

2. Resiliency: Once some nodes are captured or compromised, the rest of the network must be least affected. It is measured as the fraction of links disconnected when s number of nodes are compromised. Resiliency $E(s) = \frac{L^1}{L}$ where L is the number of links present before s nodes are compromised and L^1 is the number of links present after s nodes are compromised.

3. Storage requirement and Computational cost: Storage requirement should be as less as possible as sensor nodes are tiny devices which have lesser memory capacity. Computational cost of the algorithm also should be less.

4. Key Revocation : There should be an efficient mechanism for revoking the keys.

Key establishment process in Wireless sensor networks mainly consists of three phases.

1. *Key pre-distribution* : Pre-loading keys in sensor nodes prior to deployment. The keys present in a sensor node constitute the key ring of the sensor.

2. *Shared key discovery* : To find a common shared key between two communicating nodes.

3. *Path key establishment* : If a common key does not exists, then a path has to be found between the communicating nodes. A path key is then established between the communicating nodes.

Key pre-distribution can be of three types:

1. *Probabilistic* : Key ring of each node is made drawing keys from a key pool randomly.

2. *Deterministic* : Key ring of each node is made following some definite pattern.

3. *Hybrid* : Makes use of the above two approaches.

Our approach to key pre-distribution is based on deterministic algorithm.

1.3 Design Theory

As most of the deterministic algorithms use design theory, in this section we briefly discuss some of the design theories for the sake of completeness.

A set system or design [3] is a pair (X, A), where A is a set of subsets of X, called blocks. The elements of X are called varieties or elements. A Balanced Incomplete Block Design BIBD(v, b, r, k, λ), is a design which satisfy the following conditions:

1. $|X|= v$, $|A|= b$.

2. Each subset in A contains exactly k elements,

3. Each variety in X occurs in r blocks,

4. Each pair of varieties in X is contained in exactly λ blocks in A.

When $v = b$, the BIBD is called a symmetric BIBD (SBIBD) and denoted by SB$[v, k, \lambda]$.

An association scheme with m associate classes on the set X is a family of m symmetric anti-reflexive binary relations on X such that:

1. any two distinct elements of X are i-th associates for exactly one value of i, where $1 \le i \le m$.

2. each element of X has n_i i-th associates, $1 \le i \le m$.

3. for each i, $1 \le i \le m$, if x and y are i-th associates, then there are ${p^i}_{jl}$ elements of X which are both j-th associates of x and l-th associates of y. The numbers v, n_i $(1 \le i \le m)$ and ${p^i}_{jl}$ $(1 \le i, j, l \le m)$ are called the parameters of the association scheme.

A partially balanced incomplete block design with m associate classes, denoted by PBIBD(m) is a design on a v-set X, with b blocks each of size k and with each element of X being repeated r times, such that if there is an association scheme with m classes defined on X where, two elements x and y are i-th $(1 \le i \le m)$ associates, then they occur together in λ_i blocks. We denote such a design by PB$[k, \lambda_1, \lambda_2, ..., \lambda_m; v]$.

Let X be a set of varieties such that
$X = \cup_{i=1}^{m} G_i$, $|Gi|= n$ for $1 \le i \le m$, $G_i \cap G_j = \emptyset$ for $i \ne j$.

The G_i s are called groups and an association scheme defined on X is said to be group divisible if the varieties in the same group are first associates and those in different groups are second associates.

A transversal design TD$(k, \lambda; r)$, with k groups of size r and index λ, is a triple (X, G, A) where

1. X is a set of kr elements (varieties).

2. $G = (G_1, G_2, ..., G_k)$ is a family of k sets (each of size r) which form a partition of X.

3. A is a family of k-sets (or blocks) of varieties such that each k-set in A intersects each group G_i in precisely one variety, and any pair of varieties which belong to different groups occur together in precisely λ blocks in A.

1.4 Motivation

We have already told that key pre-distribution in sensor network is a challenging task. Eschenaur and Gligor [4] was the first to address a probabilistic solution to this problem. Then Camtepe and Yener [5, 6], Lee and Stinson [7, 8] and many others proposed deterministic solution to this problem with the help of design theory. Ruj and Roy [9] was the first to provide a solution using Reed-Solomon code. They first use the coding theory as a deterministic solution to key pre-distribution. Our motivation was to check with other coding techniques to distribute the keys so that we can get better resiliency. We have got better resiliency using BCH code. Our approach is also scalable. Our motivation for proposing the key revocation method was to design a distributed approach so that all the compromised nodes as well as all the compromised keys can be removed.

1.5 Thesis Organization

The organization of rest of the thesis and a brief outline of the chapters in this thesis is as follows.

In chapter 1 we have described about sensor network, its application and characteristics, key pre-distribution problem, its different aspects and the motivation behind our thesis.

In chapter 2 some related works on key pre-distribution and key revocation and their merits and demerits have been discussed.

In chapter 3 we have described our proposed approach on key pre-distribution. We have used BCH code for key pre-distribution and map the BCH code to key identifier and the key corresponding to each key identifier are installed into the sensor nodes before deployment. Then we have showed the scalability of our approach and analysis and comparison with some existing approach.

In chapter 4 We have described our approach to key revocation. We have shown some problems with the distributed approach of Chan *et. al.* [2] and proposed two new distributed approach which overcome those problems. We analyze our technique in terms of computational and communicational costs.

We conclude our thesis in chapter 5.

Chapter 2

Background

Background of Key Pre-distribution Schemes
Background of Key Revocation Schemes

Chapter 2

Background

This chapter is divided into two parts. In the first part we have presented a brief literature survey of different key pre-distribution schemes. In the second part we have discussed about various key revocation schemes proposed so far.

2.1 Background of Key Pre-distribution Schemes

All the key pre-distribution schemes can be divided into three according to the way of choosing keys for each node from the key pool. They are :

1. Probabilistic : Keys are drawn randomly and placed into the sensors.
2. Deterministic : Keys are drawn based on some definite pattern.
3. Hybrid : Makes use of both the above techniques.

To discuss about the schemes in a better way we have divided them into some parts and we have discussed below about each part in respective subsections.

2.1.1 Basic Schemes

First we will discuss about two basic schemes which though were not meant for wireless sensor networks, but they have been used in context of wireless sensor networks. Those two schemes are Blom's scheme and Blundo *et. al.* 's scheme.

Blom [10] proposed a key pre-distribution scheme that allows any two nodes of a group to find a pairwise key. The security parameter of the scheme is c, i.e., as long as no more than c nodes are compromised, the network is perfectly secure. They have used one public matrix and one secret symmetric matrix to construct this scheme. Each node will have the share of those matrix such that any two nodes can calculate a common key between them without knowing each other's secret matrix share. The problem with this scheme is that if more than c number of nodes are compromised, the whole network will be compromised.

In the scheme proposed by Blundo, Santis, Herzberg, Kutten, Vaccaro, Yung [11], they used a symmetric bivariate polynomial over some finite field GF(q). Symmetric bivariate polynomial is a polynomial P$(x, y) \in$ GF (q)[x, y] with the property that P (i, j) = P (j, i) for all $i, j \in$ GF (q). A node with ID U_i stores a share in P, which is an univariate polynomial $f_i(y)$ = P(i, y). In order to communicate with node U_j , it computes the common key $K_{ij} = f_i(j) = f_j(i)$; this process enables any two nodes to share a common key. If P has degree t, then each share consists of a degree t univariate polynomial; each node must then store the t + 1 coefficients of this polynomial. So, each node requires space for storing t + 1 keys. If an adversary captures s nodes, where $s \leq t$, then it can not get any information about keys established between uncompromised nodes. However, if it captures t + 1 or more nodes then all the keys of the network can be captured.

Now we will discuss about the basic schemes which were proposed for wireless sensor networks.

Eschenauer and Gligor first proposed a random key pre-distribution scheme [4] for wireless sensor networks. They divided the key pre-distribution mechanism into three steps: key pre-distribution, shared-key discovery and path-key establishment. In this approach, a key ring for a node containing some fixed number of keys are chosen randomly without replacement from a key pool of large number of keys. Each node is assigned a key ring.The key identifiers of a key ring and corresponding sensor identifiers are stored in a trusted controller node. Now a shared key may not exist between two nodes. In that case, if there exists a path of nodes sharing keys pairwise between those two nodes, they may communicate via that path. They have also shown that for a network of 10000 nodes, a key ring containing 250 keys is enough for almost full connectivity. When sensor nodes are compromised, key revocation is needed. For this a controller node broadcasts a revocation message containing the list of identifiers of keys which have been compromised and all the nodes after getting the message removes the compromised keys from the key ring. The main advantages of this scheme are that the scheme is flexible, scalable, efficient and easy to implement. However, the main disadvantages are that it cannot be used in regions which are prone to massive node capture attack.

Chan Perrig and Song [1] modified Eschenauer and Gligor scheme. According to their q-composite scheme two nodes must share at-least q number of keys to have a secure path between them. The path key will be formed by the hash of all the common keys. Though for small number of node capture, resiliency was improved, the resiliency was affected drastically as number of captured nodes increases.

2.1.2 Random Pairwise Scheme

In the random pairwise scheme, proposed by Chan, Perrig and Song [1], they have proposed that in a network of size N and minimum connection probability of two nodes is p, each node will store k number of keys where $k = N \times p$. The key pre-distribution, shared key discovery and path key establishment is done as in [4]. Node revocation for compromised nodes are done by voting of all the nodes in the network with a suitable threshold parameter. But the disadvantage of this scheme is that it is not scalable and choosing the threshold value for node revocation is very important as it can lead to other problems.

The pairwise key scheme of Liu and Ning [12] is based on the polynomial pool based key pre-distribution by Blundo *et. al.* [11]. They have shown the calculation for the probability that two nodes share a common key. They have also shown the probability that a key is compromised. Later it was extended in [13] where they modified the scheme into a hypercube based key pre-distribution.

Zhu, Xu, Setia and Jajodia [14] also proposed a random pairwise scheme based on probabilistic key sharing where two nodes can establish shared keys without the help of an online KDC and only knowing each other's key id. Communication overhead in this scheme is very low. But if any node in the path is compromised then the key establishment process has to be restarted.

2.1.3 Grid-based Key Pre-distribution Schemes

Chan and Perrig was the first to propose a grid based key pre-distribution scheme where they place all the nodes of a network in a square grid. The scheme was named as PIKE scheme [15]. In that scheme, each node will have a secret pairwise key with the nodes which lie in the same row or same column. So for a network of size N, each node has to store $2(\sqrt{N}-1)$ number of keys. If two nodes do not have any shared key, they will have exactly two intermediate nodes having shared key with both the nodes. Here any node can act as an intermediary. Hence, it reduces the battery drainage of the nodes near base station who have to serve as intermediary most of the time in other schemes. But the main disadvantage of this scheme is that it has high communication overhead. Because large number of key pairs will not have common key between them, path-key establishment will be very much time consuming.

In [16], Kalindi *et. al.* modified the PIKE scheme. They placed the nodes as well as the keys in a grid and divide the grid into some sub-grids. A node will have all the keys in its key chain which lie in its same row or column and which are in its same or neighboring sub-grids. Key needed to store in each node can be much less than [15] if number of sub-grids are more. It will increase the resiliency but decrease the connectivity. The reverse will happen if number of sub-grids is lesser. Nodes belonging to the same sub-grid and in same row or same column share more keys. But they are not allowed to use all the common keys because capturing of one node of a row or column will reveal all the keys of that row and column.

Sadi, Kim and Park [17] proposed another grid based random scheme based on bivariate polynomials. In this scheme, they will first arrange they nodes into a $m \times m$ square grid. After that some $2m\omega$ bivariate polynomials will be generated and they will be divided into some group such that each row and each column will be assigned one group of polynomials. A node then will select some 2τ number of polynomials from its row polynomial group and column polynomial group. If two nodes are in same row or in same column, they use a challenge response protocol to find whether they are sharing a common polynomial. If they a shared polynomial, they can setup a shared key. Otherwise they will have to go for path key establishment and they will have to find two other intermediate nodes such that a path can be established. In this case also the communication overhead is high.

Abedelaziz Mohaisen, YoungJae Maeng and DaeHun Nyang [18] proposed a 3-dimensional grid based key pre-distribution. According to their scheme, If the network size is N, then all the node of the network is arranged in a $m \times m \times m$ grid where $m = N^{\frac{1}{3}}$. Now $3N^{\frac{1}{3}}$ symmetric polynomials will be distributed among the nodes in such a way that all the nodes with the same axis value owns the share of same corresponding polynomial. Two nodes having same axis value will share common polynomial and key can be prepared from that. The probability of connectivity is $\frac{3}{m+1}$. Though the communication overhead is low in this scheme than the previous schemes, the resiliency is very poor.

2.1.4 Group Based Key Pre-distribution

Liu, Ning and Du observed that sensor nodes in the same group are usually close to each other and they proposed a group based key pre-distribution scheme without using deployment knowledge [19, 20]. They divide the nodes of a network into groups and then form cross groups taking exactly one sensor node from each group such that there will not be any common node between any two cross groups. They presented two instantiations of pre-distribution. In the first one, hash function were used. Two nodes will share a common key if they are in same group or in same cross group. If the number nodes in the network is N and they are divided into n groups each containing m nodes, $N = n \times m$ and each node need to store $\frac{m+n}{2}$ keys. In the second method, they used symmetric bivariate polynomials and assign a unique polynomial to each group and cross group. Every node will have share of the polynomials corresponding to their groups and cross groups. The advantages of this scheme are that it does not do not use deployment knowledge and give resiliency and connectivity similar to the deployment knowledge based schemes. The polynomial based schemes can be made scalable. The framework can be used to improve any existing pre-distribution schemes. The disadvantages of this scheme is that the probability of secure communication between cross-group neighbors is very less. The scheme is not suitable for networks which have small group size.

To overcome the problems of Liu et al's scheme [20], Martin Paterson and Stinson [21] proposed a group based design using resolvable transversal designs. To increase the cross group connectivity, they proposed that each node is contained in m cross groups rather than one. Though some additional storage is required. They did not give any algorithm for the construction of such designs.

Table 2.1: Various generalized quadrangles used by Camtepe and Yener and their different parameters

Design	s	t	v	b	k	r
GQ(q,q)	q	q	$(q+1)(q^2+1)$	$(q+1)(q^2+1)$	$q+1$	$q+1$
GQ(q,q^2)	q	q^2	$(q+1)(q^3+1)$	$(q^2+1)(q^3+1)$	$q+1$	q^2+1
GQ(q^2,q^3)	q^2	q^3	$(q^2+1)(q^5+1)$	$(q^3+1)(q^5+1)$	q^2+1	q^3+1

2.1.5 Key Pre-distribution Using Combinatorial Structures

In the schemes which use combinatorial structures, one of their greatest advantage is that almost all of them have efficient shared key discovery algorithm with which easily two nodes can find their common key.

Camtepe and Yener Scheme

Camtepe and Yener were the first to use combinatorial structures in key pre-distribution [5, 6]. They first used projective planes and then generalized quadrangles. A finite projective plane PG(2,q) (where q is a prime power) is same as the symmetric BIBD, BIBD$(q^2+q+1, q^2+q+1, q+1, q+1, 1)$. So, q^2+q+1 number of nodes can be accommodated in the network each node having $q+1$ number of

Table 2.2: Connection Probability and Resiliency(fail(1)) for different value of t In Camtepe and Yener scheme

	Connection Probability	Fail(1)
$t = 2$	1	$\frac{1}{p}$
$t = 3$	$\frac{q^2+3q+2}{2(q^2+q+1)}$	$\frac{3}{q(q+2)}$
$t = 4$	$\frac{2q^2+q+3}{3q^2+3}$	$\frac{3q^3-3q^2+13q-1}{4q^4+2q^3+6q^2}$
$t = 5$	$\frac{5q^4+10q^3+7q^2+10q8}{8(q^4+q^3+q^2+q+1)}$	$\frac{8q^5+18q^4-26q^3+73q^2-15q+2}{15q^6+15q^5+6q^4+24q^3}$
$t = 6$	$\frac{19q^4+16q^3+19q^2+6q+30}{30(q^4+q^2+1)}$	$\frac{45q^7+30q^6+145q^5-230q^4+511q^3-186q^2+51q-6}{4(19q^8+16q^7+19q^6+6q^5+30q^4)}$

keys. It ensures 100% connectivity. But the resiliency was very poor. Also for a network containing large number of nodes, storage requirement will be relatively large. For a network of size N, each node needs to store approximately $\sqrt{N}$ number of keys. To get better result, they used generalized quadrangles, GQ(s,t) where s and t are the two parameters of GQ. Three designs were used : GQ(q, q) was constructed from PG$(4, q)$, GQ(q, q^2) was constructed from PG$(5, q)$, GQ(q^2, q^3) was constructed from PG$(4, q^2)$. Camtepe and Yener have mapped these GQs in key pre-distribution $[5, 6]$ like this :
v = number of keys = (s + 1)(st + 1), b = number of nodes = (t + 1)(st + 1), r = number of keys in each node = (s + 1), and k = key chains that a key is in = (t + 1) for all the three GQs, these parameters are given in Table 2.1. Here q is taken as any prime or prime power.

Probability that two node will share a common key in these GQs are $\frac{t(s+1)}{(t+1)(st+1)}$. Though GQs do not give 100% connection probability, resiliency is much better than projective planes. Also the storage requirement in these cases is less than that of projective planes.

Lee and Stinson Scheme

Lee and Stinson [7] formalized the definitions of key pre-distribution schemes using set systems. They introduced the idea of common intersection designs [8]. They used block graphs for sensors and according to them, every pair of nodes can be connected by maximum of 2-hop path. They have shown that (v,b,r,k)-1 design or the (v,b,r,k) configuration have regular block graphs with vertex degrees maximized. So, connectivity will be largest in this case. So, they have used (v,b,r,k) configuration. In a (v,b,r,k) configuration having b-1 = k(r-1), all the nodes are connected to each other and it's same as projective planes. But for large network, the key-chain in each node will be large. So, they introduced μ-common intersection design. In that if two node's key chain, A_i and A_j are disjoint, then there will be at least μ number of nodes, A_h who has common keys with both A_i and A_j. So, $|A_h \epsilon A : A_i \cap A_h \neq \phi$ and $A_j \cap A_h \neq \phi| \geq \mu$. They have also used transversal design for key pre-distribution [7]. They have shown that for a prime number p and a integer k such that $2 \leq k \leq p$, there exists a transversal design TD(k,p). In that design, p^2 number of nodes can be arranged with k keys in each node in such a way that (i,j)th node will have the keys $(x, xi + j \bmod p) : 0 \leq x \leq k$. for $0 \leq i \leq p-1$ and $0 \leq j \leq p-1$. If two nodes want to find common keys between them they just need to exchange their node identifiers and the shared key algorithm complexity is $O(1)$. The communication overhead is $O(\log p) = O(\log \sqrt{N})$ where N is the size of the network. They also gave the estimate of probability of sharing a common key between two nodes and it is $p_1 = \frac{k(r-1)}{b-1}$ where k is the keys per node, r is the number of nodes a key is in and b is the total number of nodes in the network. The estimate for resiliency for s node capture is fail(s) $= 1 - (1 - \frac{r-2}{b-2})^s$. A multiple space has also been presented by Lee and Stinson in [22].

Chakraborty *et. al.* Scheme

Chakrabarti, Maitra and Roy [23, 24] proposed a hybrid key pre-distribution scheme by merging the blocks in combinatorial designs. They first showed that if one considers 4 Kbytes of memory space for storing keys in a sensor node, then choosing 128-bit key (16 byte), it is possible to accommodate 256 many keys. So, storage capacity can be increased in order to increase resiliency of the network. They considered Lee and Stinson construction and randomly selected some fixed number of blocks and merged them to form key chains. Though their proposed scheme increased the number of keys per node, it improved the resiliency than Lee and Stinson's Scheme [7].

Dong *et. al.* Scheme on 3-design

Dong et al in [25] proposed a scheme based on 3-design. If q is a prime power, then there exists a 3-$(q^n+1, q+1, 1)$ design with number of blocks $= \frac{q^n(q^{2n}-1)}{q(q^2-1)}$ for $n \geq 2$. They actually used a 3-$(q^2+1, q+1, 1)$ design with $n = 2$. In that, q^3+q number of nodes can be accommodated in the network with each node having $q+1$ number of keys. Maximum number of keys shared between any two nodes is two. Connectivity of this scheme is $\frac{\frac{1}{2}q^3+\frac{3}{2}q^2-1}{q^3+q^2-1}$. Resiliency of this scheme for a single node capture is $\frac{3q^2+q-2}{q^4+2q^3-q^2+2q-2}$. From the above formulas, we can tell that for large value of q, connectivity is almost $\frac{1}{2}$ and resiliency for a single node capture is almost zero. But the biggest disadvantage of this scheme is that the resiliency reduces drastically as the number of compromised nodes increases.

Dong *et. al.* on Orthogonal Arrays

Dong et al have proposed a key pre-distribution scheme based on orthogonal array. [26]. In that work, they have taken orthogonal array of index one and used Bush's construction for orthogonal array construction. from that construction, for any prime p we will get a network containing key pool size of p^2+1, number of nodes $= p^t$ for an integer t and number of keys per node $= p+1$. The authors have shown the connection probability and resiliency for different value of t. We present it in Table - 2. They have shown that when the value of t tends to infinity, the connection probability is almost 0.632121 and when the value of p tends to infinity, value of fail(1) tends to zero. They have compared their scheme with Lee and Stinson's scheme and shown that it has better connection probability and resiliency than Lee and Stinson's scheme [7]. Also, choosing a correct value of t, one can get better connectivity than [25].

Ruj and Roy Scheme

Ruj and Roy proposed a scheme using triangular PBIBD [27] and they found that for a network of size N, only about $O(\sqrt{N})$ keys per node is needed and they got a highly connected resilient and scalable network.

2.1.6 Key Pre-distribution Using Deployment Knowledge

Location dependent key pre-distribution were first proposed by Liu and Ning [28]. They proposed two schemes taking advantage of the location information. According to them, as sensors are deployed in group, nodes in the same group have higher probability of being deployed close to each other. In their first scheme, i.e., closest pairwise scheme, they proposed that a node will have pairwise keys with the nodes which are close to each other. In the second scheme, they used polynomial based key pre-distribution like [11]. They divide the nodes in groups and assign each group a unique symmetric bivariate polynomial. A node will have share of polynomials of its own group as well as its four neighbor groups. Common key can be calculated between the nodes who are in the same or neighboring groups like [11].

Du et al proposed a key pre-distribution scheme using deployment knowledge in [29]. which they extended in [30]. This scheme is based on grid group deployment scheme where sensor nodes are deployed in groups such that a group of sensors are deployed in a single deployment point. The deployment model was given in [30]. They used Blom's scheme [10] for key pre-distribution in [31, 32]. But they modified it into multiple key spaces. In their deployment scheme, If two groups are neighbors, then their will be some amount of overlap between their respective key pools, i.e., they will have some number of common keys in their key pools. But if two groups are far away from each other, then the overlap will decrease and it can be even zero. This scheme uses less number of keys and gives higher connectivity and better resiliency. But the complexity of this scheme is its main disadvantage.

Yu and Guan [33, 34] proposed a key pre-distribution scheme using deployment knowledge and compared the effect of having triangular, hexagonal and square grids. They showed that the hexagonal grids are giving better performance in case of both connectivity and resiliency. They used Blom's scheme for key pre-distribution. They divided the nodes into groups and placed them in a grid according to deployment knowledge. A public matrix is generated for all the groups and some private matrices are generated for each group. Each node will have their share from the public matrix as well as from their respective group's private matrix. That will help the nodes in the same groups to make a common key. For communication between nodes of neighbor groups, they declared some groups as basic groups and assign each of them one unique private matrix to them. Non basic groups will have all the matrices of their neighboring basic groups. Nodes of each group will have share of its own group's matrices. Any two neighboring groups will have common private matrix. So, any two nodes from two neighboring groups can establish a key with the help of that private matrix. So, this scheme produces a high connectivity between neighboring nodes.

Huang, Mehta, Medhi and Harn [35] proposed a grid-group based key predistribution scheme. This scheme is perfectly secure to random node capture as well as perfectly secure to selective node capture. Their approach is similar to Du et al using multiple space Blom's scheme.

Simonova, Ling and Wang discussed a homogeneous scheme in [36]. According to them, each grid in the network will have a disjoint key pool. Nodes from the same grid will communicate via this. There will another key pool called deployment key pool which will be constructed from neighboring key pools. Nodes from two neighboring grid can communicate via keys of the deployment key pool. Zhou, Ni and Ravishankar was first to propose a key pre-distribution scheme in [37] where sensors are mobile.

2.2 Background of Key Revocation Schemes

Key revocation problem was first addressed by Eschenaur and Gligor [4]. They proposed a centralized approach to key revocation in which a controller node broadcast a message to each node in the network informing about the compromised keys. In this scheme a signature key needs to be sent to each node a priori to the broadcast message.

A distributed approach to key revocation was first proposed by Chan et al in [1] and later it was extended by them in [2]. They have used distributed voting mechanism with the help of polynomial secret sharing. Their proposed scheme can revoke the compromised node but may not fully revoke the compromised keys of that node.

Wang *et. al.* [38] proposed a key updating techniques to obsolete the keys used by the compromised nodes. They first proposed the idea of sessions to be used. They divided the total life span of a network into some sessions. A session key is broadcast at the beginning of each session to update the keys in each node in such a way that compromised nodes can not get this session key and their keys will become obsolete. This process was a centralized one.

Park *et. al.* [39] proposed the idea of dynamic session to reduce the life time of a compromised node in the network. They modified the centralized scheme of Wang *et. al.* [38] and they proposed the duration of each session not to be fixed. It reduces the time a compromised node can stay in the network.

Moore *et. al.* [40] proposed a suicide strategy to revoke compromised nodes in the network. In their strategy, in order to revoke a compromised node, a legitimate node has to die. This is an overhead of this strategy.

Chapter 3

Key Pre-distribution Using BCH Code

Terms and Definitions

Key Pre-distribution Using BCH Code

Shared Key Discovery and Path Key Establishment

Scalability of The Scheme

Result and Comparison with Other Schemes

Conclusion

Chapter 3

Key Pre-distribution Using BCH Code

In this chapter, we discuss our proposed key pre-distribution mechanism using BCH code. We have mapped the BCH code to key identifier and the key corresponding to each key identifier are installed into the sensor nodes before deployment. We have found that our proposed scheme has a better resiliency and required the same or less number of keys to be stored in each sensor for a given number of nodes than the existing well known schemes. Our proposed scheme is also scalable too, in the sense, the addition of new nodes into the network does not require alteration, addition or modification of keys in the nodes present in the network.

The rest of the chapter is organized as follow.
In section 3.1 we have discussed few terms and definitions for the better understanding of code as well as BCH code and its characteristics. Section 3.2 discusses the detail key pre-distribution algorithm along with example. In section 3.3 we have discussed about the shared key discovery and path key establishment phase. Scalability of our scheme has been shown with a suitable example in section 3.4. In section 3.5 we have shown the results and compare them with that of some existing schemes. We have conclude the chapter in section 3.6.

3.1 Terms and Definitions

In this section we briefly discuss the BCH code along with the related terminologies and definitions for the sake of completeness.

A *code* is a pair (Q,C) such that the following properties are satisfied [3] :

(i) Q is a set of symbols.

(ii) C is a set of d-tuples of symbols called *codeword* where $d \geq 1$ and d is an integer.

A *code* is said to be a *linear code* if it posses the following properties:

(i) Sum of any two codewords belonging to same code is also a valid codeword belonging to that code,

(ii) All zero codeword is always a valid codeword, and

(iii) Minimum hamming distance will be the minimum weight of any non zero codeword.

A *code* is said to be a *cyclic code* if it is *linear* and any cyclic shift of a codeword is also a codeword belonging to the same code.

BCH code [41] is a cyclic linear block code, constructed from an alphabet set P. Length of the codewords are n = p^m - *1* where m is an integer and $|P|= p$. A *generator polynomial g(x)* is used to derive the codewords. Total number of possible codewords for an alphabet set P is p^k where $k = n - deg(g(x))$. Here *deg (g(x))* represents the highest degree of x in the generator polynomial *g(x)*. The process of finding the generator polynomial for a particular code is described in Section 4.

Galois field, GF, is a field with finite number of elements. GF(q) has q number of elements. A GF of order q^m, that is GF(q^m), can be constructed from GF(q)

where m is an integer. In such cases, GF(q) is called *base field* and GF(q^m) is called *extension field*.

Primitive polynomial f(x) over any *Galois field* is a prime polynomial over that GF with the property that in the extension field constructed from modulo *f(x)*, every element of the extension field except *zero* can be expressed as a power of x.

If GF(q) is the base field and GF(q^m) is the extension field, then $x^n - 1$ can be factorized over GF(q) where $n = q^m - 1$. Let say, $x^n - 1 = f_1(x)f_2(x)$ ....$f_p(x)$. In the extension field, $x^n - 1 = \Pi(x - \beta_j)$ where β_j are all the non-zero elements of GF(q^m). Here, we can say that each β_j is a solution of exactly one of the $f_i(x)$. This $f_i(x)$ is called the *minimal polynomial* of the corresponding β_j.

Set of elements in the extension field sharing the same minimal polynomial over base field are called *conjugates* with respect to GF(q).

If *f(x)* is the minimal polynomial of an element,say β, in the extension field then the conjugate set including β will be $(\beta, \beta^q, \beta^{q^2}, ..., \beta^{q^{r-1}})$ where $\beta^{q^{r-1}} = \beta$ for any integer r.

3.2 Key Pre-distribution Using BCH Code

In this section, we explain the key pre-distribution using Bose-Chaudhuri-Hocquenghem (BCH) codes. The proposed scheme is scalable. Key pre-distribution in the proposed scheme is carried out in two phases. First phase consists of the construction of BCH codewords. In the second phase, we derive the key identifiers for each sensor from the BCH codeword. Each node is represented by means of a unique node polynomial derived from GF(p). The codeword is obtained in the First phase. Then in the second phase identifiers for each node is derived from the codeword obtained in the First phase. We describe below the two phases in key pre-distribution.

First Phase : The following steps are carried out in this phase to construct

BCH codewords.

Step 1 : Choose the length of the codeword, n, such that $n = p^m - 1$ where p is a prime or a prime power, and m is an integer.

Step 2 : Choose a primitive polynomial over GF(p) of degree m and construct GF(p^m).

Step 3 : Find the set of conjugates from the elements of the GF(p^m). For each set of conjugates find the minimal polynomial corresponding to that set.

Step 4 : Choose a value, t, which is the maximum number of errors BCH code can correct. The generator polynomial for the codewords is given by

$$g(x) = LCM[f_1(x), f_2(x), ..., f_{2t}(x)]$$

where $f_i(x)$ is the minimal polynomial of the i^{th} element of GF(p^m). Compute the value of $k = n - deg(g(x))$. Maximum number of nodes in the network will be p^k. The value of t is chosen in such a way that p^k will cover the network size.

Step 5 : To obtain the polynomial of individual codewords, multiply each of the p^k number of node polynomials of degree *k-1* in GF(p) with the generator polynomial. Here, each p^k number of node polynomials means all the polynomials of degree *k-1* whose coefficients are from GF(p).

Second Phase : In this phase we derive the key identifiers for each sensor from the codewords formed in first phase. The codeword for each node derived in the First phase is mapped to key identifiers, which will identify the keys to be assigned to the sensor node. There is a unique key corresponding to each key identifier. We derive n key identifiers from codeword $(a_1, a_2, a_3, ..., a_n)$ where each identifier corresponds to an alphabet a_j for $1 \leq j \leq n$. Each key identifier is a triplet, consisting of (a_j, j, s) where j = 1,2,......,n and s is the relative position of appearance of the alphabet a_j in the codeword, *i.e.*, s = (number of time a_j occurred in the codeword before the current occurrence + 1) mod p^{m-1}.

We have shown the mapping from BCH code to key pre-distribution in Table 3.1. Each key identifiers are denoted by $(a_j,\ j,\ s)$ where, $0 \leq a_j \leq p-1$,
$1 \leq j \leq p^m - 1$ and
$1 \leq s \leq j$ for $j \leq p^{m-1} - 1$ and
$0 \leq s \leq p^{m-1} - 1$ for $j \geq p^{m-1}$
The number of keys in the key pool will be
$p \times (1 + 2 + ... + p^{m-1}) + P \times p^{m-1} \times (p^m - 1 - p^{m-1})$
$= \frac{p \times p^{m-1} \times (p^{m-1}+1)}{2} + p^m \times (p^m - p^{m-1} - 1)$
$= p^m \times (p^m - \frac{p^{m-1}}{2} - \frac{1}{2})$

Number of nodes in the network is p^k and number of keys per node is $p^m - 1$.

Key pre-distribution	Our construction
Key pool set (P)	$(a_j,\ j,\ s)$ where, $o \leq a_j \leq p-1$, $1 \leq j \leq p^m - 1$ and $1 \leq s \leq j$ for $j \leq p^{m-1} - 1$ $0 \leq s \leq p^{m-1} - 1$ for $j \geq p^{m-1}$
Key pool size($\|P\|$)	$p^m \times (p^m - \frac{p^{m-1}}{2} - \frac{1}{2})$
Number of key chains(|N|) *i.e.*, number of Nodes in network	p^k
Number of keys per node(k)	$p^m - 1$

Table 3.1: Mapping of parameters between key pre-distribution and their corresponding parameters from BCH codes

Example : We illustrate below the generation of BCH code and its mapping to key identifiers through an example.

First Phase :

Step 1 : Consider $p = 2$ and $m = 3$. The value of n is computed to be 7.

Step 2 : There are two primitive polynomials over GF(2) of degree 3. One is $P(z) = z^3+z+1$ and another is $P(z) = z^3+z^2+1$. We randomly choose one primitive polynomial. In this example we consider the polynomial $P(z) = z^3 + z + 1$. Then we construct GF(2^3) as follows :

$\beta^1 = z$

$\beta^2 = z^2$

$\beta^3 = z^3 = z + 1$

$\beta^4 = z^4 = z^2 + z$

$\beta^5 = z^5 = z^3 + z^2 = z^2 + z + 1$

$\beta^6 = z^6 = z^3 + z^2 + z = z^2 + 1$

$\beta^7 = z^7 = z^3 + z = 1$

Step 3 : The conjugate sets and their corresponding minimal polynomials are given in Table 3.2.

Conjugate sets	Minimal polynomial
$\beta^1, \beta^2, \beta^4$	$(x^3 + x + 1)$
$\beta^3, \beta^6, \beta^5 (= \beta^{12})$	$(x^3 + x^2 + 1)$
β^7	$(x - 1)$

Table 3.2: Conjugate sets and their corresponding minimal polynomials

Step 4 : We consider the value of $t = 1$. The generator polynomial $g(x) =$ LCM[(minimal polynomial of β^1) , (minimal polynomial of β^2)]
$= \text{LCM}[(x^3 + x + 1), (x^3 + x + 1)]$
$= (x^3 + x + 1)$.
Value of $k = 7 - 3 = 4$. Therefore, the number of nodes in the network is p^k $= 2^4 = 16$.

Step 5 : Each node have corresponding node polynomial of degree 3 in GF(2). We obtain the code polynomial for each node by multiplying the node polynomial for that node with the generator polynomial $x^3 + x + 1$. Codeword for each node is obtained from their respective code polynomial. Node ID, node polynomial, code polynomial and their corresponding codeword for the *Sixteen* nodes that we have considered in our example is shown in Table 3.3.

Second Phase : In this phase we derive the key chain for a node from its codeword. For example the key identifiers corresponding to Node ID 1 is shown in Table 3.4.

Node ID	Node Polynomial	Code Polynomial	Code Representation
0	0	0	0000000
1	1	$x^3 + x + 1$	0001011
2	x	$x^4 + x^2 + x$	0010110
3	x + 1	$x^4 + x^3 + x^2 + 1$	0011101
4	x^2	$x^5 + x^3 + x^2$	0101100
5	$x^2 + 1$	$x^5 + x^2 + x + 1$	0100111
6	$x^2 + x$	$x^5 + x^4 + x^3 + x$	0111010
7	$x^2 + x + 1$	$x^5 + x^4 + 1$	0110001
8	x^3	$x^6 + x^4 + x^3$	1011000
9	$x^3 + 1$	$x^6 + x^4 + x + 1$	1010011
10	$x^3 + x$	$x^6 + x^3 + x^2 + x$	1001110
11	$x^3 + x + 1$	$x^6 + x^2 + 1$	1000101
12	$x^3 + x^2$	$x^6 + x^5 + x^4 + x^2$	1110100
13	$x^3 + x^2 + 1$	$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$	1111111
14	$x^3 + x^2 + x$	$x^6 + x^5 + x$	1100010
15	$x^3 + x^2 + x + 1$	$x^6 + x^5 + x^3 + 1$	1101001

Table 3.3: Node ID along with its corresponding node polynomial, code polynomial and codeword for *Sixteen* number of nodes

Codeword	0	0	0	1	0	1	1
Key identifiers	(0,1,1)	(0,2,2)	(0,3,3)	(1,4,1)	(0,5,0)	(1,6,2)	(1,7,3)

Table 3.4: Key Identifiers for Node ID 1

After obtaining the key identifiers for a node the keys corresponding to the key identifiers are installed in the node before deployment. Key identifiers of all the *Sixteen* nodes considered in our example are shown in Table 3.5.

3.3 Shared Key Discovery and Path-Key Establishment Phase

In the shared key discovery phase, every node will broadcast their key identifiers to all its neighbor nodes. Each neighbor nodes, after getting the key identifiers of a particular node will match with its own key identifiers. The keys corresponding to the matched key identifier will be the shared key between those two nodes.

For example Node 11 and Node 12 have a common key identifier (1,1,1). The key corresponding to the key identifier (1,1,1) is used as the shared key for the communication between them.

After the shared key discovery phase, if two neighboring nodes find no shared key between them, then they will find a third node who is connected to both the

Node ID	Key Identifiers
0	(0,1,1),(0,2,2),(0,3,3),(0,4,0),(0,5,1),(0,6,2),(0,7,3)
1	(0,1,1),(0,2,2),(0,3,3),(1,4,1),(0,5,0),(1,6,2),(1,7,3)
2	(0,1,1),(0,2,2),(1,3,1),(0,4,3),(1,5,2),(1,6,3),(0,7,0)
3	(0,1,1),(0,2,2),(1,3,1),(1,4,2),(1,5,3),(0,6,3),(1,7,0)
4	(0,1,1),(1,2,1),(0,3,2),(1,4,2),(1,5,3),(0,6,3),(0,7,0)
5	(0,1,1),(1,2,1),(0,3,2),(0,4,3),(1,5,2),(1,6,3),(1,7,0)
6	(0,1,1),(1,2,1),(1,3,2),(1,4,3),(0,5,2),(1,6,0),(0,7,3)
7	(0,1,1),(1,2,1),(1,3,2),(0,4,2),(0,5,3),(0,6,0),(1,7,3)
8	(1,1,1),(0,2,1),(1,3,2),(1,4,3),(0,5,2),(0,6,3),(0,7,0)
9	(1,1,1),(0,2,1),(1,3,2),(0,4,2),(0,5,3),(1,6,3),(1,7,0)
10	(1,1,1),(0,2,1),(0,3,2),(1,4,2),(1,5,3),(1,6,0),(0,7,3)
11	(1,1,1),(0,2,1),(0,3,2),(0,4,3),(1,5,2),(0,6,0),(1,7,3)
12	(1,1,1),(1,2,2),(1,3,3),(0,4,1),(1,5,0),(0,6,2),(0,7,3)
13	(1,1,1),(1,2,2),(1,3,3),(1,4,0),(1,5,1),(1,6,2),(1,7,3)
14	(1,1,1),(1,2,2),(0,3,1),(0,4,2),(0,5,3),(1,6,3),(0,7,0)
15	(1,1,1),(1,2,2),(0,3,1),(1,4,3),(0,5,2),(0,6,3),(1,7,0)

Table 3.5: Key Identifiers of all the *Sixteen* nodes considered in the example

nodes and will establish the path key between the two nodes. For example, let A and B be the first two nodes and C be the third node. If k_1 is the key between A and C and k_2 is the key between B and C, then C will send k_1 key to B encrypting it with k_2, and will send k_2 key to A encrypting it with k_1. Both node A and B will create a secret key $K_{AB} = h(k_1, k_2)$ and erase k_1 and k_2 from their memory. The key K_{AB} is used as the secret key between A and B.

3.4 Scalability of the Scheme

The number of nodes N, that is addressable in the proposed scheme is p^k. For addition of nodes into the network, we need to generate the codeword for the nodes to be added. To generate the codeword for the nodes to be added, all the k degree polynomials in GF(p) are multiplied with the generator polynomial, *g(x)*. All the polynomials of k degree whose coefficients are from GF(p) are taken into account except those polynomials whose coefficient of x^k is *zero*. We are not considering polynomials whose coefficient of x^k is *zero* because a polynomial of degree k in GF(p) whose coefficient of x^k is *zero* is nothing but a $k-1$ degree polynomial in GF(p). The codeword for the new nodes will be of $n+1$ bits. However, the codeword for the existing nodes are of n bits. We perform circular right shift of

Node ID	Node Polynomial	Codeword Polynomial	Code after circular left shift
16	x^4+0	$x^7+x^5+x^4$	01100001
17	x^4+1	$x^7+x^5+x^4+x^3+x+1$	01110111
18	x^4+x	$x^7+x^5+x^2+x$	01001101
19	x^4+x+1	$x^7+x^5+x^3+x^2+1$	01011011
20	x^4+x^2	$x^7+x^4+x^3+x^2$	00111001
21	x^4+x^2+1	$x^7+x^4+x^2+x+1$	00101111
22	x^4+x^2+x	x^7+x^3+x	00010101
23	x^4+x^2+x+1	x^7+1	00000011
24	x^4+x^3	$x^7+x^6+x^5+x^3$	11010001
25	x^4+x^3+1	$x^7+x^6+x^5+x+1$	11000111
26	x^4+x^3+x	$x^7+x^6+x^5+x^4+x^3+x^2+x$	11111101
27	x^4+x^3+x+1	$x^7+x^6+x^5+x^4+x^2+1$	11101011
28	$x^4+x^3+x^2$	$x^7+x^6+x^2$	10001001
29	$x^4+x^3+x^2+1$	$x^7+x^6+x^3+x^2+x+1$	10011111
30	$x^4+x^3+x^2+x$	$x^7+x^6+x^4+x$	10100101
31	$x^4+x^3+x^2+x+1$	$x^7+x^6+x^4+x^3+1$	10110011

Table 3.6: Symbolic Code Representation and Code polynomial for all the newly added Nodes

each newly formed codeword to get the desired codeword so that we can match the key identifiers for the same power of x in the code polynomials for any two nodes. The number of new nodes that can be added into the network is $p^{k+1}-p^k$. The number of keys to be installed in the new nodes will be *n+1*, whereas the number of keys in the existing nodes will remain at *n*. We can add new nodes into the network without changing the keys of the existing nodes. Thus the scheme is scalable.

We explain the scalability by means of an example. Suppose a network has *Sixteen* number of nodes and the key identifiers associated with each node in the network is shown in Table 3.5. Number of keys installed at each node is *Seven.*

Suppose we want to add *sixteen* more nodes to the existing network. Table 3.6 shows the codeword for the newly added nodes and Table 3.7 shows the key identifiers to be installed in each node. The number of keys to be installed in the new nodes is *Eight*, shown in Table 3.7 whereas there are *Seven* keys in the existing nodes, shown in Table 3.5. No changes has taken place in the number of keys of existing nodes. This shows the scalability of the scheme. Next we show that a node in the existing network can perform a secure communication with newly added nodes.

Node ID	Key Identifiers
16	(0,1,1),(1,2,1),(1,3,2),(0,4,2),(0,5,3),(0,6,0),(0,7,1),(1,8,3)
17	(0,1,1),(1,2,1),(1,3,2),(1,4,3),(0,5,2),(1,6,0),(1,7,1),(1,8,2)
18	(0,1,1),(1,2,1),(0,3,2),(0,4,3),(1,5,2),(1,6,3),(0,7,0),(1,8,0)
19	(0,1,1),(1,2,1),(0,3,2),(1,4,2),(1,5,3),(0,6,3),(1,7,0),(1,8,1)
20	(0,1,1),(0,2,2),(1,3,1),(1,4,2),(1,5,3),(0,6,3),(0,7,0),(1,8,0)
21	(0,1,1),(0,2,2),(1,3,1),(0,4,3),(1,5,2),(1,6,3),(1,7,0),(1,8,1)
22	(0,1,1),(0,2,2),(0,3,3),(1,4,1),(0,5,0),(1,6,2),(0,7,1),(1,8,3)
23	(0,1,1),(0,2,2),(0,3,3),(0,4,0),(0,5,1),(0,6,2),(1,7,1),(1,8,2)
24	(1,1,1),(1,2,2),(0,3,1),(1,4,3),(0,5,2),(0,6,3),(0,7,0),(1,8,0)
25	(1,1,1),(1,2,2),(0,3,1),(0,4,2),(0,5,3),(1,6,3),(1,7,0),(1,8,1)
26	(1,1,1),(1,2,2),(1,3,3),(1,4,0),(1,5,1),(1,6,2),(0,7,1),(1,8,3)
27	(1,1,1),(1,2,2),(1,3,3),(0,4,1),(1,5,0),(0,6,2),(1,7,1),(1,8,2)
28	(1,1,1),(0,2,1),(0,3,2),(0,4,3),(1,5,2),(0,6,0),(0,7,1),(1,8,3)
29	(1,1,1),(0,2,1),(0,3,2),(1,4,2),(1,5,3),(1,6,0),(1,7,1),(1,8,2)
30	(1,1,1),(0,2,1),(1,3,2),(0,4,2),(0,5,3),(1,6,3),(0,7,0),(1,8,0)
31	(1,1,1),(0,2,1),(1,3,2),(1,4,3),(0,5,2),(0,6,3),(1,7,0),(1,8,1)

Table 3.7: Key Identifiers of all the newly added *Sixteen* number of nodes in the network

Suppose an existing node, say Node 11, wants to communicate with a newly added node, say Node 27. First they will exchange their key identifiers. The key identifier (1,1,1) is found to be common between them. So, they will use the key corresponding to the key identifier (1,1,1) for secure communication between them. If no common key exists between them then they will create a secret key between them during the path-key establishment phase.

3.5 Result and Comparison with Other Schemes

In this section, we have shown the result of network resiliency and number of keys per node for different values of number of nodes in the network. Here, we have assumed the value of m to 2. We have compared our result with two existing schemes , Camtepe and Yener scheme [6] and Ruj and Roy scheme [9]. The metrics used for comparison is the number of node, the number of keys required per node, number of compromised nodes, and the resiliency.

We have shown the resiliency of our scheme against random node capture attack. It can be seen from Table 3.8 that our proposed scheme is more resilient.

Proposed Scheme				R R Scheme				C Y Scheme			
k	N	s	Fail(s)	k	N	s	Fail(s)	k	N	s	Fail(s)
24	625	5	0.175251	22	529	5	0.217391	24	553	5	0.198915
24	625	10	0.316493	22	529	10	0.434783	24	553	10	0.397830
48	2401	10	0.142349	48	2401	10	0.186564	48	2257	10	0.203810
48	2401	15	0.229743	48	2401	15	0.2761	48	2257	15	0.305715
48	2401	20	0.355067	48	2401	20	0.408163	48	2257	20	0.407621
80	6561	10	0.070326	80	6561	10	0.123456	80	6321	10	0.123398
80	6561	20	0.230650	80	6561	20	0.246913	80	6321	20	0.246796

Table 3.8: Comparison of proposed key pre-distribution scheme with Ruj and Roy (R R) scheme and Camtepe and Yener (C Y) scheme. Number of nodes in the network is N, keys per node is k, number of compromised node is s and resiliency is *Fail(s)*[*Fail(s)* is the probability of affected links due to the compromise of s nodes].

Unlike deterministic schemes mentioned in [42], our scheme is scalable. In our scheme, scalability can be increased without changing the keys of existing nodes of the network.

3.6 Conclusion

In this work, we have proposed a key pre-distribution scheme using BCH codes. In the proposed scheme, we have taken key pool based approach where key identifiers of each node will be taken from a pool of key identifiers. The advantage of the scheme over the other deterministic schemes is that in this scheme, new nodes can be added to the network without changing the configuration of keys of the existing nodes. Also by varying the value of the different parameters in the scheme we can setup different sizes of network based on the requirement. Varying the values of t, which is the number of errors BCH code can correct, we can accommodate desired number of nodes with some acceptable resiliency. In future we would like to use other coding scheme and see their performance.

Chapter 4

Proposed Key Revocation Scheme

Problems with Chan *et. al.* [1,2] Key Revocation Mechanism
Network Model and Assumptions
Proposed scheme
Analysis of the Proposed Scheme
Conclusion

Chapter 4

Proposed Key Revocation Scheme

In this chapter we discuss two proposed key revocation schemes with distributed voting procedure. In chapter 2 we have already told that Chan *et. al.* [1, 2] have proposed a distributed voting procedure. But we have pointed out some problems in that scheme and we have tried to overcome that with a new key revocation scheme using a novel distributed voting mechanism. The rest of the chapter is organized as follows.

In section 4.1 we have pointed out the problems with Chan et al's scheme. We have discussed the network model used for our proposed scheme in section 4.2. Two proposed algorithms have been discussed in detail in section 4.3. We have analyzed our scheme in section 4.4. We have concluded the chapter in section 4.5.

4.1 Problems with Chan *et. al.* [1, 2] Key Revocation Mechanism

In this section we discuss the problems associated with Chan *et. al.*'s [1,2] scheme.

1. It is possible to remove a compromised node from the network, however it may not be possible to remove all its keys from the network. We give a scenario to support our claim. Let us consider two nodes, $\boldsymbol{u}$ and $\boldsymbol{v}$, sharing a common key, say k_1. Suppose they are deployed far away from each other in the network such that they are not in the communication range of each other. Let there exists a node $\boldsymbol{w}$ which is in the communication range of $\boldsymbol{v}$ but not in the communication range of $\boldsymbol{u}$ and they share the common key, k_1. Nodes $\boldsymbol{v}$

and $\boldsymbol{w}$ can discover the common key k_1 between them and can communicate via this key k_1. Suppose node $\boldsymbol{u}$ gets captured then the key k_1 gets revealed. The neighbors of $\boldsymbol{u}$ do not share the key k_1 with $\boldsymbol{u}$. Therefore, they are unaware of the fact that the key k_1 has been compromised. Hence, $\boldsymbol{v}$ and $\boldsymbol{w}$ will not be informed. Now the adversary can use the key k_1 and decrypt all the messages between $\boldsymbol{v}$ and $\boldsymbol{w}$. In the preset scenario compromised keys are not removed completely from the network in Chan *et. al.* scheme, compromising the network security.

2. Sybil attack [43] is possible in their proposed scheme. A compromised node removed from the network is known only to the neighbor. Rest of the network is not aware of the node that is revoked from the network. Therefore, a clone can be deployed elsewhere in the network; resulting in a Sybil attack.

3. Path keys established through the compromised node is not revoked.

4. Each node has to store votes for all its neighbors before the deployment. For this to happen we need to know the network topology before the deployment which is not always possible.

4.2 Network Model and Assumptions

In this section we discuss the network model and assumptions considered for the proposed key revocation scheme.

We assume the network is divided into hexagonal regions such that there will not be more than k number of nodes in each region. Each region have a unique id $< i, j >$ where i and j are the row and column of that region. Regions are further divided into two types: basic region and non-basic region. A region $< i, j >$ is called a basic region *if* $[i\%2 = 0 \ \&\& \ j\%2 = 0 \ \&\& \ i\%4 \neq 0] \ || \ [i\%4 = 0 \ \&\& \ j\%2 = 1]$. Other regions are called non-basic regions. A non-basic regions will have atmost two basic regions in their neighborhood. Basic regions have a single unique trivariate polynomial, whereas a non-basic region will have atmost two trivariate polynomials assigned to them drawn from each of its neighborhood basic

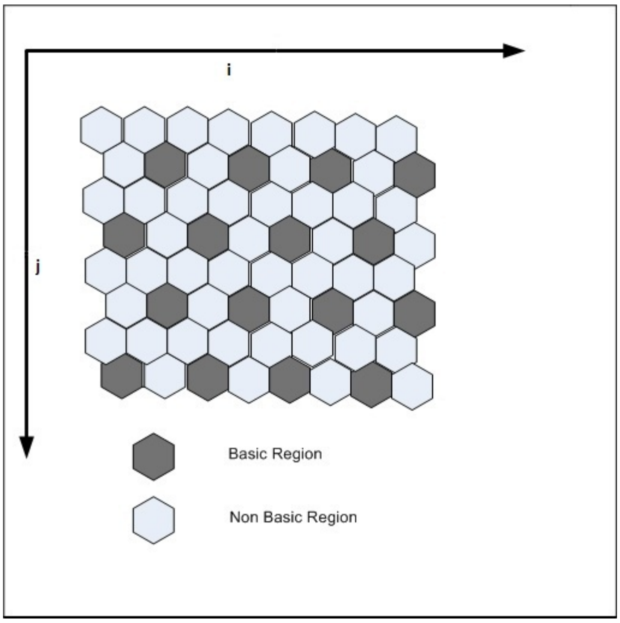

Figure 4.1: Division of network into basic and non-basic region

region. $f_1(x, y, z), f_2(x, y, z), ...f_n(x, y, z)$ are trivariate polynomials where highest degree for x, y, z will be $7k$. Figure-4.1 shows the division of network into two types of regions.

Each node will have a share of the polynomial(s) of it's region along with a unique hash function h(). If $f_1(x, y, z)$ is the polynomial for a region $< i, j >$, then for a node with node id u, the polynomial shares $f_1(u, y, z)$ is called authentication polynomial, $auth_u^1(y, z)$ and $f_1(x, u, z)$ is called the verification polynomial, $verf_u^1(x, z)$ for the node u.

We made the following assumptions :

1. Nodes have a unique identifier.

2. Each node have more than t numbers of neighborhood where t is the minimum number of neighbors who must agree to revoke a node.

3. Number of compromised node in a node's neighborhood is less than t.

4. Base station is not prone to compromise.

5. Nodes have built in intrusion detection system.

6. Each node maintains a two-hop neighborhood information.

7. Each node stores all the intermediary nodes for each path key formed.

8. The diameter of a region is greater than $2r$ where r is the radius of any node's communication range.

9. Each node maintains two lists; a Blacklist which contains the list of compromised nodes and a Suspected list which contains the list of suspected neighbor nodes along with the accuser.

Lemma 1 : **If two nodes are neighbors of a same node then they will be situated either in the same or in the neighboring regions.**

Proof : When two nodes are in the communication range of each other then they are in the neighborhood of each other. To become the neighbor of a node, distance between the two nodes should be less than or equal to r where r is the radius of node's communication range. Two nodes which are at a distance greater than r from each other can not be neighbor of each other. We have assumed that the diameter of a region to be greater that $2r$. Therefore, two nodes which are situated in non-neighboring regions, the distance between them will always be greater than $2r$. Hence, those two nodes can never be neighbors of a same node. Therefore we can conclude that nodes which are neighbors of a same node lies either in same region or in the neighboring regions.

4.3 Proposed Scheme

In this section we proposed two revocation schemes. First scheme is described in Subsection-4.3.1 and second in Subsection-4.3.2.

4.3.1 Scheme I

The proposed scheme consists of four phases. They are : *Setup, Voting, Revocation and Removal.* Action taken in each phase is described below.

1. *Setup :* In this phase, the network is divided into regions. The Blacklist and Suspected list at each node is set to empty.

2. *Voting :* In this phase a node vote against its suspected neighbors. If a node u wants to vote against one of its neighbor node, say, v, then u will prepare a message M containing the node id of the victim node v. This message is sent to all the neighbors of victim node v, and to the *base station.* Let w be one of the neighbors of v. Then according to Lemma 1, the nodes u and w are either from the same region or from the neighboring regions. Therefore, they have share of a unique polynomial. Let the polynomial be $f_i(x, y, z)$. Then u have $auth_u^i(y, z) = f_i(u, y, z)$ and w have $verf_w^i(x, z) = f_i(x, w, z)$. The node u will send the message M along with a single value $p = auth_u^i(w, h(M))$, *i.e.*, $p = auth_u^i(y, z)$ at $y = w$ and $z = h(M)$. After receiving the message, the node w will check the message's authentication. It will compute the value $q = verf_w^i(u, h(M))$, *i.e.*, $q = verf_w^i(x, z)$ at $x = u$ and $z = h(M)$. If $q = p$ then the message is authenticated. Node w updates its Suspected list by inserting the id of node v into the Suspected list if the list does not contain the id of node v. The name of the accuser that is node u in the present scenario is also inserted into the list.

   Each node maintains a counter on the number of votes registered against each of its neighbors. A node for which the number of votes registered against it crosses the threshold parameter, t, then that node is put under Blacklist.

When a node x puts a node y in the Blacklist, it performs the following actions:

(a) Stop communicating with y.

(b) Delete all the keys it shares with y.

(c) Delete all the path keys formed through y.

3. *Revocation :* The *base station*, on receiving t number of votes against a node, will prepare a key revocation message containing the node id of the compromised node and the compromised keys. Then it will broadcast this message, along with the authentication polynomials corresponding to polynomials assigned to each region. The *base station* will broadcast $\sum f_i(base.id, y, h(M_1))$ where *base.id* is the id of the base station, and $f_i(x, y, z)$ is the set of all the trivariate polynomials for $1 \leq i \leq n$ where n is the number of basic regions.

4. *Removal :* After receiving the key revocation message from the *base station*, a node will first check the message authenticity to ensure that it has come from the *base station*. A node l with region's polynomial $f_k(x, y, z)$ will compute $f_k(base.id, y, h(M_1))$ where $y = l$. It will also compute its own verification polynomial $verf_l^k(x, z)$ where $x = base.id$ and $z = h(M_1)$. If the above two values are equal, then the message is authenticated. Then the node l will delete all the keys mentioned in the message and put the victim node v in the Blacklist. Since, the list of compromised nodes exists in each node, this can prevent sybil attack and node replication attack.

4.3.2 Scheme II

This scheme also consists of four phases like Scheme-I. This differs from Scheme-I in the presence of monitor nodes in each region. Monitor nodes are more secured than the normal nodes and communicate directly among themselves. Actions performed at each step is explained below :

1. *Setup :* Action in this phase remain same as that in Scheme-I.

2. *Voting :* The mechanism of voting remains same as that in Scheme I. However, the vote is sent to the monitor node of the accuser's region. On receiving a vote, monitor node checks whether the suspected node belongs to its region or not. If not, then it will send this voting information to the monitor node of victims region. The monitor node of victim's region will update its Suspected list. When the number of votes reaches a threshold parameter t registered against a node, then the corresponding monitor node will inform other monitor nodes about the compromised node along with the keys that has been compromised.

3. *Revocation :* In this phase, the monitor node prepare a message containing the node id of the compromised node and the compromised keys. This message is sent to all the nodes in the monitor node's regions along with an authentication value for each node. For example, if the monitor node is m, sending an authentication message M_3 to node d then it will send an authentication value $f_k(m, d, h(M_3))$ where $f_k(x, y, z)$ is a tri-variate polynomial corresponding to the region of m.

4. *Removal :* After receiving key revocation message from their corresponding monitor node, a node checks its authenticity to ensure that it has come from its monitor node. If the node is l_1 then it will compute its own verification

polynomial $verf_l(x, z)$ where $x = m$ and $z = h(M_3)$. If this value is equal to the authentication value sent by the monitor node m, then the message is authenticated. Then the node, l, deletes all the keys contained in the message and put the accused node in the Blacklist.

4.4 Analysis of the Proposed Scheme

Our proposed scheme is an improvement over Chan *et. al.* scheme. It differs from their scheme in the following ways :

1. We have divided the network into hexagonal regions.
2. The idea of sessions is not used in our proposed scheme.
3. We have used trivariate polynomials whereas their voting procedure was based on secret sharing of bivariate polynomial.
4. We have used the concept of monitor node was also not present in their scheme.

In this section we analyze the proposed key revocation mechanism.

1. compromised nodes can not collude and revoke a node as there can not be more than t numbers of nodes in the neighborhood of a node.
2. To compute a trivariate polynomial, the adversary has to capture all the nodes having a share of that polynomial. Forgery of a vote is not possible

as there will not be more than k number of nodes in each region.

3. Votes are verified so that no false voting results in to revocation of a legitimate node. Neighbor nodes will not be able to authenticate the false vote by an adversary. Therefore, no revocation or updation of Suspected list will occur.

4. A listener can not replay a vote to generate additional votes.

5. The neighbor nodes do not broadcast the revocation message to the entire network. Thus, it is not vulnerable to denial of service attack.

6. As all the path keys constructed by the compromised nodes are removed after the revocation, the adversary can not affect the network computing the path keys at later stage.

7. The proposed scheme is resistant to Sybil attack or any other kind of replication attack.

In the proposed schemes, nodes need to store shares of atmost two trivariate polynomials of degree $7k$. Therefore, each node stores atmost four bivariate polynomials of degree $7k$. For each bivariate polynomial, a node has to store $(7k+1)^2$ number of values. Thus a node needs to store atmost $4 \times (7k+1)^2$ number of values.

For key revocation, we need to transmit a unique message authentication univariate polynomial of degree $7k$ for each region in Scheme I. Therefore, $n \times (7k+1)$ number of values need to be transmitted where n is the number of basic region in Scheme I. In scheme II only a single value needs to be transmitted.

Next, we calculate the computation needed for authentication of key revocation. In Scheme I, during verification, each node gets a univariate polynomial $f(y)$ of degree $7k$. Then, they compute the value of the polynomial $f(y)$ at $y =$ node *id*. This computation needs $(7k + (7k - 1) + (7k - 1) + ... + 1)$ number of multiplications $= 7k(7k + 1)/2$ number of multiplications and $7k$ number of addition. In Scheme II only a single value is transmitted, hence this computation is not required. Next step of verification is similar for both the schemes. Each node needs to compute its verification polynomial. Verification polynomial is a bivariate polynomial of degree $7k + 1$ and it is of the form $x^m(a_1y^m + a_2y^{m-1} + a_3y^{m-2} + ... + a_{m+1}) + x^{m-1}(b_1y^m + b_2y^{m-1} + b_3y^{m-2} + ... + b_{m+1}) + ...$ where $m = 7k + 1$.

Each term within the bracket is a univariate polynomial which needs $7k(7k+1)/2$ multiplications and $7k$ additions. Each of these values again needs to be multiplied by x of different power. Total number of multiplications is $= 7k(7k+1)/2 \times (7k + 1) + 7k(7k + 1)/2$

$= 7k(7k + 1)(7k + 2)/2$

Number of additions is $= 7k(7k + 1) + 7k$

$= 7k(7k + 2)$

We have shown the comparison of two of our schemes in Table 4.1.

4.5 Conclusion

We have proposed two key revocation model for wireless sensor network in this paper. In the Scheme I we overcome the problems of the existing algorithm and Scheme II have been introduced in order to further reduce the communication cost of Scheme I. In future, any other mechanism can be used for voting technique so further reduce the storage cost and time to revoke a compromised node. Also future improvements can be made in terms of reducing the computational and communication cost.

Table 4.1: Comparison between key revocation Scheme I and Scheme II

	Scheme I	Scheme II
Storage Requirement in each node	$(7k+1)^2$	$(7k+1)^2$
Number of values to be transmitted for authentication	$(7k+1)\times$ number of regions	1
Message authentication cost	polynomial computation: multiplications=$\frac{7k(7k+1)}{2}$ additions=$7k$ polynomial verification: multiplications=$\frac{7k(7k+1)(7k+2)}{2}$ additions=$7k(7k+2)$	polynomial computation: multiplications=0 additions=0 polynomial verification: multiplications=$\frac{7k(7k+1)(7k+2)}{2}$ additions=$7k(7k+2)$

Chapter 5

Conclusion and Future work

In this thesis we have discussed about the problem of key pre-distribution in wireless sensor networks and we have tried to find a solution for that in chapter 3. We have proposed a key pre-distribution mechanism with the help of BCH coding. We have got some better resiliency than some of the existing schemes. In future research can be done in order to find a suitable coding scheme which can increase the resiliency of the network. A coding technique can be found out with large number of codewords and large minimum distance so that it can be fitted to key pre-distribution. Except combinatorial designs, other designs like packing designs, cover free families and many more unexploited designs can be exploited. In our work we have concentrated on random node capture attack. In future research can be done so that selective node capture attack also can be taken care. We have also proposed a key revocation mechanism. It has overcome some of the problems existing distributed mechanism has. In future research can be done to reduce the communicational cost and computational cost further. Also a researches should be made so that revocation can be done without any help from the Base Station or monitor nodes.

One outcome of the thesis is the motivation to study designs and coding theory and find out new designs which have not been constructed so far. We hope that some new structures of the designs and coding schemes will help us in the betterment of the solution.

Bibliography

[1] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, page 197, Washington, DC, USA, 2003. IEEE Computer Society.

[2] Haowen Chan, Virgil D. Gligor, Adrian Perrig, and Gautam Muralidharan. On the distribution and revocation of cryptographic keys in sensor networks. *IEEE Trans. Dependable Sec. Comput.*, 2(3):233–247, 2005.

[3] Douglas R. Stinson. *Combinatorial Designs: Construction and Analysis.* Springer-Verlag, 2004.

[4] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47, New York, NY, USA, 2002. ACM.

[5] Seyit Ahmet Çamtepe and Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. In *ESORICS*, pages 293–308, 2004.

[6] Seyit A. Çamtepe and Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 15(2):346–358, 2007.

[7] Jooyoung Lee; D.R. Stinson. A combinatorial approach to key predistribution for distributed sensor networks,. *Wireless Communications and Networking Conference*, 2:1200–1205, 13-17 March 2005.

[8] Jooyoung Lee; D.R. Stinson. Common intersection designs,. *Journal of Combinatorial Designs*, 14:251–269, 2006.

[9] Sushmita Ruj and Bimal Roy. Key predistribution schemes using codes in wireless sensor networks. pages 275–288, Berlin, Heidelberg, 2009. Springer-Verlag.

[10] Rolf Blom. An optimal class of symmetric key generation systems. In *EUROCRYPT*, pages 335–338, 1984.

[11] Carlo Blundo, Alfredo De Santis, Amir Herzberg, Shay Kutten, Ugo Vaccaro, and Moti Yung. Perfectly-secure key distribution for dynamic conferences. In *CRYPTO*, pages 471–486, 1992.

[12] Donggang Liu and Peng Ning. Establishing pairwise keys in distributed sensor networks. In *ACM Conference on Computer and Communications Security*, pages 52–61, 2003.

[13] Donggang Liu, Peng Ning, and Rongfang Li. Establishing pairwise keys in distributed sensor networks. *ACM Trans. Inf. Syst. Secur.*, 8(1):41–77, 2005.

[14] Sencun Zhu, Shouhuai Xu, Sanjeev Setia, and Sushil Jajodia. Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach. In *ICNP*, pages 326–335, 2003.

[15] Haowen Chan and Adrian Perrig. Pike: peer intermediaries for key establishment in sensor networks. In *INFOCOM*, pages 524–535, 2005.

[16] R. Kannan S.S. Iyengar R. Kalidindi and A. Durresi. Sub-grid based key vector assignment: A key pre-distribution scheme for distributed sensor networks. *Journal of Pervasive Computing and Communications*, 2(1):35–43, 2006.

[17] Mohammed Golam Sadi, Dong Seong Kim, and Jong Sou Park. Gbr: Grid based random key predistribution for wireless sensor network. In *ICPADS (2)*, pages 310–315, 2005.

[18] Abedelaziz Mohaisen, YoungJae Maeng, and DaeHun Nyang. On grid-based key pre-distribution: Toward a better connectivity in wireless sensor network. In *PAKDD Workshops*, pages 527–537, 2007.

[19] Donggang Liu, Peng Ning, and Wenliang Du. Group-based key predistribution for wireless sensor networks. *TOSN*, 4(2), 2008.

[20] Donggang Liu, Peng Ning, and Wenliang Du. Group-based key predistribution in wireless sensor networks. In *Workshop on Wireless Security*, pages 11–20, 2005.

[21] Keith M. Martin, Maura B. Paterson, and Douglas R. Stinson. Key predistribution for homogeneous wireless sensor networks with group deployment of nodes, 2008.

[22] Jooyoung Lee and Douglas R. Stinson. On the construction of practical key predistribution schemes for distributed sensor networks using combinatorial designs. *ACM Trans. Inf. Syst. Secur.*, 11(2), 2008.

[23] Dibyendu Chakrabarti, Subhamoy Maitra, and Bimal K. Roy. A key predistribution scheme for wireless sensor networks: Merging blocks in combinatorial design. In *ISC*, pages 89–103, 2005.

[24] Dibyendu Chakrabarti, Subhamoy Maitra, and Bimal K. Roy. A key predistribution scheme for wireless sensor networks: merging blocks in combinatorial design. *Int. J. Inf. Sec.*, 5(2):105–114, 2006.

[25] Jun-Wu Dong, Dingyi Pei, and Xueli Wang. A key predistribution scheme based on 3-designs. In *Inscrypt*, pages 81–92, 2007.

[26] Jun-Wu Dong, Dingyi Pei, and Xue-Li Wang. A class of key predistribution schemes based on orthogonal arrays. *J. Comput. Sci. Technol.*, 23(5):825–831, 2008.

[27] Sushmita Ruj and Bimal K. Roy. Key predistribution using partially balanced designs in wireless sensor networks. In *ISPA*, pages 431–445, 2007.

[28] Donggang Liu and Peng Ning. Location-based pairwise key establishments for static sensor networks. In *SASN*, pages 72–82, 2003.

[29] Wenliang Du, Jing Deng, Yunghsiang S. Han, Shigang Chen, and Pramod K. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *INFOCOM*, 2004.

[30] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A key predistribution scheme for sensor networks using deployment knowledge. *IEEE Trans. Dependable Sec. Comput.*, 3(1):62–77, 2006.

[31] Wenliang Du, Jing Deng, Yunghsiang S. Han, Pramod K. Varshney, Jonathan Katz, and Aram Khalili. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Inf. Syst. Secur.*, 8(2):228–258, 2005.

[32] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *ACM Conference on Computer and Communications Security*, pages 42–51, 2003.

[33] Zhen Yu and Yong Guan. A key pre-distribution scheme using deployment knowledge for wireless sensor networks. In *IPSN*, pages 261–268, 2005.

[34] Zhen Yu and Yong Guan. A key management scheme using deployment knowledge for wireless sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, 19(10):1411–1425, 2008.

[35] Dijiang Huang, Manish Mehta 0003, Deep Medhi, and Lein Harn. Location-aware key management scheme for wireless sensor networks. In *SASN*, pages 29–42, 2004.

[36] Katerina Simonova, Alan C. H. Ling, and Xiaoyang Sean Wang. Location-aware key predistribution scheme for wide area wireless sensor networks. In *SASN*, pages 157–168, 2006.

[37] Li Zhou, Jinfeng Ni, and Chinya V. Ravishankar. Supporting secure communication and data collection in mobile sensor networks. In *INFOCOM*, 2006.

[38] Yong Wang, Byrav Ramamurthy, and Xukai Zou. Keyrev: An efficient key revocation scheme for wireless sensor networks. In *ICC*, pages 1260–1265, 2007.

[39] Chul-Hyun Park, Yi-Ying Zhang, In-Tai Kim, and Myong-Soon Park. Dls: Dynamic level session key revocation protocol for wireless sensor networks. In *Information Science and Applications (ICISA), 2010 International Conference on*, pages 1–8, 2010.

[40] Tyler Moore, Jolyon Clulow, Shishir Nagaraja, and Ross Anderson. New strategies for revocation in ad-hoc networks. In *ESAS*, pages 232–246, 2007.

[41] Ranjan Bose. *Information Theory, Coding and Cryptography*. Tata McGraw-Hill, 2008.

[42] Seyit A. Çamtepe and Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 15(2):346–358, 2007.

[43] John R. Douceur. The sybil attack. In *IPTPS*, pages 251–260, 2002.