Application of Combinatorial Structures in Wireless Communication

Thesis submitted to Indian Statistical Institute in partial fulfillment for the degree of Doctor of Philosophy

by

Samiran Bag

Applied Statistics Unit

Indian Statistical Institute

Kolkata

January 2015

Application of Combinatorial Structures in Wireless Communication

Thesis submitted to Indian Statistical Institute in partial fulfillment for the degree of Doctor of Philosophy

by

Samiran Bag

Applied Statistics Unit

Indian Statistical Institute

Kolkata

under the supervision of

Prof. Bimal Roy

Applied Statistics Unit

Indian Statistical Institute

Kolkata

"Never argue with stupid people, they will drag you down to their level and then beat you with experience."

Mark Twain

Acknowledgements

It has been a long cherished dream of mine to earn a doctoral degree. Now, while I am writing this thesis I wish to thank each and every person who helped me directly or indirectly to make this thesis possible. I consider myself lucky enough to have Prof. Bimal Roy as my supervisor. He was the Director of the Institute for most of the time of my work. He guided me like a father throughout the entire time of my work. He was the first teacher who taught me the first lessons on cryptography. I also thankfully acknowledge the constructive guidance and effective contribution of Dr. Sushmita Ruj. She was a post doctoral fellow at the University of Ottawa, Canada when I started collaborating with her. Later, she became an assistant professor at the Indian Statistical Institute, Kolkata after a brief stint as an assistant professor at the Indian Institute of Technology, Indore. We have authored two papers together which are included in this thesis. Next, I thank Dr. Kishan Chand Gupta, of Indian Statistical Institute. He had been an inspiration to me for the entire period of my Ph.D. He taught me three courses on Coding theory, Finite field theory and Combinatorial designs. He helped me from time to time whenever I had approached him be it related to my research or it was about something else. Lastly, I thank Mr. Indranil Ghosh Ray. He was my fellow researcher at the Indian Statistical Institute. He has the sharpest mind among all researchers here at ISI and all researchers including myself used to discuss with him on academic issues and he was happy to provide explanations on various topics as much as he could.

Abstract

Wireless communication has a compelling need in today's world. This need is driven by the sudden surge in the demand of ad-hoc networks like wireless sensor networks. These networks consist of many isolated devices called nodes that use the wireless medium for communication. There are three security notions attached to any wireless communication. These are data confidentiality, data integrity and data availability. Cryptographic protocols are required for achieving the first two notions which in turn require shared secret keys. The last notion viz. data availability mainly deals with protecting the data from 'denial of service attack'.

In this thesis, we apply combinatorial designs for achieving these three security notions. We study key predistribution in wireless sensor networks and jamming resistant communication and propose new schemes using combinatorial design. The designs we used here are Affine Geometry, Transversal Design, Steiner Triple System, Symmetric Balanced Incomplete Block Design, Mutually Orthogonal Latin Squares etc. Though combinatorial designs have been in use for key predistribution in wireless sensor networks for past few years, the techniques we propose in this thesis are far better than others in terms of connectivity, resilience etc.

As mentioned above, keys are required for ensuring data security in a wireless network. We propose three different schemes that can be used for key predistribution in wireless sensor networks of different kinds. The first one is for a homogeneous wireless sensor network containing identical nodes. We use affine geometry to design the key predistribution scheme. This scheme ensures constant time shared key discovery. We show that the performance of this scheme is better than several other schemes of similar kind.

We also propose another key predistribution scheme using a two-layered hybrid design having Symmetric Balanced Incomplete Block Design at the upper layer and Blom's scheme at the lower layer. Like the previous scheme this scheme also comes with a constant time shared key discovery mechanism. This key predistribution scheme is highly resilient against random node capture attack and beats all other schemes of similar kind with a huge margin. We then show how this scheme can be used in a grid-group deployment.

Again we propose a key predistribution scheme for grid-group deployment of sensor nodes. Here we assume that the sensor nodes are distributed in groups. Our scheme mainly focusses on cross-group key establishment i.e key establishment between sensor nodes belonging to two different groups. This scheme offers an improvement in terms of resiliency to other schemes in literature.

This thesis is the first one to have proposed the use of combinatorial designs in anti-jamming wireless communication. Thus, this thesis opens a new direction of research in jamming resistant wireless communication. We discuss jamming resistant communication and propose schemes that attempt to maintain steady communication under denial of service attack. We propose a scheme that enables a group of users to communicate in the presence of a jammer who tries to disrupt the communication as much as possible. This scheme enables a user of the group to meet with every other user on an exclusive channel within a bounded time called a 'session'. This scheme is the first one that enables the user to meet with another user in every session and also provide effective resistance against jamming attack. In addition, we propose a jamming resistant communication scheme for multicast communication. We also enhance the performance of an existing jamming resistant communication scheme called Uncoordinated Frequency Hopping(UFH) scheme. In UFH, two users would hop unboundedly for a rendezvous on the same frequency channel in order to exchange pending messages. We propose a scheme that would allow the nodes two meet with each other in every attempt. Thus, our scheme brings down the time needed for message communication between a pair of users.

Contents

List of Figures

List of Tables

For my mother Mrs. Monorama Bag

This page intentionally left blank

Chapter 1

Introduction

In today's world the need of information communication is of tremendous importance. Information communication system is a basic requirement for sending an e-mail to a dedicated server as well as in making a secret financial transaction with a party sitting abroad. This triggers the need of setting up communication networks. There are three types of communication networks, viz. wired network, wireless network and hybrid network.

Use of wired networks dates back to the early ages in the history of electronic message communication. Samuel F. B. Morse developed the first telegraph device in 1837. Since then wired communication networks have evolved drastically. Such networks have been extensively studied as they have wide applications in the form of telephone (LAN), telegraph, telegram, etc. Routers present in these networks fix their infrastructure in the sense that their topology is constant except for addition and deletion of nodes from time to time. Wireless networks use wireless medium for communication between various constituent components. Wireless communication did not really take place until Michael Faraday demonstrated electro magnetic induction way back in 1831. Since then it has gone through several phases of evolution. The main advantage of using wireless network over wired network is the portability of the devices. Wireless network is also extensively used in ad-hoc and sensor networks.

There are three security notions of wireless communication viz. data confidentiality, data integrity and data availability. The first two security notions viz. data confidentiality and data integrity can be achieved using cryptographic protocols whereas the last one requires different measures. This thesis discusses issues related to all three security notions and attempts to find effective solutions.

1.0.1 Classification of Wireless Networks

Wireless networks can be classified into two different groups. These two groups are Wireless Infra-structured Network and Wireless ad-hoc network.

1. **Wireless Infra-Structured Networks**: Wireless Infra-Structured Network links two or more devices using some wireless distribution method. A wireless access point (AP) is required for infrastructure mode wireless networking. The AP is then cabled to the wired network to allow wireless clients access to, for example, Internet connections or printers. Additional APs can be added to the WLAN to increase the reach of the infrastructure and support any number of wireless clients.
2. **Wireless ad hoc network (WAHN)**: A wireless ad hoc network is a decentralized network that does not rely on a preexisting infrastructure. In these networks, each node participates in routing by forwarding data for other nodes. An ad hoc network typically refers to any set of networks where all devices have equal status on a network and are free to associate with any other ad hoc network device in link range. Ad hoc network often refers to a mode of operation of IEEE 802.11 wireless networks.

1.1 Classification of Wireless Ad hoc Networks

Further wireless ad hoc networks may be of three types on the basis of their application, nature of topology and other properties. These are :

1. Mobile Ad Hoc networks (MANET)
2. Wireless Mesh Networks (WMN)
3. Wireless Sensor Networks (WSN)

1.1.1 Mobile Ad Hoc Network

A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes. The set of applications for MANETs is diverse, ranging from small,

static networks that are constrained by power sources, to large-scale, mobile, highly dynamic networks. The design of network protocols for these networks is a complex issue. Regardless of the application, MANETs need efficient distributed algorithms to determine network organization, link scheduling, and routing.

1.1.2 Wireless Mesh Network

A wireless mesh network (WMN) [2] is a communications network made up of radio nodes organized in a mesh topology. Wireless mesh networks often consist of mesh clients, mesh routers and gateways. The mesh clients are often laptops, cell phones and other wireless devices while the mesh routers forward traffic to and from the gateways which may, but need not, connect to the Internet. The coverage area of the radio nodes working as a single network is sometimes called a mesh cloud. Access to this mesh cloud is dependent on the radio nodes working in harmony with each other to create a radio network. A mesh network is reliable and offers redundancy. When one node can no longer operate, the rest of the nodes can still communicate with each other, directly or through one or more intermediate nodes. Thus, wireless mesh networks can self form and self heal. Wireless mesh networks can be implemented with various wireless technology including 802.11, 802.15, 802.16, cellular technologies or combinations of more than one type.

1.1.3 Wireless Sensor Network

A wireless sensor network (WSN) consists of spatially distributed tiny autonomous sensor nodes. These nodes gather sensory information about the surrounding environment. These networks monitor physical or environmental conditions, such as temperature, sound, pressure, etc. and to co-operatively pass their data through the network to a main location called the base station. The more modern networks are bi-directional, also enabling control of sensor activity. Today such networks are used in many industrial and consumer applications, such as industrial process monitoring and control, machine health monitoring, and so on [17, 59]. Extensive surveys can be found in [1, 12, 76].

The WSN is built of "nodes" – from a few to several hundreds or even thousands, where each node is connected to one (or sometimes several) sensors. Each such sensor network node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna, a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery or an embedded form of energy harvesting. A sensor node might vary in size from that of a shoe-box down to the size of a grain of dust, although functioning "motes" of genuine microscopic dimensions have

yet to be created. The cost of sensor nodes is similarly variable, ranging from a few to hundreds of dollars, depending on the complexity of the individual sensor nodes. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and communications bandwidth. An example of a typical sensor is the MICAz mote, that has an under powered processor with 4KB of RAM, 512 KB of program memory, an Advanced Encryption Standard (AES) cryptographic hardware and run the TinyOS operating system. The transmitter of the UC Berkley Mica platform has bandwidth of 10 Kbps.

The topology of the WSNs can vary from a simple star network to an advanced multi-hop wireless mesh network. The propagation technique between the hops of the network can be routing or flooding [20].

The information exchanged between sensor nodes can be very sensitive for some applications, like military and health care services. Hence, the communication between sensors need to be secured for those applications of wireless sensor networks. The main challenges of security in wireless sensor networks are [12] :

1. Wireless nature of communication.
2. Resource limitation on sensor nodes.
3. Very large and dense WSN.
4. Lack of fixed infrastructure.
5. Unknown network topology prior to deployment.
6. High risk of physical attacks to unattended sensors.

Cryptographic primitives need to be employed for meeting these security challenges. This necessitate loading of keys inside the sensor nodes. Hence key management becomes of utmost importance in sensor networks. A key establishment technique for WSN must incorporate these following properties [76]:

1. Availability : Ensuring that the service offered by the whole WSN, by any part of it or by a single sensor node must be available whenever required.
2. Flexibility : Key establishment technique should be useful in multiple applications and allow for adding nodes at any time.
3. Survivability: Ability to provide service in case of power failure or attacks.
4. Adaptive security service: Ability to change security levels as resource availability changes.

1.2 Key Predistribution in WSN

Wireless sensor networks use symmetric key cryptography for key establishment. This consists of three steps

1. *Key predistribution* : This means preloading the keys inside the sensor nodes prior to deploying them in the zone of deployment.
2. *Shared key discovery* : This is the method of computing the common key between a pair of nodes.
3. *Path key establishment* : If a common key does not exists, then a path has to be found between the communicating nodes. This path is a chain of nodes of minimum length such that any two consecutive nodes in the chain have a shared key. A path key is then established between the communicating nodes.

1.2.1 Basic Terminology

1. Key pool: A set of keys from which subsets of keys are selected and placed in the sensor nodes.
2. Key ring : A group of keys contained in a sensor node is called the key ring of that node.
3. Node identifier: The unique index that is assigned to a node belonging to a WSN.
4. Key identifier: The unique index that is assigned to a key of the key pool.

1.2.1.1 Key Predistribution

We know that key predistribution in wireless sensor networks is the process of loading the cryptographic keys inside the sensor network prior to deployment in the target zone. Key predistribution mechanisms are basically of three types namely probabilistic key predistribution, deterministic key predistribution and hybrid key predistribution.

In the first type of key predistribution keys are drawn from a key pool either randomly or by following a probability distribution and are placed into the individual sensor nodes. Every key drawn is replaced back into the key pool keeping the key pool unaltered throughout the entire process of key predistribution. Two sensor nodes, in probabilistic key predistribution scheme may or may not share a common key. Hence, two nodes communicate with each other with certain probability.

In the second approach i.e. for deterministic key predistribution scheme keys are loaded inside the sensor nodes following a deterministic pattern. Due to this deterministic pattern shared key discovery turns out to be computationally easy. In this scheme like the probabilistic scheme a pair of nodes may or may not share a common key. But if a pair of nodes share a common key there does exist an algorithm to find it.

Lastly, a hybrid key predistribution scheme is a composite scheme of the above two key predistribution schemes.

1.2.1.2 Shared Key Discovery

There are few methods of shared key discovery. One method is to have the key identifiers are broadcast. A pair of nodes willing to communicate first exchange their identifiers. Since node ids are public information they can be sent in an unencrypted form. Once, the nodes get to know each other, they can compute the common key. This can be done by looking into a lookup table that stores the key identifiers of the node and the identifier of the nodes that contain the same key. Alternately if the key predistribution scheme is deterministic, a deterministic algorithm may be available for shared key discovery [12, 38, 60]. Alternately, a challenge response protocol is used for shared key discovery. To find one or more common shared keys between two nodes, each node has to broadcast a list $\{(\mathcal{C}, E_{K_i}(\mathcal{C})), i = 1, 2, \ldots, k\}$, where $\mathcal{C}$ is the challenge and K_i is the i^{th} key of the node. The size of the key ring of each node is k. Upon receipt of the list, the other node would decrypt the encrypted challenges with keys from its own keyring and would try to match it with the challenge. If for some key, a match is found then the corresponding key is the common key. This type of challenge response protocol is used in [19, 27]. Another way of finding shared keys was proposed by Pietro, Macini and Mei [55], in which pseudo-random key-index transformations are used to find the common keys.

1.2.1.3 Path Key Establishment

Whenever two nodes share a common key, a secure communication channel can be established between the two nodes. However, there are key predistribution schemes in existence (e.g. [23]) where not every pairs of nodes share a common key. If two nodes do not share a common secret key, they cannot communicate securely. Hence, for such pair of nodes who do not share a key, a secret key must be established between them. In order to do so, a secure path has to be found between the two nodes. This path is a chain of nodes beginning and ending with the two nodes such that any two consecutive nodes in the chain shares a common key. If such a chain is found then it can be used by

the two nodes at the ends of the chain to agree upon a common key generated randomly by one node and communicated to the other one using the secure path of nodes.

1.2.2 Wireless Sensor Network: Broad Overview

Wireless Sensor Networks (WSN) constitute popular families of ad hoc mobile networks of recent times. Such networks typically consist of three types of nodes, namely:

1. Base Station (BS).
2. Identical ordinary sensors (or nodes or sensor nodes or motes)
3. Cluster Heads (CHs) or Gate–Way (GW) nodes: sometimes provisions are made for some special nodes having certain extra capabilities which are termed as Cluster Heads (CHs) or Gate–Way (GW) nodes.

Based on the existence of the Cluster Heads (CHs) mentioned above, WSNs are further categorized into two types:

1. Hierarchal Wireless Sensor Network(HWSN): These heterogeneous networks contain some special nodes (CH) having more capabilities than (ordinary) nodes. A natural hierarchy of at least three levels is induced by the CH in any HWSN model. At the lowermost level, the participating nodes are split into small clusters by the CH. The CHs are sometimes referred to as Gate–Way (GW) nodes as any communication involving the sensors in their cluster and nodes outside must pass through the CHs
2. Distributed Wireless Sensor Network (DWSN): In case of DWSN, there is no fixed type of architecture in the sensor nodes. The topology is unknown before the deployment. The mode of communication is mainly Unicast in this case, however Broadcasting may be invoked from time to time. Such networks are homogeneous in the sense that all (ordinary) nodes other than the Base Station are treated equally.

Capacities of each such unit in the ordinary nodes is quite limited for any WSN, be it DWSN or HWSN. For HWSN, the capacities and power of the CHs may vary while the KDS of any such network is usually quite powerful.

As the name suggests, communication in ‘wireless sensor networks’ is achieved using radio frequencies. Resource constrained nodes can communicate with each other only

within a limited range having center as the node and small radius termed as *Radio Frequency range* or *radius of communication* or *physical layer of [38]*. This range or radius is generally same for ordinary sensors and may be varied for CHs. While the KDS has quite a large *radius of communication*.

In terms of deployment strategy and location control, WSNs are categorized into 5 categories namely :

1. Fixed, full control : In this type of deployment the target position of deployment is precisely known.
2. Fixed, partial control : In this type of deployment, partial information about the deployment of sensors is known.
3. Fixed, no control : In this type of deployment the sensor nodes are randomly scattered in the deployment zone. Hence the final location of any node is not known prior to deployment.
4. Locally mobile: In this type of deployment sensors can move freely within a prescribed region only, but cannot move out of that region.
5. Fully mobile : In this type of deployment the sensor nodes can move anywhere within the network zone.

1.3 Attacks on Key Predistribution Schemes

Wireless sensor networks are deployed in adversarial environment, sometimes in an area completely under control of an enemy. Hence, WSNs are vulnerable to attacks. The following assumptions are made about the capabilities of the attacker [30] :

1. The attacker has unlimited energy and computing power.
2. The attacker has access to all information stored in a captured node.
3. The attacker is privy to all the traffic in the network and can record the same.
4. The attacker can introduce forged messages into the system.
5. The attacker has the ability to physically ascertain the location of a given sensor by listening to the traffic.
6. The attacker has the ability to deploy fabricated nodes.

The next topic of interest is the various threat models that one should consider. Among several types of models for node capture, the work in this thesis is aimed at fixing a couple of them which are being described here:

1. Random node capture attack: Nodes are captured randomly.

2. Selective Node capture: The attack is described in [30, 55]. For completeness, a brief outline is being presented here. In this attack, the goal of the attacker is to collect a subset T of the keys in the pool. Assume that the attacker has already compromised a number of sensors and hence collected all their keys in a set W. Consider a random variable $G(s)$ to denote the key information gain for capture of an additional sensor s. Thus $G(s)$ measures the number of keys in the key ring of s which are in T and are not in W. For instance, if the attacker wants to compromise the channel between sensors s_a and s_b, the attacker concentrates precisely on the subset that contains all the keys of the key ring of both s_a and s_b. According to above notation the attacker's concerned subset is $T = M_a \cap M_b$, where M_a and M_b denotes the key rings of s_a and s_b respectively. Now consider capture of another node s. Assuming that the attacker has collected a set W of keys, the random variable $G(s)$ is equal to $|(M_s \cap M_a \cap M_b) - W|$. At each step of the attack sequence, the next sensor to be tampered with is sensor s, where s maximizes $E[G(s)|I(s)]$, the expectation of the key information gain $G(s)$ given the information $I(s)$ that the attacker knows about the key ring of sensor s. It will be later established that in the present (combined) scheme an attacker does not gain in any way by launching a selective node capture attack. As such selective node capture becomes just as good as random node capture from the attacker's point of view.

We will see later in this thesis that our schemes are secure against random and selective node capture attack.

1.3.1 Definitions

Definition 1.1. $E(s)$ is defined as the fraction of edges disconnected when s nodes are compromised. This is given by,

$$E(s) = \frac{\text{Number of links present after } s \text{ nodes are compromised}}{\text{Number of links present before } s \text{ nodes are compromised}}$$

Here, N is the total number of nodes in the network and $s < N$.

Definition 1.2. $V(s)$ is defined as the fraction of nodes disconnected when s nodes are compromised. This is given by,

$$V(s) = \frac{\text{Number of nodes disconnected when } s \text{ nodes are compromised}}{N}$$

Here, N is the total number of nodes in the network and $s < N$.

Definition 1.3. The connectivity ratio in a WSN is the probability of existence of a secure link between two randomly selected nodes. Numerically it can be calculated as,

$$\text{Connectivity ratio}(\rho_c) = \frac{\text{Total number of secure links present between pairs of nodes}}{\binom{N}{2}}$$

Here, N is the total number of nodes in the network.

1.4 Jamming Resistant Wireless Communication

The networks we have discussed so far use wireless medium for communication. Communication is accomplished in these networks through transmitting messages over wireless medium. Each communicating device has one or multiple transducers allowing them to send or receive messages simultaneously. The message–sender transmits the message using its antenna and the intended recipient of the message receives it on its antenna. For successful message delivery the antennas of the two devices must be set to the same frequency. In other words, the message–sender must be transmitting the message on the very same frequency the receiver is currently listening to. But a malicious adversary can willfully launch denial of service attack to disrupt the communication. She can do this by putting a strong noisy signal on the same frequency being used for communication by a pair of devices. This noisy signal could damage the message during transmission. If sufficient damage is done to the message, the receiver would not be able to infer anything from the received signal. Thus jamming attack is disruptive and has to be taken care of. Jamming resistant wireless communication mechanisms thus deal with strategies employed for ensuring *'data availability'*.

1.4.1 Wireless Communication : The Basic Model

1. Network : Wireless communication essentially involves a network consisting multiple communicating nodes. This nodes use radio waves for message exchange. Communication can be unidirectional or bi-directional. We can consider for example a trivial network consisting of two nodes, a sender and a receiver. The sender always sends messages to the receiver but not vice-versa. Alternately, the two

nodes may be exchanging messages to each other making it a bi-directional communication. Similarly, we can consider any wireless network discussed in section 1.0.1.

2. Jammer : She is a computationally unbounded attacker having limited jamming strength. So, she can jam a finite number of communication channels. The attacker has knowledge of all public information about the network except the secret keys. The intention of the jammer is to disrupt the communication as much as possible using its jamming strength as well as the publicly known information about the system model. The attacker jams a wireless channel by putting a strong noisy signal over the channel. This signal interferes with the electro magnetic wave of any message–signal carried by the same channel. As a result the signal could become uninterpretable by any receiver. This is called 'denial of service' attack.

1.4.2 Jamming Resistant Communication

Wood *et al.* [75] presented a novel protocol for defeating energy–efficient jamming in networks based on IEEE 802.15.4 compatible hardware. Their protocol employed four defence mechanisms against jamming attack. These are as follows:

1. Frame masking : In the frame masking defence mechanism a pseudo random sequence is used as Start of Frame Delimeter (SFD). The sender and the receiver both agree upon this pseudo–random sequence that indicates the start of a frame. Unless the attacker's radio is configured to recognize the correct SFD, she cannot start jamming.

2. Channel Hopping : In this strategy the message–sender and the message–receiver keeps changing transmission channels. This makes it hard for the jammer to choose the appropriate channel to be jammed. If the communicating devices keep hopping over a sufficiently large set of channels, there will be a low probability of jamming provided the jammer has no information on the hopping pattern of the communicating devices.

3. Packet fragmentation : The third mechanism deals with splitting a bigger message into smaller fragments. A big message requires longer transmission time. So if a message takes long time in transmission it may get damaged by a jammer who hops through channels quickly. In contrary a small message packet can be delivered faster before a jammer can notice it. Hence, fragmentation can resist jamming to some extent.

4. Redundant encoding : The last mechanism discussed by Wood *et al.* is redundant encoding. Redundant encoding scheme is employed to transmit fragmented messages at the cost of transmission redundancy. In other words, some coding technique is used to send the fragmented packets so that if some of the packets get corrupted the redundancy of the information can be used to reconstruct the original message. Once such coding technique is erasure coding by means of which it is possible to fragment a message into many parts. However, among those fragments only a finite number of fragments are sufficient to reconstruct the original message. Hence, a message–sender can use this technique to fragment any message and can transmit them in any order to the message–receiver. Now, some fragments may get blocked by the jammer. But because of the redundancy, the receiver can reconstruct the message as soon as it collects sufficient number of fragments.

1.4.3 Classification of Jamming Resistant Wireless Communication Schemes

Anti-jamming wireless communication schemes mostly exploit the spectral diversity of the wireless medium. Though there are schemes like [78] that relies on spatial retreat of the nodes from the area under control of the jammer, most of the existing schemes apply spread spectrum or frequency hopping for countering jamming attack.

1.4.3.1 Direct sequence spread spectrum

In telecommunications, direct-sequence spread spectrum (DSSS) is a modulation technique. As with other spread spectrum technologies, the transmitted signal takes up more bandwidth than the information signal that modulates the carrier or broadcast frequency. The name 'spread spectrum' comes from the fact that the carrier signals occur over the full bandwidth (spectrum) of a device's transmitting frequency. Certain IEEE 802.11 standards use DSSS signaling. DSSS phase-modulates a sine wave pseudorandomly with a continuous string of pseudonoise (PN) code symbols called "chips", each of which has a much shorter duration than an information bit. That is, each information bit is modulated by a sequence of much faster chips. Therefore, the chip rate is much higher than the information signal bit rate. DSSS uses a signal structure in which the sequence of chips produced by the transmitter is already known by the receiver. The receiver can then use the same PN sequence to counteract the effect of the PN sequence on the received signal in order to reconstruct the information signal.

All jamming resistant wireless communication schemes based on DSSS use a set of spreading codes. each message is spread to a wide bandwidth and transmitted over the wireless medium. The receiver uses the spreading code to despread the message.

1.4.3.2 Frequency hopping spread spectrum

Frequency-hopping spread spectrum (FHSS) is a method of transmitting radio signals by rapidly switching a carrier among many frequency channels, using a pseudorandom sequence known to both transmitter and receiver. It is utilized as a multiple access method in the frequency-hopping code division multiple access (FH-CDMA) scheme.

By itself, frequency hopping provides only limited protection against eavesdropping and jamming. There is a simple algorithm that effectively discovers the sequence of frequencies. To get around this weakness most modern military frequency hopping radios employ separate encryption devices such as the KY-57. U.S. military radios that use frequency hopping include the JTIDS/MIDS family, HAVE QUICK and SINCGARS.

In FHSS, transmission occurs only on a small portion of this bandwidth at any given time, the effective interference bandwidth is really the same. Whilst providing no extra protection against wideband thermal noise, the frequency-hopping approach does reduce the degradation caused by narrowband interference sources. Hence, **frequency hopping can provide very effective resistance against narrowband jamming attacks**.

One of the challenges of frequency-hopping systems is to synchronize the transmitter and receiver. One approach is to have a guarantee that the transmitter will use all the channels in a fixed period of time. The receiver can then find the transmitter by picking a random channel and listening for valid data on that channel. The transmitter's data is identified by a special sequence of data that is unlikely to occur over the segment of data for this channel and the segment can have a checksum for integrity and further identification. The transmitter and receiver can use fixed tables of channel sequences so that once synchronized they can maintain communication by following the table. On each channel segment, the transmitter can send its current location in the table.

In the US, FCC part 15 on unlicensed system in the 900 MHz and 2.4 GHz bands permits more power than non-spread spectrum systems. Both frequency hopping and direct sequence systems can transmit at 1 Watt. The limit is increased from 1 milliwatt to 1 watt or a thousand times increase. The Federal Communications Commission (FCC) prescribes a minimum number of channels and a maximum dwell time for each channel.

In a real multipoint radio system, space allows multiple transmissions on the same frequency to be possible using multiple radios in a geographic area. This creates the possibility of system data rates that are higher than the Shannon limit for a single channel. Spread spectrum systems do not violate the Shannon limit. Spread spectrum systems rely on excess signal to noise ratios for sharing of spectrum. This property is also seen in MIMO and DSSS systems. Beam steering and directional antennas also facilitate increased system performance by providing isolation between remote radios.

Jamming resistant wireless communication schemes that use FHSS evade the jammer by changing the frequency on which they transmit/receive data. This thesis contains two chapters that discuss jamming resistant communication using FHSS method. We shall discuss such schemes in chapter 2.

1.5 Our Contribution & Thesis Plan

This thesis is based on papers [3–7]. The main contribution of the thesis can be divided into two parts. The first part includes chapter 3,4 and 5 that discusses key predistribution schemes for wireless sensor networks. The other part of the thesis includes chapter 6 and 7 and is dedicated to jamming resistant wireless communication. Chapter 2 provides the background study of the key predistribution schemes and anti-jamming communication schemes that are related to the contribution of this thesis. We only discuss about the key predistribution schemes which are similar in nature to those discussed in later chapters of this thesis. We also discuss several jamming resistant wireless communication schemes at the end of chapter 2.

In chapter 3, we discuss a key predistribution scheme that makes use of finite affine geometry. It is based on paper [6]. In this scheme, nodes have nearly equal number of keys stored in them. The number of keys stored in a node differ by at most 3. Also, the number of common keys between a pair of nodes are not same. We have measured the performance of our proposed scheme using $V(s)$ and $E(s)$. We have also compared our scheme with other existing schemes and have shown that our schemes offer better performance than many of them in terms of $E(s)$ defined in section 1.3.1.

In chapter 4, we discuss another key predistribution scheme. We propose a key predistribution scheme for homogeneous wireless sensor networks using the scheme of Blom [10] as well as Symmetric Balanced Incomplete Block Design. The main advantage of using this scheme for key predistribution is that for this scheme the adversary needs to capture large number of nodes in order to compromise all the keys in an uncompromised node. In other words, in order to disconnect an uncaptured node from all other

nodes, the adversary needs to capture many more nodes than other standard schemes. Next, we use this new key predistribution scheme in a grid-group deployment of sensor nodes. The entire deployment zone is broken into square regions. The sensor nodes falling within a single square region can communicate directly. Sensor nodes belonging to different square regions can communicate by means of special nodes deployed in each of the square region. We measure the resiliency in terms of fraction of links disconnected as well as fraction of nodes and regions disconnected. We show that our key predistribution scheme when applied to grid-group deployment performs better than standard models in existence in terms of standard measures.

Chapter 5 is based on paper [3], we discuss another scheme for key predistribution in wireless sensor networks for a grid-group deployment of the sensor networks. We have used finite affine planes in this purpose. Our main contribution is to develop a key predistribution scheme in the special nodes called agents that connects the sensor nodes of one region to those of a different region. This paper mainly focusses on setting up key-links between different groups which makes this scheme more analogous to the key predistribution scheme by Ruj and Roy in [62]. Ruj-Roy used only three agents per region in their scheme. In our work the number of agents per region is a variable that depends upon the size of the deployment grid. Our scheme offers better performance than well known Ruj-Roy scheme and some other standard existing schemes that uses deployment knowledge.

In this thesis we have two chapters dedicated for anti-jamming communication. These are chapter 6 and chapter 7. In these works we have used Frequency Hopping Spread Spectrum method for countering jamming attacks. The difference between the works described in chapter 6 and chapter 7 is that chapter 6 deals with two party communication whereas chapter 7 deals with multi-party communication. In other words, the work contained in chapter 6 focusses on anti-jamming communication between a pair of users and chapter 7 focusses on anti-jamming communication between many users. Two users can communicate using frequency hopping through a common pseudo random sequence generated by a pseudo random generator as in [53]. But for generating the sequence the users must first agree upon a common secret key. For communicating this key anti-jamming communication mechanism is required which creates a cyclic dependency between anti-jamming communication and anti jamming key establishment. Strasser et al. [68] proposed uncoordinated frequency hopping that allows communication between a pair of users without shared secret in the presence of jammer thus allowing them to establish a common secret required for communication. But in UFH the users hop unboundedly for establishing a shared key. We, using design theory design a frequency hopping scheme that allows a pair of users who do not share any secret key to exchange messages in a bounded time given by $O(\frac{1}{p_j})$, where p_j is the jamming probability of a

channel in the network. This scheme would allow a pair of nodes to establish a secret key in presence of the jammer faster than the UFH scheme. Also, this scheme can be used for exchanging data between the pair of nodes without bothering about establishing a secret key.

In chapter 7 we discuss two frequency hopping schemes. The first scheme allows a group of users to communicate with each other in the presence of a jammer. This scheme deals with anti-jamming communication for a set of users/nodes who communicate with each other. There are two schemes discussed in this chapter. The aim of the first scheme is to ensure that the set of nodes can communicate in such a fashion that each and every pair of nodes get an opportunity to meet on an exclusive channel within a fixed time bound which is henceforth called a 'session'. Most of the existing schemes on frequency hopping are meant for two party communication or broadcast communication. Those schemes that deal with multi party communication try to establish single common control channels for all users. But the problem of establishing separate communication channels for different pairs of users were not addressed. The classical frequency hopping scheme using pseudorandom number sequence does not guarantee a rendezvous in a fixed time when the number of nodes are many. The only frequency hopping scheme that offers a time-bounded rendezvous between any pair of users is the Quorum Rendezvous Channel Hopping scheme by Lee et al. [36]. But the time bound for Lee et al. scheme is very high, between $O(C)$ and $O(C^2)$, where C is the number of channels in the network. We, in our work have attempted to reduce this time bound to exactly $O(C)$. In our scheme a user doesn't need to remain idle for even a single time slot if it has messages to be transmitted to other nodes. In every time slot each user rendezvous with another user in this scheme contrary to the existing schemes where nodes may remain idle waiting for a rendezvous with another user. In order to evade jamming, the users keep changing frequency channels using pseudorandom number sequences. Our scheme ensures that at each time slot, every user must meet with a unique user on a dedicated channel. In other words, the sending and receiving sequences for every user is different in our scheme, whereas in [36] it may happen that two or more users end up selecting the same receiving and sending sequences and being on the same channel at the same time or trying to send messages on the same channel at the same time. Our scheme, on the other hand ensures that no more than two nodes be on the same channel at the same time, a condition necessary for avoiding collision.

We discuss another anti-jamming communication scheme for multicast communication in chapter 7. Here, there is a set of senders and a set of receivers. The number of senders is less than the number of receivers. A sender sends messages to multiple receivers at the same time. We proposed a scheme whereby any receiver is bound to meet a particular sender on a frequency channel within a bounded time. We used combinatorial designs

for developing these two schemes. We have provided a comparison of our scheme with standard channel hopping schemes of similar kind in table 2.2.

Chapter 2

Background

In this chapter we first describe combinatorial design and then present a literature survey of the research work published in this thesis. This chapter is divided into three sections. In the first section we explore combinatorial designs. We introduce the reader to several combinatorial designs used in the later chapters of this thesis. In the following section we briefly discuss several key predistribution schemes already in existence. We mention several schemes and point out their advantages and disadvantages. In the last section, we discuss some existing schemes for jamming resistant communication. These schemes use different techniques for enabling communication in the presence of jammer.

2.1 Combinatorial Design

The thesis is aimed at showing applicability of combinatorial designs for key predistribution in wireless sensor network and for providing jamming resistance in any wireless communication network. The whole thesis is dedicated to the study of different combinatorial designs and their correspondence to the two areas of application – sensor networks and jamming resistant wireless communication. In this thesis we study Unbalanced Block Design (UBD), Steiner Triple System (STS), Affine Geometry (AG), Symmetric Balanced Incomplete Block Design (SBIBD), Mutually Orthogonal Latin Square (MOLS) and Transversal Design (TD). We have used each of them either for key predistribution in a sensor network or for jamming resistant wireless communication schemes.

Combinatorial designs have very interesting patterns. By studying these patterns we can devise efficient key establishment schemes which were not possible in many randomized key predistribution schemes. We also come up with a new type of designs construction

called unbalanced design which we study in Chapter 3. On one hand we are able to design better key predistribution schemes as well as develop communication strategies that offer resistance against jamming attack, on the other hand the designs we use, have their own mathematical interest.

2.1.1 Definitions

Definition 2.1. A design (def. 1.1 of [66]) is a two tuple $(X, \mathcal{A})$ where X is a set of elements or varieties and $\mathcal{A}$ is a set of subsets (also called *blocks*) of X. Thus,

$$\mathcal{A} = \{B : B \subseteq X\}.$$

Definition 2.2. A (v, b, r, k, λ)-Balanced Incomplete Block Design(BIBD) (def. 1.2 of [66]) is a design satisfying these properties:

1. $|X| = v$,
2. $|\mathcal{A}| = b$,
3. $\forall x \in X, |\{B : B \in \mathcal{A}, x \in B\}| = r$,
4. $\forall B \in \mathcal{A}, |B| = k$,
5. $\forall x, y \in X, x \neq y, |\{B : B \in \mathcal{A},\ x, y \in B\}| = \lambda$.

A (v, b, r, k, λ)-BIBD is also denoted as a (v, k, λ)-BIBD.

Example: A (7, 3, 1)-BIBD.

$X = \{1, 2, 3, 4, 5, 6, 7\}$, and

$A = \{123, 145, 167, 246, 257, 347, 356\}$.

Definition 2.3. A Symmetric Balanced Incomplete Block Design (def. 2.1 of [66]) is a (v, b, r, k, λ) BIBD where $v = b$.

It can be shown that for a (v, b, r, k, λ) SBIBD $r = k$ holds.

Definition 2.4. A *Transversal Design* (def. 6.42 of [66]) $TD(k, n, \lambda)$, where $k \geq 2$, $n \geq 1$; is a triple $(X, \mathcal{G}, \mathcal{B})$ satisfying,

1. $|X| = kn$,
2. $\mathcal{G} = \{G_1, G_2, \ldots, G_k\}$, such that, $|G_i| = n, 1 \leq i \leq k$, $G_i \bigcap G_j = \emptyset$, $\bigcup_{i=1}^{k} G_i = X$,

3. $\mathcal{B} = \{B : B \subset X, |B| = k\}$,

4. $\forall B \in \mathcal{B}, 1 \leq i \leq k, |B \cap G_i| = 1$,

5. $\forall i, j \in \{1, 2, \ldots, n\}, i \neq j, \forall x \in G_i, \forall y \in G_j, |\{B : B \in \mathcal{B}, x, y \in B\}| = \lambda$.

Definition 2.5. Let $(X, \mathcal{A})$ be a (v, b, r, k, λ) design where $X = \{x_1, x_2, \ldots, x_v\}$ and $\mathcal{A} = \{A_1, A_2, \ldots, A_b\}$. The Incidence Matrix (Def. 1.11 of [66]) of $(X, \mathcal{A})$ is a $v \times b$ matrix $M = (m_{ij})$ where,

$$m_{ij} = \begin{cases} 1 & \text{if } x_i \in A_j \\ 0 & \text{if } x_i \notin A_j \end{cases}$$

The incidence matrix, M, of a $(v, b, r, k,)$-BIBD satisfies the following properties:

1. every column of M contains exactly k many "1"s;

2. every row of M contains exactly r many "1"s;

3. two distinct rows of M both contain "1"s in exactly λ columns.

Definition 2.6. A **Partially Balanced Block design (PBD)** is a design in which each pair of points occurs in λ blocks, for some constant λ, called the index of the design.

Definition 2.7. The **intersection number** between any two blocks is the number of elements common to the blocks.

Definition 2.8. Let the intersection numbers between any the blocks in a BIBD be $\mu_1, \mu 2, \ldots, \mu_x$. Let $M = \mu_i : i = 1, 2, \ldots, x$. Let $\mu = max\{\mu_1, \mu_2, \ldots, \mu_x\}$. μ is called the linkage of the design.

Definition 2.9. A balanced incomplete block design is said to be **resolvable** if the set of b blocks can be partitioned into t classes such that each variety appears in exactly one class. The classes are called resolution classes of the design.

A detailed discussion of resolvable designs is presented in chapter 8 of [69].

Let, $(X, \mathcal{A})$ be a set system where $X = \{x_i : 1 \leq i \leq v\}$ and
$\mathcal{A} = \{A_j : 1 \leq j \leq b\}$.

The dual set system of (X, A) is any set isomorphic to the set system (X , A) where $X' = \{x'_i : 1 \leq i \leq b\}$ and $\mathcal{A}' = \{A'_j : 1 \leq j \leq v\}$ and where

$$x_i \in A_j \iff x'_j \in A'_i.$$

Hence, the dual of a (v, b, r, k, λ) BIBD is a design D^* with b varieties, v blocks, each block containing exactly r varieties and each variety occurring in exactly k blocks. Also it can be noted that any two blocks of D^* contains λ varieties in common.

Definition 2.10. An association scheme with m associate classes [[69], Section 11] 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 $1 \leq i \leq m$,
2. each element of X has n_i i-th associates, $1 \leq i \leq m$,
3. for each i, $1 \leq i \leq m$, 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, $n_i (1 \leq i \leq m)$ and p^i_{jl} $(1 \leq i, j, l \leq m)$ are called the parameters of the association scheme.

Definition 2.11. A partially balanced incomplete block design [69] with m associate classes is a design based on a v set X with b blocks and replication number r such that there is an association scheme defined on X such that if $\exists x, y \in X$ and x and y are i'th associates, $1 \leq i \leq m$ then they occur together in precisely λ_i blocks. The parameters $v, b, r, k, \lambda_i (1 \leq i \leq m)$ are called the parameters of the design. The design is denoted as $PB[k, \lambda_1, \lambda_2, \ldots, \lambda_m; v]$ design.

Definition 2.12. Let X be a set of varieties such that

$$X = \bigcup_{i=1}^{m} G_i, |G_i| = n \text{ for } G_i \bigcap G_j = \emptyset \text{ for } i \neq j$$

The G_is 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.

Definition 2.13. A t-(v, k, λ) is a design $(X, \mathcal{A})$ such that the following properties are satisfied:

1. $|X| = v$
2. $\forall B \in \mathcal{A}, |B| = k$
3. $\forall X' \subseteq X$, such that $|X'| = t, |\{B : B \in \mathcal{A}, X' \subseteq B\}| = \lambda$

A (v, k, λ)-BIBD is a t-(v, k, λ). A detailed study of t-design appears in [[66], Chapter 9]. According to the construction given in [[66], Chapter 9] the following result can be stated.

Definition 2.14. A Steiner triple system of order v, or $STS(v)$, is a $(v, 3, 1)$-BIBD. (Section 6.2 of [66])

Definition 2.15. A finite affine plane consists of a set of points and a set of lines, having the following properties:

1. Given any two distinct points, there is exactly one line incident with both of them.
2. Given any line L and a point p not on L, there exists exactly one line L' through p that does not intersect L
3. There are four points such that no line is incident with more than two of them.

This construction of affine planes can be found in [66, 69].

Let q be a prime. Let $P = \mathbb{Z}_q \times \mathbb{Z}_q$ be a set of points. $|P| = q^2$. We define,

$$L_{\alpha,\beta,1} = \{(x, y, 1) : (x, y) \in Z_q \times Z_q, (x, y) \neq (0, 0)\},$$

$$L_{1,\beta,0} = \{(1, x, 0) : x \in Z_q\}$$

and

$$L_{0,1,0} = \{(0, 1, 0)\}.$$

Let $L = L_{\alpha,\beta,1} \bigcup L_{1,\beta,0} \bigcup L_{0,1,0}$.

It can be shown that taking P as the set of points and L as the set of lines, we have an affine plane over $GF(q)$. This affine plane is denoted as $AG(2, q)$. For a proof we refer the readers to [69].

We see that $(\alpha, \beta) \neq 0$. For every point $|L_{\alpha,\beta,1}| = q^2 - 1$, $|L_{1,\beta,0}| = q$ and $|L_{0,\beta,0}| = 1$. So, there are q^2 points and $q^2 + q$ lines.

Consider $L_{\alpha,\beta,1}$ and $L_{1,\beta,0}$. If $\alpha \neq 0$, then $x = \alpha^{-1}\gamma - \alpha^{-1}\beta y$, $\gamma = \{0, 1\}$. For every $y \in \mathbb{Z}_q$ we will have a unique value of x. Since there are q elements in $\mathbb{Z}_q$, so there are q number of points passing on the line $\alpha x + \beta y = \gamma$ where $\alpha \neq 0$.

Alternatively, if $\alpha = 0$, then β must be not equal to 0. So, from $L_{\alpha,\beta,1}$ it follows that $y = \beta^{-1}$. Here y is constant and for each element of $\mathbb{Z}_q$, we will have a distinct value of x. So in both the cases there are exactly q points on a line.

The number of lines through a point (x, y) when $(x, y) \neq (0, 0)$ is the number of lines through (x, y) in $L_{\alpha,\beta,1}$ (i.e. not passing through origin) plus the number of lines in $L_{1,\beta,0} \bigcup L_{0,\beta,0}$ which pass through the origin. Since the number of lines through two points is one, so the number of lines through (x, y) and $(0, 0)$ is 1. Now the number

of lines through (x, y) and not passing through origin is the number of solution pairs (α, β) to the equation $\alpha x + \beta y = 1$. This equation can be written as $\alpha x = 1 - \beta y$ and as $\beta \in \mathbb{Z}_q$ can take q values, α is known. So this equation has q solutions. Hence, the number of lines through a point (x, y) when $(x, y) \neq (0, 0)$ is $q + 1$.

If (x_1, y_1) and (x_2, y_2) are two distinct points, then a unique line through these points is given by

$$(y_2 - y_1)x + (x_1 - x_2)y = y_2x_1 - y_1x_2$$

Definition 2.16. A Latin Square (LS) (def. 6.1 of [66]) of order n is an $n \times n$ array L such that each element in the array belongs to a set X of cardinality n and each row of the L is a permutation of elements in X and each column of X is a permutation of elements in X.

Example 2.1. *A Latin Square of order 4 where* $X = \{1, 2, 3, 4\}$ *is given by;*

1	*2*	*3*	*4*
4	*1*	*2*	*3*
3	*4*	*1*	*2*
2	*3*	*4*	*1*

Note that all four rows and four columns are permutations of $\{1, 2, 3, 4\}$.

Definition 2.17. (def. 6.19 of [66]) Let L_1 and L_2 are Latin Squares of order n defined on the set X_1 and X_2 respectively. Then, L_1 and L_2 are *Orthogonal Latin Squares* provided that for every $x_1 \in X_1$ and for every $x_2 \in X_2$, there is a unique cell (i, j); $i, j \in \{1, 2, 3, \ldots, n\}$ such that $L_1(i, j) = x_1$ and $L_2(i, j) = x_2$.

2.2 Key Predistribution in WSN

We have seen in 1.2.1.1 that key predistribution is of three types, viz. probabilistic, deterministic and hybrid key predistribution.

We now discuss two approaches which form the basis of several key predistribution schemes. These schemes, not initially intended for WSN, have been modified and suitably applied by several researchers for key predistribution in sensor networks. This thesis too introduces a key predistribution scheme that make use of one of them. A comparison of different key predistribution scheme that are analogous to the context of this thesis is given in table 2.1.

2.2.1 Blom's Scheme

Blom [10] proposed a scheme for pairwise key establishment between a set of N users. The distribution server first chooses a $t \times N$ matrix G over a finite field $GF(q)$. t is called the security parameter of the scheme. The matrix G is considered to be a public information. Now the distribution server constructs a $t \times t$ symmetric matrix D over $GF(q)$. This matrix is a private information of the system. Now, the server computes the $t \times N$ matrix A, where $A = (DG)^T$, T being the transposition operator. Now, $AG = (DG)^T G = G^T D^T G = G^T DG = G^T A^T = (AG)^T$.

Thus AG is a symmetric matrix. Let $K = AG$, we know that $K_{ij} = K_{ji}$, where K_{ij} is the element in K located in the i^{th} row and j^{th} column. K_{ij} (or K_{ji}) is the pairwise key between node U_i and node U_j. To carry out the above computation, nodes U_i and U_j should be able to compute K_{ij} and K_{ji}, respectively. This can be easily achieved using the following key predistribution scheme, for $w = 1, 2, \ldots, N$,

1. Store the w^{th} row of matrix A in node U_w.
2. Store the w^{th} column of matrix G in node U_w.

Now, if two nodes (say U_x and U_y) want to communicate, they need to establish a common key. Node U_x has row x of A and column x of G. Node U_y has row y of A and column y of G. Now , they can establish a pairwise key this way:

1. Node U_x and U_y exchange column x and column y of matrix G respectively.
2. Node U_x calculates K_{xy} =(row x of A).(column y of G).
3. Node U_y calculates K_{yx} =(row y of A).(column x of G).

The matrix G is a public information. Therefore the rows of G could be sent without encryption. Since K is a symmetric matrix, $K_{xy} = K_{yx}$. Hence K_{xy} can be used as the common key between the two nodes.
c-secure property: It has been proved that the above scheme is t-secure [10] i.e. if any $t+1$ columns of G are linearly independent, then no member other than U_x and U_y can compute K_{xy} or K_{yx} if no more than t members are compromised.

2.2.2 Blundo Scheme

This scheme was proposed by Blundo, Santis, Herzberg, Kutten, Vaccaro, Yung [11] and was not originally used for sensor networks. It uses a symmetric bivariate polynomial

$P(x, y)$ over some finite field $GF(q)$, i.e. 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 of the polynomial, P which is $f_i(y) = P(i, y)$. In order to communicate with node Uj, it computes the common key $K_{ij} = f_i(j) = P(i, j) = P(j, i) = f_j(i)$; this process enables any two nodes to have 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. These are elements of $GF(q)$, as are the pairwise keys that are established; thus, storing a degree t share requires as much space as storing $t + 1$ keys. If an adversary captures s nodes, where $s \leq t$, then it does not learn any information about keys established between uncompromised nodes; however, if it captures $t + 1$ or more nodes then it can interpolate to compute the polynomial P and hence learn all the keys. In the next few sections we discuss several key predistribution schemes for various kinds of networks.

2.3 The Basic Scheme

This scheme was the first probabilistic key predistribution scheme ever. It was proposed by Eschenauer and Gligor in ACMCCS'02 [27]. Many probabilistic key predistribution techniques use this as the underlying scheme. In this scheme a pool of randomly generated keys is selected first. The size of the key pool is K_P. Keys are drawn from the pool at random and are loaded into the sensor nodes. Every key that is drawn is replaced back into the key pool. In this scheme two nodes may not have any key in common.

2.4 q-composite Scheme

A variation of the basic scheme was introduced by Chan, Perrig and Song [19]. This scheme is called the q-composite scheme. Here, two nodes can communicate only if they have $q' \geq q$ keys in common. The key predistribution is similar to the Basic Scheme. Here, all nodes are loaded with exactly k keys randomly drawn from the key pool of size K_P same as the basic scheme. Afterwards, two nodes can only communicate directly if they share at least q common keys and the shared key is computed from the q many common keys.

2.5 Random Pairwise Scheme

In a pairwise scheme each pair of nodes share a secret key with each other. For a network consisting of N nodes, each node has $N-1$ keys which it shares with the $N-1$ nodes. As already mentioned in chapter 1, this is a huge burden on the sensor, since sensors have very limited memory. We now discuss some pairwise schemes.

2.5.1 Chan-Perrig-Song scheme

Chan Perrig and Song [19] proposed a modification of the Random Pairwise Scheme. In this scheme a node a shares a pairwise key with another node b with some probability p_c. Hence, in this scheme every pair of nodes do not necessarily share a pairwise key. Thus, every node stores $k = Np_c$ keys. Looking the other way round, the maximum size of the network that can be supported is $N = k/p_c$. This scheme reduces the storage overhead of the pairwise key predistribution scheme at the cost of connectivity.

2.5.2 Yağan-Makowski's results for Chan et al. scheme

Yağan and Makowski [79] investigated the connectivity of wireless sensor networks under the random pairwise key predistribution scheme of Chan et al. [19] under the assumption of full visibility. Since, the key graph in Chan et al. scheme happens to be a k-out random graph, where k is the number of keys stored in each node, their study got reduced to investigating the connectivity in $\mathbb{H}(N;k)$, N being the size of the network.

2.5.3 Liu-Ning-Li polynomial-pool-based key predistribution

The pairwise key scheme of Liu and Ning [42] uses the scheme of Blundo et al. [11] as an underlying scheme. In this scheme instead of a unique polynomial, a pool of polynomial is used for key predistribution. A node is given shares from a fixed number of polynomials from this pool. Two nodes can have a common key if they have polynomial shares from the same polynomial.

2.5.4 Delgosha-Fekri scheme

Delgosha and Fekri [22] proposed Key Pre-distribution in Wireless Sensor Networks Using Multivariate Polynomials. Proposed scheme, called hypercube multivariate scheme (HMS), is an improvement to the hypercube based scheme of [42] by using multivariate

polynomials. An n-dimensional hypercube is designed. The coordinates of the hypercube correspond to a sensor nodes each. Each sensor node gets its polynomial shares from nm symmetric n-variate polynomials. Sensor nodes who have polynomial shares from a common n-variate polynomial can compute a shared key. Every two sensors at Hamming distance of one from each other can establish a direct key.

The authors proposed a grid based key predistribution scheme in the same paper. This scheme will be discussed in section 2.6.

2.5.5 Rasheed-Mahapatra's scheme

Rasheed and Mahapatra proposed "Key Predistribution Schemes for Establishing Pairwise Keys with a Mobile Sink in Sensor Networks" in [58]. In this paper, they exploit the use of either the probabilistic generation key predistribution scheme [31] or the Q-composite scheme [19] in conjunction with the polynomial pool-based key predistribution scheme [42] to establish a secure link between a mobile sink and a sensor node to improve network resilience to node captures. First, they propose a scheme that combines the polynomial pool-based key predistribution [42] with the probabilistic generation key predistribution scheme [31] to establish a pairwise key between mobile sink and any sensor node. Second, they develop a scheme that uses the Q-composite scheme in conjunction with polynomial pool-based scheme. The two proposed schemes guarantee that any sensor node can establish a pairwise key with a mobile sink with high probability and without sacrificing security.

2.5.6 MKPS scheme by Delgosha et al.

Delgosha, Ayday and Fekri proposed "MKPS: A Multivariate Polynomial Scheme for Symmetric Key-Establishment in Distributed Sensor Networks" in [21]. In this paper, the authors propose a multivariate key pre-distribution scheme (MKPS). In this scheme, a large set of symmetric multivariate polynomials is generated by the sink prior to the network deployment. Every sensor node is uniquely assigned an ID that is a d tuple consisting of non-negative integers. These IDs are used to assign d d-variate polynomials to every node. For every node, the shares of these polynomials are stored in its memory. They showed that in this setting every two nodes with IDs at the Hamming distance of one from each other have shares of the same $d-1$ multivariate polynomials. Using these shares, these nodes can establish $d-1$ common keys.

2.5.7 Probabilistic scheme of Zhu et al

Zhu, Xu, Setia and Jajodia [84] proposed a scheme for establishing pairwise keys. They used probabilistic key sharing [13] and threshold secret sharing [64]. With this scheme a pair of nodes can establish a secret key on the fly. In this scheme a deterministic algorithm is used to distribute a set of keys to a set of nodes. Prior to the commencement of communication two nodes establish a shared key on the fly by method of secret sharing whereby the key is divided into some parts and passed to the recipient through different logical paths so that none other than the intended recipient could get all shares necessary for reconstructing the key.

2.6 Grid-based predistribution schemes

2.6.1 PIKE scheme

PIKE [18] scheme was proposed by Chan and Perrig. In this scheme keys are established between two nodes with the help of other nodes acting as trusted intermediaries. Suppose the maximum number of nodes in a network is N. All the nodes are arranged in a $\sqrt{N} \times \sqrt{N}$ square grid structure. A node has identifier (x, y) if it is positioned in the point of the grid having coordinate (x, y). The deployment order of the nodes is $(0,0), (0,1), \ldots, (0, \sqrt{N}-1), (1,0), (1,1), (1,2), \ldots, (1, \sqrt{N}-1)$ and so on. A node (x, y) stores pairwise keys with all the nodes with ids in the set $\mathcal{I}_{x,y} = \{(x, y') : 0 \leq y' \leq \sqrt{N}-1\} \bigcup \{(x', y) : 0 \leq x' \leq \sqrt{N}-1\}$. Each key is unique and shared only between two nodes, hence they are called pairwise keys. Thus each node stores $2(\sqrt{N}-1)$ keys.

2.6.2 Kalindi et als Scheme

Kalindi, Kannan, Iyengar and Durresi [33] provided a modification of the PIKE scheme. In this scheme the $\sqrt{N} \times \sqrt{N}$ grid of PIKE is further divided into $l \times l$ cells containing $m \times m$ keys, where $m = \lfloor \frac{\sqrt{N}}{l} \rfloor$. Let K_{ij} be a unique key positioned at the coordinate (i, j) on the grid. Also let C_{xy} be a cell in the grid and SG_{xy} be the grid consisting C_{xy} and its 8 neighboring cells. Then the key vector for a node U_{ij} is given by

$$V_{ij} = \bigcup K_{ic'} + \bigcup K_{r'j},$$

where, $(y-1) \mod k \times m < c' < (y+1) \mod l \times m$ and
$(x-1) \mod k \times m < r' < (x+1) \mod l \times m$.

2.6.3 Sadi-Kim-Park Scheme

Sadi, Kim and Park [63] proposed another grid-based random scheme based on bivariate polynomials. This scheme is called Grid-Based Random key predistribution scheme. In this scheme the total number of sensor nodes is N. They are arranged in an $m \times m$ grid where $m = \sqrt{N}$. The setup server generates $2m\omega$ bi-variate polynomials $F = \{f_i^c(x, y) : 0 \leq i \leq m\omega - 1\} \bigcup \{f_i^r(x, y) : 0 \leq i \leq m\omega - 1\}$ of degree t. The coefficients of these polynomials belong to F_q. The server divide them to group and assigns them to the rows and columns of the grid. In these technique a pair of nodes can establish a common key if they share a common polynomial.

2.6.4 Mohaisen-Maeng-Nyang scheme

Mohaisen, Maeng and Nyang [52] introduced a 3–dimensional grid based key predistribution scheme. They considered an $m \times m \times m$ grid as deployment zone where $m = \sqrt[3]{N}$, N being the number of nodes in the network. A set P of $3\sqrt[3]{N}$ symmetric bi-variate polynomials over a finite field F_q are chosen for key predistribution. Polynomial shares are distributed to the nodes in such a fashion that every node gets polynomial shares from three polynomials and two nodes can have a common key if they belong to the same row of any of the three axes of the grid.

2.6.5 Delgosha and Fekri's scheme

Farshid Delgosha and Faramarz Fekri discussed a grid based key predistribution scheme in [22]. In this approach, they assign the points on a two-dimensional grid to the cells. In addition, they assign a unique symmetric bivariate polynomial to every cell. In order to distribute the shares of this polynomial between sensors, they divide the sensors in every cell into equal-size groups. In every cell, the shares of the corresponding polynomial are distributed among the sensors in the groups. Moreover, the sensors of a cell store the shares of the polynomials corresponding to the neighbor cells. As a result, the neighbor cells are able to establish pairwise keys.

2.7 Group-based key predistribution

Sometimes we may not know the exact location of the sensor nodes deployed in the target zone. However, we may have the knowledge of the proximity of nodes. This information may be useful in devising new key predistribution schemes. Group based

key predistribution schemes are of two types, one that uses deployment knowledge and one that does not. Here, we discuss group-based key predistribution schemes which do not use deployment.

2.7.1 Liu-Ning-Du Scheme

Group-based key predistribution scheme without using deployment knowledge was proposed by Liu, Ning and Du [45, 46]. They proposed two key predistribution schemes for their proposed framework. They put sensor nodes in close proximity to each other in the same deployment group. The sensor nodes to be deployed are divided into n groups each consisting of m sensor nodes. All the sensor nodes belonging to the same deployment group are deployed together from the same point. After they are dropped the actual position of the sensor nodes follow a probability distribution function(pdf).There are two key predistribution schemes for this deployment. One of them deals with establishing pairwise keys within each group and the other is used to establish pairwise keys in different deployment groups (called the cross-group predistribution). Within a group any existing key predistribution scheme can be used for key establishment. The cross groups are built such that each cross-group includes exactly one sensor node from the deployment group and there are no common sensor nodes between any two different cross groups.

2.7.2 Martin-Paterson-Stinson's improvement of Liu et al's scheme

Martin, Paterson and Stinson [49] proposed an improvement of Liu et al's scheme. They used resolvable transversal design for developing a group-based key predistribution scheme. They modified the cross group connectivity of Liu et al scheme by having each node contained in m cross groups instead of one. The structure of the resolvable transversal design implies that each node is contained in precisely m cross groups, each cross group contains at most one node from each group, and two cross groups have at most one node in common.

2.8 Key Predistribution using Combinatorial Designs

Combinatorial designs were first used for key predistribution by Mitchell and Piper [51]. However, Çamptepe and Yener [14] were first to apply designs in key predistribution for wireless sensor networks.

We now describe how a key predistribution scheme can be applied for key predistribution in a WSN. Let us assume that there is a set $\mathcal{N}$ of wireless sensor network consisting of N sensor nodes. Also assume that there is a combinatorial design $(X, \mathcal{A})$ where $|\mathcal{A}| \geq N$. We choose a randomly generated set $\mathcal{K}$ of $|X|$ symmetric keys. We choose two mappings $f : \mathcal{K} \to X$ and $g : N \to \mathcal{A}$. Now, the key ring of the i^{th} node of the WSN is given by:

$$\forall n_i \in N, KeyRing(n_i) = \{K : K \in \mathcal{K}, f(K) \in g(n_i)\}.$$

Now, we discuss some key predistribution scheme based on combinatorial design.

2.8.1 Çamtepe and Yener's scheme

We have already mentioned that Çamtepe and Yener [14, 15] were the first to have applied combinatorial designs in key predistribution for wireless sensor networks. They used symmetric balanced incomplete block design and projective geometry. They used projective geometry $PG(2, q)$ in the construction of SBIBD where q is a prime power. In this scheme every pair of nodes have a common key. This scheme supports a total of $q^2 + q + 1$ nodes, each node containing $q + 1$ keys. The total number of keys in the network is also equal to $q^2 + q + 1$. A key is assigned to $q + 1$ nodes.

The other predistribution scheme involves generalized quadrangles. Three known designs for generalized quadrangles have been used : $GQ(q, q), GQ(q, q^2)$ and $GQ(q^2, q^3)$. The construction of $GQ(q, q), GQ(q, q^2)$ and $GQ(q^2, q^3)$ have been done from $PG(4, q), PG(5, q)$ and $H(4, q^2)$ respectively. Although the first scheme using symmetric design results in full connectivity of the network, the resiliency is poorer than the second scheme using generalized quadrangles.

2.8.2 Lee Stinson's Scheme

Lee Stinson [38] formalized the definition of key predistribution using combinatorial design. They proposed key predistribution scheme using transversal design. In this scheme two nodes can have 0 or 1 common shared key between them. Lee Stinson generalized their scheme in [40].

2.8.3 Chakrabarti-Maitra-Roy Scheme

Chakrabarti, Maitra and Roy [16] proposed a hybrid key predistribution scheme. They used the transversal design based scheme of Lee and Stinson in [38] and randomly

merged the blocks of the original design. Then those merged blocks are used for key predistribution in the sensor nodes. This scheme uses more memory than [38] for storing the additional keys corresponding to the merged blocks. However, this scheme improves the resilience of the original key predistribution scheme.

2.8.4 Dong, Pei and Wangs scheme

Dong, Pei and Wang [23] used 3–design for developing a key predistribution scheme. They constructed a 3–design from $\mathbb{F}_{q^2}$, where q is a prime number. Dong et al. considered the $q^3 + q$ many distinct blocks of the 3– design and mapped the distinct blocks to the nodes of the WSN. Hence, the number of nodes supported by the network is $q^3 + q$. A pair of nodes can share at most two keys. The disadvantage of this method is that resiliency reduces drastically as the number of nodes compromised increases.

2.8.5 Product construction of Wei and Wu

Wei and Wu [74] formalized the method of Du et al. [25] and Liu and Ning [42] who used Blom scheme as an underlying scheme in their design. Their scheme is based on the product of key distribution scheme and set systems. They deduced conditions of the combinatorial designs that optimize the performance of the network in terms of connectivity and resiliency.

2.8.6 Key predistribution scheme of Ruj & Roy

Ruj and Roy used partially balanced incomplete block design in [60]. Ruj and Roy proposed a scheme using triangular PBIBD 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. The authors also proposed a novel key predistribution scheme using Reed Solomon code in [61]. This scheme does not offer full connectivity and but has got constant time shared key discovery algorithm.

2.9 Key predistribution using Deployment knowledge

Sometimes sensor nodes are deployed in a pre-determined way. For example, they can be dropped from an airship. In that case, sensor nodes which are dropped together remain in close proximity to each other upon deployment. Although it may not be possible to

precisely pinpoint sensors locations, it is often possible to approximately determine their locations.

There are many key predistribution schemes in literature that used deployment knowledge. In this section we provide brief description of some of the well known schemes.

2.9.1 Liu-Ning Scheme

Liu and Ning [43]. They proposed two key predistribution schemes that exploit the deployment knowledge. The first scheme was closest pairwise scheme. This scheme, as the name implies, uses pairwise keys between sensor nodes lying close to each other. The second scheme uses the polynomial-based key predistribution scheme of Blundo et al. [11]. The deployment region is broken down into equal sized squares , each of which is a cell with coordinates. Each of the cells is associated with a bivariate polynomial. The setup server distributes to the sensor the coordinates of the home cell and the polynomial shares of the home cell and its neighboring cells. For direct key establishment a node broadcasts the coordinates of its home cell. From this coordinate the destination node finds out the common polynomial that it shares with the broadcasting node if at all. Now the common key can be calculated using the same method as [11].

2.9.2 Du et al's Scheme

Du et al proposed a key predistribution scheme using deployment knowledge in [24] which they extended in [26]. They used a grid-group based deployment. The deployment zone is of the shape of a grid. In this scheme the total number of sensor nodes is N divided into $t \times n$ equal sized groups such that all sensors in the same group $G_{i,j}, 1 \leq i \leq t, 1 \leq j \leq n$ is deployed from the deployment point with index (i, j).

Du et al used Blom's scheme [10] for key predistribution among the sensor nodes. But instead of one they used multiple space Blom scheme. They used ω key spaces of Blom scheme. Each node is assigned information from $\tau, 2 \leq \tau \leq \omega$ different key spaces of Blom scheme chosen at random from the ω key spaces. Two sensor nodes can directly compute a common key if they possess information from at least one common key space. However there is not guarantee that two nodes will have some key in common.

2.9.3 Yu-Guan Scheme

Yu and Guan [81, 82] proposed another key predistribution scheme using deployment knowledge. They discussed the effect of deployment on triangular, hexagonal and square

grids. According to this scheme, the entire deployment area is split into t grids equally. The shape of grids may vary. The total number of sensor nodes is N. These N sensor nodes are divided equally into t groups, one for each grid. The sensor nodes in group i is deployed in grid i. It is assumed that the location of the nodes of each group i follows some probability distribution function. They also used Blom [10] scheme for key predistribution. Their scheme requires low storage and offer full connectivity between nodes within communication range.

2.9.4 Huang et al's scheme

Huang, Mehta, Medhi and Harn [29, 30] proposed another key predistribution that uses deployment knowledge. According to this scheme the target zone is a $i.a \times j.a$ rectangular area. This region is broken down into $i * j$ square regions each having an area equal to a^2. The total number of N sensor nodes are equally distributed among ij groups. Each group gets n_z sensor nodes, where $n_z = N/i.j$.

Keys are loaded into the sensor nodes following the multiple space Blom scheme similar to Du et al scheme [26]. For key predistribution in nodes from different groups, pairwise key predistribution is chosen. For key establishment within the same group the nodes need to identify the common key space between them.

2.9.5 Simonova-Ling-Wang Scheme

In [65], Simonova et al discussed two key predistribution schemes. In both the schemes, the deployment zone is broken down into grids. Sensor nodes are deployed in these grids following the scheme in [26]. There are two types of key pool: the original key pool and the deployment key pool. If the grid size is $m \times m$, then the number of deployment key pool is m^2, one for each group. Simonova et al used transversal design for key predistribution.

2.9.6 Zhou-Ni-Ravishankar scheme

Zhou Ni and Ravishankar [83] also used deployment knowledge. They discussed a key predistribution scheme where sensor nodes can be mobile. According to their scheme, there are two types of nodes. The static nodes are deployed in groups. Apart from them there are mobile collectors which are used to collect and aggregate sensor data and forward to the base station. There are n_s sensor nodes and n_m mobile collectors. The static sensors are arranged in g groups $G_i, 1 \leq i \leq g$. Each group contains $\gamma = n_s/g$

sensors. Group G_u comprises of sensors s_i such that $(u-1)\gamma < i \leq u\gamma$. Sensor nodes within a group are connected to each other using pairwise keys. Nodes of different groups are connected by agents.

2.9.7 Wang-Chen scheme

Neng-Chung Wang1 and Hong-Li Chen [73] proposed a grid-based pairwise key predistribution scheme for wireless sensor networks. In the proposed scheme, multiple polynomials for each row, each column, and each diagonal in the grid are constructed. Then, each sensor node in each row, column, and diagonal in the grid establishes a pairwise key with the other nodes using the predistributed symmetric polynomials. The setup server distributes identity and 4τ polynomials to the sensor node. If two sensor nodes share the same polynomial, they can establish a pairwise key.

2.9.8 Ruj-Roy scheme

Ruj and Roy proposed a key predistribution scheme that uses deployment knowledge in [62]. They considered a deployment zone which is divided into $r \times r$ square grid. Each of the group contains two types of nodes viz. common nodes and agents. Common nodes can communicate within the group to which they belong. Two common nodes from different groups can communicate through some special nodes called agents. According to Ruj-Roy scheme there are three agents per group. They chose the scheme of Çamptepe and Yener in [14, 15] for key predistribution among nodes in a group. Apart from these they used a transversal design based key predistribution scheme for the agents across the deployment zone.

Scheme	K	M	R	C	T
Blom [10]	1	$t+1$	t-secure	$t+1$	$t+1$
Blundo [11]	1	$(t+1)\log q$	t-secure	$\log N$	$t+1$
Probabilistic					
Basic Scheme	$\frac{(K_P-k)!^2}{(K_P-2k)!K_P!}$	k	$\frac{k}{K_P}$	$O(k \log K_P)$	$k \log k$
Q-composite random	$\frac{(K_P-k)!^2}{(K_P-2k)!K_P!}$	k	$\binom{k}{q}$	$O(k \log K_P)$	$k \log k$
Random Pairwise					
Chan-Perrig-Song [19]	$\frac{(K_P-k)!^2}{(K_P-2k)!K_P!}$	Np_c	$\frac{k}{K_P}$	$O(k \log K_P)$	$k \log X$
Liu-Ning-Li [42, 44]	Give in [42] section 4.1	$s'(t+1)\log q$	t-secure	$s' \log \lvert\mathcal{F}\rvert$	$O(t)$
Zhu et al. [84]	1	k	Given in section 4.1 of [84]	$O(\log N)$	no of shares
Delgosha-Fekri scheme [22]	$\frac{n(m-1)}{N-1}$	$n(t+1)\log q$	t-secure	$L-1$	$O(t)$
MKPS [21]	$\leq \frac{d(m-1)}{n-1}$	$d(t+1)\log p + d \log m$	$\lambda(r,t)-1$-secure	$d \log m$	$O((d-1)(t+1))$
Grid based scheme					
PIKE [18]	$\frac{1}{\sqrt{N}}$	$2\sqrt{N}-1$	$\frac{1}{\sqrt{N}}$	$O(\log N)$	$O(1)$
Kalindi et al. [33]	Given in [33], section IIB	$1-\frac{12m-6}{N}$	$6\frac{\sqrt{N}}{l}-1$	$O(\log N)$	$O(1)$
Sadi-Kim-Park [63]	$\frac{2}{\sqrt{N}+1}$	$2\tau(t+1)\log q$	t-secure	$O(\log N)$	$O(\tau \log N \omega)$
Mohaisen-Maeng-Nyang [52]	$\frac{3}{\sqrt[3]{N}+1}$	$3((N_c+1)\lceil \log N \rceil^{\frac{1}{3}} + \sqrt[3]{N}(t+1)q)$	Given in [52] section 3.6	$1.5\lceil \log N \rceil$	$O(t)$
Delgosha-Fekri [22]	$\frac{n(m-1)}{N-1}$	$7(t_c+1)\log q_c + n(t+1)\log q$	t-secure	$O(\log N)$	$O(t)$
Group based scheme					
Liu-Ning-Du [45, 46]	Given in [46]	$\frac{m+n}{2}$	Given in [46]	$O(\log N)$	$O(H)$
Liu-Ning-Du [45, 46] (Polynomial)	Given in [46]	$\frac{m+n}{2t}\log q$	Given in [46]	$O(t)$	$O(t)$
Martin-Paterson-Stinson [49]	Given in [49]	$(m+1)(t+1)$	Given in [49]	$O(\log N)$	$O(1)$
Combinatorial design based scheme					
Çamtepe and Yener [12, 14]	1	$q+1$	$\frac{q-1}{q^2+q+1}$	$O(\log N)$	$O(1)$
Lee-Stinson [38]	$\frac{k}{r+1}$	k	$\frac{r-2}{b-2}$	$\log N$	$O(\log r)$
Chakrabarti, Maitra and Roy [16]	Given in [16]	$zk-\binom{z}{2}\frac{k}{r+1}$	Given in [16]	$O(z \log N)$	$z \log r$
Dong et al. [23]	0.5	$q+1$	Given in [23]	$O(\log q)$	$O(q)$
Ruj-Roy [60]	1	$2(n-2)$	Given in [60]	$O(\log n)$	$O(1)$
Ruj-Roy [61]	Theorem 4.4.1 of [61]	k	given in [61]	$dim \log q$	$O(1)$
Scheme in chapter 3	1	$4q-2$ to $4q+1$	Given in section 3.3.3	$O(logq)$	$O(1)$
Scheme in section 4.3	1	$O(\sqrt{N})$	Given in section 4.4	$O(\log N)$	$O(1)$
Deployment knowledge based scheme					
DDHV [26]	Given in [26]	$\omega\tau \log q$	t-secure	$O(\tau)$	$(t+1)\omega \log \omega$
LN [42, 44]	Given in [42, 44]	$(t+1)\log q$	Given in [42, 44]	$O(\log C \log R)$	$O(t)$
YG [81, 82]	1	$N(t+1)$	t-secure	$t \log t$	$Nt \log t$
ZNR [83]	Given in [83]	$O(\gamma)$, $O(n_s)^2$	Given in [83]	$O(\log N)$	depends on node type
HMMH [30]	Given in [30]	$\tau(\lambda+1)$	Given in [30]	$O(\tau)$	ωN
SLW [65]	Given in [65]	$O(\sqrt{N/g})$	Given in [65]	$O(\log p')$	$O(1)$
RR [62]	1	$O(p)$ $O(q')$	Given in [62]	$O(\log p)$ $\log q'$	1 $O(\log q')$
Scheme in section 4.5	1 1	$O(N')$ $q+1$	given in section 4.5.2 $\frac{q-1}{q^2+q+1}$	$O(\log N')$ $O(\log N)$	$O(1)$ $O(1)$
Scheme in chapter 5	1 1	$O(q)$ $q+1$	Given in 5.3.5 $\frac{q-1}{q^2+q+1}$	$O(\log q)$ $O(\log N)$	$O(\log q)$ $O(1)$

TABLE 2.1: Table of comparison of different key predistribution schemes Here K is the connectivity ratio, M is the memory requirement per node, R is the resiliency of the key predistribution scheme, C is the communation overhead and T is computational cost to establish shared key.

2.10 Jamming Resistant Wireless Communication

In this section we discuss the existing works for countering jamming attack in wireless communication. So far a lot of work has been done in this area. As discussed in the previous chapter, there are two types of schemes that provides a remedy against jamming attacks, viz. Direct Sequence Spread Spectrum and frequency hopping spread spectrum. DSSS uses spreading codes to modulate the data signal to a much wider band while frequency hopping uses channel switching to evade a jammer. Frequency hopping is effective to counter narrowband jamming.

There are many research works on frequency hopping. These works take advantage of different techniques that does as well as does not need pre-shared secrets shared between the communicating devices. Most of these works are meant for two party communication where they depend on the use of a secret key to generate pseudorandom number(PN) sequences using the secret key as seed. Examples of such schemes can be found in [28, 53]. There are schemes like [57, 67, 68] that do not use PN sequences. The communicating devices hop over the available channels depending on this sequence. Such techniques work nicely when the number of users is two or there is one sender and a group of receivers, all listening to the same broadcast from the sender. When there are many users communicating exclusively to each other this technique fails.

There are schemes like [35] that deal with control channel jamming. Control channels are different from communication channels in the sense that control channels are frequency bands used to broadcast messages for coordinating network functions [35]. [70] uses random key predistribution scheme for establishing control channels for pair of nodes but the existence of a common channel between a pair of nodes depend on the random key graph. Li et al. [41] discussed anti-jamming communication in a single channel network. Xu et al. [77] proposed evasion strategy where nodes move from a jammed channel to a jam-free channel together. These schemes are aimed at providing a single jam-free channel for all users. There is only one scheme by Lee et al. [36], that considers multiple user network and ensures a rendezvous between a pair of users in bounded time. We have given a comparison of some frequency hopping schemes that are similar to our work in table 2.2.

2.10.1 Classical Frequency Hopping

Navda, Bohra, Ganguly and Rubenstein proposed to use channel hopping to increase 802.11 Resilience to Jamming Attacks in [53]. In this paper they considered the communication between an access point(AP) and a legitimate client over an 802.11 based

wireless network. In order to implement channel hopping in a manner that is secure from the jammer and minimize the throughput loss, they assumed that the channel hopping sequence is pseudo-random, and that this sequence is known by only the AP and the legitimate client. When there is a group of users, this scheme can be used for communication only when each of the nodes uses a different pseudorandom number sequence. In such scenario, two nodes will be able to communicate if they by chance meet on a different channel at some point of time.

2.10.2 Gummadi et al. scheme

Gummadi, Wetherall, Greenstein and Seshan proposed "Understanding and Mitigating the Impact of RF Interference on 802.11 Networks" in [28]. In this paper, they make three contributions. First, they quantify the extent and magnitude of 802.11s vulnerability to interference, and relate the causes of such vulnerability to design limitations in commodity NICs. Second, they extend the SINR model to capture these limitations, and quantify how their extended version can be used to predict the high interference degradation with even weak and narrow-band interferers seen in practice. They also use the model to show that changing 802.11 operational parameters would be ineffective at mitigating this degradation, while channel hopping can be helpful. Third, they implement and evaluate a rapid channel hopping scheme that can withstand even multiple strong interferers in a realistic setting, at a reasonable cost in terms of channel switching overheads. The novelty of our design lies in combining rapid channel hopping at the driver-level with Direct Sequence Spread Spectrum (DSSS) at the physical layer at the same time.

2.10.3 DEEJAM scheme by Wood et al

Wood, Stankovic, and Zhou [75] presented a protocol for defeating energy-efficient jamming in networks based on IEEE 802.15.4 compatible hardware. They discussed four defense mechanisms for countering jamming attack. These are frame masking, channel hopping, packet fragmentation and redundant encoding. Each of the four defense mechanisms layer atop one another to allow operation even in the presence of an ongoing jamming attack.

DEEJAM is able to recover from much of the packet loss caused by jamming. The general design approach for DEEJAM is to hide messages from a jammer, evade its search, and reduce the impact of messages that are corrupted anyway. Their protocol requires that the jammer increase its effort substantially to continue to cause disruption, which also increases the opportunity for finding and removing it by external means. The authors

showed that the neighborhood of a jammer is rendered completely unusable when no defenses are used.

2.10.4 Xu-Wood-Trappe-Zhang scheme

Xu, Wood, Trappe and Zhang [78] proposed a scheme for defending denial of service attack in wireless communication. They presented two strategies that may be employed by wireless devices to evade a MAC/PHY-layer jamming-style wireless denial of service attack. The first strategy, channel surfing, is a form of spectral evasion that involves legitimate wireless devices changing the channel that they are operating on. The second strategy, spatial retreats, is a form of spatial evasion whereby legitimate mobile devices move away from the locality of the DoS emitter.

The first escape strategy that they present is channel surfing. Typically, when radio devices communicate they operate on a single channel. When an adversary comes in range and blocks the use of a specific channel, it is natural to migrate to another channel. The second escape strategy that they propose is spatial retreats. The rationale behind this strategy is that when mobile nodes are interfered with, they should simply move to a safe location.

2.10.5 Xu-Trappe-Zhang scheme

Xu, Trappe and Zhang [77] explored two different approaches for channel surfing: coordinated channel switching, where the entire sensor network adjusts its channel; and spectral multiplexing, where nodes in a jammed region switch channels while nodes on the boundary of a jammed region act as radio relays between different spectral zones.

In coordinated channel switching, the entire network must coordinate its evasion of the interference by switching to the next channel and resuming network operation there. The strategy involves a transition phase during which an increasing amount of nodes switch to the next channel. Following the transition, the entire network resumes stable operation on the next channel. In this protocol if a node detects jamming it moves to the next interference-free searching for its lost neighbors after executing a channel switch command.

2.10.6 Li-Koutsopoulos-Poovendran's scheme

Li, Koutsopoulos and Poovendran [41] considered jamming in a single channel wireless sensor network. The jam detection algorithm proposed in this work decides whether

there is an attack based on the observation samples obtained at the monitor node. The attack detection mechanism is as follows: During normal network operation, and in the absence of a jammer, there is a large enough training period in which the monitor node 'learns' the percentage of collisions it experiences as the long-term average of the ratio of number of slots in which there was a collision over total number of slots of the training period. Assume now the network operates in the open after the training period and fix attention to a time window much smaller than the training period. An increased percentage of collisions over this time window compared to the learned long-term average may be an indication of an ongoing jamming attack or only a temporary increase of percentage of collisions compared to the average during normal network operation. A detection algorithm takes observation samples obtained at the monitor node (i.e, collision or not collision) and decides whether there exists an attack.

2.10.7 Lazos-Liu-Krunz scheme

Lazos, Liu, and Krunz [35] addressed the problem of control-channel jamming in multi-channel wireless ad hoc networks. They considered a sophisticated adversary who exploits knowledge of protocol mechanics along with cryptographic quantities extracted from compromised nodes to maximize the impact of his attack in higher layers. They defined new security metrics for quantifying the adversary's ability to localize and deny legitimate nodes access to the control channel. The authors developed a randomized distributed channel establishment scheme that allows nodes to establish a new control channel using frequency hopping. Under this scheme, network nodes are able to temporarily construct a control channel until the jammer is removed from the network. The difference between this scheme and the classical frequency hopping scheme is that the communicating nodes are not synchronized on the same hopping sequence, but each node follows a unique hopping sequence. This leads to unique identification of the set of compromised nodes by nearby nodes.

2.10.8 Tague-Li-Poovendran's scheme

Tague, Li, and Poovendran [70] proposed "Probabilistic Mitigation Of Control Channel Jamming via Random Key Distribution". In this work, they proposed the use of random key distribution for resilience to control channel jamming and statistically characterize the performance as a function of the number of colluding or compromised users. They make use of results for secure communication in [71, 72], in developing key distribution and analyzing system performance. This approach allows the system designer to choose

the degree of probabilistic resilience to collusion or user compromise without fixing a threshold number of colluding or compromised users a priori.

2.10.9 Efficient Uncoordinated FHSS by Strasser et al

Strasser, Pöpper and Čapkun addressed the problem of jamming-resistant communication in scenarios in which the communicating parties do not share secret keys [67]. Their scheme is called "Efficient Uncoordinated FHSS Anti-jamming Communication". This scheme is applicable to scenarios where the deployment of shared secret keys is unrealistic, and therefore this problem cannot be solved using existing anti-jamming solutions like FHSS and DSSS that depend on pre-shared keys.

The FHSS scheme is simple. The sender and the receiver(s) hop among a set of known frequency channels in an uncoordinated and random manner. At any instant of time the sender transmits message on c_n channels and the receiver scans c_m channels. The sender hops with a higher speed than the receiver(s). Information is transferred when the receiver happens to listen on the same frequency channel on which the sender is currently transmitting. For example consider there is a network of two users. The number of channels is 10 and are identified by the numbers in the set $\{1, 2, \ldots, 10\}$. The sender at each time slot chooses $c_n = 3$ channels out of the 10 channels. Similarly the receiver selects some $c_m = 4$ channels to listen to. If the sender and the receiver choose a common channel at the same time slot, there will be message exchange. Let, for example at time $t = 1$, the sender chooses the channels 3, 5 and 9 and the receiver chooses the channels 1,4,7,9. So, the sender and receiver will meet on channel 9 and if this channel is not jammed for the time being there could be a message exchange from the sender to the receiver at $t = 1$ provided the sender has some message to transmit. If the sender and the receiver do not have any common channel selected at the same time there would be no communication of message between the two users even if the jammer is not operative. The authors incorporated a generalized the packet verification technique using hash-linking approach of the basic UFH scheme [68]. Upon reception of a new packet m_i, the receiver must identify all packets that link to m_i or to which m_i links. This can be done by traversing all N already received packets once.

2.10.10 Jamming Resistant Key Establishment using UFH

Strasser, Pöpper, Čapkun and Mario Čagalj presented "Jamming-resistant Key Establishment using Uncoordinated Frequency Hopping" in [68] discussed in section 2.10.9. In this work, they address and describe the anti-jamming/key-establishment circular dependency problem. Anti-jamming spread-spectrum communication techniques rely on a

shared (spreading) key and key establishment relies on a jamming-resistant communication. This leads to the following question: in the presence of a communication jammer, how can two devices that do not share any secrets establish a shared secret key over a wireless radio channel? This leads to a circular dependency between the anti-jamming communication and the key establishment problem. The authors proposed Uncoordinated Frequency Hopping scheme that enables two communicating nodes to establish a secret key in presence of the jammer. Once a key is established, it can then be used to support later coordinated frequency hopping communication. This UFH scheme supports the transmission of messages of arbitrary length in a jammed environment without relying on a shared secret key.

2.10.11 Anti-jamming broadcast communicaton scheme by Pöpper et al

Pöpper, Strasser and Čapkun presented "Anti-Jamming Broadcast Communication using Uncoordinated Spread Spectrum Techniques" in [57]. In this paper, they proposed uncoordinated spread spectrum techniques that enable anti-jamming broadcast communication without shared keys. They presented three instances of Uncoordinated Spread Spectrum techniques, Uncoordinated Frequency Hopping (UFH) [68], Uncoordinated DSSS (UDSSS) [56], and hybrid UFH-UDSSS.

In UFH, the communication channels correspond to frequency channels and in UDSSS they correspond to spreading code sequences. The sender chooses the communication channels for the transmission of each message randomly from C and keeps its choice secret. The receivers try to guess the senders selection in order to receive the message. In UDSSS there is a set of spreading codes. The sender chooses one code randomly and spreads the message with that code sequence. The receivers try to guess the sender's selection in order to receive the message. They overcome their lacking knowledge of the sender's spreading operation by accepting a delay in the reception of the message during which they repeatedly try to guess the senders spreading sequence. The authors combined Frequency Hopping and Direct Sequence Spread Spectrum technique into an FH-DSSS system. In such a combined scheme, not only the spreading code but also the carrier frequency is chosen randomly from a predefined set for each message transmission.

2.10.12 Quorum Rendezvous Channel Hopping

Lee, Oh and Gerla proposed Quorum Rendezvous Channel Hopping in [36]. In this scheme they used finite quorum systems to design a channel hopping scheme for a group of users. This is the first scheme that ensured that a pair of nodes rendezvous within

a finite amount of time to exchange pending information. Their scheme evades jammer by allowing the nodes to hop over a set of N available channels and also to meet with another node on some channel within κ^2 time slots, where $\sqrt{N} \leq \kappa \leq N$.

Given a finite universal set $U = Z_N = \{0, 1, \ldots, N-1\}$ of N elements, a subset $D = \{a_1, \ldots, a_\kappa\} \subset Z_N$, $a_i \in \{0, \ldots, N-1\}$ and $\kappa \leq N$, is called a cyclic (N, κ)- *difference set* if for every $d \equiv 0(\mod N)$ there exist at least one pair of elements (a_i, a_j) such that $a_i - a_j \not\equiv d(\mod N)$. Given a (N, κ) difference set $D = \{a_1, \ldots, a_\kappa\} \subset Z_N$, a cyclic *quorum system* constructed by D is $Q = \{G_0, \ldots, G_{N-1}\}$, where $G_i = \{a_1 + i, a_2 + i, \ldots, a_\kappa + i\}(\mod N)$ and $i = 0, \ldots, N-1$. According to this scheme, the nodes randomly choose a quorum from the set Q and using this they compute their sending and receiving sequences.

Scheme	Network Type	Remedy against jamming	Dedicated channel for pair of nodes	TTR	Keyed/Keyless
Classical Frequency Hopping scheme by Navda et al. [53]	Two party Multi-party	Communication channel Communication channel	Yes depends	Bounded Unbounded	Keyed
UFH [67]	Two party	Communication channel	Yes	Unbounded	Keyless
QRCH [36]	Multi-party	Communication channel	No	Bounded	Keyless
UFH Broadcast [56]	Broadcast	Communication channel	No	Unbounded	keyless
Tague et al. [70]	Multi-party	Control channel	Yes but not for all pair of nodes	Bounded	keyed
Lazos et al. [35]	Multi-party	Control channel	No	Bounded	Keyless
Xu et al. [78]	Two party/multiparty	Communication channel	No	Unbounded	Keyless
Our scheme in chapter 6	Two party	Communication channel	Yes	Yes	Keyless
Our scheme-I in section 7.3 of Chapter 7	Multi-party	Communication channel	Yes	Bounded	Temporary Key
Our scheme-II in section 7.4 of chapter 7	Multicast	Communication channel	No	Bounded	Keyed

TABLE 2.2: Table of comparison of frequency hopping schemes. The second column shows whether the anti-jamming frequency hopping communication scheme deals with two party or multi-party communication. The third column shows whether the scheme discusses communication channel jamming or control channel jamming. The fourth column shows whether any pair of nodes can meet on an exclusive channel or not. The fifth column corresponds to the time to rendezvous(TTR) between the nodes. The last column shows if the technique used is keyed or keyless.

Chapter 3

Key Predistribution in Wireless Sensor Networks Using Finite Affine Plane

This chapter is based on the research work discussed in [6]. In this chapter, we discuss a new combinatorial scheme for key distribution that makes use of finite plane in $\mathbb{Z}_q$, where q is a prime number. We have defined finite affine plane in section 2.1.1. We implement this key predistribution scheme using an unbalanced combinatorial design developed from finite affine plane. This gives rise to a new type of combinatorial design.

Then we study and compare the connectivity and security of our scheme with respect to existing schemes. We study the security of such a network in terms of two parameters. One of the parameters consider the proportion of nodes disconnected when a certain number of nodes are compromised. The other parameter considers the proportion of links broken under node compromise. We show that our design results in much better connectivity and resiliency compared to [14, 16, 38, 60] and show that our connectivity and security results outperform all these schemes.

3.1 A key predistribution scheme using affine planes

We use a new design for key predistribution. Our design is unbalanced, which means, that our design does not satisfy the condition 4 of definition 2.2. In our design, blocks contain different number of varieties.

We now discuss our design in details. We present the construction of the design and show how our design can be applied to key predistribution in sensor networks. Let q be

a prime number. Recall the description of finite affine plane in section 2.1.1. An affine plane $AG(2,q)$ is constructed as given in Section 2.1. There are q^2 points, q^2+q lines, each line containing $q+1$ points and each point belonging to q lines. Let P_i denote the i^{th} point. We divide the set of points into $\lfloor \frac{q^2}{4} \rfloor$ disjoint groups of 4 points. Consider g_i which consists of one point from each of these groups. $g_i = \{P_i, P_{\lfloor \frac{q^2}{4} \rfloor + i}, P_{2\lfloor \frac{q^2}{4} \rfloor + i}, P_{3\lfloor \frac{q^2}{4} \rfloor + i}\}$. Let g_i be the set of elements, where $i = 0, 1, \ldots, \lfloor \frac{q^2}{4} \rfloor - 1$. A subset A_i of lines consist of all lines which pass through the points in g_i. We will calculate the value of $|A_i|$ in Theorem 3.3.

Algorithm 1 provides the construction of the unbalanced design. The output of Algorithm 1 is a combinatorial design B. B contains $\lfloor \frac{q^2}{4} \rfloor$ blocks $B_0, B_1, \ldots, B_{\lfloor \frac{q^2}{4} \rfloor - 1}$, each block contains lines of $AG(2,q)$ as varieties.

Algorithm 1 Algorithm to generate the blocks of the unbalanced design

Input: The finite affine plane $AG(2,q)$ where q is a prime number.
Output: The unbalanced design B.
Let $P_i = (\lfloor \frac{i}{q} \rfloor, i - q\lfloor \frac{i}{q} \rfloor), 0 \leq i \leq q^2 - 1$
L_i be the set of lines passing through point P_i. L_i can be computed using Algorithm 6 for each i.
for $k = 0 \rightarrow \lfloor \frac{q^2}{4} \rfloor - 1$ **do**
$B_k = \{L_k \cup L_{\lfloor \frac{q^2}{4} \rfloor + k} \cup L_{2\lfloor \frac{q^2}{4} \rfloor + k} \cup L_{3\lfloor \frac{q^2}{4} \rfloor + k}\}$
end for

We now map this design to a sensor network. The lines are mapped to keys. The size of the key pool is q^2+q. So the sensor network consists of $N = \lfloor \frac{q^2}{4} \rfloor$ sensors, a sensor n_i consists of k_i keys (k_i is given by Theorem 3.3). Two nodes n_i and n_j contain μ_{ij} keys in common. We will later show that $1 \leq \mu_{ij} \leq 16$. In our design $|X| = v = q^2+q$, $b = \lfloor \frac{q^2}{4} \rfloor$, $4q-7 \leq k \leq 4q-2$. So, we see that if N be the size of the network, then the amount of key storage is $O(\sqrt{(}N)$.

Choice of parameter

Let N be the total number of nodes that the network can support. We choose the smallest prime q, such that $N < \frac{q^2}{4}$. We construct an affine plane $AG(2,q)$ with q^2 points and q^2+q lines. We consider the set of N points $\{P_0, P_1, P_2, \ldots, P_{4N-1}\}$ and for each sensor assign keys to sensor n_i corresponding to the lines passing through the points $\{P_i$,P_{N+i}, P_{2N+i} and $P_{3N+i}\}$.

Example

Let the total number of nodes supported is $N = 6$. So we choose $q = 5$. The set of points are
(0,0), (0,1), (0,2), (0,3), (0,4), (1,0), (1,1), (1,2), (1,3), (1,4), (2,0), (2,1), (2,2), (2,3), (2,4), (3,0), (3,1), (3,2), (3,3), (3,4), (4,0), (4,1), (4,2), (4,3), (4,4).

Node no.	Points in the node	Key identifiers
1	(0,0) (1,1) (2,2) (3,3)	(1,0,0) (1,1,0) (1,2,0) (1,3,0) (1,4,0) (0,1,0) (0,1,1) (1,0,1) (2,4,1) (3,3,1) (4,2,1) (1,4,0) (0,3,1) (1,2,1) (2,1,1) (3,0,1) (4,4,1) (1,4,0) (0,2,1) (1,1,1) (2,0,1) (3,4,1) (4,3,1) (1,4,0)
2	(0,1) (1,2) (2,3) (3,4)	(0,1,1) (1,1,1) (2,1,1) (3,1,1) (4,1,1) (1,0,0) (0,3,1) (1,0,1) (2,2,1) (3,4,1) (4,1,1) (1,2,0) (0,2,1) (1,3,1) (2,4,1) (3,0,1) (4,1,1) (1,1,0) (0,4,1) (1,2,1) (2,0,1) (3,3,1) (4,1,1) (1,3,0)
3	(0,2) (1,3) (2,4) (4,0)	(0,3,1) (1,3,1) (2,3,1) (3,3,1) (4,3,1) (1,0,0) (0,2,1) (1,0,1) (2,3,1) (3,1,1) (4,4,1) (1,3,0) (0,4,1) (1,1,1) (2,3,1) (3,0,1) (4,2,1) (1,2,0) (4,0,1) (4,1,1) (4,2,1) (4,3,1) (4,4,1) (0,1,0)
4	(0,3) (1,4) (3,0) (4,1)	(0,2,1) (1,2,1) (2,2,1) (3,2,1) (4,2,1) (1,0,0) (0,4,1) (1,0,1) (2,1,1) (3,2,1) (4,3,1) (1,1,0) (2,0,1) (2,1,1) (2,2,1) (2,3,1) (2,4,1) (0,1,0) (0,1,1) (1,2,1) (2,3,1) (3,4,1) (4,0,1) (1,1,0)
5	(0,4) (2,0) (3,1) (4,2)	(0,4,1) (1,4,1) (2,4,1) (3,4,1) (4,4,1) (1,0,0) (3,0,1) (3,1,1) (3,2,1) (3,3,1) (3,4,1) (0,1,0) (0,1,1) (1,3,1) (2,0,1) (3,2,1) (4,4,1) (1,2,0) (0,3,1) (1,1,1) (2,4,1) (3,2,1) (4,0,1) (1,3,0)
6	(1,0) (2,1) (3,2) (4,3)	(1,0,1) (1,1,1) (1,2,1) (1,3,1) (1,4,1) (0,1,0) (0,1,1) (1,4,1) (2,2,1) (3,0,1) (4,3,1) (1,3,0) (0,3,1) (1,4,1) (2,0,1) (3,1,1) (4,2,1) (1,1,0) (0,2,1) (1,4,1) (2,1,1) (3,3,1) (4,0,1) (1,2,0)

TABLE 3.1: Table showing an example of key predistribution in 6 nodes

Note that the total number of points here is $q^2 = 25$

The key identifiers are
(0,1,1), (0,2,1), (0,3,1), (0,4,1), (1,0,1), (1,1,1), (1,2,1), (1,3,1), (1,4,1), (2,0,1), (2,1,1), (2,2,1), (2,3,1), (2,4,1), (3,0,1), (3,1,1), (3,2,1), (3,3,1), (3,4,1), (4,0,1), (4,1,1), (4,2,1), (4,3,1), (4,4,1), (1,0,0), (1,1,0), (1,2,0), (1,3,0), (1,4,0), (0,1,0).

Here, the total number of lines are $q^2 + q = 30$.

The individual nodes and the keys contained in them are given in the table 3.1.

Now from this table we can observe that the common lines(i.e. keys) between node 5 and node 6 are
(1,4,1),(3,0,1),(3,1,1),(3,3,1),(0,1,0),(0,1,1),(1,3,1),(2,0,1),(0,3,1), (1,1,1),(4,0,1),(1,3,0).

3.2 Shared Key Discovery

If two nodes want to communicate securely then they need to find a common secret key. In our scheme this common key can be found like this:
Let the two nodes be n_i and n_j $(i, j \in \{0, 1, \ldots, \lfloor N-1\}, i \neq j)$. The points they contain are

$$n_i = \{P_i, P_{N+i}, P_{2N+i}, P_{3N+i}\}$$

$$n_j = \{P_j, P_{N+j}, P_{2N+j}, P_{3N+j}\}$$

Let, $l_{ik} = P_{(k-1)N+i}$ and $l_{jk} = P_{(k-1)N+j}$

for $k \in \{1, 2, 3, 4\}$ Now the shared key discovery algorithm is given in Algorithm 2:

Algorithm 2 Algorithm to find shared key between two nodes n_i, n_j such that $i < j$.

Input: The node identifiers i and j
Output: Either a key if one exists or "No key found" if none exists.

```
for r = 1 : 4 do
  for s = 1 : 4 do
    Let αx + βy = γ be the line between The points l_ir and l_js
    if γ = 0 then
      if α = 0 then
        if (0, 1, 0) is not a compromised key then
          output key (0, 1, 0) and exit.
        end if
      else
        if (1, α^{-1}β, 0) is not a compromised key then
          output (1, α^{-1}β, 0) and exit.
        end if
      end if
    else
      if (γ^{-1}α, γ^{-1}β, 1) is not compromised key then
        output (γ^{-1}α, γ^{-1}β, 1) and exit.
      end if
    end if
  end for
end for
Signal "No key found".
```

This algorithm makes use of a function that given two distinct points finds the line in $AG(2, q)$ passing through both of them. From Section 2.1 the equation of a line through two distinct points (x_1, y_1) and (x_2, y_2) can be determined to be

$$(y_2 - y_1)x + (x_1 - x_2)y = y_2x_1 - y_1x_2$$

The compromised keys belonging to any compromised node can be found using the algorithm in Section 3.5.

This shared key discovery algorithm can be run in constant time since the total number of iteration is upper bounded by $4 \times 4 = 16$. Since the total number of nodes $N < \lfloor \frac{q^2}{4} \rfloor$, hence the number of bits required to represent the node identifiers is $O(\log N) = O(\log q)$.

Further, if two nodes i and j want to compute a shared key(if one exists) they will be running the algorithm identically as they check whether $i < j$ or $i > j$ eliminating the possibility of conflict between i and j. So eventually they will end up finding the same key.

3.3 Analysis of our schemes

In this section we analyze our scheme, calculating the number of keys required for each node, the connectivity and security of the network.

3.3.1 Memory requirement

The following Lemmas 3.1 and 3.2 helps us to calculate the number of keys present in each node.

Lemma 3.1. *Let n_i contain the points $\{P_i, P_{N+i}, P_{2N+i}$ and $P_{3N+i}\}$. If no three of these four points are collinear, then the number of keys in n_i is $4q-2$.*

Proof. We have mentioned in Section 2.1 that in $AG(2,q)$ there are $q+1$ lines that pass through a point. Since there are four points in each node n_i, which are $\{P_i, P_{N+i}, P_{2N+i}$ and $P_{3N+i}\}$. So, there are $4(q+1)$ lines in total. Now if all the four points in that node are such that no three of them are collinear then there are $\binom{4}{2}$ lines that pass through each pair of points in n_i and are counted twice in the previous $4(q+1)$ lines. Hence the actual number of distinct lines is $4(q+1) - \binom{4}{2}$ Hence the maximum number of distinct keys in each node is $4q-2$. □

Lemma 3.2. *Let n_i contain the points $\{P_i, P_{N+i}, P_{2N+i}$ and $P_{3N+i}\}$. If all four points are collinear, then the number of keys in n_i is $4q+1$.*

Proof. From Section 2.1 we see that in $AG(2,q)$ there are $q+1$ lines that pass through a point. If all the four points in n_i are collinear then there exactly one line passing through each of the 4 points $\{P_i, P_{N+i}, P_{2N+i}$ and $P_{3N+i}\}$. Now we know that $q+1$ distinct lines pass through each point of them. Among these $q+1$ lines passing through the 4 points one line is common and other q in all the 4 points are distinct. So. the total number of distinct lines passing through $\{P_i, P_{N+i}, P_{2N+i}$ and $P_{3N+i}\}$ is $4*q+1 = 4q+1$. So, the number of distinct keys will be $4q+1$. □

From Lemma 3.1 and 3.2 it follows that:

Theorem 3.3. *If the number of keys in a node n_i is k_i then $4q - 2 \leq k_i \leq 4q + 1$.*

Theorem 3.4. *If μ_{ij} be the number of keys in common between any two nodes n_i and n_j, then $1 \leq \mu_{ij} \leq 16$.*

Proof. Let μ_{ij} be the number of keys in common between two nodes n_i and n_j. Since any two nodes contain distinct points, there is at least one line through two distinct points in n_i and n_j. This is because by definition of affine plane, there is a line through any two distinct points. The key corresponding to this line is the common key between the nodes n_i and n_j. This situation arises, when the eight points belonging to n_i and n_j are all collinear.

When no 3 of the points in n_i and n_j are collinear then the value of for each pair of points P_i' and P_j' belonging to node n_i and n_j, there is a line. So the maximum number of lines is $4 \times 4 = 16$. The maximum number of common keys is hence 16.

It follows that $1 \leq \mu_{ij} \leq 16$. □

Theorem 3.5. *If s nodes are compromised, then no node will be disconnected if $s < \frac{q+1}{4}$.*

Proof. Let $n_1, n_2, \ldots, n_s$ are the s nodes. Let n_t $(t \neq n)$ be an uncompromised node. There are $q + 1$ number of lines through a point $x \in n_t$.

There are at the most $4s$ distinct points in $n_1, n_2, \ldots, n_s$. Two points cannot be present together in more than one distinct line. So, there can be at most $4s$ distinct lines that pass through x as well as each of the points in $n_1, n_2, \ldots, n_s$. If $4s < q + 1$, then x will contain at least one line that does not pass through the compromised nodes. So n_t will not be disconnected, if $s < (q + 1)/4$.

□

3.3.2 Connectivity

Two nodes within communication range can exchange information securely, provided they have a common key. In most probabilistic schemes, this is not possible, since key chains are chosen randomly. We see that in our scheme, any two nodes share at least one key. So, any two nodes within communication range can carry on secure message exchange. We will see later that many of the deterministic schemes like [16, 38] do not guarantee that nodes within communication will share a common key. So, our scheme has full connectivity, which reduces delays occurring in multihop communications.

N	q	s	$V(s)$ (experimental)
870	59	5	0.0057
870	59	10	0.011494
1980	89	11	0.0055
1980	89	15	0.00757
1980	89	20	0.0101

TABLE 3.2: Experimental value of $V(s)$, when N is the total number of nodes, s is the number of nodes compromised.

3.3.3 Study of security

Sensor nodes are deployed in regions where they cannot be attended to and powered by battery. The networks are self organizing. In areas of strategic importance like military areas, nodes can be compromised by adversaries. The adversaries might be passive, just overhearing the conversation between two sensors or might be active, in which it might compromise the nodes learning the keys present in the nodes and not only decrypting information between sensors, but injecting false messages. It is very important to study how a key predistribution scheme affects the network in case of node compromise and compare with existing schemes.

We study the security of the network by measuring two parameters. We have defined them in section 1.3.1. The first one $V(s)$, which measures the fraction of nodes which are completely disconnected when a given number (s) of nodes are compromised. A disconnected node cannot carry on any more communications and becomes fully ineffective. The other measure is $E(s)$ which finds out the fraction of links disconnected when s nodes are compromised.

We analyze the resilience of our scheme experimentally.

Analysis of $V(s)$ $V(s)$ is defned in section 1.3.1. We note from Theorem 3.5 that no nodes are disconnected if the number of compromised nodes s is such that $s \leq (q+1)/4$. Experimental results of the values of $V(s)$ is given in Table 3.2.

Analysis of $E(s)$

$E(s)$ is defned in section 1.3.1. Experimental results of the values of $E(s)$ is given in Table 3.3.

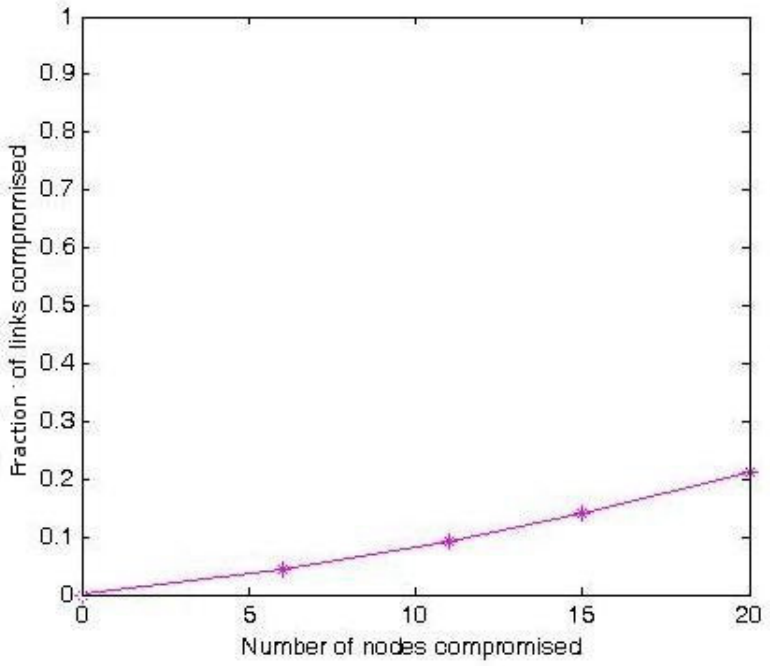

FIGURE 3.1: Experimental results for $E(s)$

N	q	s	$E(s)$ (experimental)
870	59	5	0.068958
870	59	10	0.157406
1980	89	11	0.090639
1980	89	15	0.139159
1980	89	20	0.212303

TABLE 3.3: Experimental value of $E(s)$, when N is the total number of nodes, s is the number of nodes compromised.

3.4 Comparison of our design with existing schemes

When we compare our scheme with non-deterministic schemes we see that our scheme requires computation of $O(1)$, to calculate the shared keys, whereas non-deterministic schemes require $O(k \log k)$ (k is the number of keys per node). The communication expense is $O(\log N)$ in our scheme, whereas it is $O(k \log v)$ in case of non-deterministic schemes. This is because our scheme broadcasts only the node identifier, whereas the probabilistic schemes have to share key identifiers.

We compare the security of our scheme with the basic scheme of Eschenaur and Gligor [27], Lee and Stinson's linear and quadratic schemes [40], Chakraborty et al scheme [16] and Ruj and Roy scheme [60]. We see from Table 3.4 the connectivity of the networks under different schemes. N represents the size of the network and k represents the

Scheme	N	k	Full Connectivity
Basic [27]	2415	136	No
Camtepe-Yener [14]	2257	48	Yes
Linear [40]	2209	30	No
Quadratic [40]	2197	12	No
CMR [16]	2550	28	No
PBIBD I[60]	2415	136	Yes
PBIBD II[60]	2450	96	Yes
Our	1980	350	Yes

TABLE 3.4: Schemes with parameters that we choose for our comparisons and connectivity

number of keys. We see from the Figure 3.2, that our scheme is highly secure compared to all of these schemes except [60] first scheme. Our scheme requires more memory in this example. However for networks which deploy huge number of sensors our scheme has $O(\sqrt{N})$ keys, which is the same as all the other schemes. So our scheme works best where there is large network with thousands of nodes.

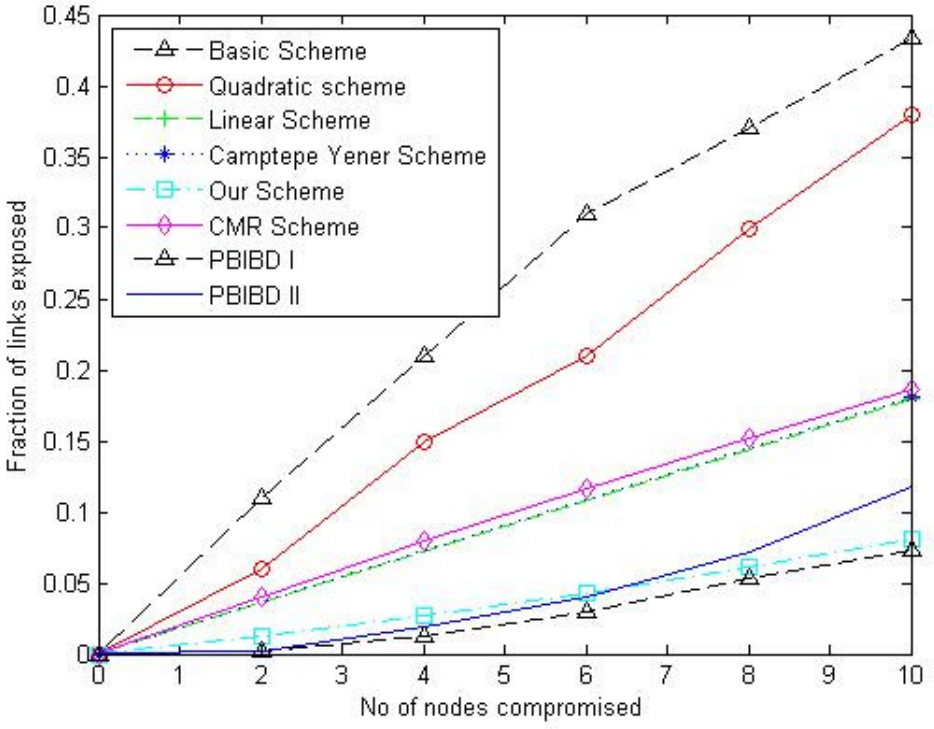

FIGURE 3.2: Comparison of the fraction of links broken for different schemes

3.5 Key Revocation

Key management has two important aspects: key distribution, which describes how to disseminate secret information to the nodes so that they can communicate securely, and key revocation, which describes how to remove secret keys that may have been compromised. In other words key revocation deals with removing the keys (by deactivating them) of the network that are shared by compromised nodes. In combinatorial strategies of key distribution one key is shared by more than one node. So, the compromised keys, to compromised nodes must be replaced with new ones at regular interval to ensure that the connectivity of the nodes remains intact. This is called key refreshing.

Revocation has two aspects *viz.* node revocation, and key revocation. In node revocation strategy the node(s) which is/are compromised is/are identified. There are some node revocation techniques that serve this purpose. Once the compromised node is identified, we need to identify the keys that this particular compromised node was containing. Once those keys are identified the network administrator can actually inform other nodes that those keys are exposed or alternately she can replace those keys with new one. The detection of compromised nodes is an whole research area of intrusion detection, which we do not discuss here.

Now, the question is how to find the keys belonging to a particular node n_i. Let v_i be the set of points representing node n_i. Let $v_{i1}, v_{i2}, v_{i3}, v_{i4}$ be the 4 points in v_i. The method of finding all keys belonging to n_i is to find all the straight lines belonging to the 4 points $v_{i1}, v_{i2}, v_{i3}, v_{i4}$. .

For a detailed explanation of how the method computes the lines one may refer to [66].
Complexity of the Algorithm All steps inside the largest loop occur four times. Since all the for loops inside the algorithm run between 0 to $q-1$, so the complexity of the algorithm is $O(4q) = O(q)$.

Once the compromised keys are known, the node n_i can tag them as exposed and it will not use them in future. This information can be used in shared key discovery as discussed in the shared key discovery algorithm in section 3.2.

3.6 Concluding Remarks

In this chapter we have discussed a new key predistribution technique using combinatorial strategy. We have discussed some properties of the design, have proven a lower as well as an upper bound for the number of distinct keys in a particular node. Again we have shown a bound for the number of distinct keys between two separate nodes. Then

we studied the resiliency of our design with respect to the model proposed by Ruj, Roy [60]. Then we have compared the experimental values of resiliency of our scheme with several deterministic schemes and shown that our scheme outperforms them. Then we have provided an algorithm for shared key discovery that works in time $O(1)$. Thereafter we have vividly discussed key revocation technique for our scheme.

Our research can be expanded in several directions. We have used a simple method for clubbing 4 points for any node n_i. In future we can generalize this to clubbing arbitrary number of points. If we club more points, then the size of the key ring will increase, through the resilience to compromise will be stronger. It is important to find the optimal number of points being clubbed. Further research can be done to invent efficient clubbing technique. This scheme can be extended from $\mathbb{Z}_q$ to $GF(q^m)$ where m is a positive integer.

Chapter 4

A New Key Predistribution Scheme for General and Grid-group Deployment of Wireless Sensor Networks

This chapter is based on the research work discussed in [4]. In this chapter, we discuss a key predistribution scheme for homogeneous wireless sensor networks using the scheme of Blom [10] as well as symmetric balanced incomplete block design (SBIBD)(def 2.3). The main advantage of using this scheme for key predistribution is that for this scheme, the adversary needs to capture large number of nodes in order to compromise all the keys in an uncompromised node. In other words, in order to disconnect an uncaptured node from all other nodes, the adversary needs to capture many more nodes than the other standard schemes.

Then, we use this new key predistribution scheme in a grid-group deployment of sensor nodes. A grid-group deployment refers to such a deployment where the entire deployment zone is broken into smaller two-dimensional square regions giving rise to an $n \times n$ grid-group structure. Equal number of sensor nodes are deployed in each of the smaller square regions of the deployment zone. Sensor nodes deployed inside one smaller square region forms a group. Sensor nodes within the same group communicate more frequently than a pair of nodes falling in two different groups. This is driven by the fact that sensor nodes in proximity to each other communicate more frequently than distant nodes. Sensor nodes deployed in this fashion grid form a heterogeneous network. This type of deployment scheme is applied in battlefields where sensors belonging to a compromised zone need to be completely disconnected from the rest of the network. Because if an

adversary compromises an area, all the sensor nodes deployed in that area are considered to be captured.

This type of deployment is proposed by Liu and Ning [42, 44]. There are two types of sensor nodes in this heterogeneous network. They mainly differ in resource. One type of nodes have a low amount of storage capacity, power, and computational power, and the other type of nodes are richer in the amount of computational resources that they posses. We shall use the name 'supernode' for the nodes which are more powerful than common nodes. Common sensors belonging to one region contain a set of keys that are completely disjoint from the sensors in some other region. This ensures that even if one region is totally disconnected, the other regions are not affected. For each sensor node, the keys are preloaded in such a way that all the nodes belonging to a particular square region (group) can communicate with each other directly. Sensor nodes belonging to different square regions (group) communicate through two or more supernodes.

Our general key predistribution scheme offers better resiliency than the schemes in [10, 14, 15]. For example, in key predistribution scheme by Blom [10], the adversary can compromise all the keys of the entire WSN merely by capturing c nodes, where c is the security parameter of the design. However, in our scheme, the adversary can only compromise few links by capturing c nodes. Our scheme also offers better resiliency than [14, 15] in terms of the number of links that get exposed when some nodes are compromised. In both key predistribution schemes based on symmetric BIBD and generalized quadrangles in [14, 15], the attacker can compromise many key links between pairs of uncaptured nodes by capturing a single node. However, in our scheme, the attacker needs to capture multiple nodes for compromising the key links between some pairs of nodes. We have compared our scheme with [62] and other similar schemes on the basis of fraction of links that gets exposed when some nodes get captured by the adversary. This is a well-known measure of the resiliency of a key predistribution scheme. Our scheme is shown to exhibit the best performance as far as the resiliency is concerned. The scheme of Ruj and Roy in [62] uses three times the number of supernodes we use in our scheme for full connectivity. Our scheme offers better resiliency using less number of supernodes.

4.1 Construction of SBIBD

We recollect the definition of SBIBD from section 2.1.1. An SBIBD is a design where the number of blocks equals the number of elements. Here we discuss construction of SBIBD. Çamptepe and Yener used mutually orthogonal Latin squares in constructing the key predistribution scheme of [14]. Another construction of the same scheme can be

found in [69]. Let $V_3(q)$ be the set of a three-dimensional vector space over a finite field F_q of q elements. A projective geometry $PG(2,q)$ over a finite field F_q is defined like the following:

1. The points are given by the one-dimensional subspaces of $V_3(q)$.
2. The lines are given by the two-dimensional subspaces of $V_3(q)$.
3. A point belongs to a line if the corresponding one-dimensional subspace of the point is contained in the two-dimensional subspace corresponding to the line.
4. Two lines are incident to each other iff the intersection of the corresponding two-dimensional subspaces of them is a nonempty one-dimensional subspace.

It can be shown that there are $(q^3-1)/(q-1)$ or q^2+q+1 number of distinct subspaces of dimension one of $V_3(q)$ [69]. Similarly, the number of distinct subspaces of dimension two of $V_3(q)$ is also q^2+q+1. Each two-dimensional subspace contains $q+1$ distinct one-dimensional subspaces. The intersection of two-dimensional subspaces is a one-dimensional subspace of $V_3(q)$. So, the number of points and lines in $PG(2,q)$ is q^2+q+1. Every line contains $q+1$ number of points. So, taking points as varieties and lines as block $PG(2,q)$ is a symmetric $(q^2+q+1, q+1, 1)$ BIBD.

Since the lines of $PG(2,q)$ are two-dimensional subspaces of $V_3(q)$, we can represent each block by the basis of the subspaces they correspond to. The basis of a two-dimensional subspace of $V_3(q)$ contains exactly two elements. So, each block in $PG(2,q)$ will be identified by two elements of $V_3(q)$.

Similarly, the points of $PG(2,q)$ are one-dimensional subspaces of $V_3(q)$. So, every variety of $(q^2+q+1, q+1, 1)$ SBIBD can be represented by the basis of the one-dimensional subspace it belongs to.

Let $L_1 = \{(1,s,t) : s,t \in GF(q)\}$

$$L_2 = \{(0,1,s) : s \in GF(q)\}$$

$$L_3 = \{(0,0,1)\}$$

Let, $\mathcal{S} = L_1 \cup L_2 \cup L_3$.

$$|\mathcal{S}| = q^2+q+1.$$

It can be shown that each element of $\mathcal{S}$ is a basis of a distinct one-dimensional subspace of $V_3(q)$. Throughout this article, we shall represent the q^2+q+1 number of varieties of the $(q^2+q+1, q+1, 1)$ SBIBD by the elements of $\mathcal{S}$.

4.1.1 Shared variety discovery of $(q^2+q+1, q+1, 1)$ SBIBD

Any two blocks of a symmetric $(q^2+q+1, q+1, 1)$ BIBD do share one and unique variety. Given a $(q^2+q+1, q+1, 1)$ SBIBD, Algorithm 3 finds the common variety of two blocks of the design. This algorithm uses the basis of the nullspace of $A.x = 0$. This basis can be computed using Gauss-Jordan elimination method [34, 50] in a constant time. Therefore, the runtime of Algorithm 3 is $O(1)$.

Algorithm 3 Computing the shared variety between two blocks of $(q^2+q+1, q+1, 1)$ SBIBD.

Input: Basis of block 1 $\{(a_1, b_1, c_1), (a_2, b_2, c_2)\}$.
Basis of block 2 $\{(a'_1, b'_1, c'_1), (a'_2, b'_2, c'_2)\}$.
Output: Find the identifier of the shared variety of the two blocks.

$$A = \begin{bmatrix} a_1 & a_2 & -a'_1 & -a'_2 \\ b_1 & b_2 & -b'_1 & -b'_2 \\ c_1 & c_2 & -c'_1 & -c'_2 \end{bmatrix}, \; x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Find the basis of the nullspace $A.x = 0$.
Let this basis be given by $(\beta_1, \beta_2, \beta_3, \beta_4)$.
$a = a_1\beta_1 + a_2\beta_2$
$b = b_1\beta_1 + b_2\beta_2$
$c = c_1\beta_1 + c_2\beta_2$
if $a \neq 0$ **then**
  The identifier of the common variety is $(1, a^{-1}b, a^{-1}c)$.
else
  if $b \neq 0$ **then**
    The identifier of the common variety is $(0, 1, b^{-1}c)$.
  else
    The identifier of the common variety is $(0, 0, 1)$.
  end if
end if

4.2 Blom's scheme

Blom [10] proposed a scheme for key predistribution where the members of a group can establish pairwise keys. Let N be the size of the network. The distribution server first chooses a $c \times N$ matrix G over a finite field $GF(q)$. The matrix G is considered to be a public information. Now, the distribution server constructs a $c \times c$ symmetric matrix D over $GF(q)$. This matrix is a private information of the system. Now, the server computes the $c \times N$ matrix A, where $A = (DG)^T$, T being the transposition operator. Now, $AG = (DG)^T G = G^T D^T G = G^T DG = G^T A^T = (AG)^T$.

Thus, AG is a symmetric matrix. Let $K = AG$, we know that $K_{ij} = K_{ji}$, where K_{ij} is the element in K located in the ith row and jth column. K_{ij} (or K_{ji}) is the pairwise

key between node U_i and node U_j. To carry out the above computation, nodes U_i and U_j should be able to compute K_{ij} and K_{ji}, respectively. This can be easily achieved using the following key predistribution scheme, for $w = 1, 2, \ldots, N$,

1. Store the wth row of matrix A in node U_w.
2. Store the wth column of matrix G in node U_w.

Now, if two nodes (say U_x and U_y) want to communicate, they need to establish a common key. Node U_x has row x of A and column x of G. Node U_y has row y of A and column y of G. Now , they can establish a pairwise key this way:

1. Node U_x and U_y exchange column x and column y of matrix G, respectively.
2. Node U_x calculates K_{xy} =(row x of A).(column y of G).
3. Node U_y calculates K_{yx} =(row y of A).(column x of G).

The matrix G is a public information. Therefore, the rows of G could be sent without encryption. Since K is a symmetric matrix, $K_{xy} = K_{yx}$. Hence, K_{xy} can be used as the common key between the two nodes.

4.2.1 c-secure property

It has been proved that the above scheme is c-secure [10], i.e., if any $c+1$ columns of G are linearly independent; then, no member other than U_x and U_y can compute K_{xy} or K_{yx} if no more than c members are compromised.

4.2.2 A construction for matrix G

We note that any $c + 1$ columns of G [25] must be linearly independent in order to achieve the c-secure property. Let α be a primitive element of a finite field $GF(q)$ where q is a prime power.

A feasible G can be designed as follows [48]:

$$G = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ \alpha & \alpha^2 & \alpha^3 & \cdots & \alpha^N \\ \alpha^2 & (\alpha^2)^2 & (\alpha^3)^2 & \cdots & (\alpha^N)^2 \\ \alpha^3 & (\alpha^2)^3 & (\alpha^3)^3 & \cdots & (\alpha^N)^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha^{c-1} & (\alpha^2)^{c-1} & (\alpha^3)^{c-1} & \cdots & (\alpha^N)^{c-1} \end{bmatrix}$$

It is well known that $\alpha^i \neq \alpha^j$ if $i \neq j$ (this is a property of primitive elements). Since G is a Vandermonde matrix, it can be shown that any $c+1$ columns of G are linearly independent when $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^N$ are all distinct. In practice, G can be generated by the primitive element α of $GF(q)$. Therefore, the wth column of G is stored at node U_w; it is only required to store the seed α^w, and any node can regenerate the column given the seed.

4.3 Proposed scheme

4.3.1 Key predistribution in the network

Here, our aim is to design a key predistribution scheme for a sensor network consisting $\mathcal{N}$ nodes where $\mathcal{N} \leq p^2+p+1$ where p is a prime number. We use the scheme in [14] and [15] by Çamtepe and Yener and [10] by Rolf Blom.

We shall be using a $(p^2+p+1, p+1, 1)$ -symmetric balanced incomplete block design $(X, \mathcal{A})$. Here, $X = \{x_1, x_2, \ldots, x_v\}, v = p^2+p+1$. $\mathcal{A} = \{B : B = \{x_{j_1}, x_{j_2}, \ldots, x_{j_{p+1}}\}, j_1, j_2, \ldots, j_{p+1} \in \{1, 2, \ldots, v\}, j_m \neq j_n, 1 \leq m, n \leq p+1\}$. $|\mathcal{A}| = p^2+p+1$. Here, B_is are the individual blocks for all $i \in \{1, 2, \ldots, p^2+p+1\}$. $|B_i| = p+1, \forall i \in \{1, 2, \ldots, p^2+p+1\}$.

4.3.1.1 The scheme

Definition 4.1. For any node $n_i \in \mathcal{N}$, and a variety $x_l \in X$ and a block $B_d \in \mathcal{A}$, $POS(B_d, x_l)$ is an integer taking values from the set $\{1, 2, \ldots, k\}$, where $f(n_i) = B_d$ and $x_l \in B_d$. The node n_i stores the values of $POS(B_d, x_l), \forall x_l \in B_d$.

Since, $|B_d| = k, \forall B_d \in \mathcal{A}$, so each node stores k number of $POS(*, *)$ values.

Definition 4.2. f is a one-to-one map from the set of nodes of the sensor network to the blocks of the symmetric $(p^2+p+1, p+1, 1)$ design. In addition to that, we assume that f^{-1} can be computed in constant time.

It can be noted that the nodes can be identified by the identifier of the blocks they correspond to. Therefore, one example of the function f is the identity mapping if $\mathcal{N} \subseteq \mathcal{A}$.

The total number of nodes in deployment be $t = |\mathcal{N}|$. Choose a prime power p such that $t \leq p^2+p+1$. Now, design a symmetric $(p^2+p+1, p+1, 1)$ BIBD using Algorithm 1 of [15]. Comparing a (v, b, r, k, λ)-design to this symmetric $(p^2+p+1, p+1, 1)$-design, we get $v = b = p^2+p+1, k = r = p+1$ and $\lambda = 1$. The varieties of the design are denoted

by $x_1, x_2, \ldots, x_{p^2+p+1}$ and the blocks as $B_1, B_2, \ldots, B_{p^2+p+1}$. We shall design our key predistribution scheme in nodes using this symmetric $(p^2+p+1, p+1, 1)$-design. Let the security parameter be c as in Section 4.2. We shall later discuss on a feasible value the integer c. Now, compute p^2+p+1 symmetric $c \times c$ matrices $D_1, D_2, \ldots, D_{p^2+p+1}$ over a finite field $GF(q)$. Now, construct a $c \times r$ matrix G using the method described in 4.2.2 i.e. if α is a primitive element of $GF(q)$, compute:

$$G = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ \alpha & \alpha^2 & \alpha^3 & \cdots & \alpha^r \\ \alpha^2 & (\alpha^2)^2 & (\alpha^3)^2 & \cdots & (\alpha^r)^2 \\ \alpha^3 & (\alpha^2)^3 & (\alpha^3)^3 & \cdots & (\alpha^r)^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha^{c-1} & (\alpha^2)^{c-1} & (\alpha^3)^{c-1} & \cdots & (\alpha^r)^{c-1} \end{bmatrix}$$

Algorithm 4 maps a (v, b, r, k, λ) design $(X, \mathcal{A})$ into a key predistribution scheme. Let $\mathcal{N} = \{n_1, n_2, \ldots, n_t\}$ be the set of nodes in the WSN. We can design a key predistribution in these nodes using Algorithm 4 and taking $v = b = p^2+p+1, r = p+1$. In Algorithm 4, we take $v = p^2+p+1$ many different key spaces of the Blom scheme [10]. We compute one $c \times r$ public matrix G and a set of v many $c \times c$ secret symmetric matrix $D_i, i \in \{1, 2, \ldots, v\}$. Thus, we can compute v many A matrices like this : $A_i = (D_i \dot{G})^T$. Hence, there are v many distinct key spaces of Blom scheme. Now, we can have a key distribution scheme by considering each of the v key space as a variety of the $(p^2+p+1, p+1, 1)$- SBIBD, where each block of the SBIBD corresponds to a node of the WSN. Since a block of a $(p^2+p+1, p+1, 1)$- SBIBD contains $p+1$ many varieties, every node will have its key share from exactly $p+1$ many key spaces.

Algorithm 4 Algorithm for key predistribution in nodes.

Input: A combinatorial design $(X, \mathcal{A})$ where
$X = \{x_1, x_2, \ldots, x_v\}$,
$\mathcal{A} = \{B_1, B_2, \ldots, B_b\}$,
$\mathcal{N} = \{n_1, n_2, \ldots, n_t\}$,
$f : \mathcal{N} \to \mathcal{A}$ is a one-one map,
A $c \times r$ Matrix G,
v number of $c \times c$ Matrices $D_1, D_2, \ldots, D_v$.
Output: A key predistribution in sensor nodes of $\mathcal{N}$.

for all $x_j \in X, 1 \leq j \leq v$ **do**
Find ordered set $S = \{B_{j_1}, B_{j_2}, \ldots, B_{j_r}\}$ be such that $B_{j_k} \in \mathcal{A}, x_j \in B_{j_k}; \forall k \in \{1, 2, \ldots, r\}$; $B_{j_k} \neq B_{j_l}, 1 \leq k, l \leq r$ and $\forall B \in \mathcal{A} \setminus S$, $x_j \notin B$.
Compute $A_j = (D_j.G)^T$
for all $i \in \{1, 2, \ldots, r\}$ **do**
if $f^{-1}(B_{j_i})$ exists **then**
Store the ith row of matrix A_j in node $f^{-1}(B_{j_i})$
Store the 2nd row of G in node $f^{-1}(B_{j_i})$
In node $f^{-1}(B_{j_i})$, store $POS(B_{j_i}, x_j) = i$.
end if
end for
end for

4.3.1.2 Memory requirement

It is easy to see that one node n_h contains one row from each matrix of the set M_h where $M_h \subset \{A_1, A_2, \ldots, A_v\}$ where $|M_h| = k$. The dimension of each row is c. Also, the node contains row 2 of matrix G which is $(\alpha, \alpha^2, \ldots, \alpha^r)$. It can be seen that for $(p^2 + p + 1, p + 1, 1)$ SBIBD $r = k$. Again, a node n_i stores $POS(f(n_i), x_i)$ for $i \in \mathbb{V}, \mathbb{V} \subset \{1, 2, \ldots, v\}, |\mathbb{V}| = k$. So, the overhead on each node is $O(kc + r + k)$. For most of the cases, c is a small constant. In this design $k = p+1$. Therefore, the memory overhead is $O(p)$ or $O(\sqrt{|\mathcal{N}|})$.

4.3.1.3 Shared key discovery between two nodes

Two nodes wishing to communicate securely need to agree upon a secret key. In the scheme discussed in Section 4.3.1.1, any two nodes can surely compute a shared key. We provide an algorithm that takes all arguments of Algorithm 4 and finds a shared key between two nodes. In addition, the algorithm takes two nodes as input and finds a common key shared by both of them.

Algorithm 5 Algorithm to compute common key between node n_i an n_j.

Input: Combinatorial design $(X, \mathcal{A})$ used in Algorithm 4 where
$X = \{x_1, x_2, \ldots, x_v\}$,
$\mathcal{A} = \{B_1, B_2, \ldots, B_b\}$,
$\mathcal{N} = \{n_1, n_2, \ldots, n_t\}$,
$f : \mathcal{N} \to \mathcal{A}$ is a one-one map,
A $c \times r$ Matrix G,
v number of $c \times c$ Matrices $D_1, D_2, \ldots, D_v$.
Output: Compute the common key between node n_i and n_j

1) Let, $B_y = f(n_i), B_z = f(n_j)$
2) Compute $x_m \in X, m \in \{1, 2, \ldots, v\}$ such that $x_m \in B_y \cap B_z$
3) Find $u = POS(B_y, x_m)$ and $w = POS(B_z, x_m)$
4) Compute wth column of matrix G from $(\alpha, \alpha^2, \ldots, \alpha^r)$.
5) $K_{n_i,n_j} = (u\text{th row of matrix } A_m).(w\text{th column of matrix } G)$

The most costly computation of Algorithm 5 is at step 3. This step reduces in finding all the blocks of a design that contains a particular variety. This can be found using a different construction of symmetric BIBD as discussed in Section 8.4 of [69].

Time complexity of Algorithm 5 The first step reduces in inverting the node ids. We assumed that f is invertible in constant time. So, the first step can be done in time $O(1)$. The second step computes a common variety belonging to two different blocks in the design used in Algorithm 4. Note that in a $(p^2 + p + 1, p + 1, 1)$-SBIBD, any two blocks will share a unique common variety. Computing such a variety in a $(p^2+p+1, p+1, 1)$-SBIBD is equivalent to computing a basis of the intersection of two-dimensional subspaces. This can be done in constant time using the Algorithm 3. The third step is a lookup of memory and is, too, of time complexity $O(1)$ if the items are stored in an indexed table. In the fourth step, the wth column of matrix G is calculated which is given by $(1, \alpha^w, (\alpha^w)^2, (\alpha^w)^3, \ldots, (\alpha^w)^{c-1})'$. Since the nodes store α^i for each $i = 1, 2, 3, \ldots, r$ and c is a constant, so computing the wth column of matrix G requires $O(1)$ computation. Finally, the fifth step can also be done in constant time since the vectors are of constant dimension. Therefore, the overall runtime of Algorithm 5 is $O(1)$.

Note that node n_i stores the value $u = POS(B_y, x_m)$ in the Algorithm 5, and node n_j stores the value of $w = POS(B_z, x_m)$. However, for computing the shared key, both the nodes need the values of u and w. So, the two nodes must exchange the values of u and w which will incur an additional communication cost of $O(1)$. To avoid this, every node can store the values of $POS(*, *)$ for other nodes. For example node $n_i = f^{-1}(B_y)$

needs to store the values of $POS(B_e, x_l) : 1 \leq e \leq v, e \neq y, x_l = B_e \bigcap B_y$. This will require a memory overhead of $O(\mathcal{N})$.

4.3.2 Proof of correctness of algorithms

Here, we establish the correctness of Algorithm 4 and Algorithm 5. It will be sufficient to show that after deployment, a pair of distinct nodes n_i and $n_j, 1 \leq i, j \leq v$ will be able to compute their common key $K_{n_i n_j} = K_{n_j n_i}$ using the shared key discovery method of Algorithm 5. According to Algorithm 5, both node n_i and node n_j will compute the blocks $B_y = f(n_i)$ and $B_z = f(n_j)$. Now, they can find the common element $x_m \in B_y \bigcap B_z : 1 \leq m \leq v$ using Algorithm 3. Now, node n_i will compute $u = POS(B_y, x_m)$. Similarly, node n_j will calculate $w = POS(B_z, x_m)$. Node n_i and n_j will exchange the values u and v. Node n_i will compute the wth column of matrix G from $(\alpha, \alpha^2, \ldots, \alpha^r)$ stored in it. Similarly, node n_j will calculate the uth column of matrix G from $(\alpha, \alpha^2, \alpha^r)$ stored in it. From Algorithm 3, we can see that node n_i and n_j have got the uth and wth row of matrix $A_m = (D_m.G)^T$. Hence, node n_i can compute $K_{uw} = (u\text{th row of matrix } A_m).(w\text{th column of matrix } G$. Node n_j will compute $K_{wu} = (w\text{th row of matrix } A_m).(u\text{th column of matrix } G$ in a similar way. Since $A_m.G$ is a symmetric matrix, $K_{uw} = K_{wu} = K_{n_i n_j}$. Hence, the two nodes will end up computing the same key using Algorithm 5. Therefore, the Algorithm 4 and 5 are correct. It can be noted that any row of matrix $A_k, 1 \leq k \leq v$ is contained only in exactly one node according to Algorithm 4. So, only node n_i contains the uth row of A_m and only node n_j contains the wth row of A_m. Hence, no other node can compute the common key $K_{n_i n_j}$.

4.4 Performance analysis of proposed scheme

In this section, we shall investigate the security aspects of the proposed scheme. As discussed in Section 1.1.3, sensor nodes are deployed in unattended environment often in area controlled by an adversary. So, an active adversary can compromise one or more sensor nodes of the deployment zone. If the sensor nodes are not tamper proof, the adversary can extract sensitive information from the set of sensor nodes compromised by the adversary and can use those informations to overhear the conversation between active sensor nodes.

Lemma 4.3. *For the proposed scheme, let* $\mathbf{S}$ *be the set of compromised sensor nodes. Let,* $f(\mathbf{S}) = \{f(n) : n \in \mathbf{S}\}$. *Two uncompromised nodes* n_1 *and* n_2 *will have an uncompromised link between them if and only if* $|\{B : B \in f(\mathbf{S}) \& x \in B\}| \leq c-1$, *where* $x = B_1 \cap B_2$ *and* $f(n_1) = B_1, f(n_2) = B_2$.

Proof. Follows from the fact that c is the security parameter of the scheme in Section 4.2.

Let, $x = x_\kappa$, where $\kappa \in \{1, 2, \ldots, v\}$. Then by Algorithm 4 and section 4.2, it can be said that if the matrix A_κ can be compromised, then the common key between node n_1 and n_2 can be computed. This can only be possible if and only if any c number of rows of the matrix A_κ are compromised. Let $\psi = \{n : n \in \mathcal{N} \& x_\kappa \in f(n)\}$. Hence, the nodes in ψ contain one distinct row of A_κ each. So, successful computation of the shared key is possible if and only if $|\mathbf{S} \cap \psi| \geq c$. In other words, the common key between the two nodes n_1 and n_2 will remain active if and only if $|\{B : B \in f(\mathbf{S}) \& x \in B\}| \leq c-1$. □

Proposition 4.4. *Let the total number of nodes be* N *and the security parameter be* c. *If* s *number of nodes are compromised and* $s \geq c$, *the probability that two uncompromised nodes will have an uncompromised link is given by* $\frac{\sum_{e=0}^{c-1} \binom{k-2}{e}\binom{N-k}{s-e}}{\binom{N-2}{s}}$.

Proof. Let C denote the event that the two nodes will share an uncompromised link. Let the two nodes be given by n_1 and n_2. Let, $f(n_1) = B_1$ and $f(n_2) = B_2$, where $B_1, B_2 \in \mathcal{A}$. There must be a unique $x_i \in X$ such that $\{x_i\} = B_1 \cap B_2$. Again, let the set of compromised nodes be $\mathbf{S}$, where $|\mathbf{S}| = s$. The adversary cannot compute the shared key between n_1 and n_2 iff $|\{B : B \in f(\mathbf{S}) \& x_i \in B\}| \leq c-1$. In a symmetric (v, k, λ) design, there are k number of blocks containing a particular variety. So, for any particular variety $x_i \in X$, $|\{B : B \in \mathcal{A} \& x_i \in B\}| = k$. Again, $B_1, B_2 \in \{B : B \in \mathcal{A} \& x_i \in B\}$. Therefore, $|\{B : B \in \mathcal{A} \& x_i \in B, B \neq B_1, B \neq B_2\}| = k-2$.

$P(|\{B : B \in f(\mathbf{S}) \& x_i \in B, n_1, n_2 \notin S\}| = e) = \frac{\binom{k-2}{e}\binom{N-k}{s-e}}{\binom{N-2}{s}}$. $\therefore P(C) = \sum_{e=0}^{c-1} P(|\{B : B \in f(\mathbf{S}) \& x_i \in B, n_1, n_2 \notin S\}| = e) = \frac{\sum_{e=0}^{c-1} \binom{k-2}{e}\binom{N-k}{s-e}}{\binom{N-2}{s}}$. □

We provide the values of $P(C)$ for different sets of parameters in Table 4.2. It can be seen that our scheme has high probability of existence of a live link between two uncaptured nodes even when large number of nodes are compromised. Here, p is the prime number of the symmetric balanced incomplete block design that is used in the scheme. c is the security parameter of Blom's scheme. s is the number of compromised nodes. Table 4.2 shows that this scheme has a high probability of existence of a key link between two nodes even when many nodes are compromised. Also, if p increases, the number of nodes increases and so does the probability of existence of a link between a pair of nodes.

TABLE 4.1: **Table of notations**

Notations	
X	The set of varieties of the design
$\mathcal{A}$	The set of blocks of the design
$x_1, \ldots, x_v$	The varieties of X
$B_1, \ldots, B_b$	Blocks of $\mathcal{A}$
$GF(q)$	The finite field of q elements
α	The primitive element of $GF(q)$
G	A $c \times r$ matrix as defined below
t	The total number of nodes in deployment
p	A prime power where $t \leq p^2 + p + 1$
$\mathcal{N}$	The set of nodes in deployment
f	One to one map $\mathcal{N} \to \mathcal{A}$
D_i	$c \times c$ symmetric matrices over $GF(q)$ for $i = 1, 2, \ldots, v$

TABLE 4.2: **Probability of existence of an active link between two uncompromised nodes in our scheme for different parameters**

p	c	s	Probability of existence of link
37	4	34	0.998651
37	4	77	0.869753
47	4	77	0.930515
61	4	89	0.948484
67	5	89	0.991089
67	4	128	0.886519
61	5	110	0.970800
71	5	223	0.810936

Here, p is the prime number, s is the number of compromised nodes, and c is the security parameter of the scheme.

4.4.1 Performance analysis in terms of known measures

We shall analyze the performance of our scheme in terms of two well-known measures viz. $E(s)$ and $V(s)$. These are the standard measures used for evaluating the resiliency of any key predistribution scheme. They have been defined in section 1.3.1.

Here, we will consider only the resiliency of the subnetwork consisting of nodes. $E(s)$ is the measure that shows the performance of the scheme in terms of it's resiliency against node captures. As defined above, $E(s)$ is the measure that shows the fraction of links that gets exposed when s number of nodes get compromised. So, the lesser the value of $E(s)$ is, the more resilient is the scheme to node capture attack.

Let $\mathbf{S}$ be the set of s sensor nodes. $\mathbf{S} \subseteq \mathcal{N}$. For two sensor nodes $n_i, n_j \in \mathcal{N}$, define

$$LNK(n_i, n_j) = \begin{cases} 0 & \text{if the adversary can compute the} \\ & \text{common key between node } n_i \text{and} \\ & n_j \text{using the information stored in} \\ & \text{nodes } n_\kappa, \kappa \in \mathbf{S} \\ 1 & \text{elsewhere} \end{cases}$$

From Lemma 4.3,

$$LNK(n_i, n_j) = \begin{cases} 0 & \text{if } |\{B : B \in f(\mathbf{S}) \& x \in B\}| \geq c, \\ & \text{where } x = B_1 \cap B_2 \text{and } f(n_1) = B_1 \\ & f(n_2) = B_2 \\ 1 & \text{if } |\{B : B \in f(\mathbf{S}) \& x \in B\}| \leq c-1, \\ & \text{where } x = B_1 \cap B_2 \text{and } f(n_1) = B_1 \\ & f(n_2) = B_2 \end{cases}$$

Let $\varphi(\mathbf{S}) = \frac{\sum_{i=1}^{t} \sum_{\substack{j=1 \\ j \neq i}}^{t} LNK(n_i, n_j)}{t(t-1)}$

Hence, $E(s) = EXP(\varphi(\mathbf{S}))$, where $EXP()$ is the expectation operator.

Theorem 4.5. *For our scheme with p^2+p+1 many nodes, $E(s) \leq \frac{c}{p^2+p+1}$ for $s \leq \frac{c(c+1)}{2}$*

Proof. The total number of nodes is $p^2 + p + 1$. That makes the number of links equal to $\binom{p^2+p+1}{2}$.

We take the attacker's point of view who would try to expose more links through compromising as less number of nodes as possible. In our design, a link can be exposed only if at least c number of nodes are compromised that contain one row of matrix A_h, each for some $h \in \{1, 2, \ldots, v\}$. If c number of rows are compromised, then the attacker would be able to reconstruct the matrix A_h. Since A is a $(p+1) \times c$, the attacker would be able to compute the common keys between $\binom{p+1}{2}$ pair of nodes or in other words $\binom{p+1}{2}$ links would get exposed. Let, $n_0, n_1, \ldots, n_p$ be $p+1$ nodes such that $x_i = \cap_{j=0}^{p} f(n_j)$ for any $x_i \in X, i \in \{1, 2, \ldots, v\}$. If any c of the nodes $n_0, n_1, \ldots, n_p$ is compromised by the adversary, then we would be able to reconstruct matrix A_i and hence, the links between nodes $n_0, n_1, \ldots, n_p$ will get exposed. So, the total number of exposed links will be $\binom{p+1}{2}$. Let the set of nodes compromised by the advisor for obtaining A_i be S. Hence, $|S| \geq c$. Since, the attacker's intention is to compromise as less number of nodes as possible, we can say, $|S| = c$. Again, the attacker would attempt to expose another set of $\binom{p+1}{2}$ links by compromising more nodes. The attacker can do

this through compromising another matrix $A_j, j \neq h, j \in \{1, 2, \ldots, v\}$. This time, the attacker needs to compromise $c-1$ nodes. First, an attacker selects a $j \neq h$ such that a node in S does contain a row A_j. Choosing such a j will ensure that the attacker will have to compromise $c-1$ more nodes. It can be proved that for any $j \neq h$, there is at most one node in S that contains a row of matrix A_j. So, the attacker would require to compromise $c-1$ additional nodes for exposing $\binom{p+1}{2}$ links. This way, it can be proved that the attacker would require to compromise $c-2$ nodes for exposing the next set of $\binom{p+1}{2}$ number of links and so on. This way, the attacker can compromise $c\binom{p+1}{2}$ number of links by capturing $c + (c-1) + (c-2) + \ldots + 1$ nodes or $\frac{c(c+1)}{2}$ nodes.

Hence, for $s \leq \frac{c(c+1)}{2}, E(s) \leq c\binom{p+1}{2}/\binom{p^2+p+1}{2}$ or, $E(s) \leq \frac{c}{p^2+p+1}$. □

Theorem 4.5 gives an upper bound of the extent of damage that occurs to the subnetwork consisting of nodes. Since $p^2 + p + 1 >> c$, so $E(s)$ is very close to zero or, in other words, the number of links that get exposed is small when less than $\frac{c(c+1)}{2}$ number of nodes are captured.

Lemma 4.6. *If a set of* $\mathbf{S}$ *sensor nodes get captured, then a node* $n_i \notin \mathbf{S}$ *will get disconnected from the rest of the network if and only if* $\forall x \in f(n_i), |\{B : B \in f(\mathbf{S}), x \in B\}| \geq c$.

Proof. The proof follows from Lemma 4.3 and the c security property. □

Theorem 4.7. $V(s) = 0, \forall s < (p+1)c$.

Proof. Let the attacker wants to disconnect a particular node $n_i, i \in \{1, 2, \ldots, v\}$ from the rest of the network. Let, S be the minimal set of nodes that the attacker needs to capture for disconnecting the first (uncompromised) node from the rest of the network. Let $B_j = f(n_i), j \in \{1, 2, \ldots, v\}$. Hence, $B_j \in X$. Let, $\{x_1, x_2, \ldots, x_{p+1}\} = B_j$. Let $\forall k \in \{1, 2, \ldots, p+1\}, C_k = \{B : B \in f(S) \& x_k \in B\}$. It can be seen that $f(S) = \cup_{k=1}^{p+1} C_k$.

We claim that $C_k \cap C'_k = \phi, k \neq k', 1 \leq k, k' \leq p+1$. If not, then suppose there exists a block $B_m \in C_k \cap C'_k$. Hence, $x_k, x'_k \in B_m$. So, $|B_m \cap B_j| \geq 2$. This is not possible since the design we used is a symmetric $(p^2+p+1, p+1, 1)$ design. So our assumption is wrong.

From the c-security property, we can say that $|C_k| = c, \forall k \in \{1, 2, \ldots, p+1\}$. Hence, $|f(S)| = |S| = (p+1)c$. Hence the result. □

The performance of our scheme in terms of $V(s)$ for certain value of parameters is shown in Figure 4.1. It can be seen that the value of $V(s)$ in Figure 4.1 is in agreement with the result stated in Theorem 4.7.

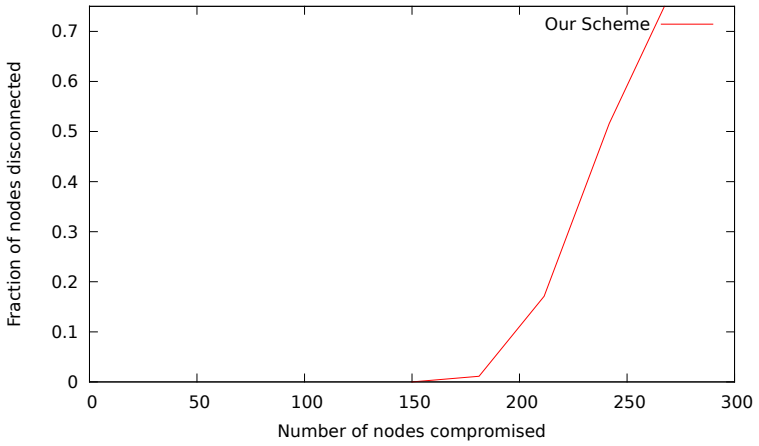

FIGURE 4.1: Graphical representation of the value of $V(s)$ with respect to the number of nodes compromised for our scheme. The parameters for this graph is $p = 29$, $c = 4$, and number of nodes $= 871$.

4.4.2 Comparative study of the scheme

Here, we compare the resiliency of our proposed scheme with other existing schemes. Some well-known standard schemes are the basic scheme of Eschenauer and Gligor [27], Lee and Stinson's quadratic and linear scheme based on transversal design in [37, 38, 40], Çamptepe and Yener's scheme in [15], the scheme of Chakrabarti et al. [16], and partially balanced incomplete block design based scheme by Ruj and Roy in [60].

The scheme of Eschenauer and Gligor in [27] is a probabilistic key predistribution scheme. This scheme uses a pool of keys. Keys are drawn randomly from the key pool with replacement and are placed in the sensor nodes. All nodes are loaded with same number of keys. This scheme does not ensure the existence of a common key between a pair of nodes. This scheme is known as the basic scheme.

Lee and Stinson [38, 40] used transversal design in key predistribution. They proposed two types of transversal design viz. linear and quadratic. In these schemes, a pair of nodes can have zero or one key in common. They used the following construction of a transversal design $TD(k, r)$ [38].

1. $X = \{(x, y) : 0 \leq x < k, 0 \leq y < r\}$.
2. $\forall i, G_i = \{(i, y) : 0 \leq y < r\}$.
3. $A = \{A_{i,j} : 0 \leq i < r \& 0 \leq j < r\}$.

They defined block $A_{i,j}$ by $A_{i,j} = (x, xi + j \mod r) : 0 \leq x < k, 0 \leq i, j < r$. Similarly for a quadratic scheme, they defined a block $A_{i,j,k}$ by
$A_{i,j,k} = (x, xi^2 + xj + k \mod r) : 0 \leq x < k, 0 \leq i, j < r$.

Each block is assigned to a node. So, the linear Lee-Stinson's scheme supports r^2 nodes, and the quadratic scheme supports as many as p^3 nodes.

Çamtepe and Yener used symmetric balanced incomplete block design in [15]. A SBIBD is a $(p^2+p+1, p+1, 1)$ design where p is a prime number. They used projective geometry for constructing the SBIBD. This scheme ensures full connectivity between nodes. Each node in this scheme contains $p+1$ keys, and every key is contained in $p+1$ nodes.

Chakrabarti et al. [16] proposed a hybrid key predistribution scheme by merging the blocks in combinatorial designs. They considered the blocks constructed from the transversal design proposed by Lee and Stinson and randomly selected them and merged them to form the sensor nodes. Though this scheme increases the number of the keys per node, it improves the resiliency of the network. The probability that two nodes share a common key is also high. Thus, it has a better connectivity.

Ruj and Roy proposed two schemes for key predistribution in [60]. They used partially balanced incomplete block design. In the first scheme, the number of nodes as well as the number of keys are equal to $n(n-1)/2$ for some positive integer n. The number of keys in a node is equal to $2(n-2)$. The number of nodes containing the same key is also $2(n-2)$. They presented another design that augments the size of the network, keeping the same number of keys in each node. The keys in the key pool also remain the same. They showed that network size can be increased in steps, keeping the same number of keys per node. However, to ensure that any pair of nodes can communicate directly, we cannot go on adding nodes in this scheme.

We have defined $E(s)$ in Section 4.4.1. $E(s)$ is the best measure of resiliency of any key predistribution scheme. A key predistribution scheme for which the value of $E(s)$ is lower offers better resiliency against node capture. So, a key predistribution scheme having low value of $E(s)$ for different values of captured nodes can withstand key compromise. Figure 4.2 shows a comparison between our scheme with these schemes in terms of $E(s)$. We measured the resiliency of the key predistribution schemes by means of simulation. The parameters of different key predistribution schemes and the number of nodes in the WSN are given in Table 4.3. We have chosen nearly equal sizes of networks for different schemes in consideration. The other parameters are chosen depending upon the network size and the system models so that the key predistribution schemes exhibit optimal performance. N is the total number of nodes in the network, and k is the number of keys per node. The value of k depends upon the other parameters of the network which in turn depend upon the network size. The last column of Table 4.3 shows whether the key predistribution scheme ensures full connectivity among the nodes or not. We used C program to evaluate the values of $E(s)$ for different values of s for all the schemes mentioned above. We compiled the source using GNU C compiler GCC

4.5.4.We considered random node capture by the adversary. In Figure 4.2, the line corresponding to the performance of our scheme almost touches the x-axis throughout the range. Hence, it can be inferred that less number of links get exposed in our scheme as compared to other schemes when same number of nodes are captured by the adversary. In other words, our scheme offers better performance than all the other schemes in terms of $E(s)$. The reason why our scheme excels in performance can be inferred from Lemma 4.3. Lemma 4.3 says that in order to compromise the links between any two nodes, the adversary is required to compromise at least c (c is the security parameter) nodes having information from the same key space as the two nodes. However, in other schemes, the same thing can be done by capturing a single node. So, even if the number of captured nodes is high enough, the value of $E(s)$ can be very low in our scheme. This fact is corroborated by the performance of our scheme as shown in Figure 4.2.

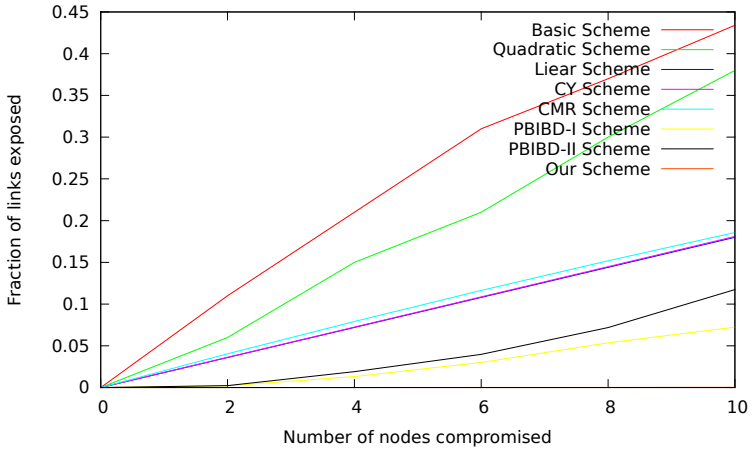

FIGURE 4.2: Graphical comparison of fraction of links exposed. With respect to the number of nodes compromised for our scheme and other schemes. The parameters for this comparison can be found in Table 4.3. The line corresponding to the performance of our scheme almost touches the horizontal axis and hence can hardly be seen.

TABLE 4.3: **Schemes with parameters that we choose for our comparisons and connectivity**

Scheme	N	k	**Full connectivity**
Basic [27]	2,415	136	No
Camtepe-Yener [14]	2,257	48	Yes
Linear [40]	2,209	30	No
Quadratic [40]	2,197	12	No
CMR [16]	2,550	28	No
PBIBD I[60]	2,415	136	Yes
PBIBD II[60]	2,450	96	Yes
Current scheme	2,257	48	Yes

N is the total number of nodes in the network, and k is the number of keys in a node.

4.5 New grid-group deployment-based design

We shall use our proposed key predistribution scheme in developing a key predistribution scheme for grid-group deployment. As mentioned earlier in Chapter 2, a grid-group deployment refers to such deployment where the entire network is broken into smaller regions called groups. The sensor nodes belonging to one group could be deemed as a mini-WSN where the sensors of a certain group communicates among themselves more frequently than with sensors of different groups. We propose a key predistribution scheme for a WSN where the network is divided into a $N \times N$ square grid. Each group in this group has got identical number of sensors.

4.5.1 The scheme

Let p be a prime number. Let $\mathcal{N} \leq p^2 + p + 1$ be the number of sensors in each group. The groups are denoted by the two tuple $(i, j), 0 \leq i, j \leq \mathcal{N}$. We shall denote the nodes of any group (i, j) as $n_{ij}^l, 0 \leq l \leq t-1$. We designate one node from each group as a supernode. This supernode has got more amount of resources than ordinary nodes in terms of memory, computational power, battery power, etc. This special node will be used for intergroup communication. The supernode of group (i, j) is denoted by $S_{i,j}$. It can be noted that a supernode $S_{i,j}$ of any group (i, j) does belong to the set $\{n_{ij}^l : 0 \leq l \leq t-1\}$. If a node $n_{i,j}^\alpha$ of group (i, j) wants to communicate with node $n_{i',j'}^\beta$ of group (i', j'), then the following steps are taken:

1. Node $n_{i,j}^\alpha$ generates a random key $\mathbb{K}$.
2. Node $n_{i,j}^\alpha$ send $\mathbb{K}$ to the supernode S_{ij}.
3. S_{ij} passes $\mathbb{K}$ to $S_{i'j'}$.
4. $S_{i'j'}$ sends $\mathbb{K}$ to node $n_{i',j'}^\beta$.

Now, the two nodes viz $n_{i,j}^\alpha$ and $n_{i',j'}^\beta$ can communicate using the key $\mathbb{K}$.

It can be noted that for accomplishing all the steps mentioned above, it is necessary to have:

1. Any two pair of nodes $n_{i,j}^\alpha$ and $n_{i,j}^{\alpha'}$ belonging to group (i, j) must be able to communicate securely $\forall \alpha \in \{0, 1, 2, \ldots, t-1\}$ and $0 \leq i, j \leq p-1$.
2. Any pair of supernodes $S_{i,j}$ and $S_{i',j'}$ belonging to two different groups (i, j) and (i', j') must be able to communicate securely where $0 \leq i, j, i', j' \leq p-1, (i, j) \neq (i', j')$.

We now state our key predistribution scheme in detail. From the above discussion, it is clear that we need to have two types of key predistribution. One type of key predistribution is for the nodes within each of the groups and the other for the supernodes belonging to distinct groups. For each of the N^2 groups, we use our key predistribution scheme discussed in Section 4.3 for key predistribution. However, we do use distinct key spaces for key predistribution in each of the groups. Hence, if all the nodes corresponding to one region get captured in the hands of the adversary, the keys in sensor nodes in other groups remain unaffected. It should be kept in mind that a supernode belongs to the group corresponding to the square region they are deployed in. Hence, a supernode contains two types of keys, one that allows it to communicate securely with other nodes in the same group they belong to and the other that allows it to communicate with other supernodes belonging to different groups. Therefore, the key predistribution in the whole network looks like the following :

1. Key predistribution for each of the N^2 groups is done by using the scheme of Section 4.3 using exclusive key spaces for all the groups.
2. A separate key predistribution using the same scheme of Section 4.3 is done for all the supernodes belonging to all the groups.

We assume that it is hard to capture a supernode until the entire square region where the supernode is located is compromised. We have assumed that the nodes within the same square region communicate more frequently than the two nodes each belonging to a separate square region. Hence, one supernode per group is sufficient to handle the burden of intergroup communication.

4.5.2 Resiliency of the network

When it comes to the resiliency of the key predistribution scheme in a grid-group deployment of the sensor network, there are three types of resiliency:

1. Intragroup resiliency : resiliency within a certain group.
2. Resiliency of the interlinks : resiliency in the set of supernodes.
3. Overall resiliency : resiliency of the entire network.

Within a group, the nodes work as a single WSN. Hence, the resiliency of the key predistribution is same as in Section 4.4. In this section, we study the resiliency of the interlinks in our key predistribution scheme. Here, too, similar to Section 4.4, we shall be using the standard measures for evaluating the resiliency of our scheme. The two measures we shall be using are $E'(s)$ and $V'(s)$.

Definition 4.8. $E'(s)$ is defined to be the fraction of interlinks between groups that get exposed when s number of supernodes are captured by the adversary. In other words, $E'(s)$ is the ratio of the interlinks present in the grid after s many supernodes are captured to the number of interlinks present in the network before s many supernodes are captured.

Let $\mathbb{S} = \{(i,j) : 0 \leq i,j \leq N-1\}$

$$\mathsf{K}_{(i,j)}(h,k) = \begin{cases} 1 & \text{if the common keybetween } S_{i,j}\text{and}S_{h,k} \\ & \text{exists} \\ 0 & \text{elsewhere} \end{cases}$$

Also, let for any group (i,j),

$$T(i,j) = \sum_{\substack{(i',j')\in\mathbb{S} \\ (i',j')\neq(i,j)}} \mathsf{K}_{(i,j)}(i',j')$$

It can be seen that in our design, all the supernodes have a common key between each other. Hence,
$T(i,j) = N^2 - 1 \qquad \forall (i,j) \in \mathbb{S}.$

Let $S \subseteq \mathbb{S}$ and $|S| = s$. Let

$$Adv^S_{(i,j)}(h,k) = \begin{cases} 1 & \text{if the adversary can compute the} \\ & \text{common key between supernode} \\ & S_{i,j}\text{and } S_{h,k}\text{using the information} \\ & \text{stored in supernode } S_{m,n}, (m,n) \in S \\ 0 & \text{elsewhere} \end{cases}$$

Let us denote,

$$P(S) = \frac{\sum_{(h,k)\in\mathbb{S}\setminus S} \sum_{\substack{(i,j)\in\mathbb{S}\setminus S \\ (i,j)\neq(h,k)}} (\mathsf{K}_{(i,j)}(h,k) - Adv^S_{(i,j)}(h,k))}{\sum_{(i,j)\in\mathbb{S}\setminus S} T(i,j)}$$

Then,

$$E'(s) = EXP(P(S)),$$

where EXP is the expectation over all $S \subseteq \mathbb{S}$ of size $|S| = s$.

We compare the experimental values of $E'(s)$ of our scheme with the experimental values of the key predistribution scheme for grid-group deployment by Ruj and Roy in [62]. Ruj and Roy considered similar deployment of sensor nodes as we did except that they used three supernodes per region whereas we used a single one. The supernodes are meant to provide interregion connectivity similar to our scheme. Both the schemes offer full connectivity between regions through supernodes. Ruj and Roy used transversal designs for key predistribution in supernodes. Figure 4.3 shows the comparison of the performance of our scheme with the scheme by Ruj and Roy in terms of $E'(s)$. The parameters of this graph can be found in Table 4.4. We considered a 37×37 square grid as the deployment zone in both the cases. In our scheme every square region contains one supernode and in Ruj and Roy scheme the number of supernodes per region is 3. Hence, the total number of supernodes is 1369 in our scheme and 4107 in Ruj-Roy scheme. The value of the security parameter of our key predistribution scheme is taken to be 4. We used C program to evaluate the values of $E'(s)$ for different values of s for both schemes. We compiled the source using GNU C compiler GCC 4.5.4. Figure 4.3 shows that our scheme is better than the scheme in [62] in terms of the number of interlinks broken when same number of supernodes are compromised in the hand of the adversary. So, for our scheme, less number of links will get broken than the Ruj-Roy scheme when the same number of nodes are captured. So, in our scheme, more interregion links remain intact than the Ruj-Roy scheme when some supernodes are captured. Thus, our scheme exhibits better performance than the Ruj-Roy scheme though it makes use of only one-third of the number of supernodes used in Ruj-Roy scheme. Our scheme reduces the cost incurred due to the deployment of large number of supernodes and also enhances the resiliency of the network against node capture.

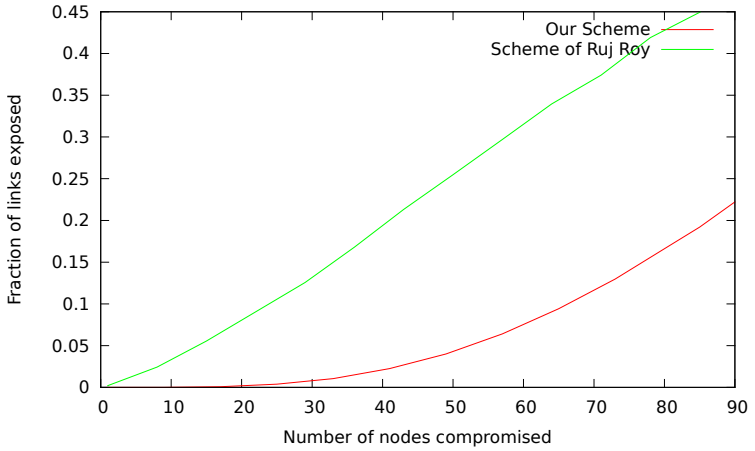

FIGURE 4.3: Graphical comparison of fraction of interlinks disconnected. This comparison is done with respect to the number of supernodes compromised for our scheme and the scheme in [62].

TABLE 4.4: **Parameters used in comparison of the proposed scheme and the Ruj and Roy scheme in Figure 4.3**

Parameters	**Ruj-Roy scheme**	**Scheme of the current study**
Number of square regions	1,369	1,369
Security parameter	-	4
Number of keys per node	13	-
Total number of nodes	4,107	1,369

Definition 4.9. $V'(s)$ is the fraction of groups that are disconnected from the rest of the groups with respect to the total number of groups when s number of supernodes are captured. In other words $V'(s)$ is the ratio of the number of groups that do not have any link to other groups after the s number of supernodes are captured to the total number of active supernodes present in the network before s many supernodes are captured.

The result proved in Theorem 4.7 is also applicable for the interlinks between supernodes in different groups. Hence, for our scheme, individual groups do not get disconnected from the rest of the network unless a large number of supernodes get captured.

Figure 4.4 shows the comparative performance of our scheme, and the Ruj-Roy scheme where the comparison is done in terms of $V'(s)$. The parameters of the graphical plot of Figure 4.4 is shown in Table 4.5. As defined above, $V'(s)$ is the fraction of nodes that get entirely disconnected from the rest of the network when s number of nodes get exposed. We used a 37×37 square grid in each case. The total number of supernodes in the entire network is $4,107$ in Ruj-Roy scheme and $1,369$ in our scheme. We have taken the security parameter of our scheme to be 4. The value of p in our scheme is 37. The number of keys (k) in a supernode is 23 in Ruj-Roy scheme. We used C program to evaluate the values of $E'(s)$ for different values of s for both schemes. We compiled the source using GNU C compiler GCC 4.5.4. Figure 4.4 shows that in our scheme, less number of nodes get detached from the network than the Ruj-Roy scheme in [62] when same number of nodes get captured by the adversary. Hence, our scheme is better than the Ruj-Roy scheme as it can keep more nodes connected to the network.

TABLE 4.5: **Parameters used in comparison of the proposed scheme and the Ruj and Roy scheme in Figure 4**

Parameters	**Ruj-Roy scheme**	**Scheme of the current study**
Number of square regions	1,369	1,369
Security parameter	-	5
Number of keys per node	23	-
Total number of nodes	4,107	1,369

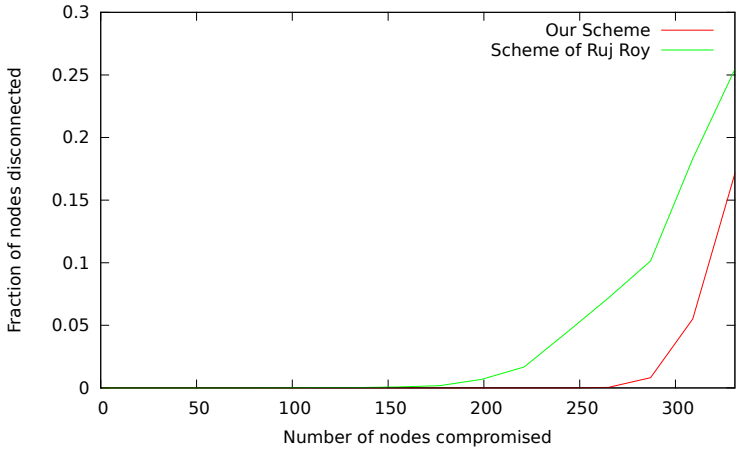

FIGURE 4.4: Graphical comparison of fraction of nodes disconnected. This comparison is done with respect to the number of nodes compromised for our scheme and the scheme in [62].

4.5.3 Overall resiliency

We shall now study the resiliency of the entire network taking into account all the groups, nodes, and supernodes.

We define $E''(s)$ as a new measure of overall resiliency in the entire network. It is defined to be the weighted average of the fractions of links exposed in every region $(i,j), 0 \leq i,j \leq N-1$ as well as the fraction of links exposed among the pair of supernodes when some nodes are compromised by the adversary in the entire network. The weight corresponding to the fraction of exposed links in a region (i,j) is equal to the number of pairs of uncompromised nodes present in that region (i,j). The weight corresponding to the fraction of exposed links between the supernodes are equal to the number of pairs of uncompromised supernodes remaining in the network. We are the first to propose this as a measure of overall resiliency in terms of fraction of links exposed in the entire network. In this measure, we separately compute the values of fraction of links exposed$(E(s_{ij}))$ in every region $(i,j) : 0 \leq i,j \leq N-1$. We also measure the value of $E(s)$ among the set of supernodes in the network. Then, we compute the weighted average of all these values of $E(s)$.

Here, we take into account the entire network consisting of all the nodes and supernodes in all the regions. Let s_{ij} be the number of nodes compromised in group (i,j) and $s = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} s_{ij}$. Also, let s_g be the number of supernodes compromised. Hence, $0 \leq s_g \leq N^2$.

Let $E(s_{ij})$ be the value of fraction of links exposed in group (i,j) when s_{ij} many nodes are captured in group (i,j). Also, let $E^g(s_g)$ be the fraction of links exposed when s_g many supernodes are compromised. After s_{ij} many nodes are compromised in region

(i, j), the number of uncompromised nodes present in region (i, j) is $\mathcal{N} - s_{ij}$. Hence, the weight corresponding to any region (i, j) is $\binom{\mathcal{N}-s_{ij}}{2}$ which is equal to the number of pairs of uncompromised nodes in region (i, j). Similarly, for the set of supernodes, the weight assigned is $\binom{N^2-s_g}{2}$. Therefore,

$$E''(s) = \frac{(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2}E(s_{ij})) + \binom{N^2-g}{2}E^g(s_g)}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2} + \binom{N^2-g}{2}}. \tag{4.1}$$

Hence, when the number of nodes captured from different groups is fixed, the overall $E''(s)$ is the weighted average of the value of $E(s_{ij})$ of all groups and the group of all supernodes.

Lemma 4.10. *When s_{ij} number of nodes are compromised in group $(i, j), 0 \leq i, j \leq N-1$ then $E''(s) < \max_{0\leq i,j<N}(E(s_{ij}))$ with a high probability where $s = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} s_{ij}$ and s is not-so-large.*

Proof. $E''(s) = \frac{(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2}E(s_{ij}))+\binom{N^2-g}{2}E^g(s_g)}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2}+\binom{N^2-g}{2}}$ Hence,

$$E''(s) < \frac{(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2}E(s_{ij})) + \binom{N^2-g}{2}E^g(s_g)}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2}}$$

Now, there are $p^2 + p + 1$ many nodes in any group which includes one supernode. If s_{ij} number of nodes are captured in group (i, j), the probability that the supernode will get captured is $\frac{s_{ij}}{p^2+p+1}$. In order to expose at least one link between two uncompromised supernodes, the adversary will have to compromise at least c nodes containing informations from the same key space of our scheme. The probability of compromising c many supernodes containing information from the same key space is very close to zero. Hence, $E^g(s_g) = 0$ with a high probability. So,

$$E''(s) < \frac{(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2}E(s_{ij}))}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\binom{\mathcal{N}-s_{ij}}{2}}$$

with a high probability, and the result follows from this. □

Corollary 4.11. *When s_{ij} number of nodes are compromised in group $(i, j), 0 \leq i, j \leq N-1$ then $E''(s) < \frac{c}{p^2+p+1}$ with a high probability where $s = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} s_{ij}$ and s is not-so-large and for all $(i, j) : 0 \leq i, j < N, s_{ij} \leq \frac{1}{2}c(c+1)$.*

Proof. Follows directly from *Lemma* 4.10 and *Theorem* 4.5. □

Corollary 4.11 gives an upper bound of the numeric value of fraction of links disconnected in the set of all uncompromised nodes of the network.

Definition 4.12. $V''(s)$ is defined to be the weighted average of the fractions of nodes disconnected from the rest of the network in a region (i, j) or in the set of supernodes when some nodes get compromised. Here, the weights are proportional to the number of pairs of uncompromised nodes present among the nodes in any region or among the supernodes. We propose and apply this measure for the first time for measuring the resiliency for such deployment of wireless sensor network.

Let $V(s_{ij})$ be the value of the fraction of nodes disconnected in region (i, j) when s_{ij} many nodes are captured. Again, let $s = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} s_{ij}$. Also let s_g be the number of supernodes captured by the adversary and $V^g(s_g)$ be the fraction of supernodes disconnected from other supernodes when s_g many supernodes are captured. After s_{ij} many nodes are compromised in region (i, j), the number of uncompromised nodes present in region (i, j) is $\mathcal{N} - s_{ij}$. Hence, the weight corresponding to any region (i, j) is $\binom{\mathcal{N}-s_{ij}}{2}$ which is equal to the number of pairs of uncompromised nodes in region (i, j). Similarly, for the set of supernodes, the weight assigned is $\binom{N^2-s_g}{2}$. Therefore,

$$V''(s) = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \binom{\mathcal{N}-s_{ij}}{2} V(s_{ij}) + \binom{N-s_g}{2} V^g(s_g)}{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \binom{\mathcal{N}-s_{ij}}{2} + \binom{N-s_g}{2}}$$

.

Lemma 4.13. *When s_{ij} number of nodes are compromised in group $(i, j), 0 \leq i, j \leq N-1$ then $V''(s) < \max_{0 \leq i,j < N}(V(s_{ij}))$ with a high probability where $s = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} s_{ij}$ and s is not so large.*

Proof. The proof is same as Lemma 4.10. □

Corollary 4.14. *When s_{ij} number of nodes are compromised in group $(i, j), 0 \leq i, j \leq N-1$ then $V''(s) = 0$ with a high probability where $s = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} s_{ij}$ and s is not-so-large and for all $(i, j) : 0 \leq i, j < N, s_{ij} \leq (p+1)c$.*

Proof. Follows immediately from Lemma 4.13 and Theorem 4.7. □

Corollary 4.14 provides a bound for the value of fraction of uncompromised nodes that get totally disconnected from the network.

We have done simulation of the performance of the key predistribution scheme for grid-group deployment taking $E''(s)$ and $V''(s)$ as the measure of the performance in the entire network. In this simulation, we randomly chose/compromised s many nodes from the entire network and then computed the values of $E''(s)$ and $V''(s)$ for them. Hence, it is equally probable for every chosen node to belong to a certain region. We measured

the values of $E''(s)/V''(s)$ for any value of s by repeating the process 100 times and taking averages of the calculated values of the $E''(s)/V''(s)$ for this 100 iterations.

The value of $E''(s)$ for different values of s can be found in Table 4.6.

TABLE 4.6: **Values of $E''(s)$ for different values of s, size of grid and number of nodes in each group**

Size of grid	Number of nodes in each group	Number of supernodes	Security parameter	s	Value of $E''(s)$
13	553	169	4	4,801	0.033041
14	307	196	4	3,001	0.010839
15	183	225	4	4,001	0.042171
18	183	324	4	5,001	0.025552
11	553	121	4	3,001	0.021120
15	381	225	3	3,126	0.034396
18	307	324	3	3,886	0.031764
16	381	256	3	5,626	0.105274
18	307	324	3	6,106	0.095601

The values of $E''(s)$ for different values of the system parameters are obtained through simulation of the key predistribution model using C program. The first column of Table 4.6 shows the dimension of the grid used as deployment zone. The second column gives the number of nodes contained in a single group. The third column shows the number of supernodes in the entire network which is equal to $R \times R$, R being the dimension of the square grid. The fourth column corresponds to the security parameter c. The fifth column gives the number of nodes compromised. The last column shows the values of $E''(s)$. It can be seen in Table 4.6 that as the grid size increases, the value of $E''(s)$ decreases while other parameters remain the same. So, the adversary needs to capture more nodes to damage the communication model considerably if the grid size is high enough. This happens as when the grid size increases, the total number of nodes in the network increases and the number of links between nodes also increases. It can be noted in this table that if the value of the security parameter is kept as low as 3 or 4, the key predistribution model can offer sufficient resiliency against node capture.

Table 4.7 gives the values of $V''(s)$ for different values of the number of captured nodes. It can be seen from Table 4.7 that the value of $V''(s)$ is very low even if a high number of nodes are captured. So, the key predistribution model is highly resilient as far as the $V''(s)$ is concerned. Also, if the size of the grid is increased, the value of $V''(s)$ gets reduced.

TABLE 4.7: **Values of $V''(s)$ for different values of s, size of grid and number of nodes in each group**

Size of grid	Number of nodes in each group	Number of supernodes	Security parameter	s	Value of V''(s)
14	553	196	3	24,000	0.114369
15	307	225	3	20,000	0.187892
14	183	196	3	18,009	0.841926
11	381	121	4	24,000	0.959935
15	307	225	4	21,002	0.027675
13	871	169	3	25,000	0.033112
7	553	49	3	6,000	0.112729
9	553	81	4	11,000	0.030479
14	307	196	4	14,000	0.000322
7	381	49	3	10,000	0.976503

4.5.4 Comparison with other schemes

Next, we compare our proposed scheme with some other key predistribution schemes that use deployment knowledge. These schemes include Du et al. 2004 [24] and 2006 [26], Liu and Ning 2003 [42] and 2005 [44], Yu and Guan 2005 [81] and 2008 [82], Zhou et al. 2006 [83], Huang et al. 2004 [30], Huang and Medhi 2007 [29], Chan and Perrig 2005 [18], Simonova et al. 2006 [65]. We have discussed them in chapter 2. Here we recapitulate the description of each of them.

Huang et al. [29, 30] used rectangular deployment zone which is divided into equal-sized regions of smaller size. In this scheme, the sensors randomly choose the keys. Huang et al. used multispace Blom scheme [10] for key predistribution. In this scheme, all nodes are identical with respect to the amount of resources they possess. This is where this scheme is different from ours. In our scheme, there are two different types of nodes viz. common nodes and agents giving rise to a heterogeneous network. Moreover, in Huang et al. scheme, the nodes in a region can communicate directly with each other with probability of >0.5; whereas, in our scheme, they can do so with a probability equal to 1 as our scheme ensures full interregion connectivity. Hence, in this scheme, more amount of computation will be required for communication than our scheme. The scheme of Huang et al. is perfectly secure against selective and random node capture attack. Hence, capture of some number of nodes by an adversary will have negligible effect to the links among the uncompromised nodes. However, if we take all the links of compromised and uncompromised nodes into account, then the fraction of links compromised will be higher.

Zhou et al. [83] used two types of sensor nodes viz. static and mobile. This scheme uses pairwise keys with each sensor within the same region. Hence, it requires high amount of memory to hold the pairwise keys if the number of sensors within a region is high

enough. If there are n number of nodes within a region, then the number of keys to be stored in a node is $O(n^2)$ under the Zhou et al. scheme; whereas, it is $O(\sqrt{n})$ in Çamptepe and Yener scheme which is used in our key predistribution scheme. Hence, our scheme is much better than Zhou et al. in terms of memory efficiency.

Liu and Ning [42, 44] used deployment knowledge. There, the whole deployment zone is split into smaller square regions like our scheme. However, in their schemes, only a single node is deployed in a square region as opposed to our scheme where there are a group of nodes deployed in a region. They used the polynomial-based scheme of Blundo et al. [11]. The deployment region is broken down into equal-sized squares $\{C_{i_c,i_r}\}i_c = 0, 1, \ldots, C-1, i_r = 0, 1, \ldots, R-1$, each of which is a cell with coordinates (i_c, i_r) denoting row i_r and column i_c . Each of the cells is associated with a bivariate polynomial. For a $R \times C$ grid, the setup server generates RC t-degree polynomials $\{f_{i_c,i_r}(x, y)\}i_c = 0, 1, \ldots, C-1, i_r = 0, 1, \ldots, R-1$, and assigns $f_{i_c,i_r}(x, y)$ to cell C_{i_c,i_r}. For each sensor, the setup server determine its home cell and its four neighboring cells which lie adjacent to the home cell in the same row and column. The setup server distributes to the sensor the coordinates of the home cell and the polynomial shares of the home cell and its neighboring cell. For example, for a sensor U_u in the cell with coordinate (r', c'), the polynomial shares $f_{r'-1,c'}(u, y), f_{r',c'-1}(u, y), f_{r'+1,c'}(u, y), f_{r',c'+1}(u, y), f_{r,c}(u, y)$ are given. For direct key establishment, a node broadcasts the coordinates of its home cell. From this coordinate, the destination node finds out the common polynomial that it shares with the broadcasting node if at all. Now, the common key can be calculated using the same method as [11].

In Simonova et al.'s [65] scheme, the number of specialized nodes depends upon the size of the network unlike ours which is constant (=1). The resiliency as given in the graph is much lower compared to our scheme. Also, resiliency in terms of nodes or regions disconnected has not been presented.

Du et al. [26] proposed another key predistribution using deployment knowledge that uses multiple space Blom scheme [10]. Under this scheme, sensors randomly choose keys from a set of different instances of Blom space. Unlike our scheme, this scheme does not guaranty full connectivity.

As we have discussed earlier, the key predistribution scheme of Ruj and Roy in [62] uses deployment knowledge. Similar to our scheme, this scheme uses the Çamptepe and Yener scheme for key predistribution within the same region. This scheme exhibits lower resiliency among the set of agents that provide interregion connectivity as discussed in previous sections. In other words, our scheme offers more resilient interregion connectivity than Ruj and Roy scheme.

Figure 4.5 shows a pictorial comparison of our scheme with standard schemes that use deployment knowledge. This comparison is based on the values of fraction of total links broken when some nodes get captured. This comparison takes into account all the links in the network which includes the links in compromised nodes as well. The parameters of the different schemes are following:

DDHV scheme has parameters $k = 200, \omega = 11$, and $\tau = 2$. LN scheme has parameters $k = 200, m = 60$, and $L = 1$; YG scheme has parameters $k = 100$; ZNR scheme has parameters $k = 100$; HMMH scheme has parameters $k = 200, \omega = 27$, and $\tau = 3$; SLW scheme has parameters $k = 16, p = 11$, and $m = 4$; Ruj-Roy scheme has parameters $k = 12$. Our scheme has parameters $p = 11$ and $c = 4$. The size of the network in DDHV, LN, YG, ZNR, and HMMH is 10,000; for SLW, it is 12,100. It is 16,093 for Ruj-Roy scheme and in our scheme. We simulated the behavior of the key predistribution schemes for random node capture attack. All schemes are implemented identical network. It can be seen in Figure 4.5 that our scheme offers better performance than similar schemes that make use of deployment knowledge up to a certain limit of the number of nodes captured by the adversary. We used C program for running the simulation.

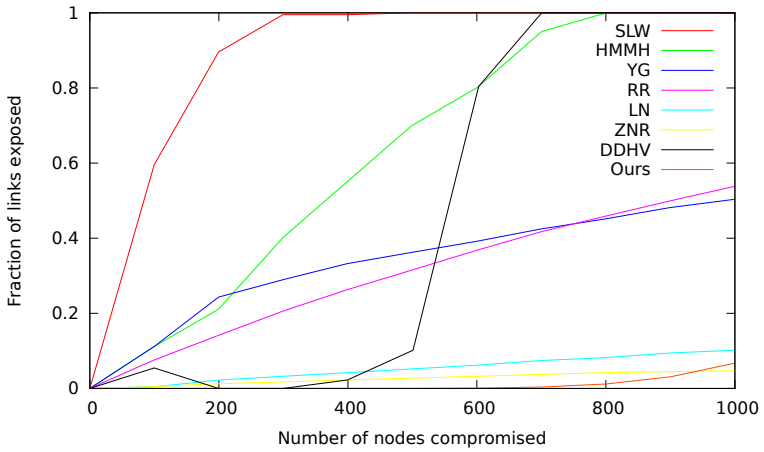

FIGURE 4.5: Graphical comparison of fraction of links disconnected. This comparison is done with respect to the number of nodes compromised for our scheme and the schemes in [18, 24, 26, 29, 30, 42, 44, 62, 65, 81–83].

The reason why our scheme excels in performance can be inferred from Lemma 4.3 and Proposition 4.4. Lemma 4.3 says that in order to compromise the links between any two nodes, the adversary is required to compromise at least c (c is the security parameter) nodes having information from the same key space as the two nodes. However, in most of the other schemes, the same thing can be done by capturing a single node. Again, Proposition 4.4 says that the probability of existence of a link between a pair of nodes is high even if many nodes are compromised. So, even if the number of captured nodes is high enough, the value of fraction of broken links can be very low in our scheme. This fact is corroborated by the performance of our scheme as shown in Figure 4.5.

We present a comparative study of communication, storage, and scalability of several schemes in Table 4.8. This table gives a comparison with respect to communication, storage cost, etc. of our scheme and the schemes in [18, 26, 29, 30, 42, 44, 62, 65, 81–83]. The first column of Table 4.8 shows the name of the scheme. The second column corresponds to the type of deployment used by the key predistribution scheme. The third column shows the type of nodes in the WSN. There are two types of sensor nodes viz. homogeneous and heterogeneous. All the nodes in a homogeneous network are identical in terms of the resources they possess. However, in heterogeneous networks, there are different types of nodes who mainly differ in the amount of computational resource built inside them. The fourth column shows the communication cost of each key predistribution scheme. When two nodes wish to communicate, they need to exchange some information before a secure communication can start. This information may be their unique identifiers or something else that is required to compute the shared key between them. The storage column gives the amount of memory needed to store the keys a node. Here, N is the number of sensors in the network, and g is the number of groups. The last column says whether the key predistribution scheme is scalable or not. The communication cost of our scheme is $O(\log N)$, and the storage overhead is $O(N^{\frac{1}{4}})$. Our scheme consumes less amount of memory than other schemes except the DDHV scheme in [24, 26] and the Yu-Guan scheme in [81, 82] that uses constant amount of storage. However, our scheme outperforms both of them in terms of resiliency measure used in the comparison in Figure 4.5.

TABLE 4.8: **Comparison of schemes with respect to type of deployment, node, communication, and storage overhead and scalability Here, N is the total number of sensors in the network. g is the number of groups in the network. [a]the storage for small sensor nodes, and [b] the storage for agents.**

Schemes	Deployment	Nodes	Communication cost	Storage	Scalability
DDHV [24, 26]	Grid-group	Homogeneous	$O(1)$	$O(1)$	Scalable
LN [42, 44]	Grid	Homogeneous	$O(\log N)$	$O(\sqrt{N})$	Not scalable
YG [81, 82]	Grid-group	Homogeneous	$O(1)$	$O(1)$	Not scalable
ZNR [83]	Group	Heterogeneous	$O(\log N)$	$O(N/g)^{a}$ $O(N)^{b}$	Not scalable
HMMH [30]	Grid-group	Homogeneous	$O(1)$	$O(\sqrt{N})$	Scalable
HM [29]	Grid-group	Homogeneous	$O(1)$	$O(\sqrt{N})$	Scalable
PIKE [18]	Grid	Homogeneous	$O(\log N)$	$O(\sqrt{N})$	Not scalable
SLW [65]-2	Grid-group	Heterogeneous	$O(\log N)$	$O(\sqrt{N/g})$	Scalable
Ruj-Roy [62]	Grid-group	Heterogeneous	$O(\log N)$	$O(N^{\frac{1}{4}})^{a}$ $O(N^{\frac{1}{4}})^{b}$	Not scalable
Current scheme	Grid-group	Heterogeneous	$O(\log N)$	$O(N^{\frac{1}{4}})^{a}$ $O(N^{\frac{1}{4}})^{b}$	Not scalable

4.6 Concluding Remarks

In this chapter, we have presented a key predistribution scheme for a wireless sensor for a grid-group-based deployment. Here, the entire deployment zone is a square which is

divided into a number of smaller squares. Each square is identical in terms of physical area and number of sensor nodes. The sensor nodes belonging to a smaller square form a group among themselves. All the groups contain two types of nodes viz. ordinary and nodes. A node within a group can make direct communication to any other node in the same group or region. Nodes belonging to two different group communicate via special nodes called nodes. These nodes are more resourceful than ordinary nodes in terms of memory, computational power, and energy. We used two types of different key predistribution schemes for this deployment. The ordinary sensor nodes and the node within a group use symmetric design-based key predistribution scheme proposed in [14] for within group communication. The nodes contain two types of keys. It can communicate to other sensor nodes belonging to the same group. Moreover, it can communicate with other nodes by means of a separate key predistribution scheme. Our scheme offers better resiliency than other existing schemes like the most notable scheme by Ruj & Roy [62] and the Zhou et al. scheme in [83]. We have shown that our scheme ensures that there will be high probability of existence of a common unexposed link between two nodes belonging to two different groups even if a considerable number of nodes are compromised by the adversary.

Chapter 5

A New Key Predistribution Scheme for Grid-Group Deployment of Wireless Sensor Networks

This chapter is based on the work discuused in [3]. In this chapter, we discuss a new scheme for key predistribution in wireless sensor networks for a grid-group deployment of the sensor networks. We have used combinatorial designs in this purpose. Our main contribution is to develop a key predistribution scheme in the special nodes called agents that connects the sensor nodes of one region to those of a different region. This work mainly focusses on setting up key-links between different groups which makes this scheme more analogous to the key predistribution scheme by Ruj and Roy in [62]. Ruj-Roy used only three agents per region in their scheme. In our work the number of agents per region is a variable that depends upon the size of the deployment grid. Our scheme offers better performance than well known Ruj-Roy scheme and some other standard existing schemes that uses deployment knowledge.

5.1 Finite affine plane

We have discussed finite affine geometry in section 2.1. Here, we recapitulate the discussion before moving into describing the actual key predistribution. A finite affine plane consists of a set of points and a set of lines, having the following properties:

1. Given any two distinct points, there is exactly one line incident with both of them.

2. Given any line L and a point p not on L, there exists exactly one line L' through p that does not intersect L

3. There are four points such that no line is incident with more than two of them.

This construction of affine planes can be found in [66, 69].

Let q be a prime. Let $P = \mathbb{Z}_q \times \mathbb{Z}_q$ be a set of points. $|P| = q^2$. We define,

$$L_{\alpha,\beta,1} = \{(x, y, 1) : (x, y) \in Z_q \times Z_q, (x, y) \neq (0, 0)\}$$

,

$$L_{1,\beta,0} = \{(1, x, 0) : x \in Z_q\}$$

and

$$L_{0,1,0} = \{(0, 1, 0)\}.$$

Let $L = L_{\alpha,\beta,1} \bigcup L_{1,\beta,0} \bigcup L_{0,1,0}$.

It can be shown that taking P as the set of points and L as the set of lines, we have an affine plane over $GF(q)$. This affine plane is denoted as $AG(2, q)$. For a proof we refer the readers to [69].

We see that $|L_{\alpha,\beta,1}| = q^2 - 1$, $|L_{1,\beta,0}| = q$ and $|L_{0,1,0}| = 1$. So, there are q^2 points and $q^2 + q$ lines.

Let $(a, b, c) \in L$ be an arbitrary line of $AG(2, q)$. The points belonging to this line are given by the solution pair (x, y) where $ax + by = c$ in $GF(q)$ If $a \neq 0$, then $x = a^{-1}c - a^{-1}by$, $c = \{0, 1\}$. For every $y \in \mathbb{Z}_q$ we will have a unique value of x. Since there are q elements in $\mathbb{Z}_q$, so there are q points on the line $ax + by = c$ where $a \neq 0$.

Alternatively, if $a = 0$, then $b \neq 0$. So, it follows that $y = b^{-1}c$. Here y is constant and for each element of $\mathbb{Z}_q$, we will have a distinct value of x. So, in both the cases there are exactly q points on a line. Hence, the number of points in a line of $AG(2, q)$ is q.

The number of lines through a fixed point (x, y) when $(x, y) \neq (0, 0)$ is the number of lines through (x, y) not passing through origin plus the number of lines which pass through the origin. Since the number of lines through two points is one, so the number of lines through (x, y) and $(0, 0)$ is 1. Now, the number of lines through (x, y) and not passing through origin is the number of solution pairs (δ, γ) to the equation $\delta x + \gamma y = 1$. This equation can be written as $\delta x = 1 - \gamma y$ and as $\gamma \in \mathbb{Z}_q$ can take q values, δ is known. So this equation has q solutions. Hence, the number of lines through a point (x, y) when $(x, y) \neq (0, 0)$ is $q + 1$. Algorithm 6 discusses a method to find all lines passing through a single line.

Algorithm 6 Algorithm to find the set of lines passing through a point in $AG(2, q)$

Input: The design D of $(q^2, q, 1)$ BIBD based on $AG(2, q)$. A point (x, y) in $AG(2, q)$
Output: The set $\mathbb{S}$ of lines passing through (x, y)

```
if (x, y) = (0, 0) then
  for i = 0 → q − 1 do
    S = S ⋃ (1, i, 0)
  end for
  S = S ⋃ (0, 1, 0)
else if x = 0 then
  for i = 0 → q − 1 do
    S = S ⋃ (i, y^{-1}, 1)
  end for
  S = S ⋃ (1, 0, 0)
else if y = 0 then
  for i = 0 → q − 1 do
    S = S ⋃ (x^{-1}, i, 1)
  end for
  S = S ⋃ (0, 1, 0)
else
  for i = 0 → q − 1 do
    S = S ⋃ (i, (1 − ix)y^{-1}, 1)
  end for
  S = S ⋃ (1, −xy^{-1}, 0)
end if
```

It can be seen that each line in $L_{1,\beta,0} \bigcup L_{0,1,0}$ passes through origin (i.e. they satisfy the equation $ax + by = c$ where $(x, y) = (0, 0)$ and $(a, b, c) \in L_{1,\beta,0} \bigcup L_{0,1,0}$). Hence, the number of lines passing through origin is equal to $|L_{1,\beta,0} \bigcup L_{0,1,0}| = q + 1$.

So, the number of lines through any point in $AG(2, q)$ is $q + 1$.

Hence, taking points as varieties and lines as blocks $AG(2, q)$ is an $(q^2, q^2 + q, q + 1, q, 1)$ design. Let us denote this design as $\mathcal{D}$.

So, the dual design of $\mathcal{D}$ will have $q^2 + q$ varieties, q^2 number of blocks, each block containing $q + 1$ varieties. It can be verified that two blocks will have exactly one common variety.

5.2 New Design From $(q^2, q, 1)$ design

Here, we shall discuss a new combinatorial design developed using an affine plane $AG(2, q)$ where q is a prime number. We have seen in section 5.1 that an affine plane $AG(2, q)$ is a $(q^2, q, 1)$-BIBD. This design contains q^2 many varieties, $q^2 + q$ many blocks each of size q. The replication number of any variety is $q+1$. Any two varieties do occur

together in exactly one block. The dual of a $(q^2, q, 1)$ design contains $q^2 + q$ varieties, q^2 number of blocks, each block containing $q + 1$ varieties. Moreover, any two blocks of this design will contain exactly one common variety. This common variety is not unique. the replication number of any variety in the dual design is q. So, there are $\binom{q}{2}$ many pairs of blocks that share a unique variety. We shall use algorithm 7 for generating a new design $\Re$ from the affine geometry $AG(2, q)$.

Definition 5.1. $\forall x, y \in \{0, 1, \ldots, q-1\}$, $U_{x,y}$ is the ordered set of lines in $AG(2, q)$.

The set of lines passing through a point in $AG(2, q)$ can be found in Algorithm 6. The elements of the set can be sorted using some convention.

Definition 5.2. $\mathcal{K} = \{K_{ij} : 0 \leq i, j \leq q^2 - 1, i < j\}$ is a set of $\binom{q^2}{2}$ elements or varieties.

We claim that the set $\mathbf{B}$ in the output of the Algorithm 7 is a design. The blocks are B_{ij} where $0 \leq i \leq q^2 - 1$ and $0 \leq j \leq q$. Hence, there are $q^3 + q$ many blocks in this design. The number of varieties is $\binom{q^2}{2}$. We prove some properties of the new design below.

Lemma 5.3. *Given any* $i, j \in \{0, 1, \ldots, q^2 - 1\}$ *and* $i < j, |\{B_{mn} : 0 \leq m \leq q^2 - 1, 0 \leq n \leq q, K_{ij} \in B_{mn}\}| = 2$.

Proof. Algorithm 7 implies that there exists an $l, 0 \leq l \leq q$ such that $K_{ij} \in B_{il}$. Similarly, there exists an $r, 0 \leq r \leq q$ such that $K_{ij} \in B_{jr}$. Hence, the result. □

Lemma 5.4. *For all* i, j *such that* $0 \leq i \leq q^2 - 1$ *and* $0 \leq j \leq q$, $|B_{ij}| = q - 1$.

Proof. It can be seen from Algorithm 7 that for any point (x_1, y_1), $x_1, y_1 \in \{0, 1, 2, \ldots, q-1\}$, $|U_{x_1,y_1}| = q + 1$. Let $j \in \{0, 1, \ldots, q\}$ and the j^{th} line in U_{x_1,y_1} be (x, y, z). Then according to Algorithm 7, $B_{ij} = \{K_{ik} : k = x_2 q + y_2, 0 \leq k \leq q^2 - 1, k \neq i, x_2 x + y_2 y = z\}$ where $i = x_1 q + y_1$ and $0 \leq x_2, y_2 \leq q - 1$. Hence, $|B_{ij}| = |\{k : k = x_2 q + y_2, 0 \leq k \leq q^2 - 1, k \neq i, x_2 x + y_2 y = z\}| = |\{k : k = x_2 q + y_2, 0 \leq k \leq q^2 - 1, x_2 x + y_2 y = z\}| - 1$. Therefore, $|B_{ij}| = |\{(x_2, y_2) : 0 \leq x_2, y_2 \leq q - 1, x_2 x + y_2 y = z\}| - 1$. Now, the set $\{(x_2, y_2) : 0 \leq x_2, y_2 \leq q - 1, x_2 x + y_2 y = z\}$ is the set of points passing through the line (x, y, z). So, $|\{(x_2, y_2) : 0 \leq x_2, y_2 \leq q-1, x_2 x + y_2 y = z\}| = q$. Hence, $|B_{ij}| = q - 1$. □

Lemma 5.5. $\forall i \in \{0, 1, \ldots, q^2 - 1\}, 0 \leq j, j' \leq q, j \neq j', B_{ij} \bigcap B_{ij'} = \phi$

Proof. Let us assume that there exist i, j, j' where $0 \leq i \leq q^2 - 1$ and $0 \leq j, j' \leq q$ such that $B_{ij} \bigcap B_{ij'} \neq \phi$. Hence, from algorithm 7 it can be said that there exists an $x : 0 \leq x \leq q^2 - 1, x \neq i$ such that $K_{ix} \in B_{ij}$ and $K_{ix} \in B_{ij'}$. Let (a_1, b_1, c_1)

Algorithm 7 Algorithm to generate the blocks of the combinatorial design

Input: The design D of $(q^2, q, 1)$ BIBD based on $AG(2, q)$. The varieties are given by $(a, b) : a, b \in \{0, 1, 2, \ldots, q-1\}$. The blocks of D are given by $L_1 \bigcup L_2 \bigcup L_3$, $L_1 = \{(\alpha, \beta, 1) : \alpha, \beta \in \{0, 1, 2, \ldots, q-1\}\}$,$L_2 = \{(1, \beta, 0) : \beta \in \{0, 1, 2, \ldots, q-1\}\}$ and $L_3 = \{(0, 1, 0)\}$. The set $\mathcal{K} = \{K_{ij} : 0 \leq i, j \leq q^2 - 1, i < j\}$.

Output: A combinatorial design $(\mathcal{K}, \mathbf{B})$, where $\mathbf{B} = \{B_{ij} : 0 \leq i \leq q^2 - 1, 0 \leq j \leq q\}$

```
for x1 = 0 → q − 1 do
  for y1 = 0 → q − 1 do
    U_{x1,y1} be the ordered set of lines that pass through point (x1, y1)
    for x2 = 0 → q − 1 do
      for y2 = 0 → q − 1 do
        if (x1, y1) = (x2, y2) then
          continue
        else
          a = y2 − y1, b = x1 − x2 and c = y2x1 − y1x2
          if c ≠ 0 then
            x = ac^{−1}, y = bc^{−1} and z = 1
          else if a ≠ 0 then
            x = 1, y = ba^{−1} and z = 0
          else
            x = 0, y = 1 and z = 0
          end if
          Let j be the position of (x, y, z) in U_{x1,y1}
          m = x1q + y1, n = x2q + y2
          if m < n then
            B_mj = B_mj ⋃ K_mn
          else
            B_mj = B_mj ⋃ K_nm
          end if
        end if
      end for
    end for
  end for
end for
```

and (a_2, b_2, c_2) be the two lines in j^{th} and j'^{th} position in the ordered set U_{x_1,y_1} where $i = x_1 q + y_1$. Let $x_2 = \lfloor x/q \rfloor$ and $y_2 = x \mod q$. Then from Algorithm 7, it can be written that $x_2 a_1 + x_2 b_1 = c_1$ and $x_2 a_2 + x_2 b_2 = c_2$. Hence, both the lines (a_1, b_1, c_1) and (a_2, b_2, c_2) pass through point (x_2, y_2). But both the lines (a_1, b_1, c_1) and (a_2, b_2, c_2) pass through point (x_1, y_1) as $(a_1, b_1, c_1), (a_2, b_2, c_2) \in U_{x_1,y_1}$. Two distinct lines cannot pass through two common points. Hence, our assumption was wrong. □

Lemma 5.6. *$\forall i, j \in \{0, 1, \ldots, q^2 - 1\}, i \neq j$ there exists a unique pair m, n where $0 \leq m, n \leq q$ such that $B_{im} \bigcap B_{jn} \neq \phi$. Also, $|B_{im} \bigcap B_{jn}| = 1$.*

Proof. We prove the fact for a pair of arbitrary $i, j \in \{0, 1, \ldots, q^2 - 1\}$. We assume that $i < j$. Let $x_1 = \lfloor i/q \rfloor, y_1 = i \mod q$. Similarly, let $x_2 = \lfloor j/q \rfloor, y_2 = j \mod q$. Since,

$i \neq j$, $(x_1, y_1) \neq (x_2, y_2)$. Let the common line passing through (x_1, y_1) and (x_2, y_2) be $(a, b, c), 0 \leq a, b \leq q-1, c \in \{0, 1\}$. Then $(a, b, c) \in U_{x_1,y_1}$ and $(a, b, c) \in U_{x_2,y_2}$. Let m, n be the position of (a, b, c) in U_{x_1,y_1} and U_{x_2,y_2} respectively. This implies that $K_{ij} \in B_{im}$ and $K_{ij} \in B_{jn}$. □

Theorem 5.7. *Let* $CL_i = \bigcup_{j=0}^{q} B_{ij}$. *Then* $\forall i, j \in \{0, 1, \ldots, q^2 - 1\}$ *such that* $i \neq j$, $CL_i \bigcap CL_j \neq \phi$. *Also,* $\forall i, j \in \{0, 1, \ldots, q^2 - 1\}$ *such that* $i \neq j$, $|CL_i \bigcap CL_j| = 1$.

Proof. Follows immediately from lemma 5.6. □

The above discussions imply that the design $(\mathcal{K}, \Re)$, where $\mathcal{K} = \{K_{ij} : 0 \leq i, j \leq q-1, i < j\}$ and $\Re = \{B_{ij} : 0 \leq i \leq q^2 - 1, 0 \leq j \leq q\}$ is not a BIB Design. This is because of the fact that not all the pairs of nodes occur together in the blocks of $\Re$. A particular variety occurs in only two blocks (from lemma 5.4) each of which contains $q - 2$ other distinct varieties. Thus, a particular variety occurs together once with only $2(q - 2)$ varieties and does not occur together with any of the other $\binom{q^2}{2} - 2q + 4$ varieties of the design. Hence, $(\mathcal{K}, \Re)$ is not a BIBD. It is rather a partially balanced design with two associate classes. In other words, $(\mathcal{K}, \Re)$ is a $PB[q-1, 1, 0; \binom{q^2}{2}]$ design where $\lambda_1 = 1, \lambda_2 = 0$. For any variety $K_{ij} \in \mathcal{K}$, the first associates of K_{ij} are those $2(q - 2)$ varieties belonging to the two blocks where K_{ij} occurs. All $\binom{q^2}{2} - 2q + 4$ many other varieties are the second associates.

We now explain the construction of $(\mathcal{K}, \Re)$ according to Algorithm 7. We store the blocks in $B_{mj}, 0 \leq m \leq q^2-1, 0 \leq j \leq q$. So the number of blocks is $q^2(q+1)$. We explain this by showing the construction of an arbitrary block $B_{mj}, 0 \leq m \leq q^2-1, 0 \leq j \leq q$. Let (x, y) be a point in $AG(2, q)$ such that $m = qx + y$. There is a unique point (x, y) in $AG(2, q)$ such that $m = qx + y$ holds. Now, in Algorithm 7, we denoted by $U_{x,y}$, the ordered set of lines passing through (x, y) in $AG(2, q)$. Let the j^{th} line in $U_{x,y}$ be (a, b, c). We know that there are $q - 1$ other points than (x, y) that are contained in the line (a, b, c). Let these points be $(x_1, y_1), (x_2, y_2), \ldots, (x_{q-1}, y_{q-1})$. Also let $m_i = qx_i + y_i, 1 \leq i \leq q - 1$. Now, B_{mj} is constructed as follows:

$$B_{mj} = \{K_{\min(m_i,m),\max(m_i,m)} : 1 \leq i \leq q - 1\}.$$

Before we actually move into describing our key predistribution scheme let us discuss why we map a design into a key predistribution scheme. So far it is clear that a combinatorial design contains a set of varieties and a set of blocks which are nothing but the subsets of the set of varieties. Now if we replace the varieties by the keys then the blocks will be sets of keys. The general method of key predistribution using combinatorial designs is to load the sensor nodes with the blocks of keys. Each of the blocks form the key ring

of a node. Number of keys in a node is equal to the size of each block. Two nodes have a common key only if their corresponding blocks do share a common variety. If such a common key exists then the pair of nodes can communicate secretly using that key. This is the general idea behind any key predistribution scheme based on a combinatorial design.

In the following sections we shall see how this design can be used to develop a key predistribution scheme for the grid-group deployment of sensor nodes.

5.3 New Key Predistribution Scheme

Here, our aim is to build a key predistribution scheme for a sensor node consisting of N number of nodes. In many cases, the deployment area of the sensor network is physically vast. Due to power-limitations every pair of nodes can not communicate with each other. Hence, the deployment zone is broken into smaller regions as in [42, 44] and in [29]. Then sensor nodes are deployed in those smaller regions such that the nodes falling into one particular square form a group among themselves. The deployment zone is split into a $q \times q$ square grid. Here q is a prime number. So, there are q^2 many square regions in the grid. Each square of the grid is identified by its two tuple co-ordinate (i, j). A square region with co-ordinate (i, j) consists of $n_{i,j} \geq q+1$ nodes. We place two types of sensor nodes in each of the q^2 square regions. Each square region contains $q + 1$ many special nodes called agents. All other nodes are common nodes. Hence, any square region with identifier (i, j) contains $q+1$ many agents and $n_{ij} - q - 1$ many common nodes. Agents have more amount of memory than the common nodes. All the nodes in a group can communicate directly. If a pair of nodes from two different groups want to communicate then this is done via the agents. Agents have a bigger transmission range. An agent can communicate with any other node within the same group and agents belonging other groups that lie within its own transmission range.

5.3.1 Key Predistribution Within a Region

Within a group key predistribution is done using the scheme in [14] by Çamptepe and Yener. This scheme ensures existence of a common key between every pair of nodes. Hence, all nodes within a region can communicate directly with any other node within the same region. For all the n_{ij} number of nodes which includes all the common nodes and agents in region $(i, j) : 0 \leq i, j \leq q - 1$, key predistribution is done using the same scheme but using a distinct set of keys. Hence, the key-set belonging to two distinct groups are disjoint. The scheme of Çamptepe and Yener is based on symmetric

balanced incomplete block design. With this scheme the total number of nodes that can be supported equals $p^2 + p + 1$, where p is any prime number. Hence, for each region (i, j), we need to select a prime number p such that p is the least prime number satisfying $n_{ij} \leq p^2 + p + 1$. Since in our design there are $q + 1$ agents in a region/group, the number of common nodes is $n_{ij} - q - 1$ per group. The total number of keys is equal to $p^2 + p + 1$. Since the dimension of the grid is $q \times q$, total number of nodes that can be supported is $q^2 \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} n_{ij}$. For the sake of simplicity we assume that all the regions contain $p^2 + p + 1$ many nodes. Then the number of nodes that can be supported is equal to $q^2(p^2 + p + 1)$. We choose a distinct set of $p^2 + p + 1$ many keys for each region. Keys are distributed to all the nodes in a group according to Algorithm 1 of [14]. So, the size of the key-pool will be $q^2(p^2 + p + 1)$. If all the nodes in a region get captured by the adversary and the keys loaded into them get exposed then the keys in the nodes belonging to other regions will remain unaffected. Ruj and Roy devised a technique in algorithm 2 of [62] for determining the common keys for the key predistribution scheme of Çamptepe and Yener. This algorithm computes the shared key between two nodes in constant time. Using this algorithm any two nodes belonging to the same region can compute their common key in $O(1)$ time.

5.3.2 Key Predistribution Among Agents

Let us denote the region with co-ordinate (i, j) by R_{ij} where $0 \leq i, j \leq q - 1$. As we mentioned before each region/group contains $q + 1$ number of agents. The agents in region R_{ij} are denoted by a_{ij}^k where $0 \leq k \leq q$. Each of the agents contains two types of keys. First type of keys allow them to communicate with other nodes in the same region they belong to as mentioned in previous section. The other type of keys are for communication with agents belonging to other regions. The key predistribution among the agents are done with the new block design proposed in section 5.2. The varieties are replaced by the key identifiers. Hence, there are $\binom{q^2}{2}$ many keys. Then we assign the output blocks of Algorithm 7 to the agents belonging to different regions. Agent a_{ij}^k gets the block B_{rk} where $r = iq + j$. For example the $q + 1$ agents of region $(0, 0)$ get blocks $B_{00}, B_{01}, \ldots, B_{0q}$ respectively. Similarly the agents of region $(q - 1, q - 1)$ get blocks $B_{(q^2-1)0}, B_{(q^2-1)1}, \ldots, B_{(q^2-1)q}$ respectively. Hence, each agent gets $q - 1$ additional keys other than the $p + 1$ keys of the within region key predistribution scheme mentioned in section 5.3.1. Therefore a total of $p + q$ keys are stored into an agent.

Lemma 5.8 proves that in any two regions $R_{i,j}$ and $R_{i',j'}$, there is a unique pair of agents $a_{ij}^k, a_{i'j'}^{k'}$ such that agent a_{ij}^k and $a_{i'j'}^{k'}$ do share a common secret key. This fact ensures the existence of a common secret key between two regions. If two nodes belonging to two different regions wish to communicate, they need to find their respective agents who

have a shared key. Since a node does have common secret keys with agents in the same region, a secret key-link can be established between any two nodes belonging to two different regions via the agents sharing a common key. This secret key-link can be used to communicate a new secret key between the two nodes from two different groups. This new key can thereafter be used for secret message transfer between the same nodes. So, if two nodes from two different regions wish to communicate they need to identify the two agents in their respective regions who share a common key. From Algorithm 7 it can be interpreted that the agents having a common key between them correspond to the same line in the affine plane $AG(2, q)$. The line can be found by computing the common line between the two points (i, j) and (i', j'). We discuss this in details in section 5.3.4.

5.3.3 Study of Connectivity

In this section we shall study the connectivity of our key predistribution scheme. Similar to our key predistribution scheme, this study has two parts. One for the key predistribution scheme among the sensor nodes within a region and the other for the key predistribution among the agents. As we have used the Çamptepe and Yener scheme [14] for key predistribution within any region and as Çamptepe and Yener scheme offers full connectivity, each pair of sensor nodes within a region are bound to share a common secret key. Now, we shall discuss the connectivity offered by our key predistribution scheme. Alternately, we shall be discussing the connectivity among the agents of different regions.

Lemma 5.8. *For any two regions R_{ij} and $R_{i'j'}$ there exists unique agents a^k_{ij} and $a^{k'}_{i'j'}$ such that a^k_{ij} and $a^{k'}_{i'j'}$ have a common key.*

Proof. Follows directly from Lemma 5.6 and the key predistribution strategy in the agents. □

Lemma 5.9. *For all $i, j, 0 \leq i, j \leq q-1, 0 \leq j \leq q$ and for all $k, 0 \leq k \leq q$, agent a^k_{ij} contains $q-1$ many keys for inter region communication.*

Proof. Follows directly from Lemma 5.4 and the key predistribution strategy. □

Note: Apart from these $q-1$ keys each agent also contains the keys that allow it to securely connect to other nodes in the same group.

Lemma 5.10. *If two agents have a shared key, then it is a pairwise key i.e. the key is present only in those two agents.*

Proof. Follows directly from Lemma 5.3 and the key predistribution strategy. □

Lemma 5.11. *If an agent belonging to any region R_{ij} is captured by the adversary, then $q-1$ many inter-region links get exposed.*

Proof. From Lemma 5.9 and Lemma 5.8 we can say that an agent contains $q-1$ many pairwise keys with $q-1$ many agents belonging to $q-1$ many distinct regions. There is no other agent from R_{ij} sharing a key with any agent of those regions. Hence, the result. □

Corollary 5.12. *In order to disconnect one agent at least $q-1$ agents must be compromised.*

From the discussion above, it is clear that with our proposed scheme every pair of regions are bound to be connected by a secure key link via a pair of agents one from each region. Hence, this scheme offers full inter-region connectivity. It can also be noted that in order to disconnect a region from all other q^2-1 region without actually capturing any of the $q+1$ agents of the region the adversary needs to capture at least q^2-1 agents. These are some of the benefits of choosing $PB[q-1,1,0;\binom{q^2}{2}]$ as the basic combinatorial design for building our key predistribution scheme. In later section we shall investigate some more advantages of our scheme in terms of performance.

5.3.4 Shared Key Discovery

As we mentioned in previous section, two nodes from two different region do not share common keys. Hence, if two nodes from two different regions wish to communicate securely they must first establish a common secret key. This secret key can be established using a technique called path-key discovery. Since for any two region there exists a pair of agents who do have a common secret key. Algorithm 8 provides a method for identifying the agents corresponding to two regions that share a common key.

Algorithm 8 Agent Discovery Algorithm

Input: The design D of $(q^2,q,1)$ BIBD based on $AG(2,q)$.
Output: The identifier of the agent in region (x_1,y_1) that share a common key with an agent in region (x_2,y_2)

```
Let, a = y2 − y1, b = x1 − x2 and c = y2x1 − y1x2
if c ≠ 0 then
    x = ac^{-1}, y = bc^{-1} and z = 1
else if a ≠ 0 then
    x = 1, y = ba^{-1} and z = 0
else
    x = 0, y = 1 and z = 0
end if
```

The 3-tuple (x, y, z) in the output of Algorithm 8 correspond to the line in $AG(2, q)$ that passes through the points (x_1, y_1) and (x_2, y_2). Recall the construction of the agents in Algorithm 7. An agent in the region with co-ordinate (i, j) corresponds to a line in $AG(2, q)$ passing through the point (i, j). Hence, we can map each agent to a line in $AG(2, q)$ and vice-versa. Hence, in order to find the agent that has a shared key with an agent of another region, one should find the line passing through the two points identical to the co-ordinate of the regions and then use Algorithm 8 to compute the common line. Once the common line is found the same can be mapped to the identifier of an agent. Each node in a region can maintain a sorted list of the lines along with the agent-id they correspond to. In order to get the identifier of an agent, a node can do a binary search for the line id and if found the corresponding agent id. can be obtained immediately. Since there are $q + 1$ many agents in a region, the time-complexity of this binary search is $O(\log q)$. Again all the steps of Algorithm 8 can be done in $O(1)$ time. Hence, the time required by a node to identify the agent containing a common key with an agent in another region is $O(\log q)$.

Now, if two nodes (say n^s_{ij} and $n^r_{i'j'}$) both belonging to two different regions R_{ij} and $R_{i'j'}$ wish to communicate securely, then node n^s_{ij} can use Algorithm 8 to determine the agent in the same region that is sharing a common key with an agent in region $R_{i'j'}$. Similarly, the node $n^r_{i'j'}$ can find the agent that has a common key with the said agent in region R_{ij}. Thus a secure link can be established between node n^s_{ij} and $n^r_{i'j'}$. A randomly chosen secret key can be passed to node $n^r_{i'j'}$ by node n^s_{ij} through this secure link. This key can thereafter be used in all subsequent communication between the two nodes.

5.3.5 Study of Resiliency

Wireless sensor nodes are sometimes deployed in hostile environment where they are exposed to the risk of physical capture. An adversary may capture some nodes of a sensor network. She will then be able to extract all the information stored inside the sensor nodes. In all key predistribution schemes cryptographic keys are stored in the memory of the sensors and hence an adversary can get to know the keys in a sensor node through examining the memory of a captured node. Since in conventional key predistribution schemes one key is stored in multiple nodes, compromising one node can lead to exposure of some keys in some uncompromised nodes. Thus, it may be possible to expose all the keys present in a node without actually capturing the node by compromising a sufficiently large number of nodes that share common key(s) with the node. Here, we study the performance of the proposed key predistribution scheme in terms of standard measures. The known measures of performance of a key predistribution scheme attempt to measure

the resiliency of any key predistribution scheme against node capture attack. The well known measures of resiliency of any key predistribution scheme are called $E(s)$ and $V(s)$.

$E(s)$ is defined as the fraction of links disconnected when s many nodes are compromised.

5.3.6 Study of intra-region resiliency

Our key predistribution scheme is primarily aimed at establishing inter group connectivity through agents. But we have proposed to apply the Çamptepe & Yener scheme in [14] for key predistribution within each of the q^2 many regions. Hence, we should discuss the resiliency offered by the intra-region key predistribution scheme or the key predistribution scheme of Çamptepe & Yener. Now, let N_{ij} be the number of nodes in any region which includes the common nodes and the $q+1$ agents installed in every region. For the sake of simplicity, let us assume that all the regions contain equal number of nodes, i.e. $N_{ij} = N, \forall i,j \in \{0,1,2,\ldots,q-1\}$. Also let p be the least prime number satisfying $p^2+p+1 \geq N$. Thus every node in a region will contain $p+1$ many secret keys in accordance with the scheme of Çamptepe & Yener [14]. Moreover, every pair of nodes do contain one and only one common key. Under this scheme every pair of nodes do share a common key. Hence, when no node is compromised by the adversary then every node is connected to every other node. Therefore, the number of links in the region is $N(N-1)/2$. Let k_i be the number of distinct keys exposed when s nodes are compromised. Now, if $s(s>1)$ many nodes are compromised, then the number of keys that get exposed is $(p+1)s$, but all of them are not distinct. This is due to the fact that every pair of nodes share a unique common key. Hence, if two nodes are compromised then the number of distinct keys exposed is $2(p+1)-1 = 2p+1$. Hence, if $s(s>1)$ nodes are captured, the number of distinct keys exposed is given by $k_i < (p+1)s$. Now, if one key is exposed, then $p(p+1)/2$ many links are exposed. So, for k_i keys the number of links exposed is given by $L_s = k_i p(p+1)/2 < sp(p+1)^2/2$. Now, in the calculation of $E(s)$, we need to take into account only the exposed links between pairs of uncompromised nodes. Since s nodes are compromised and each of them were having $N-1$ many key-links with $N-1$ many other nodes in the same region, the total number of links are thus equal to $(N-1)s$. Among these $(N-1)s$ many links, we have twice counted the $\binom{s}{2}$ many links between $\binom{s}{2}$ pairs of compromised nodes. Hence, the total number of distinct links between the compromised nodes and any other node in the same region is $(N-1)s - s(s-1)/2$. This number should be subtracted from L_s to get the actual number of exposed links in the set of uncompromised nodes. Hence, the number of exposed links among the set of uncompromised nodes is $L_s - (N-1)s + s(s-1)/2 < sp(p+1)^2/2 - (N-1)s + s(s-1)/2$. Hence, the fraction

of links exposed is given by

$$E(s) < \frac{sp(p+1)^2/2 - (N-1)s + s(s-1)/2}{N(N-1)/2}$$

Lemma 5.13. $E(s) < \frac{s(p+1)}{p^2+p+1}$

Proof. $(N-1)s > s(s-1)/2$. Hence, $E(s) < \frac{sp(p+1)^2/2}{N(N-1)/2}$. In Çamptepe & Yener scheme , $N \geq p^2+p+1$ [14]. So, $E(s) < \frac{sp(p+1)^2/2}{(p^2+p+1)(p^2+p)/2}$. Hence, the result. □

5.3.7 Study of resiliency of the inter-region links

Our proposed key predistribution scheme builds the connectivity between the q^2 regions via the agents. Here we shall be analysing the effects of compromise of the agents on the inter-region connectivity. For this purpose, we use another measure $E'(s)$ presented in definition 4.8.

In definition 4.8, we used $E'(s)$ to denote the fraction of interlinks disconnected when s many agents are compromised. An interlink exists between two regions R_{ij} and $R_{i'j'}$ if and only if there are agents $a_{ij}^k, a_{i'j'}^{k'}$ where $0 \leq k, k' \leq q$ such that a_{ij}^k and $a_{i'j'}^{k'}$ do share a common key. Lemma 5.8 ensures existence of an interlink between every pair of agents. Also, Lemma 5.10 proves that such a pair of agents will share a pairwise key. Hence, an interlink can get broken only if at least one of the two agents get captured in the hand of the adversary.

Theorem 5.14. $E'(s) \leq \frac{2s}{q^2(q+1)}$

Proof. If the adversary wants to compromise maximum number of interlinks, he would try to select a set agents such that no two agents in the set does have a common key, thus increasing the number of keys in his hand. Let τ_{ij}^k be the set of keys contained in agent $a_{ij}^k, 0 \leq i, j \leq q-1$. From lemma 5.9 it is evident that $|\tau_{ij}^k| = q-1$. Let, $\Psi = \{a_{i'j'}^{k'} : 0 \leq i', j' \leq q-1, 0 \leq k' \leq q, (i,j) \neq (i',j'), \tau_{ij}^k \bigcap \tau_{i'j'}^{k'} \neq \phi\}$. Then ϕ is the set of agents sharing a common pairwise key with agent a_{ij}^k. Corollary 5.12 implies that $|\Psi| = q-1$. So, if the agent a_{ij}^k gets compromised then it would lead to the exposure of the pairwise interlinks between region R_{ij} with all region $R_{i'j'}$ such that $a_{i'j'}^{k'} \in \Psi$ for some $k' \in \{0, 1, \ldots, q\}$. Hence, $q-1$ number of interlinks will get exposed. Compromise of a single agent will lead to the exposure of $q-1$ interlinks. In the worst case all the agents captured by the adversary may have no keys in common. Hence, if s many agents are compromised $(q-1)s$ many links will get exposed. Now, in the proposed scheme the

total number of interlinks is $\binom{q^2}{2}$. Hence, the fraction of links exposed in the worst case is $\frac{2s}{q^2(q+1)}$.

□

Figure 5.1 shows the variation of experimental values of $E'(s)$ with the theoretic bound of $E'(s)$ as proven in theorem 5.14. It is evident from figure 5.1 that as the number of compromised agents increase, the rate of increase of the experimental value of $E'(s)$ tends to reduce with respect to the rate of increase of the theoretic upper bound. The values of different parameters used in figure 5.1 is shown in table 5.1.

Size of the grid	Size of the Lee Square	No. of agents compromised	Fraction of interlinks broken (Exp)	Fraction of interlinks broken(Theoretic bound)
37×37	32×32	10	0.000386	0.000384
37×37	32×32	1610	0.060966	0.061897
37×37	32×32	3210	0.119663	0.123409
37×37	32×32	4810	0.176398	0.184922
37×37	32×32	6410	0.231186	0.246434
37×37	32×32	8010	0.284220	0.307947
37×37	32×32	9610	0.335408	0.369459
37×37	32×32	11210	0.384561	0.430972
37×37	32×32	12810	0.431888	0.492484
37×37	32×32	14410	0.477329	0.553996

TABLE 5.1: Table of values of different parameters used for comparison in Figure 5.1.

Let s_{ij} be the number of nodes belonging to region R_{ij} that are captured by the adversary. We had previously assumed that n_{ij} is the total number of nodes in R_{ij}. The probability that x many agents will be compromised in R_{ij} when s_{ij} many nodes are captured by the adversary is $\frac{\binom{q+1}{x}\binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}}$. Then the expected number of agents compromised in R_{ij}, when s_{ij} many nodes are compromised is given by, $EXP(AG_{ij}) = \sum_{x=0}^{q+1} x\frac{\binom{q+1}{x}\binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}}$. Hence, $EXP(AG_{ij}) = \frac{\sum_{x=0}^{q+1} x\binom{q+1}{x}\binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}} = \frac{\sum_{x=1}^{q+1} x\binom{q+1}{x}\binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}}$. Hence we get,

$$EXP(AG_{ij}) = \frac{(q+1)\sum_{x=1}^{q+1}\binom{q}{x-1}\binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}} \tag{5.1}$$

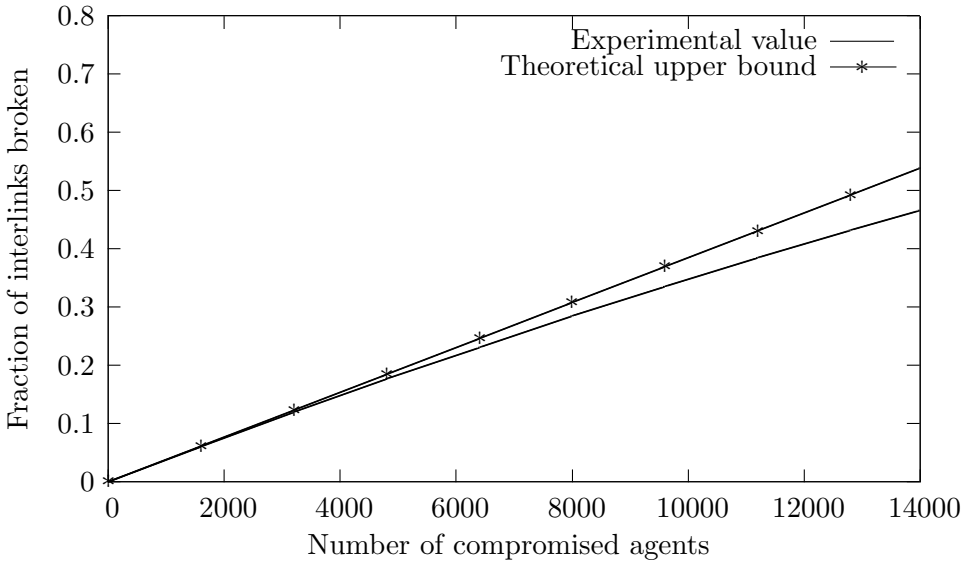

FIGURE 5.1: Graphical presentation of experimental values and theoretical upper bound of $E'(s)$ when the grid size is 37×37 and the size of Lee square is 32×32.

Expected number of agents captured in the whole grid is therefore given by

$$EXP(AG) = \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} E(AG_{ij})$$

So,

$$EXP(AG) = \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} \frac{(q+1) \sum_{x=1}^{q+1} \binom{q}{x-1} \binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}}$$

Hence, using Theorem 5.14,

$$E'(EXP(AG)) \leq \frac{2}{q^2} \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} \frac{\sum_{x=1}^{q+1} \binom{q}{x-1} \binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}} \tag{5.2}$$

Lemma 5.15. *Consider a deployment zone where every region contains n many nodes including the agents. If s nodes are randomly captured from each region, then the value of $E'(EXP(AG)) \leq \frac{2}{\binom{n}{s}} \sum_{x=1}^{q+1} \binom{q}{x-1} \binom{n-q-1}{s-x}$*

Proof. From 5.2, we can write $E'(EXP(AG)) \leq \frac{2}{q^2} \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} \frac{\sum_{x=1}^{q+1} \binom{q}{x-1} \binom{n_{ij}-q-1}{s_{ij}-x}}{\binom{n_{ij}}{s_{ij}}}$. Now, replacing n_{ij} by n and s_{ij} by s, we get $E'(EXP(AG)) \leq \frac{2}{q^2} \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} \frac{\sum_{x=1}^{q+1} \binom{q}{x-1} \binom{n-q-1}{s-x}}{\binom{n}{s}} = 2 \frac{\sum_{x=1}^{q+1} \binom{q}{x-1} \binom{n-q-1}{s-x}}{\binom{n}{s}}$. □

We have performed simulation for computing the values of $E'(EXP(AG))$ for different values of the parameters. We chose identical values of n_{ij} and s_{ij} for all regions. That

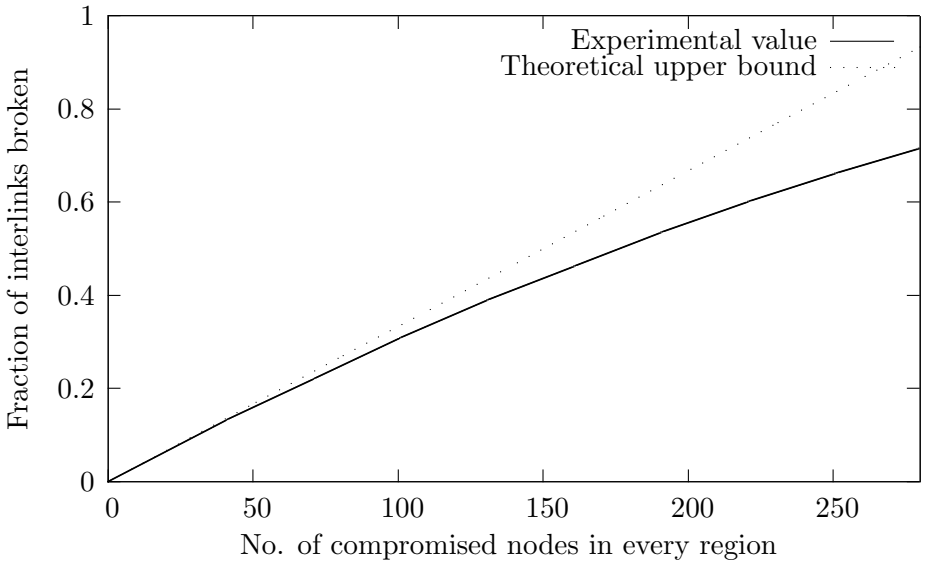

FIGURE 5.2: Graphical representation of experimental values and theoretical upper bound of $E'(s)$ when the grid size is 31×31 and the size of Lee square is 31×31.

means in our simulation, we selected $s_{ij} = s, n_{ij} = n, \forall i, j \in \{0, 1, 2, \ldots, p-1\}$. Figure 5.2 provides a graphical representation of the experimental values of $E'(EXP(AG))$ and the theoretical upper bound of the same for different values of s. The values of different parameters used in figure 5.2 can be found in table 5.2.

Size of the grid	Size of the Lee Square	No. of nodes in each region	No. of agents compromised	Fraction of interlinks broken (Exp)	Fraction of interlinks broken(Theoretic bound)
31×31	31×31	600	0	0.000000	0.000000
31×31	31×31	600	51	0.169999	0.163271
31×31	31×31	600	81	0.269999	0.251926
31×31	31×31	600	111	0.370000	0.335219
31×31	31×31	600	141	0.470000	0.414384
31×31	31×31	600	171	0.570000	0.488289
31×31	31×31	600	201	0.670000	0.557540
31×31	31×31	600	231	0.769999	0.622413
31×31	31×31	600	261	0.869999	0.681452
31×31	31×31	600	291	0.970000	0.736301

TABLE 5.2: Table of values of different parameters used for comparison in Figure 5.2.

When a node gets compromised, the adversary gets to know all the keys loaded into a sensor node. Now, in combinatorial strategy of key predistribution, a single key may be

shared by multiple nodes and in that case the same key will act as the shared key between many node-pairs. If such a key gets compromised then the compromise would break the key link between pairs of nodes that use the same key as their common key. If there is no other key link between those nodes then the nodes will get completely disconnected from each other. Hence, less the value of the fraction of links exposed, better will be the key predistribution scheme. In other words a key predistribution scheme can be deemed to exhibit better performance if the value of $E(s)$ is less for different values of s i.e. the number of compromised keys. In our scheme any key K_{ij} is present in any two agents as shown in lemma 5.3. So, one key acts as a common key between a unique pair of nodes. Hence, when one node gets captured by the adversary resulting in the exposure of all the keys stored in it, the key-links between other node-pairs remain unaffected.

Now, we define another measure for evaluating the resiliency of our key predistribution scheme. This measure is motivated by another measure of similar kind which was proposed by Ruj and Roy in [60]. This measure is called $V(s)$. $V(s)$ is defined as the fraction of nodes that get disconnected from the rest of the network when s many nodes are compromised by the adversary leading to the exposure of all keys stored in them. This measure was proposed by Ruj & Roy in [60]. In this chapter, we use a modified measure $V'(s)$ presented in definition 4.9 that takes only the agents into account.

In definition 4.9, We used $V'(s)$ to denote the fraction of regions that get disconnected from the rest of the regions on an average when s many agents are compromised by the adversary. These s many compromised agents may come from the set of $q^2(q+1)$ agents deployed in q^2 regions. One region gets disconnected from the rest of the regions only if all the $q+1$ many agents in that region get compromised or all the keys stored in them get exposed. The adversary can suitably choose a set of agents for capturing in order to maximise the number of disconnected regions provided she has full knowledge of the key predistribution strategy.

From lemma 5.8 and lemma 5.9, we know that an agent contains $q-1$ distinct keys and it shares every key with an unique agent of a distinct region. In order to disconnect one region from the rest of the regions, the easiest way would be to capture all the $q+1$ many agents present in the region. Hence, lemma 5.16 holds.

Lemma 5.16. *$V'(s) = 0$ if $0 \leq s \leq q$.*

Theorem 5.17. *$V'(s) \leq \frac{s}{q^2(q+1)}$ where $s \leq q^2 - 1$*

Proof. As we have discussed in previous section, the easiest way to disconnect a region from other regions is to capture all the $q+1$ many agents in that region. Hence, the best case for the adversary will be to keep capturing the agents of a region in order

to disconnect that region. If s many agents are compromised, the number of regions that can be disconnected is $\lfloor \frac{s}{q+1} \rfloor$. Hence, $V'(s) \leq \frac{\lfloor \frac{s}{q+1} \rfloor}{q^2} \leq \frac{s}{q^2(q+1)}$. Upon careful observation, it can be seen that this bound does not hold for all values of s. Say, for example an agent $a^k_{i,j}$ of region (i,j) shares all its $q-1$ keys with agents in the set $\{a^{k_1}_{i_1,j_1}, a^{k_2}_{i_2,j_2}, \ldots, a^{k_{q-1}}_{i_{q-1},j_{q-1}}\}$. Now, if an adversary compromises all agents in regions $R_{i_1,j_1}, R_{i_2,j_2}, \ldots, R_{i_{q-1},j_{q-1}}$ leading to complete disconnection of these regions, then this would compromise all keys present in $a^k_{i,j}$. Hence, if the adversary decides to disconnect region $R_{i,j}$, she would need to capture all agents in R_{ij} except agent $a^k_{i,j}$. Hence, it would be sufficient to compromise only q many agents to disconnect $R_{i,j}$ from other regions. Hence, this bound of $V'(s)$ holds only when the number of disconnected regions is less than $q-1$. Since the number of disconnected regions is $\lfloor \frac{s}{q+1} \rfloor$, the bound will hold for any $s \leq q^2 - 1$. □

We have conducted experiments on the performance of our key predistribution using $V'(s)$ as a measure. The experiment was done by means of simulation of the key predistribution scheme. Figure 5.3 shows a pictorial representation of the performance of our key predistribution scheme in terms of $V'(s)$. The values of different parameters used in plotting the curve of figure 5.3 can be found in table 5.3.

Size of the grid	Size of the Lee Square	No. of agents in each region	No. of agents compromised	Fraction of regions disconnected (Exp)
29×29	19×19	30	15000	0.000000
29×29	19×19	30	22000	0.017994
29×29	19×19	30	22323	0.030519
29×29	19×19	30	22646	0.042925
29×29	19×19	30	22969	0.074316
29×29	19×19	30	23292	0.119025
29×29	19×19	30	23615	0.187277
29×29	19×19	30	23938	0.299009
29×29	19×19	30	24261	0.461474
29×29	19×19	30	24584	0.676774
29×29	19×19	30	24907	0.890408

TABLE 5.3: Table of values of different parameters used in Figure 5.3

Define,

$$C(i,j)[k] = \begin{cases} 1 & \text{if agent } k \text{ in region } R_{ij} \text{ is not compromised} \\ 0 & \text{elsewhere} \end{cases}$$

Also Define, $\mathcal{C}(i,j) = |\{k : k \in \{0,1,\ldots,q\}, C(i,j)[k] = 0\}|$

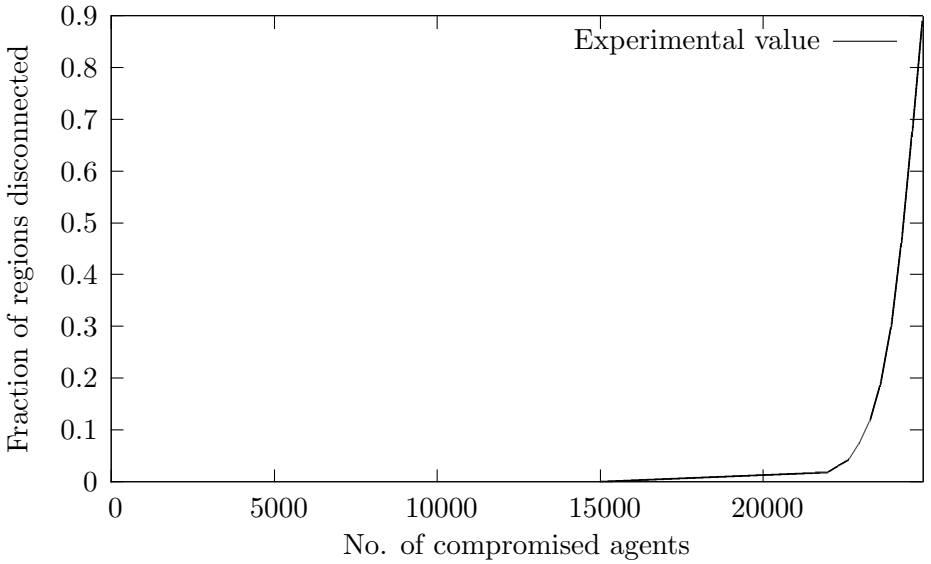

FIGURE 5.3: Graphical representation of experimental values of $V'(s)$ when the grid size is 29×29 and the size of Lee square is 19×19.

Theorem 5.18. *If $0 \leq g \leq q+1$ be the number of agents compromised in region $R_{i,j}$, then $P[D(i',j') = 0 | \mathcal{C}(i,j) = g, \forall i,j \in \{0,1,\ldots,q-1\}] = (\frac{g}{q+1})^{(q+1-g)(q-1)}$. In other words, when g number of agents are captured from all regions, the probability that any particular region gets disconnected from all other regions is given by $(\frac{g}{q+1})^{(q+1-g)(q-1)}$.*

Proof. The adversary needs to compromise all keys present in the $q+1$ agents of region R_{ij}. Since it has captured g many agents in R_{ij}, it requires to compromise other keys stored in the rest of $q+1-g$ many agents through capturing agents belonging to other regions who share common keys with the $q+1-g$ many agents in R_{ij}. Let $S = \{a^l_{i,j} : 0 \leq l \leq q\}$ be the set of agents that belong in region $R_{i,j}$. Now, the adversary may capture agents belonging in the set $U \subset S$, where $|U| = g$. There can be $\binom{q+1}{g}$ many such subsets. Now, for any U, if the adversary captures all the agents in U, then she would need to compromise the keys contained in the rest of the $q+1-g$ agents by capturing agents of other regions who share those keys with them. Each of the $q+1-g$ uncompromised agents has $q-1$ many pairwise keys that it shares with $q-1$ many agents belonging in $q-1$ many distinct regions. Hence, to compromise all the keys stored in an agent of $S \setminus U$ the adversary needs to capture $q-1$ many agents each from a separate region. Moreover, from theorem 5.7 we know that there is a unique pair of agents between two regions who share a common key. Therefore to compromise the keys in all the $q+1-g$ many uncompromised agents, the adversary will have to capture $(q+1-g)(q-1)$ many agents from $(q+1-g)(q-1)$ many regions that share common keys with the uncompromised agents in $R_{i,j}$. Since g number of agents are captured from each region, the probability of capturing a particular agent

of a region that shares a common key with an agent of $R_{i,j}$ is equal to $\frac{g}{q+1}$. Hence, the probability of compromising $(q+1-g)(q-1)$ many particular agents each from a distinct region is equal to $(\frac{g}{q+1})^{(q+1-g)(q-1)}$. Hence, for a particular composition of set U, the probability of compromising $(q+1-g)(q-1)$ many particular agents each from a distinct region is equal to $(\frac{g}{q+1})^{(q+1-g)(q-1)}$. There can be $\binom{q+1}{g}$ many different composition of of U with equal probability of occurrence. Hence, the required probability is $\sum_{x=1}^{\binom{q+1}{g}} \frac{1}{\binom{q+1}{g}} (\frac{g}{q+1})^{(q+1-g)(q-1)} = (\frac{g}{q+1})^{(q+1-g)(q-1)}$. □

5.3.7.1 Comparison with other schemes

We have compared our scheme with an existing scheme of similar kind. This scheme was proposed by Ruj & Roy [62]. Figure 5.4 gives a pictorial comparison of the relative performances of the two schemes in terms of $E'(s)$. The values of different parameters used in figure 5.4 can be found in table 5.4.

Figure 5.5 gives pictorial comparison of the relative performances of the two schemes in terms of $V'(s)$. The values of different parameters used in figure 5.5 can be found in table 5.5.

Size of the grid	Size of the Lee Square	No. of nodes in each region	Fraction of interlinks broken (Our Scheme)	Fraction of interlinks broken(Ruj-Roy Scheme)
37×37	16×16	0	0.000000	0.000000
37×37	16×16	1	0.000038	0.002059
37×37	16×16	36	0.001387	0.166546
37×37	16×16	71	0.002736	0.378596
37×37	16×16	106	0.004083	0.558819
37×37	16×16	141	0.005423	0.687052
37×37	16×16	176	0.006764	0.786011
37×37	16×16	211	0.008106	0.859707
37×37	16×16	246	0.009444	0.902046
37×37	16×16	281	0.010780	0.919285
37×37	16×16	316	0.012124	0.958004

TABLE 5.4: Table of values of different parameters used for comparison in Figure 5.4.

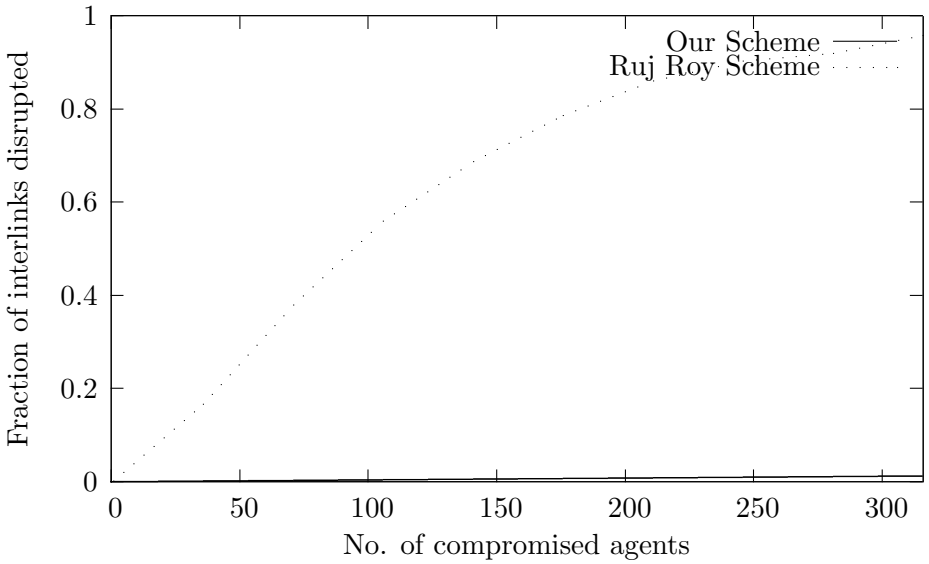

FIGURE 5.4: Graphical comparison between the performance in terms of $E'(s)$ of our scheme & the scheme of Ruj-Roy.

Size of the grid	Size of the Lee Square	No. of nodes in each region	Fraction of regions disconnected (Ruj-Roy)	Fraction of regions disconnected(Ours)
31×31	31×31	0	0.000000	0.000000
31×31	31×31	1	0.000000	0.000000
31×31	31×31	261	0.147867	0.000000
31×31	31×31	521	0.839126	0.000000
31×31	31×31	781	0.916025	0.000000
31×31	31×31	1041	0.941623	0.000000
31×31	31×31	1301	0.958897	0.000000
31×31	31×31	1561	0.969095	0.000000
31×31	31×31	1821	0.974714	0.000000
31×31	31×31	2081	0.983663	0.000000
31×31	31×31	2341	0.991051	0.000000

TABLE 5.5: Table of values of different parameters used for comparison in Figure 5.5.

5.4 Comparison with other schemes that use deployment knowledge

In this section we compare our proposed scheme for key predistribution using deployment knowledge with other existent schemes that use deployment knowledge. Some standard key predistribution schemes that use deployment knowledge are Huang, Mehta, Medhi and Harn, [30], Huang and Medhi [29], Zhou Ni Ravishankar [83], Liu Ning [42, 44],

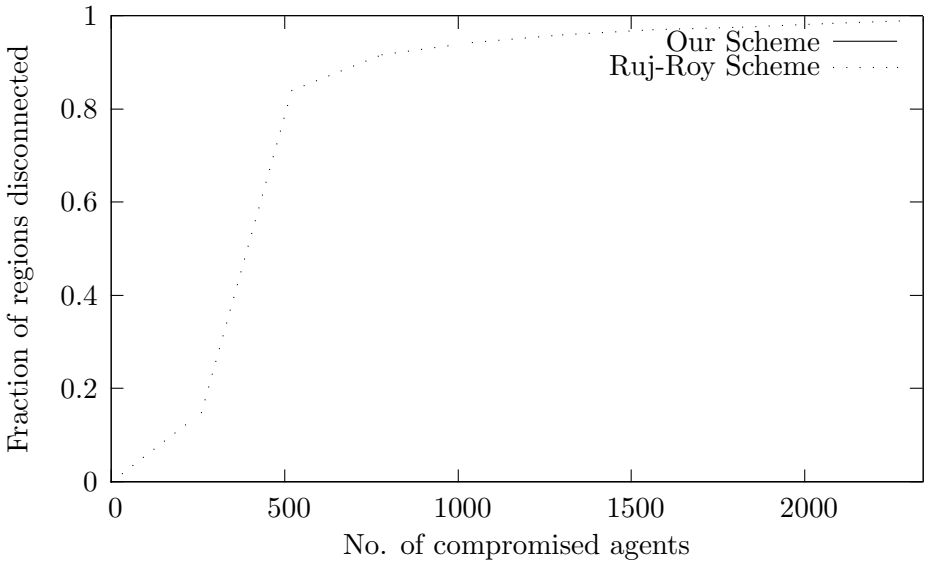

FIGURE 5.5: Graphical comparison between the performance in terms of $V'(s)$ of our scheme & the scheme of Ruj-Roy.

Younis, Ghumman and Eltoweissy [80], Du, Deng, Han and Varshney [26], Simonova, Ling and Wang [65], Ruj and Roy [62]. We have discussed these schemes in chapter 2. Here, we provide a brief recapitulation for the reader.

Huang, Mehta, Medhi and Harn [29, 30] used rectangular deployment zone which is divided into equal sized regions of smaller size. In this scheme the sensors randomly choose the keys. Huang *et al.* used multi-space Blom scheme [10] for key predistribution. In this scheme all nodes are identical with respect to the amount of resources they possess. This is where this scheme is different from ours. In our scheme there are two different types of nodes viz. common nodes and agents giving rise to a heterogeneous network. Moreover in Huang *et al.* scheme the nodes in a region can communicate directly with each other with probability > 0.5 whereas in our scheme they can do so with a probability equal to 1 as our scheme ensures full inter-region connectivity. Hence, in this scheme more amount of computation will be required for communication than our scheme. The scheme of Huang *et al.* is perfectly secure against selective and random node capture attack. Hence, capture of some number of nodes by an adversary will have negligible effect to the links among the uncompromised nodes. But if we take all the links of compromised and uncompromised nodes into account then the fraction of links compromised will be higher.

Zhou, Ni and Ravishankar [83] used two types of sensor nodes viz. static and mobile. This scheme uses pairwise keys with each sensor within the same region. Hence, it requires high amount of memory to hold the pairwise keys if the number of sensors within a region is high enough. If there are n number of nodes within a region, then the

number of keys to be stored in a node is $O(n^2)$ under the Zhou *et al.* scheme whereas it is $O(\sqrt{n})$ in Çamptepe and Yener scheme which is used in our key predistribution scheme. Hence, our scheme is much better than Zhou *et al.* in terms of memory efficiency.

Liu and Ning [42, 44] and Blackburn *et al.* [9] used deployment knowledge. There, the whole deployment zone is split into smaller square regions like our scheme. But in their schemes only a single node is deployed in a square region as opposed to our scheme where there are a group of nodes deployed in a region.

Simonova *et al.* [65] proposed a key predistribution scheme that uses deployment knowledge. The resiliency in this is much lower than our scheme.

Du, Deng, Hang, Varshney [26] proposed another key predistribution using deployment knowledge that uses multiple space Blom scheme [10]. Under this scheme sensors randomly choose keys from a set of different instances of Blom space. Unlike our scheme this scheme does not guaranty full connectivity.

As we have discussed earlier, the key predistribution scheme of Ruj and Roy in [62] uses deployment knowledge. Similar to our scheme, this scheme uses the Çamptepete and Yener scheme for key predistribution within the same region. This scheme exhibits lower resiliency among the set of agents that provide inter-region connectivity as discussed in section 5.3.7. In other words our scheme offers more resilient inter-region connectivity than Ruj and Roy scheme.

Figure 5.6 gives a pictorial comparison of the relative performances of the different key predistribution schemes that use deployment knowledge including our scheme. Here it can be visualised that our key predistribution scheme offers better performances than many schemes of its kind. It can be seen that only Zhou *et al.* [83] and Liu-Ning [42, 44] offers better performance than our scheme. But as we have stated earlier that Zhou *et al.* uses pairwise keys for node-connectivity within a region. Hence, it requires very high amount of memory inbuilt into the sensor nodes making the scheme very costly for usage in sensor networks. Liu and Ning scheme uses only one node at any region in contrast to our scheme where a group of sensor nodes are deployed in any region. Therefore in such deployments where a group of nodes are needed to be deployed at any region, the scheme of Liu and Ning is not applicable. Hence, it can be precisely stated that our scheme offers best performance among all the key predistribution of similar kind.

Table 5.6 provides a comparative study of communication, storage and scalability of several key predistribution schemes that use deployment knowledge.

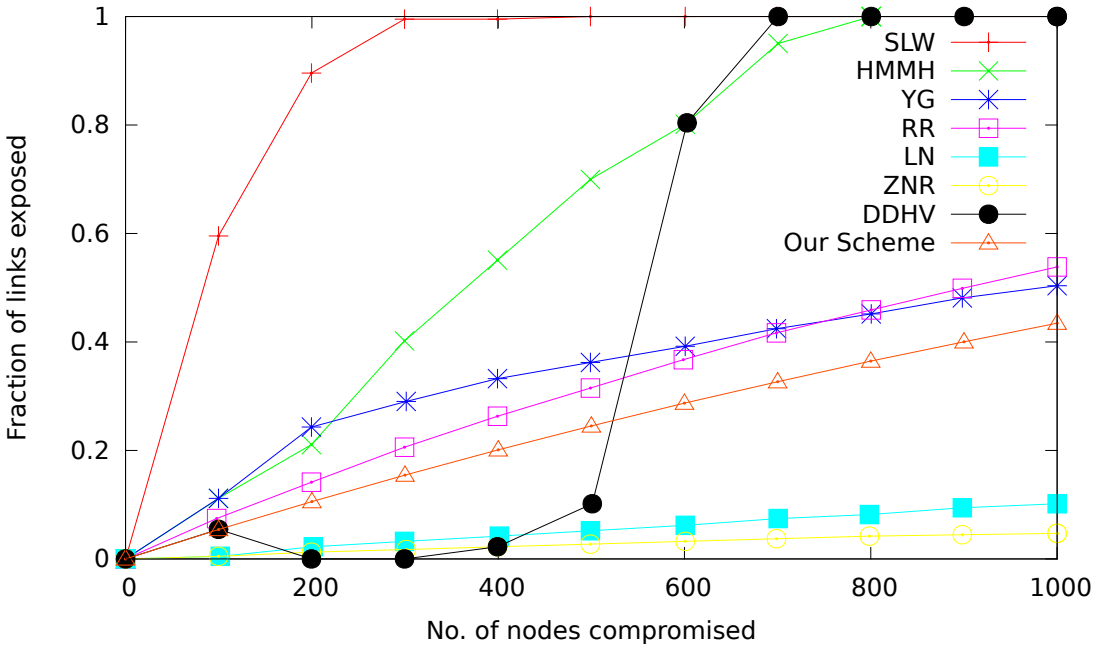

FIGURE 5.6: Comparison of Du *et al.* [26] (DDHV), Liu-Ning [44] (LN), Yu-Guan [81] (YG), Zhou *et al.* [83] (ZNR), Huang *et al.* [30] HMMH, Simonova *et al.* (SLW),Ruj-Roy [62] (RR) and our scheme. (i)DDHV scheme has parameters $k = 200$, $\omega = 11$ and $\tau = 2$, (ii)LN scheme has parameters $k = 200$, $m = 60$ and $L = 1$, (iii)YG scheme has parameters $k = 100$, (iv)ZNR scheme has parameters $k = 100$, (v)HMMH scheme has parameters $k = 200$, $\omega = 27$ and $\tau = 3$ (vi)SLW scheme has parameters $k = 16$, $p = 11$ and $m = 4$ (vii)Ruj-Roy scheme has parameters $k = 12$. (viii) Our scheme has parameter $q = 13$. The size of the network in DDHV, LN, YG, ZNR, HMMH is 10000, for SLW it is 12100, 16093 for Ruj Roy scheme and 16055 for our scheme.

5.5 Concluding Remarks

In this chapter we have proposed a scheme for key predistribution for a grid-group deployment of Wireless Sensor Network. We assumed that the deployment zone is a square of size $q \times q$ where q is a prime. The entire deployment zone is further divided into smaller square regions giving rise to a grid like structure. There are groups of nodes deployed in all the smaller regions. Our principal objective was to devise a scheme to establish the "inter-region" secure key-links. As we have mentioned earlier, our scheme uses an existing key predistribution scheme i.e. the Çamptepe and Yener scheme for establishing "intra-region" key links. Ruj and Roy [62] also applied the same key predistribution scheme for establishing the secure key-links among the sensor nodes within a region of the deployment zone. Thus our scheme is applicable for the same deployment as in the Ruj and Roy scheme. We have discussed in section 5.3.7 that our scheme is more resilient against node capture than the Ruj and Roy scheme in terms of the well known measures of resiliency. Also, the storage overhead of our scheme in the agents are of the same order. Hence, our scheme improves the state of the art without increasing the cost. In our scheme, any two nodes within a region can communicate directly as

Schemes	Communication cost	Storage	Scalability
DDHV [24, 26]	$O(\tau)$	$\tau(\lambda+1)$	Scalable
LN [42, 44]	$O(\log C \log R)$	$(t+1)q$	Not scalable
YG [81, 82]	$\lambda(\ \log g)$	$(\lambda+1)\omega$	Not scalable
ZNR [83]	$O(\log N)$	$O(\gamma)^1$ $O(n_s)^2$	Not scalable
HMMH [30]	$O(\tau)$	$\tau(\lambda+1)$	Scalable
HM [29]	$O(\tau)$	$\tau(\lambda+1)$	Scalable
PIKE [18]	$O(\log \sqrt{N})$	$O(\sqrt{N})$	Not Scalable
SLW [65]-1	$O(\log p')$	$O(\sqrt{N/g})$	Scalable
SLW [65]-2	$O(\log p')$	$O(\sqrt{N/g})$	Scalable
RR [62]	$O(\log p)^1$ $O(\log q')^2$	$O(p)^1$ $O(q')^2$	Not Scalable
Ours	$O(\log n)^1$ $O(\log q')^2$	$O(\sqrt{n})^1$ $O(q')^2$	Not Scalable

TABLE 5.6: Comparison of the different key predistribution schemes with respect to communication cost, storage overhead and scalability. Here τ denotes the number of key spaces selected out of ω spaces of Blom's scheme in DDHV scheme, λ denotes the security parameter for Blom scheme, ω denotes the number of key spaces of Blom's scheme as in YG scheme. t denotes the degree of symmetric bivariate polynomial whose coefficients are in F_q used in LN scheme. $C \times R$ is the area of the region for LN scheme. g is the number of groups in YG and SLW scheme, γ is the number of nodes in each group in ZNR scheme. n_s is the total number of sensors in the same scheme. N is the total number of sensors. p and p' are parameters in RR scheme and that of SLW schemes respectively. $q' \times q'$ is the size of the deployment grid in both our scheme and the RR scheme. ^{1} is the storage for small sensor nodes and ^{2} is the storage for agents.

they do share a common key. When a pair of nodes each from two separate regions wish to communicate they need to do so with the help of a pair of agents belonging to their own regions and sharing a common secret pairwise key. Our scheme ensures existence of a pair of agents sharing a common key in any two regions. We have analysed the performance of our key predistribution scheme with some standard key predistribution schemes in literature that use deployment knowledge where the basis of measurement of resiliency is the fraction of links broken. In addition, we have also compared schemes with respect to their associated costs like communication cost or storage overhead. We have shown that our key predistribution scheme is better than those scheme with respect to some measure or the other.

In future more research can be done to improve the resiliency of the inter-region connectivity. Also, initiative could be taken to use our scheme in conjunction with some other scheme than the Çamptepe and Yener scheme in order to improve the resiliency of the network against node capture.

Chapter 6

Two Channel Hopping Schemes for Jamming Resistant Wireless Communication

Jamming resistance is crucial for reliable wireless communication. Most of the existing schemes offering countermeasures of jamming depend on the use of a secret key shared between the communicating devices. This secret key is used as a seed to generate a random hopping sequence. The message–sender and the message–receiver hop over different wireless channels depending upon this generated sequence. This creates a cyclic dependency between the jamming resistant key establishment and jamming resistant communication. To break this dependency, Strasser et al. proposed a jamming resistant key establishment mechanism in [68] that uses Uncoordinated Frequency Hopping. But this scheme has a major disadvantage that under this scheme a sender and a receiver need to hop randomly over a number of channels and they can only communicate a message only if by chance they meet over the same channel at any instant. This limitation makes communication under UFH very slow.

This chapter is based on the research work discussed in [5]. In this chapter, we discuss a new countermeasure against jamming. This chapter discusses two schemes based on combinatorial designs that allow a pair of devices to communicate in presence of an active jammer. In our anti-jamming communication strategy a pair of communicating devices (sender and receiver) hop over multiple communication channels based on a design theoretic approach in order to evade the jammer. We used combinatorial designs for developing these channel hopping schemes. Our schemes ensure that the jammer does not gain anything by knowing the channel hopping algorithm. There is no secret key associated with the scheme. Our schemes beat the scheme in [68] in the sense that

in our schemes the sender and the receiver do not require to hop unboundedly waiting for a rendezvous. Our schemes guaranty that a pair of communicating devices must meet every time on a certain channel for communication. Hence, this scheme can be used for key establishment in the presence of jammer as an alternative to [68]. Also, this scheme can be used for exchanging data between the pair of nodes without bothering about establishing a secret key similar to [67].

6.1 System Model and Adversary Model

6.1.1 System Model

Here we consider unicast communication scenario. In unicast communication there is a pair of nodes. One of them is the sender(A) and the other is a receiver(B). The sender transmits messages over the wireless medium. B is the intended recipient of the message. Both the nodes have multiple transducers that allow them to transmit messages over more than one channels as well as to listen to multiple channels. Suppose the nodes wish to communicate at some point of time. Let C be the set of wireless channels available for communication. Let node A be sending the same message over the set of channels in C_S ($C_S \subseteq C$). Also, let node B be scanning all the channels in set C_R ($C_R \subseteq C$). So, the message can be successfully communicated to B only if $C_S \bigcap C_R \neq \emptyset$, i.e there is at least one channel on which node A and B rendezvous. But if all the common channels are jammed by a malicious adversary then obviously no communication will take place. We assume that the message to be communicated is large enough. In order to send it, we must first split the message into a number of fragments of smaller size. Then the fragments are sent one by one from the sender to the receiver. In this study we assume that $|C_S| = |C_R|$, i.e. the number of channels over which a particular sender can transmit message at the same time is same as the number of channels to which the receiver can listen simultaneously. The amount of time needed to transmit a message-fragment is called a time slot. The timeline is divided into smaller time slots that occur consecutively. End of one time slot follows commencement of the next one and so on. We also assume that length of each time slot is constant and all the devices are time-synchronized. We further assume that the information about the start and the end of time slots are publicly known.

6.1.2 Attacker Model

The attacker can jam a finite number of channels at any instant. It does this by putting a strong signal on these channels. The strong signal distorts any message currently being

carried by the channel. The attacker has knowledge of the set of channels C used by the sender and the receivers. It knows the starting and ending time of a time slot. At the onset of each time slot t the attacker chooses a set of channels $C_t^J \subset C$ and jams every channel in C_t^J for that time slot t. Then in next time slot the jammer again chooses some channels for jamming. Here we ssume that $|C_t^J| = J$ for all time slot t where J is the jamming strength of the attacker. The attacker is assumed to be computationally unbounded. The attacker's aim is to disrupt the communication as much as possible by attempting to damage more message-fragments during transmission. The attacker chooses its jamming strategy depending upon its jamming strength and the information that it possesses about the system.

6.1.3 Message Passing Mechanism

In the presence of an active jammer, transmission of messages from sender to receiver may not always be successful. Because depending upon the jammer's strength, it has a non-zero probability of successfully blocking a message fragment that is being communicated over a certain channel. This probability depends upon the jammer's strength and the communication mechanism. Therefore, additional measures are necessary to ensure that a message is properly communicated to it's intended recipient. In order to ensure this, a message is fragmented into a number of fragments using some coding technique and the message fragments are sent to the receiver. It may so happen that some of the fragments are lost because of the jammer. So we need to send redundant packets to ensure that the receiver is successfully able to reconstruct the original message from the packets it receives. One such coding technique is erasure code. Message sending using erasure coding technique in similar scenario has been explored in [47], [57]. With this coding technique it is possible to fragment a message to infinitely many fragments such that the message can be reconstructed if at least l such fragments can be received for some fixed value of l. The sender can send its messages using this technique by fragmenting them and sending the fragments one by one to the receiver whenever they meet on the same channel. The receiver keeps collecting these message fragments until l distinct fragments are collected after which it can immediately reconstruct the message. If p_m be the probability that a message sent by the sender will be successfully received by the receiver, then under rateless erasure coding technique, the expected number of transmissions required is given by $N(p_m) = \sum_{i=0}^{\infty} \binom{i-1}{l-1} (p_m)^l (1-p_m)^{i-l} i = \frac{l}{p_m}$ [67]. On the other hand if the number of fragments of the message is finite and smaller than $\frac{l}{p_m}$, then the sender can repeatedly send all the fragments in a sequence. It can be shown that in this case $N(p_m) \approx \frac{\log(n-l)-\log n}{\log(1-p_m)} n \in O(\log \frac{n}{n-l} n)$ [67].

6.2 The Scheme

6.2.1 General Idea for using set system for channel hopping

Let, $(X, \mathcal{A})$ be a set system. Let A and B are two communicating devices. Let A be the sender and B be the receiver. The total number of wireless channels available to them is $C = |X|$. We identify each of the channels by an element from the set X. Let, $\mathcal{A} = \{B_1, B_2, \ldots, B_b\}$. B_i's are the blocks of the set system for $i \in \{1, 2, \ldots, b\}$. We also assume that $\forall i \in \{1, 2, \ldots, b\}, |B_i| = k$. Now sender's channel hopping scheme is the following :

1. In every time slot t, the sender randomly chooses a block B_i from $\mathcal{A}$.
2. For each $x \in B_i$, the sender transmits the message fragment through all the channels having identifier x in time slot t.

Similarly, the channel hopping scheme of the receiver is the following :

1. In every time slot t, the receiver randomly chooses a block B_j from $\mathcal{A}$.
2. For each $y \in B_j$, the receiver listens to the channel having identifier y for incoming message fragment in time slot t.

It is easy to see that the necessary condition for successful message transmission from sender to the receiver in absence of the jammer is :

$\forall B_1, B_2 \in \mathcal{A}, B_1 \neq B_2, |B_1 \bigcap B_2| \geq 1$

If the above condition holds then the above channel hopping scheme ensures rendezvous of the sender and the receiver on each time slot. In other words, both the sender and the receiver will meet on some channel on all time slots. Note that the sender and the receiver may not know which block the other one is selecting for the current time slot. Despite this, they end up meeting on some channel. So, we impose another constraint on the set system $(X, \mathcal{A})$ which is $\forall B_1, B_2 \in \mathcal{A}, B_1 \neq B_2, |B_1 \bigcap B_2| = 1$.

Now, what if the jammer comes into the picture? We assume that the jammer can jam J channels simultaneously.

For the time being let us consider that the jammer's strategy is to choose J random channels out of the C channels. Let $\mathcal{J}^t$ denotes the set of channels jammed by the attacker in a time slot t. Also, let $\mathcal{S}^t$ and $\mathcal{R}^t$ be the blocks chosen by the sender and

receiver respectively in time slot t. Then the probability of jamming is given by $P(JAM^t) = P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t) = P(\mathcal{J}^t \supseteq \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) * P(\mathcal{S}^t = \mathcal{R}^t) + P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) * P(\mathcal{S}^t \neq \mathcal{R}^t)$ Since, there are b blocks in $\mathcal{A}$, $P(\mathcal{S}^t = \mathcal{R}^t) = \frac{1}{b}$. Since $|\mathcal{J}^t| = J$ and $|\mathcal{S}^t| = k$, $P(\mathcal{J}^t \supseteq \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) = \frac{\binom{C-k}{J-k}}{\binom{C}{J}}$. Now if $\mathcal{S}^t \neq \mathcal{R}^t$, then $\mathcal{S}^t \bigcap \mathcal{R}^t = \{x\}, x \in X$. Hence, $P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) = P(x \in \mathcal{J}^t) = \frac{J}{C}$. Hence, $P(JAM^t) = \frac{1}{b}\frac{\binom{C-k}{J-k}}{\binom{C}{J}} + (1 - \frac{1}{b})\frac{J}{C}$. From this we get,

$$P(JAM^t) = \frac{J}{C}[1 - \frac{1}{b}(1 - \prod_{i=1}^{k-1} \frac{J-i}{C-i})] \tag{6.1}$$

Note that if $b >> 1$, then $P(JAM^t) \approx \frac{J}{C}$.

This is the general idea of our channel hopping strategy. Now we shall be exploring the applicability of two different types of combinatorial designs for proposed channel hopping scheme. We shall also investigate the performance of our channel hopping scheme for those two different combinatorial designs.

6.2.2 Channel Hopping scheme using Steiner Triple System

We have discussed Steiner Triple System in section 2.1. Here we apply this design in our channel hopping scheme. Let D be a $(v, 3, 1)$ STS design containing b blocks. Also let r be the replication number. Hence, $b = \frac{v(v-1)}{6}, r = \frac{v-1}{2}$. Let D^* be the dual design of D. Hence, D^* contains b varieties and v blocks. Each block in D^* contains r varieties and each variety occurs exactly in 3 blocks. Also, any two blocks in D^* contain a single common variety which is a necessary condition for ensuring rendezvous of the sender and the receiver on every time slot.

Now, the communicating devices can communicate using the channel hopping strategy of section 6.2.1 and using D^* as the combinatorial design for hopping.

We clarify this with an illustration. Let us choose the $STS(9)$(Steiner Triple System of order 9) as D. Therefore, the dual design D^*, used for the channel hopping scheme will be this design :
$(X', \mathcal{A}')$, $X' = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$,
$\mathcal{A}' = \{B'_0, B'_1, \ldots, B'_8\}$. The blocks of $\mathcal{A}'$ are given by;

$$B'_0 = \{0, 3, 4, 11\}$$
$$B'_1 = \{0, 5, 6, 7\}$$
$$B'_2 = \{0, 8, 9, 10\}$$

$$B'_3 = \{1, 3, 5, 10\}$$
$$B'_4 = \{1, 4, 6, 8\}$$
$$B'_5 = \{1, 7, 9, 11\}$$
$$B'_6 = \{2, 4, 5, 9\}$$
$$B'_7 = \{2, 3, 7, 8\}$$
$$B'_8 = \{2, 6, 10, 11\}$$

Hence, the number of available channels should be equal to 12 with ids ranging from 0 to 11. Now, let us assume that this blocks are loaded into the memory of the communicating devices. Now let us assume that the sender has a message to be sent to the receiver. The sender fragments the message using the method described in section 6.1.3. The sender then starts transmitting the fragments to the receiver over the wireless channels. In any arbitrary time slot t, the sender chooses the block B'_3 and the receiver chooses the block B'_7 at random. So, the sender sends a message-fragment through the four channels with id $1, 3, 5, 10$. Similarly the receiver scans the channels with id $2, 3, 7, 8$ for incoming message-fragment. It is easy to see that they meet over channel 3. That is the receiver will read the message-fragment on channel 3 provided the channel is not jammed. It can be seen in the above example that any pair of blocks in the above example do share a common variety. Hence, the communicating devices will surely meet on at least one channel in all time slots irrespective of the blocks they choose.

Now, using equation 6.1, we can calculate the jamming probability by replacing block number by v and block size by k as follows:

$$P(JAM^t) = \frac{J}{C}[1 - \frac{1}{v}(1 - \prod_{i=1}^{r-1} \frac{J-i}{C-i})] \tag{6.2}$$

Now, the attacker may choose another strategy. It may get to know the blocks of D^*. As such, it can randomly choose a block from D^* and jam all the channels whose ids match with the entries of the varieties of the block. If it has more jamming strength it could choose more than one block from D^* and jam all the channels with same ids as the varieties in those blocks. We have assumed that the attacker can jam J channels. Since r is the size of each block in D^*, the attacker may choose $\lfloor \frac{J}{r} \rfloor$ distinct blocks of D^* for jamming.

Now, let us calculate the jamming probability in this case.

Theorem 6.1. *Let us assume the above design D^* is used for channel hopping by the sender and the receiver. If the attacker chooses to select a random block from D^* and jams all the channel with same ids as the varieties in the block, then the jamming probability is given by $P(JAM^t) = \frac{3}{v} - \frac{2}{v^2}$.*

Proof. $P(JAM^t) = P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t) = P(\mathcal{J}^t \supseteq \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) * P(\mathcal{S}^t = \mathcal{R}^t) + P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) * P(\mathcal{S}^t \neq \mathcal{R}^t)$ Since, D^* contains v blocks, $P(\mathcal{S}^t = \mathcal{R}^t) = \frac{1}{v}$. So, $P(\mathcal{S}^t \neq \mathcal{R}^t) = 1 - \frac{1}{v}$. Since the attacker can choose only a single block of D^*, it can not disrupt the communication without selecting the same block that is chosen by the sender and the receiver. Hence, $P(\mathcal{J}^t \supseteq \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) = P(\mathcal{J}^t = \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) = \frac{1}{v}$. On the other hand if the sender and the receiver choose distinct blocks of D^*, then the attacker will have only these options for successful jamming :
1) the attacker chooses the same block chosen by the sender.
2) the attacker chooses the same block chosen by the receiver.
3) the attacker chooses a third block containing the common element between the previous two blocks.

Now, we have stated earlier that in D^*, one variety occurs in only 3 blocks. Hence, the common variety between the sender's block and the receiver's block will occur only in one distinct block other that these two blocks. Hence, for successful jamming the attacker will have to choose only 3 out of the v blocks of D^*. Hence, $P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) = \frac{3}{v}$. Hence, $P(JAM^t) = P(\mathcal{J}^t \supseteq \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) * P(\mathcal{S}^t = \mathcal{R}^t) + P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) * P(\mathcal{S}^t \neq \mathcal{R}^t) = \frac{1}{v^2} + \frac{3}{v}(1 - \frac{1}{v}) = \frac{3}{v} - \frac{2}{v^2}$. □

A more powerful attacker can choose multiple blocks from D^* and jam all the channels having ids same as the varieties contained in those blocks. Let us consider an attacker who can jam $n(< r)$ blocks. Each block of D^* contains r varieties. Hence, the jamming strength of the attacker $J \leq nr$ as all the n blocks do not have distinct varieties. Now, we can calculate jamming probability as below :
$P(\mathcal{S}^t = \mathcal{R}^t) = \frac{1}{v}$. $P(\mathcal{J}^t \supseteq \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) = (1 - \frac{\binom{v-1}{n}}{\binom{v}{n}}) = (1 - \frac{(v-1)!n!(v-n)!}{n!(v-1-n)!v!}) = (1 - \frac{v-n}{v}) = \frac{n}{v}$. Now, if $\mathcal{S}^t \neq \mathcal{R}^t$, i.e if the sender and the receiver chooses different blocks of D^*, then in order to successfully jam the communication, the attacker will have to choose at least one of these three blocks :
1) the block chosen by the sender.
2) the block chosen by the receiver.
3) the third block containing the common element between the previous two blocks.

If the attacker chooses none of these blocks, then the attack will not be successful for the current time slot. Hence, $P(\mathcal{J}^t \not\supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) = P(\mathcal{S}^t \bigcap \mathcal{R}^t \notin \mathcal{J}^t | \mathcal{S}^t \neq \mathcal{R}^t) = \frac{\binom{v-3}{n}}{\binom{v}{n}} = \frac{(v-3)!n!(v-n)!}{n!(v-3-n)!v!} = \frac{(v-n)(v-n-1)(v-n-2)}{v(v-1)(v-2)}$. Hence, $P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) = 1 - P(\mathcal{J}^t \not\supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) = 1 - \frac{(v-n)(v-n-1)(v-n-2)}{v(v-1)(v-2)}$. This implies, $P(JAM^t) = P(\mathcal{J}^t \supseteq \mathcal{S}^t | \mathcal{S}^t = \mathcal{R}^t) * P(\mathcal{S}^t = \mathcal{R}^t) + P(\mathcal{J}^t \supseteq \mathcal{S}^t \bigcap \mathcal{R}^t | \mathcal{S}^t \neq \mathcal{R}^t) * P(\mathcal{S}^t \neq \mathcal{R}^t) = \frac{n}{v^2} + (1 - \frac{1}{v})(1 - \frac{(v-n)(v-n-1)(v-n-2)}{v(v-1)(v-2)}) = \frac{n}{v^2} + (1 - \frac{1}{v})(1 - \prod_{i=0}^{2}(1 - \frac{n}{v-i}))$.

We have compared the jamming probabilities for these two strategies. Table 6.1 provides a comparison of jamming probabilities with respect to different parameters. In Table 6.1, v is the number of varieties of the Steiner Triple System whose dual design is used in the proposed channel hopping scheme. J is the number of channels the attacker can jam simultaneously. C is the total number of available channels. C = No. of varieties in $D^* = \frac{v(v-1)}{6}$. It can be seen that an attacker who uses strategy 1 is more likely to be able to jam the communication than the attacker who chooses the second strategy. Therefore, the best strategy for an attacker is to jam J randomly chosen channels in each time slot.

v =No of blocks of D^*	C =No of channels	J =jamming strength	Probability of Jamming (Strategy 1)	Probability of Jamming (Strategy 2)
15	35	14	0.373333	0.355556
15	35	21	0.560000	0.495385
19	57	18	0.299169	0.288089
19	57	27	0.448753	0.408180
21	70	20	0.272109	0.263039
21	70	30	0.408163	0.374866
21	70	40	0.544218	0.474520
25	100	36	0.345600	0.322017
25	100	48	0.460800	0.411270
33	176	80	0.440771	0.392044

TABLE 6.1: table of jamming probabilities for two jamming strategies in scheme I

6.2.2.1 Comparison with UFH scheme

In this section, we shall be evaluating the performance of our channel hopping scheme with respect to the performance of Uncoordinated Frequency Hopping scheme in [67, 68]. The basis of comparison between these two schemes is the number of transmissions required to fully communicate a message from the sender to the receiver for both the schemes. We have used simulation for evaluating quantitative performances of our scheme and the UFH scheme. Figure 6.1 gives a pictorial representation of the relative performances of our scheme and the UFH scheme in absence of the jammer. The horizontal axis shows the number of fragments of a message. The vertical axis shows the total number of required transmissions of message-fragments including retransmissions of lost fragments. Note that our scheme ensures rendezvous in each and every time slot. Therefore, in absence of the jammer a message-fragment could be delivered to the receiver in every time slot. Hence when there is no jammer, the number of time slots required to transmit a message is equal to the number of fragments of the message in our scheme. On the other hand, The UFH scheme does not guaranty rendezvous of

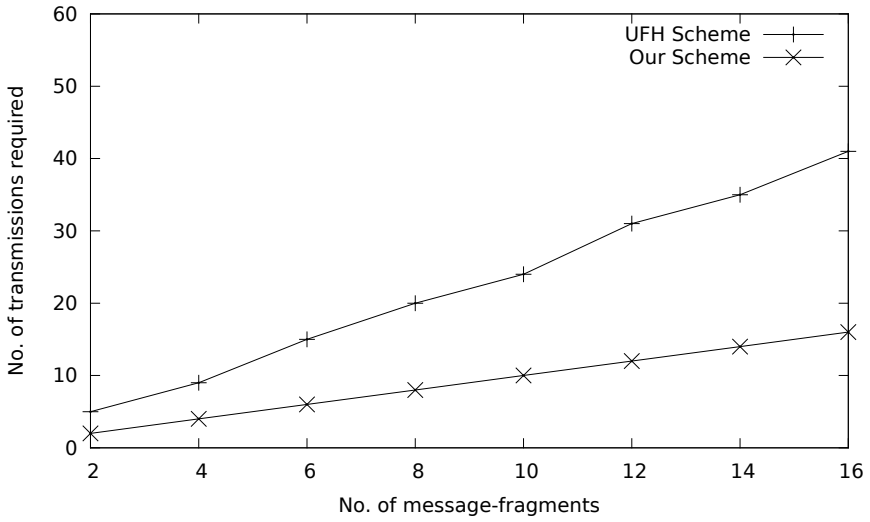

FIGURE 6.1: Graphical comparison of performances of Our scheme and the Uncoordinated Frequency Hopping scheme in [67, 68] in absence of the jammer. The comparison is done in terms of number of transmissions required to communicate a message from sender to receiver under the above two schemes. Total number of available channels is 35. Size of each block in our scheme is 7. The receiver and the sender can listen to/transmit over 7 channels simultaneously. In both the two cases the message is fragmented using rateless erasure coding technique.

communicating devices in every time slot. Hence a longer time is required to transmit the same message using UFH scheme as many message-fragments will not be delivered to the receiver necessitating retransmission of those fragments. Hence, for the same message our scheme requires less number of transmissions of message fragments than the UFH scheme.

Figure 6.2 gives a pictorial representation of the relative performance of these two schemes in presence of the jammer. The strength of jammer is same in both the cases. We used the dual design of $STS(15)$ for channel hopping scheme. In every time slot the sender can send message-fragments over 7 channels. Similarly the receiver can listen to 7 channels at a time. Figure 6.2 shows the number of transmissions required to fully communicate messages of different lengths from the sender to the receiver. The horizontal axis shows the number of fragments of a message. The vertical axis shows the number of transmission-attempts needed to send all the fragments to the receiver including the number of retransmissions of the lost fragments. Figure 6.2 shows that our scheme requires less number of transmissions of message-fragments than the UFH scheme. Hence, our scheme exhibits better performance than the UFH scheme in [67, 68].

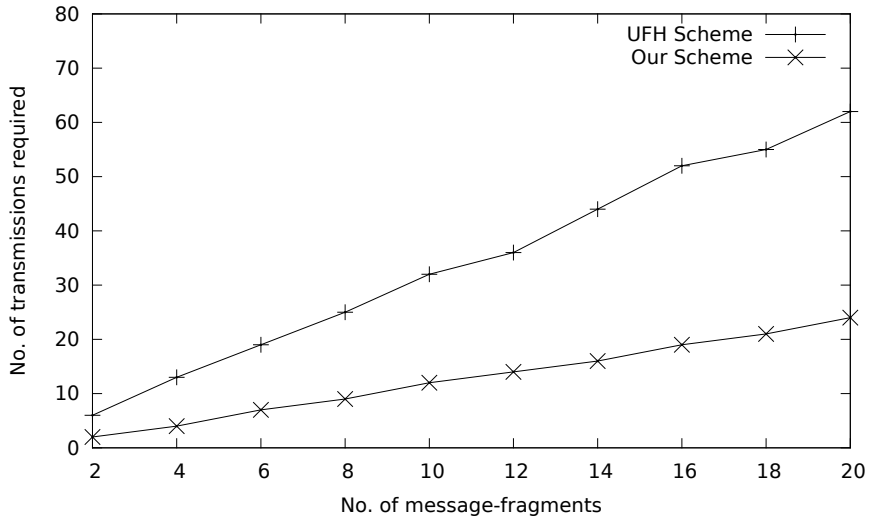

FIGURE 6.2: Graphical comparison of performances of Our scheme and the Uncoordinated Frequency Hopping scheme in [67, 68]. The comparison is done in terms of number of transmissions required to communicate a message from sender to receiver under the above two schemes. Total number of available channels is 35. The attacker has strength to jam 7 channels simultaneously. Size of each block in our scheme is 7. The receiver and the sender can listen to/transmit over 7 channels simultaneously. In both the two cases the message is fragmented using rateless erasure coding technique.

6.2.3 Alternate scheme using Transversal design

Here, we shall be applying another combinatorial design for channel hopping i.e. Transversal design. We have come across Transversal Design in definition 2.4. We present a construction of a Transversal design [38].

1. p is a prime number.
2. $X = \{(x, y) : 0 \leq x, y \leq p - 1\}$.
3. For all $i, 0 \leq i < p$, $G_i = (i, y) : 0 \leq y < p$.
4. $A_{ij} = \{(x, xi + j \mod p) : 0 \leq x \leq p - 1\}$.
5. $\mathcal{A} = \{A_{ij} : 0 \leq i, j \leq p - 1\}$

It can be shown that $(X, \mathcal{A})$ is a Transversal design. For a proof of this fact one may refer to [38].

Lemma 6.2. *The replication number of any variety in the above design is p. In other words, for any $(x, y) \in X, 0 \leq x, y < p$, $|\{A_{ij} : A_{ij} \in \mathcal{A}, (x, y) \in X\}| = p$*

Proof. We know that for a combinatorial design $vr = bk$, where v is the number of blocks, r is the replication number, b and k are the number of blocks and the size of blocks. In our design $v = |X| = p^2$, $b = |\mathcal{A}| = p^2, k = p$. Hence, $r = p$. □

Let $\mathcal{A}_1, \mathcal{A}_2$ be two partitions of $\mathcal{A}$ i.e. $\mathcal{A}_1 \bigcup \mathcal{A}_2 = \mathcal{A}$. $\mathcal{A}_1 = A_{ij}, i = 2m + 1$ and $\mathcal{A}_2 = A_{ij}, i = 2m$, m positive integer. Let S be the set of p^2 available channels. Let f be a one-to-one map $X \longrightarrow S$. Now the channel hopping scheme is as follows:

1. In every time slot t, the sender chooses a block B_t from $\mathcal{A}_1$.
2. For each element $z \in B_t$, the sender transmits message fragment through channel with id $f(z)$ in time slot t.

Similarly, the channel hopping scheme of the receiver is as follows:

1. In every time slot t, the receiver chooses a block B'_t from $\mathcal{A}_2$.
2. For each element $z' \in B'_t$, the receiver scans channels with id $f(z')$ in time slot t'.

In some arbitrary time slot t, the sender chooses a block $B_t = \{(x, xi+j) : 0 \leq x \leq p-1\}$ where $0 \leq i, j \leq p-1, i = 2m+1$. It transmits message-fragments to all the channels $f(z)$ such that $z \in B_t$. In time slot t, the receiver chooses a block $B'_t = \{(x, xi' + j') : 0 \leq x \leq p-1\}$ where $0 \leq i, j \leq p-1, i = 2m$. The receiver scans all the channels with id $f(z')$ such that $z' \in B'_t$. The sender and the receiver will rendezvous if $B_t \bigcap B'_t \neq \phi$. This can happen only if $\exists x, 0 \leq x \leq p-1$, such that $xi+j = xi'+j'$ i.e. if $x = (j'-j)(i-i')^{-1}$. A solution for x will exist if $i \neq i'$ which is true since i is odd and i' is even number. Hence, a rendezvous is ensured in every time slot under this hopping strategy.

This channel hopping scheme can be used by the sender and the receiver who wish to communicate over wireless medium in an adversarial environment. The sender fragments the message as described in section 6.1.3. Then it transmits the fragments through all the channels whose ids correspond to the elements in a block chosen randomly from $\mathcal{A}_1$ in a certain time slot. Similarly, in each time slot the receiver chooses a block from $\mathcal{A}_2$ and listens to all the channels whose ids correspond to the elements of the block. It is shown above that in each time slot there is a common channel on which the sender transmits and to which the receiver listens. If that particular channel is not jammed for that particular time slot, there will be a successful delivery of the message-fragment. On the other hand, if that particular channel is being jammed for that particular time slot, then the communication will fail for that time slot.

Let us illustrate this using an example. We choose $X = \{(x, y) : 0 \leq x, y < 3\}$. The individual groups are given by:

$$G_1 = \{(0,0), (0,1), (0,2)\}$$
$$G_2 = \{(1,0), (1,1), (1,2)\}$$
$$G_3 = \{(2,0), (2,1), (2,2)\}$$

Now, the blocks are given by:

$$A_{00} = \{(0,0)(1,0)(2,0)\}$$
$$A_{01} = \{(0,1)(1,1)(2,1)\}$$
$$A_{02} = \{(0,2)(1,2)(2,2)\}$$
$$A_{10} = \{(0,0)(1,1)(2,2)\}$$
$$A_{11} = \{(0,1)(1,2)(2,0)\}$$
$$A_{12} = \{(0,2)(1,0)(2,1)\}$$
$$A_{20} = \{(0,0)(1,2)(2,1)\}$$
$$A_{21} = \{(0,1)(1,0)(2,2)\}$$
$$A_{22} = \{(0,2)(1,1)(2,0)\}$$

Let us assume that this Transversal design is used by a pair of sender and receiver. Since the number of varieties of this design is 9, the number of available wireless channels should be equal to 9. According to our scheme all the blocks $A_{1,0}, A_{1,1}, A_{1,2}$ will be loaded into the sender and all other blocks i.e. $A_{0,0}, A_{0,1}, A_{0,2}, A_{2,0}, A_{2,1}, A_{2,2}$ will be loaded into the receiver. If the sender wishes to send a piece of message to the receiver it should first fragment the message using the erasure coding technique detailed in section 6.1.3. Then it can send individual fragments to the receiver. According to our scheme it sends one fragment in a time slot. In some arbitrary time slot the sender randomly chooses a block from its memory. Let this block be $A_{1,2}$. Now the sender transmits the message fragment to all the channels whose ids correspond to the varieties present in the block $A_{1,2}$ i.e. $(0,2)(1,0)(2,1)$. Similarly the receiver chooses one block from its memory. Let this be $A_{0,1}$. Hence, the receiver scans all the channels whose ids correspond to the varieties present in $A_{0,1}$ which are nothing but the varieties $(0,1)(1,1)(2,1)$. It can be seen that both the communicating devices meet on channel with id $(2,1)$. Hence, if this channel is jam-free for the current time slot, then there will be a successful communication of a message-fragment. On the other hand if the channel is being jammed for the current session, the message-fragment will get blocked.

The attacker attempts to fully utilise it's capability to block the communication as much as possible. Since, it is assumed that it has the knowledge of the hopping mechanism based on transversal design and does also have the knowledge of the mapping between the varieties of the design and the channels, it could actually employ different strategies. We enlist the strategies that the attacker could choose in this context:

1. The attacker could jam J randomly chosen channels out of the p^2 channels available for communication. Here J is the jamming strength.
2. The attacker could choose some n blocks from the set of blocks $\mathcal{A}$ and could jam all the channels whose ids correspond to the varieties of the n blocks where $J \leq np$.
3. The attacker could choose n blocks from the set of blocks $\mathcal{A}_1$ and could jam all the channels whose ids correspond to the varieties of the n blocks where $J \leq np$.
4. The attacker could choose n blocks from the set of blocks $\mathcal{A}_2$ and could jam all the channels whose ids correspond to the varieties of the n blocks where $J \leq np$.

Now, we can calculate the jamming probability for a random jammer who jams a set of J channels chosen in random from the set S of available channels. Note that here, $|S| = C = p^2$. Hence, using equation 6.1,

$P(JAM^t) = \frac{J}{C}[1 - \frac{1}{b}(1 - \prod_{i=1}^{k-1} \frac{J-i}{C-i})]$
Replacing the parameters with their values for our Transversal design we get

$$P(JAM^t) = \frac{J}{p^2}[1 - \frac{1}{p^2}(1 - \prod_{i=1}^{p-1} \frac{J-i}{p^2-i})] \tag{6.3}$$

If the attacker knows the hopping strategy of the communicating devices, i.e. if it gets to know the parameters of the Transversal design which is being used for channel hopping and also the mapping between the varieties of the design and the channel ids then it could use different strategy for jamming. It could randomly choose n blocks from the $(X, \mathcal{A})$ and jam all channels with ids corresponding to the varieties of the block. Since each block has p varieties, the jamming strength of the attacker J must be at least np. Lemma 6.3 calculates the jamming probability of an attacker who uses this jamming strategy.

Lemma 6.3. *If the above Transversal design is used by the communicating devices, and the attacker chooses n blocks out of the p^2 blocks of $(X, \mathcal{A})$ and jams all the np channels corresponding to the varieties of the n blocks, the jamming probability is* $= 1 - \prod_{i=1}^{n}(1 - \frac{p}{p^2-i+1})$.

Proof. Since, the sender and the receiver choose two blocks from two different partitions of $\mathcal{A}$, they choose two different blocks of $\mathcal{A}$. Hence, the common variety between the two selected blocks is a variety which is equally probable to be any of the p^2 varieties of the design. Let this common variety be denoted as (α, β) where $0 \leq \alpha, \beta < p$. Now, the replication number of any variety in $(X, \mathcal{A})$ is p (lemma 6.2). Hence, (α, β) occurs in p distinct blocks of $(X, \mathcal{A})$. The attacker chooses n blocks. Hence, the probability of not choosing a block containing the variety (α, β) is $\frac{\binom{p^2-p}{n}}{\binom{p^2}{n}} = \frac{(p^2-p)(p^2-p-1)...(p^2-p-n+1)}{p^2(p^2-1)...(p^2-n+1)} = \prod_{i=1}^{n}(1 - \frac{p}{p^2-i+1})$. Hence, the result. □

Thirdly, we calculate the jamming probability if the attacker uses the third strategy i.e. if the attacker in each time slot chooses some n random blocks from $\mathcal{A}_1$ and jams all the channels with ids corresponding to the varieties in those n chosen blocks. We calculate the jamming probability for strategy 3 in lemma 6.4.

Lemma 6.4. *If the attacker chooses the third strategy, then the jamming probability is* $1 - \prod_{i=0}^{n-1} \frac{p^2-1-2i}{p^2+p-2i}$.

Proof. Let us first find the value of $r_1 = |\{A : A \in \mathcal{A}_1, (x, y) \in A\}|$ for some $(x, y) \in X$. Hence, $r_1 = |\{(i, j) : 0 \leq i, j < p, i = 2m + 1, y = xi + j \mod p\}|$. So, $r_1 = |\{(i, j) : 0 \leq i, j < p, i = 2m + 1, j = y - xi \mod p\}| = |\{j : 0 \leq j < p, j = y - xi \mod p, 0 \leq i < p, i = 2m + 1\}| = \frac{p+1}{2}$.

Hence, the common variety between the sender's block and the receiver's block occurs in $\frac{p+1}{2}$ blocks of $\mathcal{A}_1$. Hence, for successfully jamming the communication, the attacker needs to choose at least one of them out of the $\frac{p(p+1)}{2}$ blocks in $\mathcal{A}_1$. On the other hand, the attack will be unsuccessful if the attacker chooses n blocks from the other $\frac{p(p+1)}{2} - \frac{p+1}{2} = \frac{p^2-1}{2}$ blocks of $\mathcal{A}_1$. Hence, the jamming probability is equal to $1 - \frac{\binom{\frac{p^2-1}{2}}{n}}{\binom{\frac{p(p+1)}{2}}{n}} = 1 - \frac{\prod_{i=0}^{n-1}(\frac{p^2-1}{2}-i)}{\prod_{i=0}^{n-1}(\frac{p^2+p}{2}-i)} = 1 - \prod_{i=0}^{n-1} \frac{(p^2-1-2i)}{(p^2+p-2i)}$. □

Finally, we calculate the jamming probability if the attacker chooses the fourth strategy i.e. if it chooses n distinct blocks of $\mathcal{A}_2$ at random in each time slot and jams all channels whose ids correspond to the varieties of the blocks. We calculate the jamming probability in lemma 6.5.

Lemma 6.5. *If the attacker chooses the fourth strategy, then the jamming probability is* $1 - \prod_{i=0}^{n-1} \frac{p^2-2p+1-2i}{p^2-p-2i}$.

Proof. The proof is similar to that of lemma 6.4. Let $r_2 = |\{A : A \in \mathcal{A}_2, (x, y) \in A\}|$ for some $(x, y) \in X$. So, $r_2 = r - r_1 = p - \frac{p+1}{2} = \frac{p-1}{2}$. Hence, the common variety between the sender's block and the receiver's block occurs in $\frac{p-1}{2}$ blocks of $\mathcal{A}_2$.

Hence, for successfully jamming the communication, the attacker needs to choose at least one of them out of the $\frac{p(p-1)}{2}$ blocks in $\mathcal{A}_2$. On the other hand, the attack will be unsuccessful if the attacker chooses n blocks from the other $\frac{p(p-1)}{2} - \frac{p-1}{2} = \frac{p^2-2p+1}{2}$ blocks of $\mathcal{A}_2$. Hence, the jamming probability is equal to $1 - \frac{\binom{\frac{p^2-2p+1}{2}}{n}}{\binom{\frac{p(p-1)}{2}}{n}} = 1 - \frac{\prod_{i=0}^{n-1}(\frac{p^2-2p+1}{2}-i)}{\prod_{i=0}^{n-1}(\frac{p(p-1)}{2}-i)} = 1 - \prod_{i=0}^{n-1}\frac{p^2-2p+1-2i}{p^2-p-2i}$. □

Value of p	C =No of channels	J =jamming strength	Prob. of Jamming (Strategy 1)	Prob. of Jamming (Strategy 2)	Prob. of Jamming (Strategy 3)	Prob. of Jamming (Strategy 4)
7	49	28	0.559767	0.471719	0.481026	0.488722
7	49	14	0.279883	0.267857	0.269841	0.271429
7	49	21	0.419825	0.376900	0.382173	0.386466
5	25	10	0.384000	0.366667	0.371429	0.377778
5	25	15	0.576000	0.504348	0.516483	0.533333
5	25	20	0.768000	0.616996	0.637363	0.666667
11	121	55	0.450789	0.384326	0.388883	0.390944
11	121	66	0.540947	0.442708	0.448992	0.451850
11	121	33	0.270473	0.250573	0.252185	0.252906

TABLE 6.2: table of jamming probabilities for 4 jamming strategies in scheme II

Table 6.2 provides a comparison of the jamming probabilities for the 4 jamming strategies discussed above. We used different parameters for calculating the jamming probabilities. Table 6.2 shows the jamming probabilities under different set of parameters. It can be seen in table 6.2 that strategy 1 ensures higher jamming probability than other 3 strategies. So the attacker could maximise the probability to jam any message fragment if it employs the first strategy. Hence, an attacker's best strategy is to jam some J randomly chosen channels in any time slot. Thus, we can say that the attacker does not gain anything by knowing the channel hopping mechanism using transversal design. So, the channel hopping mechanism i.e. the parameters of the transversal design and the mapping between the varieties of the design and the channel can be made public. In other words it is a "keyless communication scheme" where there is no shared secret between the communicating devices.

6.2.3.1 Comparison with other schemes

Like the first channel hopping scheme using Steiner Triple System, we shall be comparing this scheme with the well known UFH scheme in [67, 68] by Strasser *et al.* The basis of comparison is the number of transmissions required for communicating a message from the sender to the receiver. We used simulation of the communication model for our scheme and the Strasser *et al.* scheme. The message is fragmented using erasure coding technique as discussed in section 6.1.3. Thereafter, the fragments are transmitted from the sender to the receiver over the wireless medium. Communication will be accomplished if the receiver is able to collect a fixed number of fragments depending upon some parameters. We have calculated the number of transmissions required for communicating messages to the receiver both in our scheme and in UFH scheme. We considered identical jammer in both the two cases. The number of available channels is same in both the cases. Communicating devices too are of identical capacity.

Figure 6.3 provides a pictorial comparison between our transversal design-based channel hopping scheme and the Uncoordinated Frequency Hopping scheme in [67, 68] in absence of the attacker. We used $TD(5, 5)$ design discussed in [38] with this parameter: $p = k = 5$. It can be seen in figure 6.3 that in absence of the jammer all the fragments get transmitted to the receiver and no redundant transmission is required. Hence, the number of transmissions required to send a message is equal to the number of fragments of a message. In contrary, the UFH scheme requires much more transmissions of message fragments than the actual size of the message in terms of number of fragments.

Figure 6.4 provides a pictorial comparison between our transversal design-based channel hopping scheme and the Uncoordinated Frequency Hopping scheme in [67, 68]. We used $TD(5, 5)$ design discussed in [38] with this parameter: $p = k = 5$. It can be seen in figure 6.4 that our scheme requires much less number of transmission for the same message than the UFH scheme. In other words, our scheme offers faster message delivery than the UFH scheme. This is because of the fact that our scheme ensures rendezvous of the pair of communicating devices on all time slots. So, a message fragment can be delivered over a channel in any time slot if the channel is not being blocked by the jammer. On the other hand, the UFH scheme, as discussed before does not guaranty rendezvous in any time slot and depending upon the parameters of the system the probability of a rendezvous may be very low. Hence, much more transmissions will be required for communicating a message in UFH scheme.

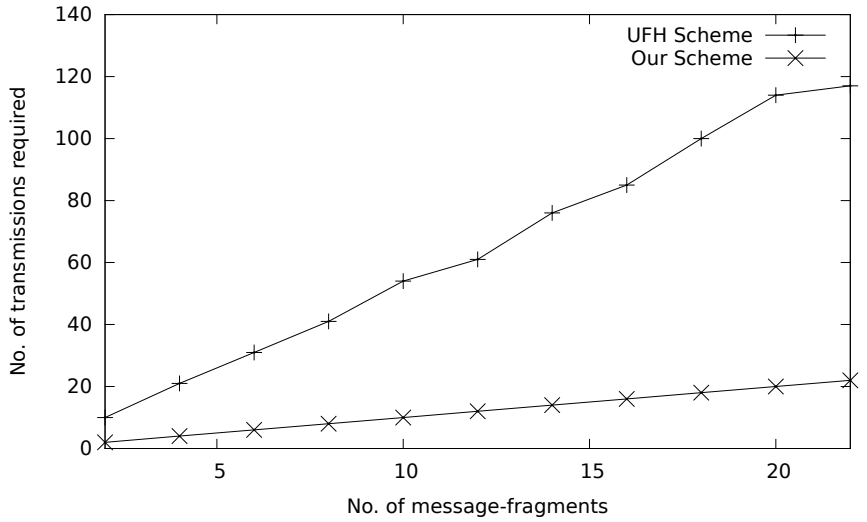

FIGURE 6.3: Graphical comparison of performances of our transversal design based scheme and the Uncoordinated Frequency Hopping scheme in [67, 68] in absence of the jammer. The comparison is done in terms of number of transmissions required to communicate a message from sender to receiver under the above two schemes. Total number of available channels is 25. Size of each block in our TD based scheme is 5. The receiver and the sender can listen to/transmit over 5 channels simultaneously. In both the two cases the message is fragmented using rateless erasure coding technique.

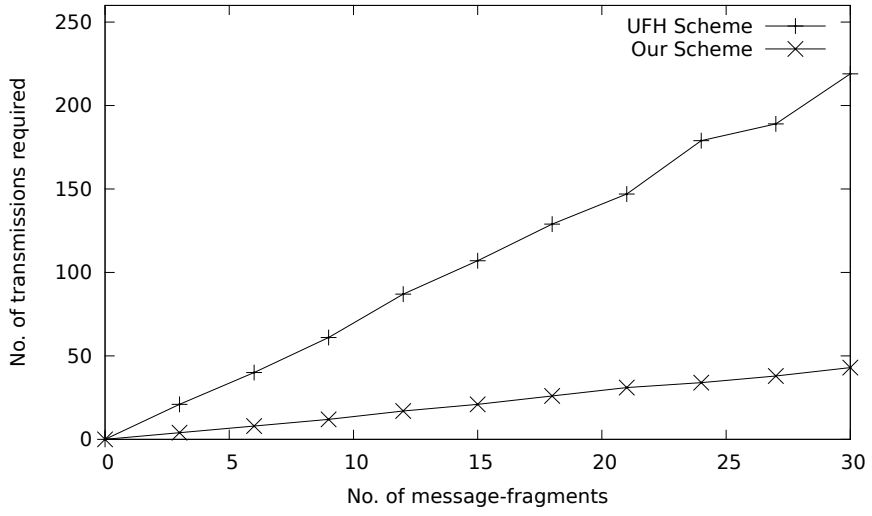

FIGURE 6.4: Graphical comparison of performances of our transversal design based scheme and the Uncoordinated Frequency Hopping scheme in [67, 68]. The comparison is done in terms of number of transmissions required to communicate a message from sender to receiver under the above two schemes. Total number of available channels is 25. The attacker has strength to jam 7 channels. Size of each block in our TD based scheme is 5. The receiver and the sender can listen to/transmit over 5 channels simultaneously. In both the two cases the message is fragmented using rateless erasure coding technique.

6.3 Concluding Remarks

In this chapter we discuss two schemes for wireless communication in presence of the jammer. We used combinatorial design for designing the framework of our communication schemes. The first scheme uses Steiner Triple System and the other uses Transversal Design. These two schemes perform better than Uncoordinated Frequency Hopping scheme proposed by Strasser *et al.* in [67, 68]. Our communication schemes are elegant in the sense that they ensure rendezvous of the communicating devices in every time slot. In UFH scheme the communicating devices would wait unboundedly for a rendezvous of the communicating devices in order to deliver a message fragment. Our schemes thus bring on an improvement to the earlier scheme by making a rendezvous certain on all time slots. Hence our proposed schemes outperform the UFH scheme. This enhancement of performance offered by the proposed scheme is also corroborated by experimental results.

Chapter 7

Jamming Resistant Schemes for Wireless Communication : A Combinatorial Approach

The work described in this chapter is based on the work discussed in [7]. In this chapter we discuss two frequency hopping schemes, one for unicast communication and the other for multicast communication. The first scheme allows a group of users to communicate with each other in the presence of a jammer. This scheme deals with anti-jamming communication for a set of users/nodes who communicate with each other. The aim of this scheme is to ensure a that a pair of nodes/users meet within a fixed amount of time on a dedicated frequency channel. Most of the existing schemes on frequency hopping are meant for two party communication or broadcast communication. For two party communication, a pair of nodes/users only need to have a secret key shared by both of them. This shared key is used as a seed to generate pseudorandom number (PN) sequence which are used by these users to hop through a set of available frequencies evading the jammer who does not know the key and hence can not anticipate the channel used for communication on a certain time slot. The same technique is used for broadcast communication. Those schemes that deal with multi party communication try to establish single common control channels for all users. But the problem of establishing separate communication channels for different pairs of users were not addressed. The classical frequency hopping scheme offers good performance when there is a sender and a group of receivers all listening to the same sender. But the classical frequency hopping scheme using PN sequence does not guarantee a rendezvous in a fixed time when the network contains many identical users communicating to each other. For classical frequency hopping scheme when there are multiple users, the users need to hop through many frequency channel using different PN sequences and in such scenario two nodes

can meet over a channel depending upon chance. The only frequency hopping scheme that offers a time-bounded rendezvous between any pair of users is the Quorum Rendezvous Channel Hopping scheme by Lee et al. [36]. But the time bound for Lee et al. scheme is very high, between $O(c)$ and $O(c^2)$, where c is the number of channels in the network. We, in our work have attempted to reduce this time-bound to exactly $O(c)$. In our scheme a user does not need to remain idle for even a single time slot if it has messages to be transmitted to other nodes. In every time slot each user rendezvous with another user in this scheme contrary to the existing schemes where nodes may remain idle waiting for a rendezvous with another user. In order to evade jamming, the users keep changing frequency channels using PN sequences.

The second scheme is for multicast communication. This scheme assumes that there is a set of senders and a set of receivers. The senders send messages to the receivers but not vice-versa. Moreover, all the senders may or may not send the same message over different frequencies. We discuss a technique that allows the receivers to listen to the transmission from the senders while the jammer is actively present in the scenario. This scheme ensures that a pair of sender and receiver will meet on some channel in a bounded time called a session. We have used combinatorial designs in both the schemes.

7.1 Preliminaries

Lemma 7.1. *If L_1 and L_2 are two orthogonal $n \times n$ Latin Squares defined on the set $\{0, 1, 2, \ldots, n-1\}$ and $x, y \in \{0, 1, 2, \ldots, n-1\}$ then $L_1 + x \mod n$ and $L_2 + y \mod n$ are orthogonal LS, where $L + x \mod n$ is the LS obtained by adding an integer x to each element of the Latin Square L modulo n.*

Proof. It is easy to see that $L_1 + x \mod n$ is a Latin Square and so is $L_2 + y \mod n$. Let $L_3 = L_1 + x \mod n$ and $L_4 = L_2 + y \mod n$. Now let us assume they are not orthogonal. So, there are $i, j : 1 \leq i \leq n, 1 \leq j \leq n$ and $i', j' : 1 \leq i' \leq n, 1 \leq j' \leq n, (i, j) \neq (i', j')$ such that $L_3(i, j) = L_3(i', j')$ and $L_4(i, j) = L_4(i', j')$. It implies $L_1(i, j) + x \mod n = L_1(i', j') + x \mod n$ and $L_2(i, j) + y \mod n = L_2(i', j') + y \mod n$. This means $L_1(i, j) = L_2(i', j')$ and $L_2(i, j) = L_2(i', j')$ which is contradictory to our assumption that L_1 and L_2 are orthogonal. □

Theorem 7.2. *If $n > 1$ is odd, then there exist two orthogonal Latin Squares of order n.*

Proof. Ref. Theorem 6.23 of [66]. □

7.1.1 Construction of Orthogonal Latin Squares

We will now show how we can construct two orthogonal LS of order $n > 1$ where n is an odd integer. We construct two $n \times n$ LS L_1 and L_2 as given below:

$L_1(i, j) = i + j \mod n$
$L_2(i, j) = i - j \mod n$

We refer the reader to Theorem 6.23 of [66] for a proof that L_1 and L_2 are orthogonal Latin Squares.

7.2 Network Description

7.2.1 System Model for Scheme-I and Scheme-II

We consider a network with a maximum of N nodes which are within communication range. The main goal is to allow the nodes to communicate with each other using the set of available frequencies $\mathcal{C}$. Each node is equipped with a transceiver which is capable of sending and receiving communication signal over a range of frequencies. We assume that the transceiver does not leak information about its active reception channels. Hence, the channels on which the transceiver is actively listening on cannot be detected by monitoring its radio signal emissions. The nodes also have a clock and are loosely time synchronized in the order of seconds using GPS. The nodes are capable of computation and storage. The nodes can hop over $\mathcal{C}$ (where, $|\mathcal{C}| = c$) communication frequencies. Each node has a secret key/public key pair (SK, PK) which is used for public-key encryption/decryption purpose and (K, K^{-1}) key for verification and signature. There is a base station which is also a certification authority which issues certificates to bind the public key to the respective nodes. Apart from this, each node holds a secret key $\mathcal{K}$. This key is used for randomising the channel hopping sequences of the nodes. The usage of this key $\mathcal{K}$ will be discussed in later sections. The timeline is divided into time slots. A time slot is a small quantum of time during which a certain node remains on the same channel i.e it sends/receives signals on the same channel. In other words, a time slot is the minimum amount of time needed to transmit a message packet. A session consists of some number(say, ω) of consecutive time slots. A time slot t belongs to session s, where $s = (t \mod \omega)$ and $0 \leq t < \infty$. There is a base station that controls the network. All the nodes are connected to the outside world through the base station.

We consider a different type of network model for the jamming resistant communication scheme discussed in section 7.4. Here we assume that there is a set $\mathcal{S}$ of senders and a set $\mathcal{R}$ of receivers. The senders only send messages whereas the receivers only collect these messages but not the other way round. For example, the senders can be a base station sending some broadcast messages simultaneously over multiple frequencies and thus posing as a set of senders. The receivers may be a set of nodes belonging to a wireless network. This is the basic modelling difference between the two schemes. In this scheme communication is unidirectional as opposed to the previous model where we considered bidirectional message sending. Everything else is same as discussed in section 7.2.1.

7.2.2 Attacker Model

We consider an omnipresent adversary who wants to disrupt the communication as much as possible. It is capable of eavesdropping on the network, alter messages and jam the channels by blocking the propagation of radio signals. The attacker is computationally bounded, but is present all the time. At a particular instant it can jam at most J channels where $J < c$. It jams a channel by emitting high power signal on that channel. We consider that the attacker's clock is time synchronized with the system. In every time slot t_i, the attacker jams a set of channels $S^J_{t_i}$, where $S^J_{t_i} \subset \mathcal{C}, |S^J_{t_i}| = J$. Thus, at the start of a time-slot the jammer starts jamming J randomly chosen channels. It continues to jam those channels until the beginning of the next time-slot. Thereafter, it chooses a new set of channels for jamming. In our jamming resistant communication scheme the sending sequence as well as the receiving sequence of any node changes in each session. This ensures that the jammer cannot totally disconnect the communication between two nodes by jamming a particular frequency channel. If a particular node is sending message to another node through a frequency channel x in session s, then the probability that in next session they will not be using the frequency channel i for message transfer is $\frac{c-1}{c}$. Therefore, a jammer who keeps on jamming the same set of channels at every time slot will not be able to disrupt communication between a pair of nodes every time.

7.2.3 Protection against eavesdropping and message modification

Each message fragment (ref. section 7.2.4) M is encrypted with public key of the receiver. The ciphertext is $C = E_{PK}(M)$. The ciphertext is signed with the secret key. The signature is given by $\sigma = sign(C, K^{-1})$. The message (C, σ) are then sent on the channel chosen according to the Algorithm 9 given in the next section. Due to encryption, an

adversary is unable to get any information by eavesdropping. The receiver checks the signature of the message to verify that it indeed was sent by a valid sender and not modified by an adversary. We will not discuss encryption and decryption techniques here, but assume that secrecy and authentication can be achieved using known protocols (as in [67]). Throughout the rest of the chapter we will address only jamming attacks, in which the adversary blocks the radio signals and makes communication difficult.

7.2.4 Message Passing Mechanism

Since the random jammer jams randomly chosen J channels in each time-slot, there will be some messages that the adversary will be able to jam. Since there are $c = N$ channels out of which the jammer is able to jam J channels per time-slot, the probability that a particular channel will be jammed by the adversary is equal to J/c. This probability is reduced for large c. However there is a non-zero probability that the jammer will disrupt the communication between a particular pair of nodes.

To alleviate this, each message is fragmented into a number of parts and sent to the intended recipient whenever the sender and the receiver rendezvous on the same channel. The receiver collects those fragments and reassembles them to get the original message. Since the attacker has a positive probability to jam a packet, it may so happen that some of the fragments are lost because of the jammer. So, we need to send redundant packets to ensure that the receiver is successfully able to reassemble the original message from the packets it receives. Erasure code is a technique to fragment messages and has been studied in [47], [57]. Through erasure codes it is possible to fragment a message into many parts so that the receiver can reassemble the message if it is able to receive at least l of the fragments, where l is an integer whose value depends upon the coding technique used. With this technique, it has to be ensured that the authenticity and integrity of each fragment is preserved. This can be done using one way accumulators based on bilinear maps [54], [8]. So, each node willing to send a message to another node, fragments the message into a number of parts through erasure coding technique and computes witnesses for each of them using one-way accumulators. Then, the sender sends the (witness, message) pair to the receiver whenever they rendezvous (which happens exactly once in each session). The receiver tries to verify each message fragment with the witness. If it is able to verify then it accepts the packet. Otherwise it rejects the packet. This way if sufficient message fragments are received, the receiver will be able to regenerate the original message.

7.3 Combinatorial anti-jamming protocol for unicast communication : Scheme-I

7.3.1 Definitions and Assumptions

Here we discuss some basic assumptions relating to the network model which is in addition to the system model discussed in section 7.2.1. Firstly, the number of available channels (c) must be more than or equal to the maximum number of nodes or users (N) in the network. In this chapter, we assume $c = N$. Secondly, the size of the network must be odd. If the network size is an even number then we shall have to make it odd by adding a virtual node to the network. This virtual node will cause no intricacy to the communication model.

Definition 7.3. A time slot is the period during which a certain node transmits a message-fragment over a certain frequency channel. Alternately it can be defined as the period during which a certain node listens to a certain frequency channel. A session consists of N many time slots.

Definition 7.4. *Sending Sequence* for a node n_i is a set $\mathcal{P}$ whose elements are 3-tuples (α, β, j) such that $\alpha, \beta \in \{0, 1, \ldots, N-1\}, j \in \{\{0, 1, \ldots, N-1\} \setminus \{i\}\}$ and for all 3-tuple $(\alpha, \beta, j) \in \mathcal{P}$, node n_i sends data to node n_j at time-slot β through frequency channel α. Also sender node n_i and receiver node n_j rendezvous at time slot β on frequency channel α.

A *Receiving Sequence* for a node n_i is a set $\mathcal{Q} = \{(\delta, \gamma, j) : \delta, \gamma \in \{0, 1, \ldots, N-1\}, j \in \{\{0, 1, \ldots, N-1\} \setminus i\}\}$, such that $\forall(\delta, \gamma, j) \in \mathcal{Q}$ node n_i receives data from node n_j at time-slot γ through frequency channel δ and for $i = j$, node n_i receives data from base station at time γ through frequency channel δ. Also receiver node n_i and sender node n_j rendezvous at time slot γ on frequency channel δ.

In our jamming resistant communication scheme, we map combinatorial design to frequency hopping sequence. Each element of the design is a two tuple (x, t), where x corresponds to the channel id. and t corresponds to the time slot.

Mathematically, the mapping between designs and channel allocation occurs in the following way:

Let, $(X, \mathcal{A})$ be a design where X is a set of varieties. The elements in X are two tuples. Let, $\mathcal{A} = \{B_1, B_2, \ldots, B_g\} \cup \{B'_1, B'_2, \ldots, B'_g\}$ be a set of blocks such that:

1. $\forall i, j \in \{1, 2, \ldots, g\}, B_i \cap B_j \neq \phi \Rightarrow i = j$,

2. $\forall i, j \in \{1, 2, \ldots, g\}, B'_i \cap B'_j \neq \phi \Rightarrow i = j$,

3. $\forall i, j \in \{1, 2, \ldots, g\}, |B_i \cap B'_j| \geq 1$.

We assign two blocks B_i, B'_i of $\mathcal{A}$ to node n_i of the network. These two blocks correspond to the sending and receiving sequences of the node n_i respectively. Let n_j be another node whose sending and receiving sequences correspond to block B_j and B'_j. The sending sequence of node n_i and the receiving sequence of node n_j are B_i and B'_j. $\forall (x, t) \in B_i \cap B'_j$, (x, t) corresponds to the rendezvous point of node n_i and node n_j. It means at time slot t node n_i will be sending some message to node n_j on frequency channel identified by x. Similarly, node n_i will be receiving message from node n_j in time slot t' on frequency channel y only if $(y, t') \in B_j \cap B'_i$.

Condition 1) ensures that no two senders transmit over the same channel at the same time. Similarly, condition 2) ensures that no two receivers listen to the same channel at the same time. This restriction prevents collision between two senders (or two receivers) sending (receiving) messages over the same channel at the same time. Condition 3) ensures that the design should ensure that there is atleast one element in common to two blocks of the design (used for sending and receiving). This is to guarantee that a sender and a receiver rendezvous within a finite amount of time. In the rest of the chapter the terms node and user will bear the same meaning.

7.3.2 Channel Hopping

We now describe a channel hopping technique using Latin Squares in details. If N is odd then we can construct two orthogonal Latin Squares of order N using the method described in Theorem 6.23 of [66]. Let these two Latin Squares be L_1 and L_2. Let L_3 be the Latin Square obtained through superposition of L_1 and L_2. Load node $n_i, 1 \leq i \leq N$ with the i^{th} row and the i^{th} column of array L_3. A key $\mathcal{K}$ is loaded in all the nodes. This key will be used as a seed to pseudo random generators to generate pseudo random numbers modulo N. There are $c = N$ frequency channels with identifiers $0, 1, 2, \ldots, N-1$. The entire time-line is divided into sessions. A session consists of N consecutive time slots. The time slots in every session is identified as $0, 1, \ldots, N-1$.

Algorithm 9 is used by each node of the network for hopping through different channels. The Algorithm generates the sending and receiving sequences of any user in a session. In the Algorithm, X' and Y' are two sets that contain the sending and receiving sequences of any node. The Algorithm also uses two pseudo random-integer generators. The output generated by these two pseudo random-integer generators are used to randomize the sending and receiving sequences of a user in each session as shown in Algorithm 9.

Algorithm 9 updates the content of the row X and column Y in each session using the random integers generated by the pseudo-random generators. This is to ensure that the sending and the receiving sequences become unpredictable by the jammer who does not know the key $\mathcal{K}$.

Algorithm 9 Latin Square based Channel Hopping Algorithm for unicast communication

Input: Key $\mathcal{K}$.
L_1 and L_2 are two orthogonal LS. L_3 is the array obtained by superposing L_1 and L_2. r^{th} row of Superposed LS L_3 is $X = \{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_N, \beta_N)\}$ and r^{th} column is $Y = \{(\delta_1, \gamma_1), (\delta_2, \gamma_2), \ldots, (\delta_N, \gamma_N)\}$.
Lifetime T of key $\mathcal{K}$. T is the number of sessions after which the key $\mathcal{K}$ needs to be refreshed.
Output: Channel Hopping Sequence of the r^{th} node n_r.
for $(i = 0; i < T; i++)$ **do**
  $X' = Y' = \phi$
  $x = Pseudorand_1(\mathcal{K}, i) \mod N$
  $y = Pseudorand_2(\mathcal{K}, i) \mod N$
  for $(j = 1; j \leq |X|; j++)$ **do**
    $\alpha'_j = \alpha_j + x \mod N, \beta'_j = \beta_j + y \mod N$
    $\delta'_j = \delta_j + x \mod N, \gamma'_j = \gamma_j + y \mod N$
    if $j \neq r$ **then**
      $X' = X' \cup (\alpha'_j, \beta'_j, j)$
    end if
    $Y' = Y' \cup (\delta'_j, \gamma'_j, j)$
  end for
  X' is the sending sequence of node r for session i.
  Y' is the receiving sequence of node r for session i.
end for

We prove in section 7.5 that Algorithm 9 computes the sending and receiving sequences in such a manner that there is no collision between multiple nodes over the same frequency channel. In other words, no two nodes try to send messages over the same channel and no two nodes read the same channel at the same time. In addition to this, lemma 7.9 shows that any sender and any receiver must rendezvous over a certain channel on all the sessions. Hence, the average rendezvous time is bounded by $O(c)$.

The secret key $\mathcal{K}$ is used to randomise the sending and receiving sequences of a node in a session. It is applied to ensure that the adversary who does not know the key cannot compute the sending and receiving sequence of any node for any session. So, a jammer who chooses J channels out of $c = N$ available channels has a jamming probability equal to J/N as proved in Theorem 7.12. It is assumed that after a certain time the key might get exposed to the adversary who can take advantage of some weakness of the base station. It can be noted from Algorithm 9 that the receiving sequence of each node n_r of the network contains a triplet (α'_j, β'_j, j) and hence according to definition

7.4, β'_j is time slot when node n_j expects to receive some data from the base station over frequency channel α'_j in a particular session. This data is nothing but information about the new key. The key is sent at regular interval to all the nodes after encrypting it with the public key of the individual nodes. The lifetime of any key $\mathcal{K}$ is T sessions. Hence new key must be communicated to all the nodes before T many sessions have elapsed. This key refreshing is done to ensure that the secret key does not get compromised by the attacker. Once T sessions have elapsed, Algorithm 9 starts using the fresh key.

The proof of correctness is presented in Section 7.5.1.

Example 7.1. *Let the number of nodes in a sensor network be* 4. *We take* L_1 *and* L_2 *given in Section 2.16 as the Latin Square to construct hopping sequence for our scheme. The superposition of* L_1 *and* L_2 *yields* L_3. *Now distribute* i^{th} *row and* i^{th} *column of* L_3 *to one node. Node* n_1 *will get the first row of* L_3 *i.e.,* $(1,5),(2,4),(3,3),(4,2),(5,1)$ *and the first column of* L_3 *i.e.,* $(1,5),(2,1),(3,2),(4,3),(5,4)$ *and so on.*

Let us consider that the pseudo random integer generated by $Pseudorand_1()$ *function at session* $i = 1$ *be 1 modulo* 5. *Similarly, let us consider that the pseudo random integer generated by* $Pseudorand_2()$ *function at session* $i = 1$ *be 4 modulo* 5. *According to the algorithm the sending sequence of node 1 is* $\{(3,3,2),(4,2,3),(0,1,4),(1,0,5)\}$. *Also, the receiving sequence of node 1 will be* $\{(3,0,2),(4,1,3),(0,2,4),(1,3,5)\}$.

Hence, at session 1, node 1 should send a message packet to node 2 through frequency channel 3 at time slot 3 if it has some message packet to be delivered to node 2. If node 1 does not have anything to send to node 2 then it will remain idle at time slot 3. Similarly, node 1 can send message packets to node 3 through channel 4 at time slot 2, can send long channel 0 at time slot 1 to node 4. Lastly node 1 can send message packet to node 5 at time slot 0 on frequency channel 1. Therefore, node 1 will be transmitting to different nodes on different channels at different time slots.

Node 1 will listen to channel 3 at time slot 2 for incoming packets from node 2. Similarly, node 1 will listen to channel 4 at time slot 1 for message packet sent by node 3, will listen to channel 0 at time slot 2 for message packet sent by node 4 and listen to channel 1 at time slot 3 for message packet sent by node 5.

7.4 Anti-jamming protocol during multicast communication : Scheme-II

In this section, we propose a channel hopping scheme for multicast communication using combinatorial design. In other words, this scheme is designed for such scenarios where

one sender sends a message that is received by a group of receivers. We shall be using transversal design for developing this channel hopping scheme. Before we move into the details of the scheme we first state the assumptions of the scheme below.

7.4.1 The Scheme

Let us consider a network consisting of N nodes where $p < N \leq p^2$ and p is a prime number that equals the number of available channels. If N is less than p^2 we can regard $p^2 - N$ many nodes as virtual nodes having no physical existence. These virtual nodes will have their own receiving sequences. Let the identifier of the nodes be $(0,0),(0,1),\ldots,(p-1,p-1)$.

Recall the definition of Transversal Design stated in section 2.1.1. Let p be a prime number. We can compute a transversal design as in [39].

1. $X = \{(x,y) : 0 \leq x < p, 0 \leq y < p\}$.
2. $\forall i \in \{0,1,2,\ldots,p-1\}, G_i = \{(i,y) : 0 \leq y < p\}$.
3. $\mathcal{A} = \{A_{ij} : 0 \leq i < p \quad \& \quad 0 \leq j < p\}$.
4. $A_{ij} = \{(x, xi + j \mod p) : 0 \leq x < p\}$.

Now we develop a multicast channel hopping scheme for wireless communication using transversal design. The map from a transversal design to a multicast scheme is the following: each block of the design is used to generate the receiving sequence in Algorithm 10. The two tuple elements of each block will correspond to the (time slot, channel id.) pair. This mapping is discussed in detail below. This channel hopping scheme enables a set $\mathcal{S}$ of p nodes to multicast any message to a set $\mathcal{R}$ of $p^2 - p$ nodes. During one time slot one node of set $\mathcal{S}$ multicasts a message to p nodes of set $\mathcal{R}$. Let the nodes be identified by $(0,0),(0,1),(0,2),\ldots(p-1,p-1)$. This set of nodes are partitioned into two disjoint sets *viz.* $\mathcal{R}$ and $\mathcal{S}$. $\mathcal{R} = \{(i,j) : 1 \leq i \leq p-1, 0 \leq j \leq p-1\}$ and $\mathcal{S} = \{(0,j) : 0 \leq j \leq p-1\}$. Thus, $|\mathcal{R} \cup \mathcal{S}| = p^2$.

Definition 7.5. A receiving sequence for a receiver node (i,j) is a set $U_{ij} = \{(\alpha,\beta,s) : 0 \leq \alpha \leq p-1, 0 \leq \beta \leq p-1, s \geq 0\}$ such that node (i,j) listens to channel β at time slot α at any session s.

Remark : The definition 7.5 of receiving sequence of any node is not to be

confused with the definition 7.4 of receiving sequence.

Now, we discuss the channel hopping algorithm for this scheme. We show how to find the receiving sequence of any node belonging to the receiver's group. As stated above, there are $p^2 - p$ receivers. These receivers only listen to channels for messages. The number of channels is p. Time is divided into consecutive sessions like the scheme-I in section 7.3. There are p timeslots in each session denoted by $0, 1, 2, \ldots, p-1$.

Algorithm 10 Transversal design based Channel Hopping Algorithm for Multicast Communication

Input: The set R of receivers.
Output: The frequency hopping scheme for a node r with id (i, j) in R.

```
for (s = 0; s < T; s++) do
    κ = Pseudorand(K, s)
    U_ij = φ
    for (x = 0; x < p; x++) do
        α = x, β = xi + j + κ  mod p
        U_ij = U_ij ∪ (α, β, s)
    end for
    U_ij is the receiving sequence of node r for session s.
end for
```

Algorithm 10 uses Transversal design for generating the receiving sequences of the receivers for any session. T is the lifetime of the key which is fed to the pseudo-random-integer generator. The Algorithm outputs the receiving sequences of all receivers for all sessions 1 upto session T after which the key $\mathcal{K}$ needs to be refreshed. The set U_{ij} stores the receiving sequences of a node with id (i, j) for all the p timeslots in a particular session. Each block of the Transversal design is used to generate the receiving sequence of one receiver node. The output of the pseudo-random-integer generator is used to randomize the receiving sequences of a user. The integer generated by the pseudo-random-integer generator is added to β which makes it impossible for the adversary to anticipate the id of the channel to which the receiver (i, j) listens at time slot α at any session s. Here, the sender nodes have no sending sequences unlike the first scheme. This is because we have assumed that the p many senders keep sending the same messages over all the p many available channels at any time slot. A receiver node reads the messages from one of the p senders that sends identical message at any particular time slot.

Example 7.2. *We now give an example of the channel hopping scheme discussed above. We choose a prime* $p = 5$. *The sender nodes are given by* $(i, j) : \forall i, j \in \{0, 1, \ldots, p-1\}$. *Let the* $Pseudorand()$ *function in Algorithm 10 returns the value* 3 *for some session* s. *Hence, the receiving sequence for the node* $(2, 3)$ *at session* s *will be given by* $U_{23} = \{(x, (2x + 3 + 3) \mod 5), 0 \leq x \leq 4\} = \{(0, 1, s), (1, 3, s), (2, 0, s), (3, 2, s), (4, 4, s)\}$. *It*

can be seen that there is no pair of 3-tuples (α_1, β_1, s) *and* (α_2, β_2, s) *such that* $\beta_1 = \beta_2$. *Lemma 7.10 gives a proof of this property of the receiving sequences. This property ensures that each sender listens to distinct channels at different time slots and thus attempt to evade a jammer.*

The proof of correctness of this algorithm is given in Section 7.5.1.

7.5 Analysis of our Schemes

We will first prove that the Algorithms 9 and 10 are correct. We then show that Scheme in Section 7.3.2 guarantees protection from the jammer.

7.5.1 Theoretical Analysis

Here we give a proof of correctness of the Algorithm 9 discussed in Section 7.3. We state the following theorem

Theorem 7.6. *Algorithm 9 is correct, which means that it satisfies the following properties:*

1. *No two nodes transmit messages through the same channel at the same time.*
2. *No two nodes listen to the same channel at the same time.*
3. *Two nodes rendezvous once in a particular session.*

We prove these properties of the channel hopping scheme below.

Lemma 7.7. *Algorithm 9 satisfies the condition that no two nodes transmit messages through the same channel at the same time.*

Proof. Let X'_m and X'_n be the sending sequence of node m and n respectively for some session s. Also, let the two orthogonal LS used in Algorithm 9 be L_1 and L_2 and let L_3 be the superposition of them. Let us assume that there exists a pair of nodes which transmit through the same channel at the same time. This implies that there exist tuples $(\alpha_1, \beta_1, i) \in X'_m$ and $(\alpha_2, \beta_2, j) \in X'_n$ where $(\alpha_1, \beta_1) = (\alpha_2, \beta_2)$. This further implies that for some $x, y \in \{0, 1, \ldots, N-1\}$, there exists $L_1(m,i) \mod N$ and $L_2(m,i) \mod N$, such that $\alpha_1 = x + L_1(m,i) \mod N, \beta_1 = y + L_2(m,i) \mod N$, and there exists $L_1(n,j)$

mod N and $L_2(n,j) \mod N$, such that $\alpha_2 = x + L_1(n,j) \mod N, \beta_2 = y + L_2(n,j) \mod N$. Here x, y are the output of $Pseudorand_1()$ and $Pseudorand_2()$ respectively.

Now since $(\alpha_1, \beta_1) = (\alpha_2, \beta_2)$, $x + L_1(m,i) \mod N = x + L_1(n,j) \mod N$. This implies that $L_1(m,i) = L_1(n,j) \mod N$ and $L_2(m,i) = L_2(n,j) \mod N$. This contradicts the fact that L_1 and L_2 are orthogonal. Hence, the first property is proved.

□

In similar way, we can prove that:

Lemma 7.8. *Algorithm 9 satisfies the condition that no two nodes listen to the same channel at the same time.*

Lemma 7.9. *Algorithm 9 guarantees that two nodes rendezvous once in a particular session.*

Proof. By the construction of LS for a row $i = \{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_N, \beta_N)\}$ and column $j = \{(\gamma_1, \delta_1), (\gamma_2, \delta_2), \ldots, (\gamma_N, \delta_N)\}$ intersect at when $\alpha_j = \gamma_i$ and $\beta_j = \delta_i$. At the s-th session, $x = Pseudorandom_1(\mathcal{K}, s)$ and $y = Pseudorandom_2(\mathcal{K}, s)$. So, node i sends signal to j on channel $\alpha_j + x$ at time $\beta_j + y$. Node j receives signal from node i in session s on channel $\gamma_i + x$ at time $\delta_i + y$. Since $\alpha_j = \gamma_i$ and $\beta_j = \delta_i$, nodes i and j rendezvous on channel $\alpha_j + x$ at time $\beta_j + y$ which is the same as $\gamma_i + x$ at time $\delta_i + y$, where the additions are done modulo N. □

From Lemmas 7.7, 7.8 and 7.9, it follows that Theorem 7.6 is correct.

We next prove the correctness of Algorithm 10. For this, we need to prove that all node belonging to the Receivers' group listens to distinct channels on different time slots. Since there are p time slots and p frequency channel, one receiver node listens to each of the p channels exactly once at a particular session. Lemma 7.10 proves this fact and hence proves the correctness of algorithm 10.

Lemma 7.10. *In the output of algorithm 10 for every receiver node* (i,j), $\forall \gamma \in \{0, 1, \ldots, p-1\}$, $\exists (\alpha, \beta, s) \in U_{ij}$ *such that* $\gamma = \beta$.

Proof. for all receiver node (i,j) $|U_{ij}| = p$. It will be sufficient to prove that for any two $(\alpha_1, \beta_1, s), (\alpha_2, \beta_2, s) \in U_{ij}$, $\beta_1 = \beta_2 \Rightarrow \alpha_1 = \alpha_2$.
$\beta_1 = \alpha_1 i + j + t \mod p$ and $\beta_2 = \alpha_2 i + j + t \mod p$. If $\beta_1 = \beta_2$ then $\alpha_1 i = \alpha_2 i$. Since $i \neq 0$, $\alpha_1 = \alpha_2$. □

Theorem 7.11. *The scheme in section 7.4.1 ensures that any two node* (i,j) *and* (i',j') *such that* $i \neq i'$ *must listen to the same channel at a unique time slot in every session.*

Proof. Let s be some session for which t is the output of $Pseudorand(\mathcal{K}, s)$. At time slot x the node (i, j) will be listening to channel $xi + j + t \mod p$ while the node (i', j') will be listening to channel $(xi' + j' + t)$. Then, these two nodes will be listening to the same channel iff $xi + j + t = xi' + j' + t \mod p$ or $x = (j' - j)(i - i')^{-1} \mod p$. If $i \neq i'$ then $(i - i')^{-1}$ exists an is unique. Hence, such a time slot will exists. □

We will next analyze the probability with which an attacker can jam a channel.

Theorem 7.12. *The scheme in Section 7.3 guarantees secure message exchange between a sender and a receiver with probability J/N in presence of an active jammer who jams J out of N channels.*

Proof. We first observe that all messages are encrypted and signed, which prevents an attacker to eavesdrop or change the message content, without being detected. From Lemma 7.9, we see that the two nodes rendezvous in finite amount of time, bounded by c. From the design, we note that the i-th sender rendezvous with the j-th receiver at time y on channel x. A jammer can jam any J channels at one time. If a sender and receiver are supposed to rendezvous on any of these channels, then they will not be able to communicate. The probability of such a situation is J/c. Since a new sending and receiving sequence is chosen at the next iteration, the jammer can jam it with a probability $J/c = J/N$. □

7.5.2 Comparison With Other Schemes

Here, we shall provide a theoretic comparison of our scheme with other similar schemes in existence which include the Quorum Rendezvous Channel Hopping scheme by Lee et al [36] and the broadcast Uncoordinated Frequency Hopping scheme in [57] by Strasser et al. Firstly, we shall be comparing our scheme with the QRCH scheme. For this purpose let us assume that there is a network of N nodes, N being an odd number. Let (N', κ) be the minimum difference set available where N' is the least number greater than or equal to N. The (N', κ)–difference set is used in the construction of the quorum system for channel hopping in the network as discussed in [36]. The QRCH scheme requires as much as κ^2 time slots for a rendezvous. In order to compare this scheme with our scheme we should assume that whenever the two nodes meet, the first half of the time slot should be used for message transmission from the first node to the second and the other half for the message transmission from the second node to the first one. Thus, the size of a time slot will be double of that of our scheme. Let, ϵ be the length of a time slot in our scheme. Then for the current scheme is should be 2ϵ. Now if an (N', κ) difference set is used to obtain the cyclic quorum system then κ should be such

that $\sqrt{N'} < \kappa \leq N'$ according to Jiang et al. [32]. So, the rendezvous time should be in between $2N'\epsilon$ and $2N'^2\epsilon$. In our scheme the time to rendezvous is $N\epsilon$. Since, $N \leq N'$, our scheme has much less time to rendezvous than the QRCH scheme.

Now, we shall compare this scheme with the traditional frequency hopping scheme [53]. Here, we assume that each node uses a different pseudorandom generator to generate the PN sequence. This ensures that all nodes do not end up listening to or sending messages to the same channel at the same time slot. Under this circumstance, two nodes will be able to communicate if they select the same channel at the same time slot and no other node selects the same frequency channel for that time slot. Let, there be N nodes and c frequency channels in the network. In absence of jammer, the probability of an exclusive rendezvous between a pair of nodes (i.e. only those two nodes meet on a channel and the rest of the $n-2$ nodes select other channels) is given by $P(R) = \frac{1}{c}(1-\frac{1}{c})^{N-2} = f(c)$ (Let). So, $f'(c) = \frac{1}{c^2}(1-\frac{1}{c})^{N-3}(\frac{N-1}{c} - 1) \geq 0$ for $N \geq 3$. f reaches maxima at $c = N-1$. For the sake of fair comparison, we choose a network where the number of channels $c = N-1$. Hence, the highest probability of successful rendezvous that can be achieved using this scheme is given by $P(R) = \frac{1}{c}(1-\frac{1}{c})^{c-1}$ in a time slot. So, rendezvous of two nodes on an exclusive channel can be seen as a binomial experiment with probability of success equal to $\frac{1}{c}(1-\frac{1}{c})^{c-1}$. In N time slots the expected number of rendezvous is $\frac{N}{c}(1-\frac{1}{c})^{c-1} = \frac{c+1}{c}(1-\frac{1}{c})^{c-1} = g(c)$ (Let). g is a decreasing function on $c, c > 1$.

$$\lim_{c\to+\infty} g(c) = \lim_{c\to+\infty} \frac{c+1}{c}(1-\frac{1}{c})^{c-1} = \frac{1}{e}.$$

Since, $N \geq 3$, $c \geq 2$. At $c = 2$, $g(2) = \frac{3}{4}$. Hence, $\frac{3}{4} > g(c) > \frac{1}{e}$. Therefore, the expected number of rendezvous of two nodes in N time slots is between 0.368 and 0.750 in this scheme, whereas it is exactly 2 in our scheme-I.

In section 7.4, a multicast scheme is discussed. In this scheme there is a set of p senders and a set of $p^2 - p$ receivers. The number of available channels is p. Let us assume a scenario where only one sender is active and the rest of the $p-1$ senders are inactive. All the $p^2 - p$ receivers are listening to the broadcast from the very sender. Now, our multicast communication protocol ensures a rendezvous between a sender and a receiver in a session consisting of p time slots. We assume a comparable scenario for the broadcast UFH [57] scheme where each receiver listens to a single randomly chosen channel out of the p available channels at any time slot. Also, the sender transmits its messages on a single randomly chosen channel at any time slot. Therefore, the modelling difference between the the two scenarios(our model and the UFH model) in this setting is that in the broadcast UFH, the communicating devices randomly choose a single channel at each time slot for sending/reading message.

Let, E_i denotes the event that a receiver i, $i \in \{1, 2, \ldots, p^2 - p\}$ rendezvous with the

sender in a certain session s.
Also let, $E_i^j, 0 \leq j \leq p-1$ denotes the event that a receiver i, $i \in \{1, 2, \ldots, p^2 - p\}$ rendezvous with the sender in a certain time slot j in a certain session s.

$$E_i = \bigcup_{j=0}^{p-1} E_i^j.$$

In our scheme $P(E_i) = 1$. For the broadcast UFH scheme, $P(E_i) = 1 - P(E_i^c) = 1 - P((\bigcup_{j=0}^{p-1} E_i^j)^c)$. Since, in UFH the communicating devices select channels randomly in each time slot, E_i^j's are independent. Hence,

$$P(E_i^c) = 1 - P(E_i) = P((\bigcup_{j=0}^{p-1} E_i^j)^c) = P(\bigcap_{j=0}^{p-1} (E_i^j)^c) = \prod_{j=0}^{p-1} P((E_i^j)^c) = \prod_{j=0}^{p-1} (1 - P(E_i^j)).$$

Since, in this setting of UFH the receiver randomly chooses a single channel out of p channels to read in each time slot, $P(E_i^j) = 1/p$. So,

$$P(E_i) = 1 - \prod_{j=0}^{p-1} (1 - \frac{1}{p}) = 1 - (1 - \frac{1}{p})^p.$$

$$\lim_{p \to +\infty} P(E_i) = 1 - \lim_{p \to +\infty} (1 - \frac{1}{p})^p = 1 - \frac{1}{e} \approx 0.632$$

Now, for a normal network it could be assumed that the number of communicating devices should be more than 3, or $p > 3$. Since, p is a prime number, p should be at least 5. For $p = 5$, $P(E_i) = 1 - (1 - \frac{1}{5})^5 \approx 0.672$.

Hence, the probability that a particular receiver would meet with a particular sender in a certain time slot in absence of jammer is equal to 1 for our scheme, whereas it is between 0.632 and 0.672 for the broadcast UFH scheme.

In the presence of jammer, $P(E_i) = 1 - k$ for our scheme where k is the jamming probability. For the broadcast UFH scheme,

$$P(E_i^j) = \frac{1-k}{p}.$$

Hence, $P(E_i) = 1 - (1 - \frac{1-k}{p})^p$.

$$\lim_{p \to +\infty} P(E_i) = 1 - \lim_{p \to +\infty} (1 - \frac{1-k}{p})^p = 1 - \frac{1}{e^{1-k}}.$$

Hence, for high value of p, the probability of jam-free rendezvous is approximately equal to $1 - \frac{1}{e^{1-k}}$ for the broadcast UFH scheme.

It can be proved that $1 - (1 - \frac{1-k}{p})^p < 1 - k$, $0 \leq k < 1$ (Proof given below).

Hence, in presence of the jammer our scheme ensure higher probability of message delivery than the broadcast UFH scheme.

Proof of the fact: Let, $f(k) = (1 - \frac{1-k}{p})^p - k$. Hence, $f(1) = 0$.

$$f'(k) = (1 - \frac{1-k}{p})^{p-1} - 1.$$

For $p \geq 2$ and $0 \leq k < 1$, $f'(k) < 0$. Hence, for $0 \leq k < 1$, f is a decreasing function on k. Therefore, for $0 \leq k < 1$, $f(k) > f(1) = 0$. Whence we get, $k < (1 - \frac{1-k}{p})^p$ or, $1 - (1 - \frac{1-k}{p})^p < 1 - k$.

Now, let us calculate the expected amount of time required to transmit a message in both our scheme and the broadcast UFH scheme. We assume that the message is fragmented using erasure coding technique [47], [57] as discussed in section 7.2.4. We assume that in both the schemes one session consists of p time slots. Let, l be the number of fragments of a particular message fragmented using erasure coding technique. The sender transmits each fragment over the p many time slots in a session i.e in all the p time slots in first session, it would send the first fragment. Then in the next session, it would send the second fragment on all the p time slots and so on. A receiver can reconstruct the message from the fragments if only it can collect at least l message fragments. In our scheme the expected number sessions required to get l many message fragments when the jamming probability is k is given by

$$E(T_k) = \sum_{r=l}^{\infty} rP(T_k = r) = \sum_{r=l}^{\infty} r\binom{r-1}{l-1}(1-k)^l k^{r-l} = \frac{l}{1-k}$$

For the broadcast UFH scheme, this is given by

$$E(T_k) = \frac{l}{1 - (1 - \frac{1-k}{p})^p}$$

For $p \geq 2$, $1-(1-\frac{1-k}{p})^p < 1-k$. So, the amount of time needed to transfer the message is less than the broadcast UFH scheme.

We give a comparison of our schemes discussed in chapter 6 and 7 with channel hopping schemes of similar kind in table 2.2.

7.5.3 Experimental Analysis

We now study the performance of our scheme with respect to other schemes in literature. So far it is clear that our scheme provides a better bound in rendezvous time than the scheme in [36]. Again, in our scheme-I the adversary cannot anticipate the channel that is being used by a pair of nodes for communication. Moreover, since the nodes send/listen to randomly selected channels in [53], it is uncertain when the two communicating nodes will rendezvous. On the contrary, our scheme ensures a rendezvous within an average time bound of $O(c)$ (c is the number of channels). Hence, any pair of communicating nodes will surely meet on some channel within a fixed time. Figure 7.1 shows a graphical comparison of the classical frequency hopping scheme and our scheme in section 7.3. Here, we choose a network having 23 nodes. A session consists of 23 time slots. In the CFH scheme each node uses a different PN sequence for hopping to a frequency channel in a time slot. We simulated the performance of the two schemes for this network and have plotted the experimental data. The horizontal axis corresponds to the probability that a certain channel is being jammed by the adversary. The vertical axis corresponds to the probability of successful transmission of a message packet in a session. Figure 7.1 shows that the proposed scheme can achieve lower probability of message loss than the well known CFH scheme.

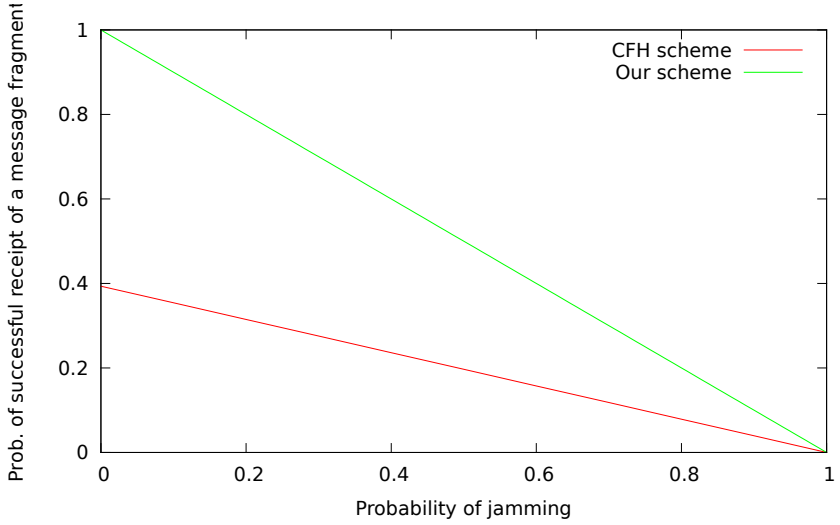

FIGURE 7.1: Graphical comparison of the probability of successful packet transmission of of our scheme and the classical frequency hopping scheme in [53] in a session containing 23 time slots. The size of the network is 23 for both the schemes.

Next, we do a comparative study of the time required for each node to rendezvous with other nodes for our scheme, the classical frequency hopping scheme [53] and [36]. It can be noticed that our scheme ensures an $O(c)$ time for the rendezvous of any pair of nodes. On the other hand, the scheme in [36] offers a maximum time to rendezvous of order $O(c^2)$ ($c = N =$ no. of nodes). Similarly, the TTR for the CFH scheme in

[53] is discussed in section 7.5.2. Figure 7.2 shows graphical comparison of the three schemes. This graph represents the experimental data obtained through simulation of the performance of these three schemes. The continuous lines corresponds to our proposed scheme in section 7.3 and the CFH scheme and the discrete points correspond to the scheme by Lee et al. in [36]. It should be noted that the quorum based scheme in [36] requires a difference set for constructing the hopping sequences. Since, difference sets are available only at discrete points, the performance can only be evaluated at some discrete points. It can be seen that our scheme offers a lower time to rendezvous as compared to the other two schemes. It can also be noticed that at some points the average rendezvous time for the scheme in [36] appears very close to that of our scheme whereas at other points they appear far away from that of our scheme. This is due to the fact that the scheme in [36] yields a maximum time to rendezvous of order $O(c^2)$ but the same for our scheme is always of order $O(c)$. So, our scheme is much better than [36] as it always ensures a time to rendezvous linear in the size of the network. Also, the time to rendezvous of our scheme is better than CFH scheme as discussed in section 7.5.2.

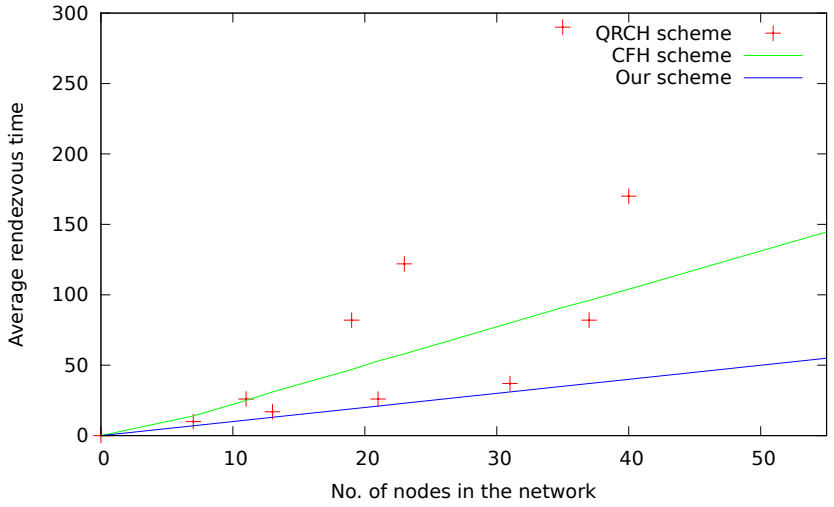

FIGURE 7.2: Graphical comparison of the average time taken for the rendezvous of a pair of nodes for three different frequency hopping schemes. QRCH denotes the quorum based channel hopping scheme in [36] and CFH stands for the classical frequency hopping scheme in [53]. The graph shows that in our scheme the time to rendezvous is less than that of QRCH and CFH.

Again, the performance of the channel hopping scheme in section 7.4.1 is compared with that of the UFH scheme [57, 67] in figure 7.3. Here, we consider fragmentation of a message as discussed in section 7.2.4. Here the X-axis shows the total number of fragments of the original message that must be correctly transmitted to the receiver so that the receiver can successfully reconstruct the original message from the fragments. The Y-axis corresponds to the jammer's strength in terms of number of channels it can

jam simultaneously. The vertical axis gives the total number of transmissions required to successfully transmit the minimum number of message fragments from which the receiver can successfully reconstruct the original message. In presence of jammer all transmission attempts of the senders will not be successful and hence some fragments need to be retransmitted. The vertical axis is showing the expected number of transmissions required to communicate all the fragments shown along the X-axis. This graph is drawn by plotting the experimental data obtained through simulating the performance of the two schemes. We used C program to simulate the performance of these schemes. The upper surface in the three dimensional figure 7.3 corresponds to the number of message transmissions required for the UFH scheme for different size of message packets and different strength of the jammer. Similarly, the lower surface represents the number of transmissions required for our scheme. Hence, figure 7.3 shows that the scheme in section 7.4.1 requires lesser number retransmissions than the UFH scheme in [57]. In other words, the probability that a particular message-fragment will be received undamaged by the receivers is higher in our scheme than the UFH scheme.

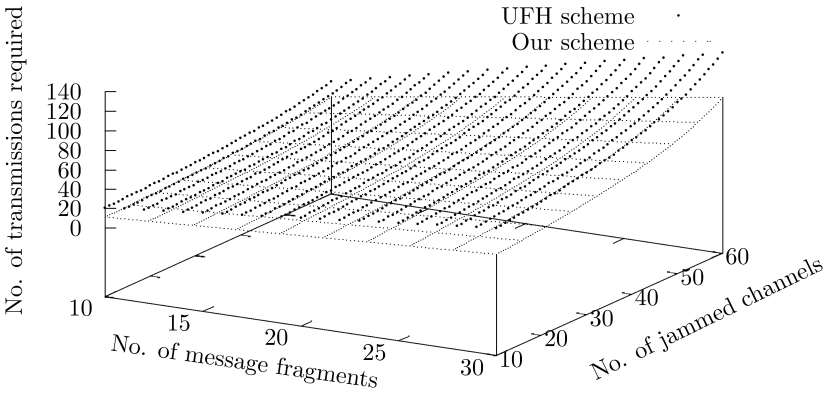

FIGURE 7.3: Graphical comparison of the performances of the channel hopping scheme discussed in section 7.4.1 with the UFH scheme. The graph shows the number of transmissions required for message fragments for different jamming strength of the attacker. X-axis corresponds to the total number of fragments of the original message. Y-axis corresponds to the number of jammed channels. The vertical axis shows the number of transmissions required for communicating the fragmented message. Here we assume that the sender/receiver can send/receive messages on 8 channels simultaneously.

Figure 7.3 shows how the jamming strength affects the performance of the scheme in section 7.4.1 for different number of message fragments. Similarly, figure 7.4 depicts the associativity of the jamming strength, number of nodes and the jamming probability in the scheme described in section 7.3. The X-axis and the Y-axis corresponds to number of nodes and the number of channels the attacker can jam respectively. The vertical axis gives the probability that a message-fragment will be jammed by the attacker. It can be seen that if the number of nodes increase and the jamming strength remains unaltered, then it shows that the probability of jamming decreases steadily.

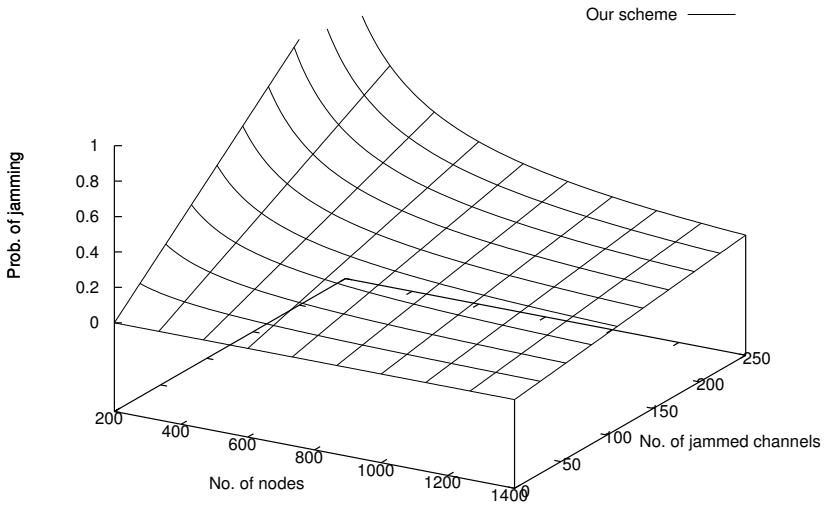

FIGURE 7.4: Graphical presentation of dependence of parameters on our proposed scheme.

7.6 Concluding Remarks

In this chapter we discussed two channel hopping schemes for wireless communication using combinatorial designs. This work opens a new area of research in the field of frequency channel hopping. In the first scheme of section 7.3 two secret keys are used to randomize the hopping of nodes through different channels. If an attacker gets to know the keys then it will be able to disrupt the communication of some of the nodes depending upon its jamming capability. The question is whether we can design a combinatorial hopping scheme that does not require a pre-shared key? Is it possible to have distinct keys stored in different nodes and still make these nodes hop over frequency channels independently without any collision? If such a channel hopping scheme can be found then if an adversary gets to know one key or some keys by compromising some nodes then other keys will remain unaffected and hence nodes who use those keys for channel hopping can continue to do so. We leave it as an open problem to find such a channel hopping scheme. In our first channel hopping scheme, every pair of nodes may not be communicating in a particular session. Hence, at any particular time slot there could be some channels not used by any user/node. This idle channels can be given to some secondary users. In other words, we can develop a channel hopping scheme for the primary users of the radio networks and can let the secondary users use the channel only when a channel is idle. This opens a new field of research on the application of our channel hopping algorithm in cognitive radio networks.

The second scheme discussed in this chapter deals with multicast communication. In this

scheme a set of nodes send messages to a distinct set of nodes in presence of a jammer. We have used transversal design for developing the hopping scheme. There are many other designs that could be used in this scenario. For example, if the nodes have multiple transceivers for listening and sending messages over multiple channels then symmetric balanced incomplete block design can be used for broadcast communication. Here the blocks can be replaced by channels. The sender and the receivers can randomly choose blocks and send/listen to all the channels in the block. This will ensure that there will always be a common channel between the sender and a receiver. We leave the application of other designs including symmetric BIBD as a scope for future research.

Chapter 8

Conclusion & Scope of Further Research

There are many research problems that can be investigated in future. Key predistribution in wireless networks have been on the focus of many research works. A lot of work has been done in this field using some of the combinatorial designs available in literature e.g. Balanced incomplete block design, transversal design, Partially Balanced Incomplete Block Design, Transversal Design, Generalized Quadrangles, 3-designs, Costas Arrays, Unbalanced design, Affine geometry etc. Yet a lot of designs remain untouched.

Throughout this thesis, we have taken $E(s)$ and $V(s)$ to be the measure of resiliency of a key predistribution. In future significant research work can be done to explore and invent new measures for evaluating performance of a key predistribution scheme. One such measure could be the diameter of the biggest component of the remaining key graph after some nodes are compromised. A longer path between two nodes requires more energy for communication. So, it is meaningful to study the key graph of a wireless sensor network after some nodes have been compromised and propose suitable key predistribution schemes accordingly that keep the diameter of the key graph shorter. Also, both $E(s)$ and $V(s)$ depend on the choice of s. This makes the measures non-comprehensive as it is not possible to estimate the values of these measures for higher values of s from lower values of them. Again, for calculating $E(s)$ and $V(s)$, we randomly compromise a number of nodes. This can be improved by observing the realistic scenario of the network. It can be observed that the attacker may not randomly compromise nodes. The nodes which are in close proximity to it are more prone to be captured. Thus, we could redefine those measures taking into account the fact that there is a center of attack in the network and those nodes whose physical position is close to the attack center have a much larger probability of being captured rather than those who lie far

away. We can investigate the resiliency of the existing key predistribution schemes with respect to this measure. We also can try to apply key predistribution techniques for other ad-hoc networks like Internet of Things (IoT), Vehicular Networks etc. We also can study heterogeneous key predistribution in a clustered network. In such scenarios, the network is divided into clusters with each cluster having a cluster-head(CH). Nodes within a cluster can communicate with the Base Station or with nodes of other clusters with the help of the CH. We can use symmetric key among nodes within a cluster and public key among the CHs and study the performance of the network.

In Chapter 4, we proposed a key predistribution scheme using the scheme of Blom [10] and SBIBD. We then used this scheme in grid group deployment of wireless sensors. We applied hybrid design for key predistribution. In this chapter, we used Blom's scheme [10] as underlying technique. Instead of this, someone could attempt using any polynomial based scheme and can compare its performance with our scheme.

In chapter 6, we discuss a jamming resistant wireless communication scheme. In this scheme a pair of communicating devices can hop over multiple wireless frequency channels in order to evade the jammer. This work can further be extended to broadcast communication where a transmitter sends wireless messages to multiple recipients. We applied Steiner Triple System and Transversal design in this context. Symmetric Balanced Incomplete Block Designs can also be used for this purpose, as each pair of blocks in an SBIB design surely contain a shared variety, a condition necessary to ensure a rendezvous of communicating devices on every time slot.

In chapter 7, we discussed two jamming resistant wireless communication scheme. In the first scheme a group of users can communicate with each other in the presence of the jammer. The other scheme is meant for multicast communication. Like the previous scheme, this scheme also makes use of combinatorial design. But this scheme assumes requirement of a shared secret key. Attempts could be taken to communicate the secret key to the individual nodes on the fly and in presence of the jammer. This will surely serve as a substantial work in this area as it would break the circular dependency between the anti-jamming communication and the establishment of secret key in presence of the jammer.

Bibliography

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38:393–422, 2002.

[2] Ian F. Akyildiz, Xudong Wang, and Weilin Wang. Wireless mesh networks: a survey. *Computer Networks and ISDN Systems*, 47(4):445–487, 2005.

[3] Samiran Bag. A new key predistribution scheme for grid-group deployment of wireless sensor networks. *Ad Hoc & Sensor Wireless Networks (To appear)*, 2013.

[4] Samiran Bag and Bimal Roy. A new key predistribution scheme for general and grid-group deployment of wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2013(145), 2013.

[5] Samiran Bag and Bimal Roy. Two channel hopping schemes for jamming resistant wireless communication. In *WiMob*, pages 659–666. IEEE, 2013.

[6] Samiran Bag and Sushmita Ruj. Key distribution in wireless sensor networks using finite affine plane. In *AINA Workshops*, pages 436–441. IEEE Computer Society, 2011.

[7] Samiran Bag, Sushmita Ruj, and Bimal K. Roy. Jamming resistant schemes for wireless communication: A combinatorial approach. In Aditya Bagchi and Indrakshi Ray, editors, *ICISS*, volume 8303 of *Lecture Notes in Computer Science*, pages 43–62. Springer, 2013.

[8] Niko Bari and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *EUROCRYPT*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–494. Springer, 1997.

[9] Simon Blackburn, Tuvi Etzion, Keith Martin, and Maura Paterson. Efficient key predistribution for grid-based wireless sensor networks. In *ICITS, LNCS 5155*, pages 54–69, 2008.

[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 Ernest F. Brickell, editor, *CRYPTO*, volume 740 of *Lecture Notes in Computer Science*, pages 471–486. Springer, 1992.

[12] Seyit A. Camtepe and Blent Yener. Key distribution mechanisms for wireless sensor networks: A survey, 2005. Technical Report TR-05-07 Rensselaer Polytechnic Institute,Computer Science Department, March 2005.

[13] Ran Canetti, Juan A. Garay, Gene Itkis, Daniele Micciancio, Moni Naor, and Benny Pinkas. Multicast security: A taxonomy and some efficient constructions. In *INFOCOM*, pages 708–716, 1999.

[14] Seyit Ahmet Çamtepe and Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. In *ESORICS*, volume 3193 of *Lecture Notes in Computer Science*, pages 293–308. Springer, 2004.

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

[16] Dibyendu Chakrabarti, Subhamoy Maitra, and Bimal K. Roy. A key predistribution scheme for wireless sensor networks: Merging blocks in combinatorial design. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *ISC*, volume 3650 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 2005.

[17] Dibyendu Chakrabarti and Jennifer Seberry. Combinatorial structures for design of wireless sensor networks. In Jianying Zhou, Moti Yung, and Feng Bao, editors, *ACNS*, volume 3989 of *Lecture Notes in Computer Science*, pages 365–374, 2006.

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

[19] Haowen Chan, Adrian Perrig, and Dawn Xiaodong Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, pages 197–213. IEEE Computer Society, 2003.

[20] Waltenegus Dargie and Christian Poellabauer. *Fundamentals of Wireless Sensor Networks: Theory and Practice*. John Wiley and Sons, 2010.

[21] Farshid Delgosha, Erman Ayday, and Faramarz Fekri. Mkps: a multivariate polynomial scheme for symmetric key-establishment in distributed sensor networks. In

Mohsen Guizani, Hsiao-Hwa Chen, and Xi Zhang, editors, *IWCMC*, pages 236–241. ACM, 2007.

[22] Farshid Delgosha and Faramarz Fekri. Key pre-distribution in wireless sensor networks using multivariate polynomials. In *SECON*, pages 118–129. IEEE, 2005.

[23] Junwu Dong, Dingyi Pei, and Xueli Wang. A key predistribution scheme using 3-designs. In *INSCRYPT*, 2007.

[24] 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.

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

[26] 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.

[27] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In Vijayalakshmi Atluri, editor, *ACM Conference on Computer and Communications Security*, pages 41–47. ACM, 2002.

[28] Ramakrishna Gummadi, David Wetherall, Ben Greenstein, and Srinivasan Seshan. Understanding and mitigating the impact of rf interference on 802.11 networks. In *SIGCOMM*, pages 385–396. ACM, 2007.

[29] Dijiang Huang and Deep Medhi. Secure pairwise key establishment in large-scale sensor networks: An area partitioning and multigroup key predistribution approach. *TOSN*, 3(3), 2007.

[30] Dijiang Huang, Manish Mehta, Deep Medhi, and Lein Harn. Location-aware key management scheme for wireless sensor networks. In Sanjeev Setia and Vipin Swarup, editors, *SASN*, pages 29–42. ACM, 2004.

[31] Sajid Hussain, Firdous Kausar, and Ashraf Masood. An efficient key distribution scheme for heterogeneous sensor networks. In *Proceedings of the 2007 international conference on Wireless communications and mobile computing*, IWCMC '07, 2007.

[32] Jehn-Ruey Jiang, Yu-Chee Tseng, Chih-Shun Hsu, and Ten-Hwang Lai. Quorum-based asynchronous power-saving protocols for ieee 802.11 ad hoc networks. *MONET*, 10(1-2):169–181, 2005.

[33] R. Kalindi, Rajgopal Kannan, S. Sitharama Iyengar, and Arjan Durresi. Sub-grid based key vector assignment: A key pre-distribution scheme for distributed sensor networks. *Int. J. Pervasive Computing and Communications*, 2(1):35–45, 2006.

[34] David C Lay. *Linear Algebra and Its Applications*. Addison Wesley, 3rd edition, August 22, 2005.

[35] Loukas Lazos, Sisi Liu, and Marwan Krunz. Mitigating control-channel jamming attacks in multi-channel ad hoc networks. In David A. Basin, Srdjan Capkun, and Wenke Lee, editors, *WISEC*, pages 169–180. ACM, 2009.

[36] Eun-Kyu Lee, Soon-Young Oh, and Mario Gerla. Randomized channel hopping scheme for anti-jamming communication. In *Wireless Days*, pages 1–5. IEEE, 2010.

[37] Jooyoung Lee and Douglas R. Stinson. Deterministic key predistribution schemes for distributed sensor networks. In Helena Handschuh and M. Anwar Hasan, editors, *Selected Areas in Cryptography*, volume 3357 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2004.

[38] Jooyoung Lee and Douglas R. Stinson. A combinatorial approach to key predistribution for distributed sensor networks. In *IEEE Wireless Communications and Networking Conference, WCNC 2005, New Orleans, LA, USA*, 2005.

[39] Jooyoung Lee and Douglas R. Stinson. A combinatorial approach to key predistribution for distributed sensor networks. In *IEEE Wireless Communications and Networking Conference, WCNC 2005, New Orleans, LA, USA*, 2005.

[40] 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.

[41] Mingyan Li, Iordanis Koutsopoulos, and Radha Poovendran. Optimal jamming attacks and network defense policies in wireless sensor networks. In *INFOCOM*, pages 1307–1315. IEEE, 2007.

[42] Donggang Liu and Peng Ning. Establishing pairwise keys in distributed sensor networks. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM Conference on Computer and Communications Security*, pages 52–61. ACM, 2003.

[43] Donggang Liu and Peng Ning. Location-based pairwise key establishments for static sensor networks. In Sanjeev Setia and Vipin Swarup, editors, *SASN*, pages 72–82. ACM, 2003.

[44] Donggang Liu and Peng Ning. Improving key predistribution with deployment knowledge in static sensor networks. *TOSN*, 1(2):204–239, 2005.

[45] Donggang Liu, Peng Ning, and Wenliang Du. Group-based key pre-distribution in wireless sensor networks. In Markus Jakobsson and Radha Poovendran, editors, *Workshop on Wireless Security*, pages 11–20. ACM, 2005.

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

[47] Michael Luby. Lt codes. In *FOCS*, pages 271–280. IEEE Computer Society, 2002.

[48] Florence Jessie MacWilliams and Neil J. A. Sloane. *The theory of error correcting codes*. Northland Holland, 1988.

[49] Keith M. Martin, Maura B. Paterson, and Douglas R. Stinson. Key predistribution for homogeneous wireless sensor networks with group deployment of nodes. *IACR Cryptology ePrint Archive*, 2008:412, 2008.

[50] Carl D Meyer. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics (SIAM), February 15, 2001.

[51] Chris J. Mitchell and Fred Piper. Key storage in secure networks. *Discrete Applied Mathematics*, 21(3):215–228, 1988.

[52] Abedelaziz Mohaisen, YoungJae Maeng, and DaeHun Nyang. On grid-based key pre-distribution: Toward a better connectivity in wireless sensor network. In Takashi Washio, Zhi-Hua Zhou, Joshua Zhexue Huang, Xiaohua Hu, Jinyan Li, Chao Xie, Jieyue He, Deqing Zou, Kuan-Ching Li, and Mário M. Freire, editors, *PAKDD Workshops*, volume 4819 of *Lecture Notes in Computer Science*, pages 527–537. Springer, 2007.

[53] Vishnu Navda, Aniruddha Bohra, Samrat Ganguly, and Dan Rubenstein. Using channel hopping to increase 802.11 resilience to jamming attacks. In *INFOCOM*, pages 2526–2530. IEEE, 2007.

[54] Lan Nguyen. Accumulators from bilinear pairings and applications. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 275–292. Springer, 2005.

[55] Roberto Di Pietro, Luigi V. Mancini, and Alessandro Mei. Energy efficient node-to-node authentication and communication confidentiality in wireless sensor networks. *Wireless Networks*, 12(6):709–721, 2006.

[56] Christina Pöpper, Mario Strasser, and Srdjan Capkun. Jamming-resistant broadcast communication without shared keys. In *USENIX Security Symposium*, pages 231–248. USENIX Association, 2009.

[57] Christina Pöpper, Mario Strasser, and Srdjan Capkun. Anti-jamming broadcast communication using uncoordinated spread spectrum techniques. *IEEE Journal on Selected Areas in Communications*, 28(5):703–715, 2010.

[58] Amar Rasheed and Rabi N. Mahapatra. Key predistribution schemes for establishing pairwise keys with a mobile sink in sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, 22(1):176–184, 2011.

[59] Kay Römer and Friedemann Mattern. The design space of wireless sensor networks. *Wireless Commun.*, 11(6):54–61, December 2004.

[60] Sushmita Ruj and Bimal K. Roy. Key predistribution using partially balanced designs in wireless sensor networks. In Ivan Stojmenovic, Ruppa K. Thulasiram, Laurence Tianruo Yang, Weijia Jia, Minyi Guo, and Rodrigo Fernandes de Mello, editors, *ISPA*, volume 4742 of *Lecture Notes in Computer Science*, pages 431–445. Springer, 2007.

[61] Sushmita Ruj and Bimal K. Roy. Key predistribution schemes using codes in wireless sensor networks. In Moti Yung, Peng Liu, and Dongdai Lin, editors, *Inscrypt*, volume 5487 of *Lecture Notes in Computer Science*, pages 275–288. Springer, 2008.

[62] Sushmita Ruj and Bimal K. Roy. Key predistribution using combinatorial designs for grid-group deployment scheme in wireless sensor networks. *TOSN*, 6(1), 2009.

[63] 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. IEEE Computer Society, 2005.

[64] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

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

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

[67] Mario Strasser, Christina Pöpper, and Srdjan Capkun. Efficient uncoordinated fhss anti-jamming communication. In *MobiHoc*, pages 207–218. ACM, 2009.

[68] Mario Strasser, Christina Pöpper, Srdjan Capkun, and Mario Cagalj. Jamming-resistant key establishment using uncoordinated frequency hopping. In *IEEE Symposium on Security and Privacy*, pages 64–78. IEEE Computer Society, 2008.

[69] Anne Penfold Street and Deborah J. Street. *Combinatorics of Experimental Design*. Clarendon Press, Oxford, 1987.

[70] Patrick Tague, Mingyan Li, and Radha Poovendran. Probabilistic mitigation of control channel jamming via random key distribution. In *in Proc. 18th Annual IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC'07)*, 2007.

[71] Patrick Tague and Radha Poovendran. A canonical seed assignment model for key predistribution in wireless sensor networks. *TOSN*, 3(4), 2007.

[72] Patrick Tague and Radha Poovendran. Modeling adaptive node capture attacks in multi-hop wireless networks. *Ad Hoc Networks*, 5(6):801–814, 2007.

[73] Neng Chung Wang and Hong Li Chen. Improving pairwise key predistribution in wireless sensor networks. In *Advances in Intelligent Systems and Applications*, volume 1, pages 521–530. Springer Berlin Heidelberg, 2012.

[74] Ruizhong Wei and Jiang Wu. Product construction of key distribution schemes for sensor networks. In Helena Handschuh and M. Anwar Hasan, editors, *Selected Areas in Cryptography*, volume 3357 of *Lecture Notes in Computer Science*, pages 280–293. Springer, 2004.

[75] Anthony D. Wood, John A. Stankovic, and Gang Zhou. Deejam: Defeating energy-efficient jamming in ieee 802.15.4-based wireless networks. In *SECON*, pages 60–69. IEEE, 2007.

[76] Yang Xiao, Venkata Krishna Rayi, Bo Sun, Xiaojiang Du, Fei Hu, and Michael Galloway. A survey of key management schemes in wireless sensor networks. *Computer Communications*, 30(11-12):2314–2341, 2007.

[77] Wenyuan Xu, Wade Trappe, and Yanyong Zhang. Channel surfing: defending wireless sensor networks from interference. In Tarek F. Abdelzaher, Leonidas J. Guibas, and Matt Welsh, editors, *IPSN*, pages 499–508. ACM, 2007.

[78] Wenyuan Xu, Timothy Wood, Wade Trappe, and Yanyong Zhang. Channel surfing and spatial retreats: defenses against wireless denial of service. In Markus Jakobsson and Adrian Perrig, editors, *Workshop on Wireless Security*, pages 80–89. ACM, 2004.

[79] Osman Yagan and Armand M. Makowski. Connectivity results for sensor networks under a random pairwise key predistribution scheme. In *ISIT*, pages 1797–1801. IEEE, 2012.

[80] Mohamed F. Younis, Kajaldeep Ghumman, and Mohamed Eltoweissy. Location-aware combinatorial key management scheme for clustered sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, 17(8):865–882, 2006.

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

[82] 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.

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

[84] 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. IEEE Computer Society, 2003.