Key Pre-distribution Schemes in Distributed Sensor Networks

A thesis presented
by

Xiaonan He

to
The Department of Computer Science
in partial fulfillment of the requirements
for the degree of
Master of Science
in the subject of

Computer Science

The University of Manitoba
Winnipeg, Manitoba
June 2006

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES
*****
COPYRIGHT PERMISSION

Key Pre-distribution Schemes in Distributed Sensor Networks

BY

Xiaonan He

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

OF

MASTER OF SCIENCE

Xiaonan He © 2006

Thesis advisor

Pak Ching Li

Author

Xiaonan He

Key Pre-distribution Schemes in Distributed Sensor Networks

Abstract

Distributed Sensor Networks (DSN) are ad-hoc networks consisting of sensor nodes with limited computational and communication capabilities, and limited power supply. They are often deployed in hostile environments for monitoring and data collection purposes, where node capture by non-friendly elements is a possibility. In order to protect the network from eavesdropping from either active or passive attacks, cryptography is used to secure communications in the network. One of the issues when using cryptography is the distribution of keys. For DSNs, Key Pre-distribution Schemes (KPS) are the preferred technique for distributing keys to sensor nodes. Thesis contains a survey of existing KPSs in DSNs. In addition, I studied the product construction as a methodology for generating new KPS from existing ones. In particular, I will propose the μ-CID Blom's schemes. The proposed scheme achieves enhanced security in terms of the resiliency against node capture.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to take this opportunity to say thanks, first of all, to my supervisor Dr. Ben Li. Without his advice and encouragement this thesis would never exist.

I would also like to thank other committee members, Dr. John van Rees and Dr. David S. Gunderson, for their reading of this thesis and suggestions for improving it.

I wish to thank all my friends that I have known these years, in Winnipeg, Fredericton and St. John, for all the friendships we have. I am also grateful to all my best friends in China for their emotional support.

At last, and most importantly, I would never forget my parents. Their love, understanding and encouragement are always with me, no matter where I was, where I am, where I will be.

Chapter 1

Introduction to Key Pre-distribution Schemes in Sensor Networks

In this chapter, I introduce some fundamental concepts and terms in the research area of designing and implementing key pre-distribution schemes for sensor networks. I start with introducing sensor networks and their constraints. In addition, I will explain why key pre-distribution schemes are the only feasible type of key distribution solution for sensor networks. At the end of this chapter, I will show the organization of this thesis.

1.1 What is a Sensor Network?

A *Sensor Node* is an electrical device that retrieves environmental information. A *sensor network* [Akyildiz et al., 2002] is a computer network consisting of a large number of tiny, inexpensive, battery powered *sensor nodes* that are deployed and distributed into an area to collect desired information.

Each sensor node is composed of four basic components: a battery power supply, a microprocessor, a sensing unit and a communication unit [Çamtepe and Yener, 2005]. Other application-dependent components may exist in various applications, such as location finding systems, power generators, and mobilizers [Perrig et al., 2004].

Although some implementations allow the use of cables as communication media among sensor nodes, most scenarios require sensor nodes to communicate wirelessly using radio frequency. In this thesis, we will only focus on *wireless sensor networks (WSN)*. Since a very large number of sensor nodes are distributed in the target area, the network is also called a *Distributed Wireless Sensor Network (DWSN)*. For simplicity, I use the term *Distributed Sensor Networks (DSN)* throughout this thesis.

Another type of sensor networks, *Hierarchical Wireless Sensor Networks (HWSN)*, as shown in Figure 1.1(a) [Çamtepe and Yener, 2005], have a hierarchical network infrastructure that deploys *Cluster Head(s)* and *Base Station(s)*. A *Cluster Head* [Younis and Fahmy, 2003] is a node that are deployed to be the center of a group of sensor nodes and is responsible for intra-group coordinations and inter-group communications. The *Base Station* of a HWSN is a resource-rich node in terms of computational ability and energy lifetime, acting as the root of the network, collecting and routing information as a gateway. HWSN is a typical design for certain scenarios but is beyond the research scope of this thesis. A *Data Sink* is a device that subscribes to general or specific data streams. All sensor nodes in the network are physically identical and data is sent directly to a *Data Sink* from sensor nodes. Normally, there is only one data sink to collect all types of information, although there could be more than one data sink in the network such that one data sink collects only certain type of information, such as only temperatures or only pressures. Figure 1.1 shows the difference between a HWSN and a DWSN.

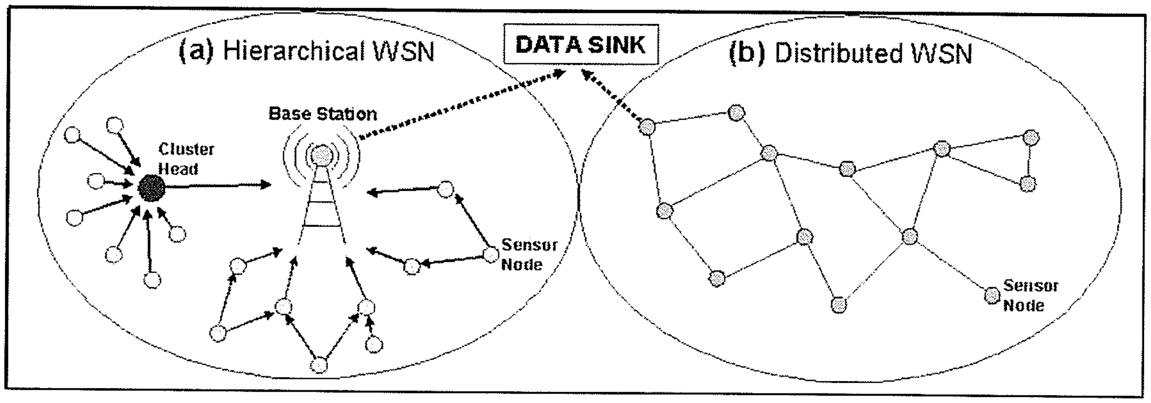


Figure 1.1: Network Models: Hierachical and Distributed Wireless Sensor Networks [Çamtepe and Yener, 2005]

Sensor networks are usually deployed into domains are either inaccessible or hazardable to humans. For example, sensor networks can be deployed for detecting environmental changes in forests, oceans and plains, etc. Military sensor networks can obtain information on hostile entities, such as troop movements. Sensor networks can also detect or monitor biological, chemical and nuclear materials. Examples of sensor networks include *Wireless Integrated Network Sensors* [WIN] and *SmartDust* [Kahn et al., 2003].

1.2 Terms in KPS Designs

Some terms used in this thesis are common in various research areas, such as network designs, graph theory and cryptography. In this section, I introduce some concepts related to KPS designs and highlight some terms, which are used thoroughly in this thesis.

I start from terms related to the concept of "key". In a key pre-distribution scheme, a *key* is a security element used to encrypt communication link(s) between two or more sensor nodes. Note that keys are symmetric in most KPSs. That is, for any two nodes N_i and N_j, if messages from N_i to N_j are encrypted using key k_{ij}, then messages from N_j to N_i use the key k_{ij} too. Keys that are used for securing this kind of bidirectional communications are called *pair-wise keys*. A sensor node stores a list of keys to communicate with others and these nodes are named the *key ring* or the *key chain* of that node. The size of a key ring is the number of keys in the key ring. A node's key ring is selected from a collection of keys, called a *key pool*, in a KPS design. A key pool contains all the keys used for a KPS.

In a DSN, every sensor node has a limited communication range and can only connect to a part of other nodes in the network directly. Since all nodes in a DSN are physically identical devices, it is reasonable to regard the communication range of a node as a circle area, which is called the neighborhood of a node. When two nodes are located within each other's neighborhood, they are physically able to communicate to each other directly. In this case, they are neighbor nodes of each other. When two nodes are not within each other's neighborhood, it could take multiple hops to connect them. A possible sequence of nodes along these links is call a *path* between these two nodes. The sequence of keys that are used to secure intermediate links between two indirectly connected nodes is called a *path-key*. For example, if two nodes a and b are connected indirectly via an intermediate node c, then the path from node a to node b is $a \rightarrow c \rightarrow b$. If the key used to secure link $a \leftrightarrow c$ is $K_{a,c}$ and the key used to secure link $c \leftrightarrow b$ is $K_{b,c}$, then the path-key from node a to b is $\{K_{a,c}, K_{b,c}\}$, from node b to a is $\{K_{b,c}, K_{a,c}\}$. A path-key also refers to a re-established key between two nodes when they satisfy some certain conditions. I will explain this in Chapter 2. After a network has been deployed, some applications require key updates. The process of updating new keys over existing secured path is called *key reinforcement*.

We now introduce terms related to networks. The *size* of a DSN is defined as the number of nodes in the network. In a DSN, if each node is regarded as a vertex and each link is regarded as an edge of a graph, then the network can be presented by a graph. I consider the graph of a DSN from the view of two models: *the Network Models* and *the Physical Model*. The network view is the view of a DSN before the deployment, where a node has unlimited communication range such that an edge exists between two nodes if they are able to establish a secure link. The corresponding graph in the network view of the DSN is called a *network graph.* The physical view is the view of the network after the deployment, where a node's physical communication range is added such that an edge exists between two nodes if they can establish a secure link and the distance between them is less than the communication radius of a node. A graph for such a physical view is called a *physical graph.* When studying a KPS, before the deployment of a DSN, I consider problems in the network graph; after the deployment, in the physical graph. Some DSNs may deploy mobile sensor nodes, but in this thesis, we only consider sensor nodes with essential components such that their locations are fixed once distributed.

1.3 Constraints of Typical Sensor Networks

Before the introductory of *key pre-distribution schemes (KPS)*, it is necessary to show some constraints on typical DSNs. The illustration of these constraints is helpful for understanding KPS designs.

A typical DSN may consist of thousands of sensor nodes. For such a network to be economically viable, the cost of each node must be inexpensive. Usually the cost of a typical sensor node is lower than $10 [Rabaey et al., 2000]. This low-cost requirement means that DSNs suffer from the following constraints:

Computational Capability Constraint Each sensor node in DSNs is equipped with an integrated microprocessor, running at a very low frequency (e.g., AT-

MEL 90$LS$8535 runs at 4 MHz [Eschenauer and Gligor, 2002]), so that the computational capability of each node is very limited. This will greatly impact the encryption/decryption processes in DSNs. Most methods used in traditional networks involve such heavy-load computations that they are not suitable to be deployed in DSNs.

Power Supply Constraint Sensor nodes are small and battery powered devices. They are not suited to perform long-term or heavy-load tasks. In most situations, it is impossible to replace or recharge the battery. This means that algorithms (not only for key distributions) in DSNs have to be very efficient and mindful of their consumption of power.

Memory Capacity Constraint Corresponding to the low-frequency CPU and very limited power supply, sensor nodes often have very small memory capacity. (e.g., Smart Dust sensors have only 8Kb program memory and 512 bytes data memory [Kahn et al., 2003]). This implies that security elements (keys, identities, key rings, etc.) have to be limited to a certain length.

Wireless Communication Range Constraint Each sensor node contains a wireless communication unit. The communication range of a sensor node is dependent on power consumptions, which means increasing the power output may extend the communication range of a node. Unfortunately, sensor nodes have limited power supply such that this kind of power boost is not allowed. Usually a node can only communicate with a small part of the network directly. It may communicate with the rest of nodes via multiple hops, if possible.

These constraints make DSNs different from traditional wireless ad hoc networks. Ad hoc networks are centerless computer networks that are deployed for mobile devices, where nodes could be frequently moved or added. When designing a KPS, some ideas may be borrowed from ad hoc networks, but most key distribution schemes are not well suited to the unique features of sensor networks. Furthermore, these constraints affect the type of cryptosystem that can be used. I now describe some well known key agreement schemes and why all but the key pre-distribution scheme are not well-suited for DSNs.

1.4 The Selection of Key Agreement Schemes

There are three types of key agreement schemes that exist in traditional networks: *the trusted-server scheme, the self-enforcing scheme, and the key pre-distribution scheme.*

The *trusted-server scheme* is a client-server architecture that uses a *trusted authority (TA)* that holds long-term keys to encrypt communications for all other nodes. It is impossible to deploy a TA in DSNs because acting as the center of communications will quickly use up the power of a sensor node. Moreover, the capture of the TA will expose the entire network, which is not an affordable risk in data sensitive applications.

The *Self-enforcing scheme*, such as public key certification [Stinson, 2002], is also impractical in sensor networks because it is an energy-intensive and computation-intensive algorithm. The research in [Carman et al., 2000] indicates that the power consumption for a RSA encryption (Self-enforcing public key) is 400 times more than a AES encryption (symmetric pre-distributed key), operating on the same CPU. Currently, the power supply and the computational capability of sensor nodes prohibit the implementation of the self-enforcing scheme.

The final option is the *key pre-distribution scheme* (KPS) [Eschenauer and Gligor, 2002]. Key pre-distribution is a procedure where keys are allocated to each node, known as the node's "key ring" or "key chain" [Eschenauer and Gligor, 2002] before the sensors are deployed in the field. After the deployment, nodes negotiate with each other using these keys to establish secure links. In resource limited environments, KPS has many advantages over other type of key agreement schemes. It is more computationally efficient and consumes less energy than others, which is the most desired feature in designing security schemes for DSNs. In Section 1.5, I give further analysis about KPSs and DSNs, and describe other security concerns.

1.5 Several Critical Concerns for KPS Designs

The key pre-distribution of a KPS occurs before the network is deployed. Usually, users of a scheme do not participate in key management after the deployment. That is, a KPS has to make the network self-organized. The network with an effective KPS must have the capability to deliver messages among nodes and handle malicious attacks without human interactions. To design an effective KPS, several factors need to be considered.

The first concern is connectivity [Eschenauer and Gligor, 2002]. A node's *local connectivity* [Wei and Wu, 2004] is defined as the connectivity within the node's physically communication range. However, the communication range of a sensor node is always limited. Although the communication range of a node can be extended to a longer distance, it is still unlikely to reach all the rest nodes directly when distributed in a large area. Even when two nodes are within each other's communication range, they still may not be able to communicate with each other directly because they are not able to establish a pairwise secure link. A "good" KPS should construct a connected physical graph so that messages can be delivered from any node to any other part of the network. That is, the KPS should distribute keys to sensor nodes so that for any two nodes a, b, there is a sequence $a_1, a_2, ..., a_n$ of nodes with $a_1 = a$, $a_n = b$ so that a_i and a_{i+1} share a common key and are within a communication range. Such a connectivity among the entire network is called *global connectivity* [Wei and Wu, 2004]. Since providing a connected network is critical to a KPS, I will use the term *global connectivity* thorough this thesis. Unless specially specified, all the "connectivity" mentioned in this thesis means global connectivity. Connectivity is a prerequisite for all the schemes, since a designer has to guarantee that messages are delivered among the network securely. We define a value Pr, normally very close to 1. When the probability that any two nodes are connected (maybe via multiple hops) is larger than P_c in the network, we say that the network is "connected".

The second concern is the resiliency against node capture [Eschenauer and Gligor, 2002]. Resiliency is a measure of the amount of communication links that are compromised by the capture of a certain amount of sensor nodes. In some scenarios, such as military missions, adversaries may actively capture some sensor nodes, which means the data stored in these captured nodes will be obtained by the adversary. Since a node may carry sensitive information about other nodes, such as node identities and keys, the adversary may use this information to decrypt and monitor other communication links. In a KPS with *perfect resiliency*, the only links that are compromised by the capture of a node are those links which involve that node. A good KPS should not allow adversaries to infer any information from the captured node about the rest of the communication links in the network. In the case where many nodes are captured, a good KPS should still allow most of the remaining nodes to communicate securely. Since connectivity is the essential requirement of a KPS design, the resiliency against node capture is always the most important factor in data sensitive applications. I use it as a measure to evaluate the security level of a scheme.

The third concern is scalability [Eschenauer and Gligor, 2002]. DSNs often deploy with a large number of sensor nodes (e.g., thousands of nodes). It is important for a KPS to accommodate very large number of nodes. This is another difference between

DSNs and traditional wireless ad hoc networks. Currently, most KPS designs are feasible to accommodate 10^3 to 10^5 of sensor nodes. Before I introduce these schemes in detail, it is easy to tell the relationship between the memory usage and the scalability. Normally, to support a larger-size network, more memory space is required. I will introduce this numerical relationship in detail in Chapter 2. In applications with very large number of sensor nodes deployed, a good KPS should support this kind of large network, while keeping a balance between the largest supported network size and memory usage.

The fourth concern is locations of sensor nodes. In most traditional networks, nodes have fixed locations and their topology are predictable in the key distribution phase. Thus, location-related schemes or optimizations can be performed in the network. Unfortunately, In a DSN, the network topology is not known prior to the deployment. That is, the locations of sensor nodes in the target plain are unknown during the key pre-distribution stage [Du et al., 2003]. Since sensor nodes are often randomly distributed into the target area, the precise location of a node is not predictable. Therefore, any KPS depending on the locations of nodes is not feasible for DSNs. This concern is critical to KPS designs. Another location related concern is about the limited communication range of a sensor node. Two sensor nodes may be predefined as neighbor nodes in the network graph, but they may not able to communicate directly because they are distributed out of each other's physical communication range.

All these factors should be considered carefully when designing a KPS. To illustrate this point, we consider two simple KPS designs. The first design only considers the KPS issue from the view of resource usage. It distributes a single key among all the nodes, which is very efficient in terms of the cost of computational capabilities and memory capacity. However, the resiliency against a malicious attack is very weak in this *single-key scheme*. The capture of a single node exposes the entire network.

Another extreme only considers the KPS issue from the view of resiliency. This $(n-1)$-*key scheme* allocates a unique key for each pair of sensor nodes in the network. In this scheme, each node needs to hold $n-1$ keys in a network of size n. In the network graph, each node is adjacent to every other node (that is, the network graph is the complete graph K_n). This scheme has perfect resiliency, but requires each node store $n-1$ keys. Hence this exhaustive solution is only suitable for small-sized DSNs because it requires a large amount of memory in large-scaled networks. These two examples illustrate that designing a KPS is not an easy task. All the constraints and the concerns need to be considered carefully to produce a feasible KPS. A KPS should efficiently use limited resources (memory, processor, power) to produce desirable characteristics such as connectivity, scalability and good resiliency for DSNs.

1.6 Thesis Organization

Firstly, this thesis contains a thorough survey for key pre-distribution schemes in distributed sensor networks. I introduce some of the important achievements in this area. Secondly, I will illustrate my attempts at constructing new KPSs.

In Chapter 2, I introduce the first well-designed KPS for DSNs, the basic random scheme. I show the random key pre-distribution in this first scheme, then present some results in random graph theory and how it is used to determine the parameters in the basic random scheme. A numerical analysis will be given at the end of Chapter 2 to show how the basic random scheme works in a DSN.

In Chapter 3, I introduce some KPSs using probabilistic methods for key pre-distributions, similar to the basic random scheme. These random schemes achieve either better resiliency against node capture or optimized memory usage compared to the basic random scheme. I show the algorithms used in these new schemes and give analytical results of resiliency.

In Chapter 4, I introduce another approach for designing KPSs, deterministic schemes, in which keys are selected from the key pool in a deterministic manner. The research of deterministic KPSs involves the areas of set system, graph theory and combinatorial designs. I also present the theoretical construction of these schemes and show the advantages and disadvantages of deterministic schemes compared to probabilistic schemes.

In Chapter 5, I introduce a methodology, the product construction, that can be used to derive new KPSs from existing ones. I illustrate the method by showing the first generalized product construction, then introduce some other schemes using product constructions.

In Chapter 6, I follow the product construction and show my attempts of producing new KPSs from existing schemes. Specifically, I combine Blom's scheme with the μ-CID scheme.

Chapter 2

The Basic Random Key Pre-distribution Scheme

In Chapter 1, I explained why KPS is currently the only practical solution for securing communications among sensor nodes in DSNs. The research of applying KPS to DSN was initiated in [Eschenauer and Gligor, 2002]. Their scheme is a probabilistic one. The term *probabilistic* means that keys are distributed to sensor nodes in a random manner. Alternatively, a probabilistic scheme is call a *random scheme.*

Eschenauer and Gligor's scheme is often referred to as "the basic random scheme" or "the basic scheme". The basic scheme could be regarded as the foundation of KPS designs and had great effects on later research in this field. Several successful random schemes were either designed on top of the basic scheme or as extensions of the basic scheme. In addition, the basic scheme also influenced some deterministic schemes and product constructions, which will be introduced later in this thesis. In this chapter, we will introduce the basic random scheme in detail. I will focus on the procedure of *random key distribution* and the method used for *parameter determination* in the basic scheme.

2.1 Random Key Distributions

Eschenauer and Gligor presented that, in the basic random scheme, key distribution consists of three phases: *key pre-distribution, shared-key discovery, and path-key establishment.* I now describe these phases.

2.1.1 The Key Pre-distribution Phase

In a KPS, keys are distributed to sensor nodes before the deployment of the network, such that the key pre-distribution phase takes place before sensor nodes are distributed into the operational environment. Eschenauer and Gligor illustrated five off-line steps in the key pre-distribution phase. However, some steps may not be necessary to all DSNs. In the sensor network that Eschenauer and Gligor presented, some special nodes, called controllers, are introduced as an essential part of the key distribution. These controllers store IDs of nodes and their associated key rings to assistant discovering the common key between two nodes after the deployment. In fact, in a DSN with some special nodes, those nodes only receive data passively which they then transmit to remote data sink(s), thus they are named, *sensor gateway*, and normally are different devices from sensor nodes. Certainly, system architects could let sensor gateway nodes actively join the communications, such as performing some key management tasks as in [Eschenauer and Gligor, 2002], but in that case, nodes are no longer equal to each other. Furthermore, even without these controller nodes, the basic random scheme is still suitable to be implemented in sensor networks. [Pietro et al., 2003] improved the basic random scheme by selecting key rings pseudo-randomly. Although it is not their main purpose, the new scheme, known as *the direct protocol*, is capable of establishing secure links without controllers. In order to generalize the basic scheme, we simplify the key pre-distribution phase into two steps.

Step 1 : A very large size key pool P_l is set up, consisting of P keys, denoted as $key_1, key_1, ..., key_P$.

Step 2 : In a network of size b, k keys are randomly chosen without replacement from P_l to create a key ring B_i for each sensor node N_i, where $i = 1, 2, ..., b$.

Note that key_i, for $i = 1$ to P, are not actual keys, but are key identifiers instead. Certainly, we could regard a key and its key identifier equally, but note that the key information delivered among nodes is key identifiers. Under no circumstance are keys allowed to be delivered or exchanged, except when they are used in encrypted messages. The actual keys are only stored in sensor nodes and used for encryption. For example, assume node A contains key_7 and node B has key_7 in memory too. Suppose they can communicate directly, then they may broadcast the key identifier of key_7 in order to establish a secure link. The actual key associated with key_7 is only used to encrypt the message through channel $A \leftrightarrow B$ but would never be broadcasted.

Let N_i and N_j denote any two nodes in the sensor network. Eschenauer and Gligor defined that, if N_i and N_j share a common key, they can use that key to secure communications between them. Thus, an encrypted link is established between N_i and N_j. In the key pre-distribution phase, nodes can be regarded as in the network graph with unlimited communication ranges. That is, any node is able to reach any other nodes in the network physically. Under this condition, sharing a common key is the only requirement to establish a secured link between two nodes. However, after nodes are distributed into the target area, two nodes connected in the network graph may be located out of each other's neighborhood such that the link between them disappears in the physical graph.

After the key pre-distribution phase, there are b prepared sensor nodes, each storing k keys. However, a node knows its neighbors only in the network graph but not in the physical graph, so that it is necessary for sensor nodes to discover its neighbors after the deployment. I now describe the shared-key discovery phase.

2.1.2 The Shared-key Discovery Phase

After the key pre-distribution phase, these b sensor nodes are randomly dropped into the target area. However, in the physical graph, some adjacent nodes in the network graph may be disconnected due to the limited communication ranges of sensor nodes. Those "out-of-range" nodes can not communicate directly, even if they share a common key. Therefore, the requirement of establishing a secured link directly between any two nodes N_i and N_j in the physical graph becomes: sharing a common key in the network graph and being located in each other's neighborhood after the deployment. In the shared-key discovery phase, each node broadcasts a list of key identifiers of the keys in its key ring. Until all the nodes obtain enough key distribution information, the topology of the sensor network is established in the physical model. If node N_i and its directly reached node N_j share a common key key_t, N_i and N_j will regard each other as a neighbor node and encrypt all messages between them using key key_t. It is possible that key_t is used in more than one link so that the decryption of one may expose other links. It is also possible that N_i and N_j share more than one common key. Eschenauer and Gligor do not provide detailed solution for this, but very likely the nodes will choose one key from all common keys, either randomly, or by the keys' priorities. A more secure method is to use all the common keys between two nodes to encrypt one link. This idea is introduced by [Chan et al., 2003] for enhanced security, which will be presented in Chapter 3.

2.1.3 The Path-key Establishment Phase

In the basic random scheme, when two nodes are located within each other's communication range of each other and are connected via multiple hops, they can set up a key as the common key between them. This process is called the path-key establishment. In other probabilistic schemes, key pre-distribution may only contain two phases. However, in [Eschenauer and Gligor, 2002], path-key establishment is regarded as the third phase of key distributions.

Eschenauer and Gligor suggested using one of the keys, which does not appear in either key ring of the two nodes, to secure the communication. Note that this process requires the participation of controller nodes. This idea does not appear in later papers. The newly selected key will be used to secure messages between the two nodes. Path-key establishment phase increases the connectivity of the network (adds more adjacent nodes) and saves energy on involved nodes. Eschenauer and Gligor did not show how to update the key. It is reasonable that the path-key is selected randomly from the key ring of one node, encrypted and sent to another node via the original multiple intermediate nodes. After the other end receives the new path-key, a new direct secure link is established between the two nodes.

2.1.4 Revocation and Re-Keying

Eschenauer and Gligor have introduced several ideas on dealing with KPS issues. They also presented *Revocation* and *Re-keying* in the basic scheme. Revocation is the process of abandoning captured keys in the network. Whenever some sensor nodes are captured, which means the adversary obtains all the information stored in these nodes, other nodes should stop using all the keys in their key rings of these captured nodes. Since some keys are removed from the key ring, some links may not exist after the revocation, so that another shared-key discovery and, possibly path-key establishment, is necessary. This key revocation process is heavily dependent on the "controller" nodes, which have a large communication range and are mobile so that this revocation may not practical in general DSNs.

Re-keying is the process of two adjacent nodes updating their common key when the key is expired. However, in most DSNs, keys have infinite lifetime and re-keying will not take place.

In fact, revocation and re-keying are methods used to enhance the basic random scheme. Since the basic scheme is the very first KPS for DSNs, we introduced it in detail. In later chapters, I only show the the essentials of KPSs. We now start to introduce some results in random graph theory and how these results help to determine the parameters in the basic random scheme.

2.2 Random Graph Theory and Parameter Determination

In the $(n-1)$-key scheme, each node connects to all other nodes in its communication range. The network graph is a complete graph in this scheme. However, the high memory usage makes the $(n-1)$-key scheme impractical for most DSNs. Based on the observation that not all the links are necessary to make the network graph connected, the basic scheme only keeps some links in a node's communication range. However, in another view, to gather and transfer desired information, a *connected network* is required, which means, any two nodes in the network should be connected either directly or via multiple hops. So an important question about the basic random scheme is: How do we know the network with the basic scheme deployed is connected in the network graph, and finally, in the physical graph?

Eschenauer and Gligor defined a desired probability P_c for graph connectivity and regard $P_c = 0.9999$ as "the network will almost certainly be connected". This value seems reasonable for most applications. For applications that need higher connectivity, the value of P_c could be closer to one. A network with the basic scheme is regarded as "connected" in this thesis if the probability that any two nodes are connected is larger than $P_c = 0.9999$.

Now consider all the parameters in the basic scheme: the network size n, the key pool size P, the key ring size k, the pre-defined desired probability for graph connectivity P_c. Since the communication range of a sensor node is always limited, another parameter, or requirement, for the basic scheme is the *density of the network*. The density of the network is the average number of nodes in a communication range. Eschenauer and Gligor declared it as "a neighborhood connectivity", but in fact not all nodes in a neighborhood are connected to the center node of the neighborhood. The density of the network, denoted by n', should be predefined as a requirement for a DSN. Among all the nodes in a neighborhood, a node only needs to directly connect to some of them, which is denoted as the degree (d) of this node.

In order to make the sensor network connected with the probability P_c in a network with density n' after the deployment, we need to find out several parameters:

- What corresponding value should a key ring size (k) be? - What value should the degree (d) of a sensor node be? - What value should the pool size (P) be?

Random graph theory helps to answer these questions. A random graph is a graph in which graph vertices, graph edges are generated by some random process. Different random graph models will generate graphs with different probability distributions. The most commonly studied model is $G(n, p)$. I now introduce this model.

Definition 2.2.1. *$G(n,p)$ consists all random graphs with vertex set $V = \{1, 2, ..., n\}$ in which edges are generated with an independent probability p.*

When a sensor network is deployed in the target area, all nodes and secured communication links form a random graph [Pietro et al., 2003]. A (random) graph is connected if for any node in the graph there exists at least one path from that node to any other node. In a sensor network with a random scheme, nodes and encrypted links should form a connected graph. Messages could be safely delivered between any two nodes in the network.

Theorem 2.2.1. *Let $G(n,p)$ denote a random graph of n nodes, where the probability that a link exists between any two nodes is p. Let d denote the expected degree of a node. Then d is given by:*

$$d = p(n-1)$$

Theorem 2.2.2 shows that, there exists a d corresponding to each p, the probability that makes the global connectivity larger than the desired connectivity P_c in a network of density n'.

To obtain the probability p that a link exists between any two nodes in a neighborhood, Eschenauer and Gligor referenced a result from random graph theory. Random graph theory, more specifically the property of connectivity, has been intensively studied. [Paul Erdös and Alfred Rényi, 1959] showed that in a very large random graph there exists a value of p such that the connectivity moves from "nonexistent" to "certainly true". This value p is called the *threshold* in the random graph and the function used to obtain p is called the threshold function of that property. In this thesis, p represents the threshold of connectivity. Eschenauer and Gligor used p to estimate the connectivity of the random scheme, then calculated the expected degree of each node.

Theorem 2.2.2. *In a very large random graph, with a predefined probability of connectivity (P_c), the threshold function defining p is:*

$$P_c = \lim_{n \to \infty} Pr[G(n,p) \textit{ is connected}] = e^{-e^{-c}},$$

where $c \in$ is a fixed real number and

$$p = \frac{ln(n)}{n} + \frac{c}{n}.$$

Given n and P_c, we can compute p.

Let $Pr[G(n,p)$ is connected] represent the probability that a random graph $G(n,p)$ is connected. Figure 2.1 from [Eschenauer and Gligor, 2002] shows the expected degree of each node as a function of the network size n. When the desired connectivity P_c increases by one order of magnitude, a node only requires several more connected neighbor nodes to keep the expected connectivity of the entire network. For example, in a network of size 10000, when the desired global connectivity is 0.9999, every node needs to store 18 keys. Now if the connectivity requirement increases to 0.99999, every node only needs to store 2 more keys.

In a sensor network, the key ring size k is constrained by the memory capacity of each node, which means the value of k has to be less than the usable memory capacity. In an application, k is assigned with a certain value. When the key ring size k, the desired network size n, and the expected probability of connectivity P_c are

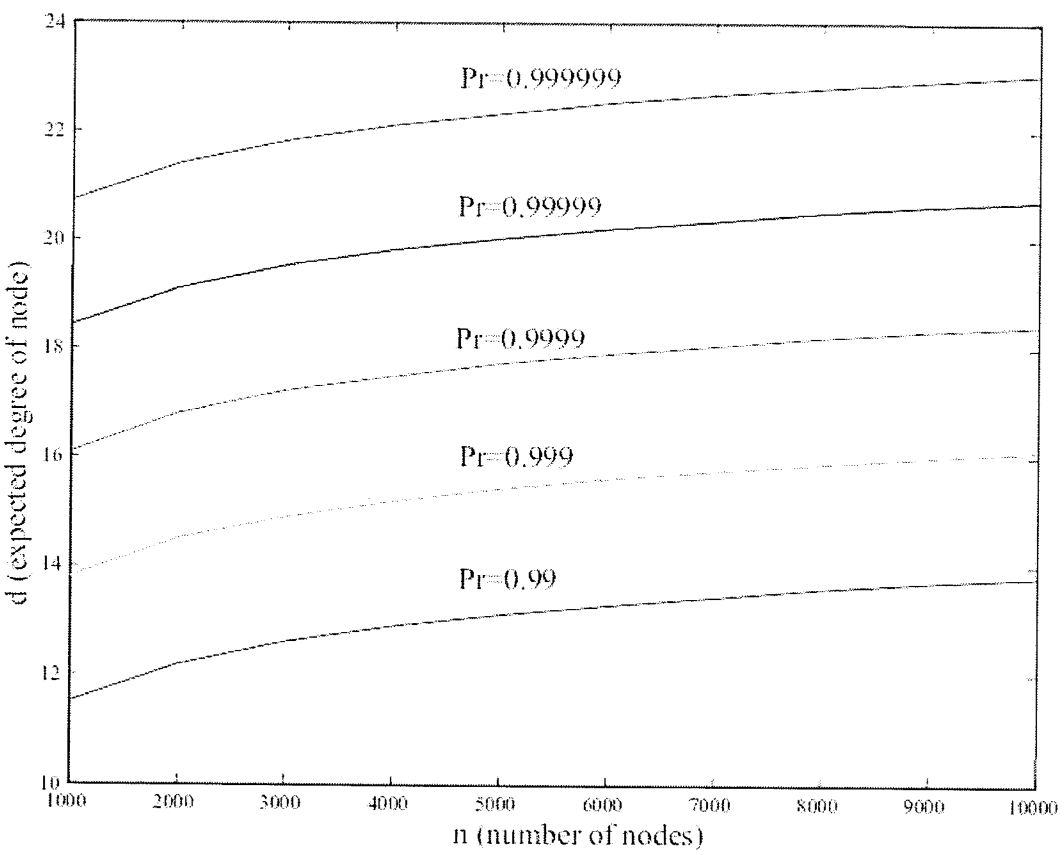


Figure 2.1: The expected degree of a node as the network size increases, with different global connectivity, where $Pr = Pr[G(n,p)$ is connected] [Eschenauer and Gligor, 2002]

all specified, it is possible to calculate the pool size P.

Lemma 2.2.1. *Let n' denote the number of neighbors that a node can physically reach in the communication range, let d denote the required degree of a node, then in most cases we have $n' \ll n$. It follows that,*

$$p' = \frac{d}{(n'-1)} \gg p,$$

where $\gg$ means "greatly less than" and p' is the probability that any two nodes in a communication range share at least one common key.

This formula indicates that in a very large sensor network where the communication range of each sensor node covers only a small part of the network ($n' \ll n$), the probability that any two nodes share at least one common key in a communication range is much larger than that probability in the entire network ($p' \ll p$). To illustrate the relationship between k, p and P, Eschenauer and Gligor showed the following result.

Lemma 2.2.2. *The probability that any two nodes share at least one common key is:*

$$p = 1 - \frac{\binom{P-k}{k}}{\binom{P}{k}}.$$

Proof. The probability that any two nodes share at least one common key equals to $1 - Pr$[any two nodes do not share any key]. To compute Pr[any two nodes do not share any key], we choose k keys out of a pool of size P to create the first key ring. The number of possible choices are $\binom{P}{k}$. The second key ring will not repeat any keys in the first key ring. Then the possible choices of the second key ring are $\binom{P-k}{k}$. Totally, there are $\binom{P}{k}^2$ ways to choose k keys twice from the key pool. Therefore, we have the probability that no common key exists between the two key rings (two nodes) is:

$$\frac{\binom{P}{k}\binom{P-k}{k}}{\binom{P}{k}\binom{P}{k}} = \frac{\binom{P-k}{k}}{\binom{P}{k}}.$$

□

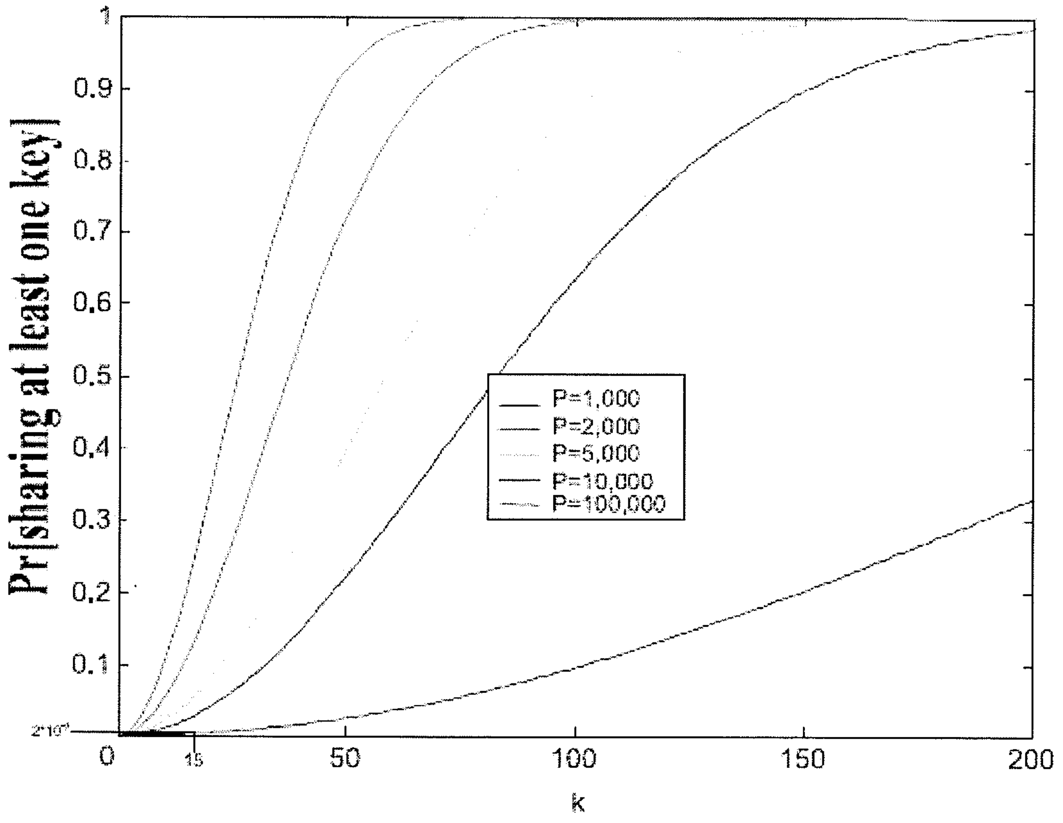


Figure 2.2: Probability that any two nodes share at least one common key when choose k keys from a pool of size P [Eschenauer and Gligor, 2002]

Figure 2.2 from [Eschenauer and Gligor, 2002] shows various values of P for the above function. With a pool size of 1000, $k = 50$ will make the global connectivity close to 1; with a pool size of 10000, key ring size has to increase to 200 to make the network connect. Figure 2.2 indicates that the pool size has great effect on the key ring size such that a proper pool size is important to a KPS. Although this plot only illustrates the relationship between p and k with a fixed pool size P, it is also suitable to present p' as the function of k (just extend a communication range as the entire network). Now we use an example to illustrate this point.

2.3 A Numerical Example

Figure 2.3 shows different ranges of k and p in Figure 2.2. Specifically, Figure 2.3(a) shows the relationship between k and p when pool size $P = 100000$. Figure 2.3(b) shows part of the amplified curve in Figure 2.3(a). They will be used for the numerical example in this section.

Assume that in a sensor network of size $n = 100000$, the probability that a link exists between any two nodes in the network is $P_c = 0.99999$, such that the network is "almost certainly" connected. By using Theorem 2.2.2, we get $c = 11.5$. For $c = 11.5$, we have $p = 2.3 \times 10^{-4}$ and $d = 2.3 \times 10^{-4} \times 99999 = 230$. $d = 230$ indicates that every node needs at least 230 neighbor nodes to make the entire network connected.

To achieve $p = 2.3 \times 10^{-4}$, we only need a key ring size of 5 based on figure 2.3(b). These value implies an extreme situation, when $p' = p$, where the communication range of each node will physically reach all other nodes in the network. More practically, only very small number of nodes will be covered in a communication range, but a node still needs to directly connect to 230 other nodes in the network. To satisfy this condition, the number of nodes that a node's communication range can physically reach, d, has to be larger than 230. Assume $d = 921$ in this case, then we have $p' = \frac{d}{n'-1} \approx 0.33$. From Figure 2.3, we find $k = 200$.

Therefore, in a sensor of network of size $n = 100000$, $P_c = 0.99999$ and $d = 931$, we need the pool size $P = 100000$ and the key ring size $k = 200$. However, these parameters in the network can always be adjusted to suit the implementation.

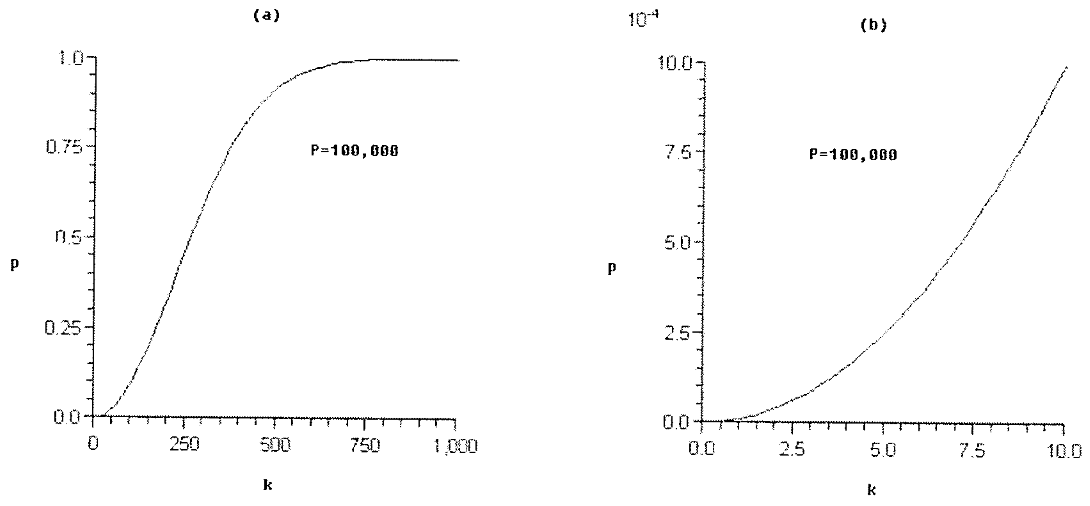


Figure 2.3: Probability that any two nodes share at least one common key when k keys are chosen from a pool of size P, two different ranges

I now give a brief review of the basic random scheme. In the basic scheme, keys are randomly selected to create a key ring for a node. If two nodes have at least one common key, then they can use the key to establish a secure link between them. Parameters of the basic random scheme are computed based on Theorem 2.2.2, which will be frequently used in later chapters.

Chapter 3

Other Random Key Pre-distribution Schemes

As the first KPS for sensor networks, the basic random scheme has an impact on the development of later KPS designs. All random KPSs use the same procedure to generate key rings, which was described in Chapter 2: for creating a key ring, each sensor node is assigned a subset of keys that are randomly chosen from a key pool. Following this procedure, two similar random techniques are introduced in [Chan et al., 2003] and [Pietro et al., 2003], as enhancement of the basic random scheme. In addition, [Du et al., 2003] and [Liu et al., 2005] are another two approaches involving random key pre-distribution, but [Du et al., 2003] and [Liu et al., 2005] combine the basic random scheme with other techniques and therefore they are more suitable to be classified as *product constructions*, which will be introduced and discussed in

Chapter 5. In this chapter, I first introduce the evaluation of KPSs, then describe the two random schemes in [Chan et al., 2003] and [Pietro et al., 2003].

3.1 Evaluations of Key Pre-distribution Schemes

As the first KPS for sensor networks, the basic random scheme does not provide any benchmarks for evaluating the resiliency of a KPS. Techniques for evaluating resiliency were introduced in several later papers [Chan et al., 2003; Pietro et al., 2003; Du et al., 2003; Liu et al., 2005]. In order to make analyses of other random schemes more understandable, I illustrate these evaluation techniques first. That is, I have to answer several questions: What is meant by a "good" KPS? Formally, what should we use to evaluate the performance of a KPS? And how? We now start to answer these questions.

When thinking about the evaluation of a KPS, memory usage is a straightforward property to compare amongst different KPSs. Since sensor nodes are devices with very limited memory capacity, in most applications, users need the memory usage of a KPS to be as small as possible. But they have to consider other properties, such as connectivity and resiliency, which makes evaluation a complex problem. Usually, I choose parameters for all schemes such that they have same memory usage, then I start to compare the resiliency among them. In this thesis, I assume all schemes use the same length (64 bits, 128 bits, etc.) of memory space for all keys. Thus the memory usage of a key ring is determined by its length. A KPS with the memory usage of k means that in the scheme every node needs to load k keys in its memory. However, when a KPS is able to save considerable memory usage while guaranteeing the predefined connectivity, I will compare the memory usage with other schemes. Usually there is a sacrifice on the resiliency when a scheme focuses on memory saving. I state that one scheme has *less memory cost* than another if it requires less keys in each node.

Connectivity is another property that can be used to evaluate KPSs. As shown in Chapter 2, connectivity should satisfy a predefined probability P_c. In most scenarios, connectivity is a requirement of a KPS because a disconnected network could not perform data collecting tasks. In this thesis, unless specified, we let the global connectivity $P_c = 0.9999$ for all the schemes. Thus connectivity becomes a fixed parameter of all KPSs. we do not compare the connectivity of two schemes but every scheme should satisfy at least $P_c = 0.9999$.

Resiliency, the most important property of a KPS, will be compared among different schemes. There is always trade-off among resiliency, connectivity and memory usage. Based on Theorem 2.2.1, in order to increase the network connectivity P_c, each node must have more directly connected neighbors. Therefore, nodes have to carry more keys, which increases the memory usage. Moreover, when a node storing more keys is captured, an adversary obtains more information about the scheme and the network, which indicates that the resiliency against node capture is weakened. In this thesis, we compare resiliency of KPSs by: 1) using same memory usage 2) using same predefined network connectivity 3) analyzing when x nodes are captured 4) investigating the fraction of additional communication that an adversary can compromised.

In addition, there are other properties that can be used to evaluate KPSs. Some properties are more engineering-oriented, such as power consumption. We have to consider specific hardware to give the evaluation, which is beyond the scope of this thesis. In the next section, I describe another probabilistic KPS that uses multiple keys to secure one link.

3.2 Using Multiple Keys to Secure One Link

In the basic random scheme, multiple keys may be shared by two nodes, however the solution for this situation is vague. Eschenauer and Gligor only define that two nodes are capable of establishing a secure link if they share *at least* one common key. But they ignore the detail on dealing with the situation where multiple common keys are shared between two nodes. Very likely, they will choose one of these common keys, either randomly or by priorities of keys.

In [Chan et al., 2003], Chan et al. attempt to use $q(q \geq 1)$ common keys to secure a direct link between two nodes. The key pool set up and the key pre-distribution procedure are the same as in the basic random scheme. However, any two nodes are able to establish a secure link if they share at least q common keys, where q is predefined. In case that two nodes share q' keys, where $q' \geq q$, a new key is generated by hashing all shared keys, $K_{new} = hash(k_1||k_2||...||k_{q'})$. These keys to be hashed are arranged by the index order in the key pool. Then this K_{new} will be used to secure the link between the two nodes. Chan et al. named this scheme with multiple-to-one key construction as the *q-composite scheme.* Except the requirement of q-common keys to establish a direct secure link between two nodes, the q-composite scheme is no difference than the basic random scheme. I will not introduce the procedure of random key pre-distribution again in this chapter. However, parameters for the q-composite schemes should vary from the basic scheme. I now describe the computation of parameters (key pool size, key ring size, etc.) in the q-composite scheme.

3.2.1 Computation of Key Pool Size

Like the basic random scheme, the key pool size needs to be computed before we generate the key ring for each node. Because multiple keys are used to secure one link in the q-composite scheme, the key pool size and the key ring size should be different than the basic scheme. Assume the network in which the q-composite scheme is evaluated uses the same sensor nodes as in the basic random scheme, the memory space reserved for storing keys should be same. That is, the key ring size should be the same. Hence I only describe the computation of key pool size in the q-composite scheme.

Lemma 3.2.1. *Let P denote the key pool size, k denote the key ring size, and p denote the probability that any two nodes share at least q keys in common. In the* *q-composite scheme, we have:*

$$p = 1 - \sum_{i=0}^{q-1} \frac{\binom{P}{i}\binom{P-i}{2(k-i)}\binom{2(k-i)}{k-i}}{\binom{P}{k}^2}.$$

Proof. Each node stores k keys, hence, there are $\binom{P}{k}^2$ ways to choose keys for any two nodes N_s and N_t. Assume N_s and N_t have i keys in common. There are $\binom{P}{i}$ ways to choose these i common keys. After that, N_s and N_t each need $k-i$ distinct keys to create a key ring. These $2(k-i)$ are picked up from $P-i$ keys in the key pool. The number of ways is $\binom{P-i}{2(k-i)}$. The $2(k-i)$ keys are divided into two equal parts for N_s and N_t. There are $\binom{2(k-i)}{k-i}$ ways to divide these keys. Hence we have the probability that N_s and N_t share exact i keys is

$$p(i) = \frac{\binom{P}{i}\binom{P-i}{2(k-i)}\binom{2(k-i)}{k-i}}{\binom{P}{k}^2}.$$

N_s and N_t need at least q common keys to establish a link between them. That is, $i = 0, 1, ..., q-1$, so it is proved. □

For a given k, q, we can compute the largest P to satisfy the desired p. It is important to choose the largest possible P. Based on the theorem 2.2.1, there certainly are $P' < P$ that makes the desired connectivity $p' > p$, but $p' > p$ indicates each nodes maybe have extra direct connected neighbors, which allow an adversary to compromise more links with one captured node. Therefore, we need the largest P to just satisfy the desired connectivity p. In the following subsection,I show the resiliency analysis of the q-composite scheme.

3.2.2 Resiliency Analysis of the q-composite Scheme

In this subsection we will review how Chen et al. evaluate the resiliency of the new q-composite scheme. I would like to determinate the probability that an adversary can decrypt messages between nodes A and B if x nodes are captured and neither A or B are captured.

Assume each node in a network with q-composite scheme deployed stores k keys and the key pool size is P. Then after x nodes are captured, the probability that a key in a particular nodes's key ring has not been compromised is $(1-\frac{k}{P})^x$. Thus the probability that a key in common between A and B is compromised is $1-(1-\frac{k}{P})^x$. Consider that at least q keys are required to establish a secured communication link. Then after x nodes are captured, the probability that a q-composite link is compromised is $(1-(1-\frac{k}{P})^x)^i$, where $i(i \geq q)$ is the number of shared keys between two nodes. In order for A and B to communicate directly, they must have at least q keys in common, and at most k keys in a node. Thus, the number of keys used to secure the link between A and B varies from q to k. Therefore, the probability that the capture of x nodes compromises a link between two uncaptured nodes is given by:

$$\sum_{i=q}^{k}(1-(1-\frac{k}{P})^x)^i\frac{p(i)}{p},$$

where p is the requirement of local connectivity and $p(i)$ is the probability that exactly i keys are shared between two nodes. We know that the local connectivity $p = p(q) + p(q+1) + ... + p(k)$. Hence, $\frac{p(i)}{p}$ represents the probability that two directly connected nodes have exactly i keys in common. $P(i)$ has been presented in section 3.2.1.

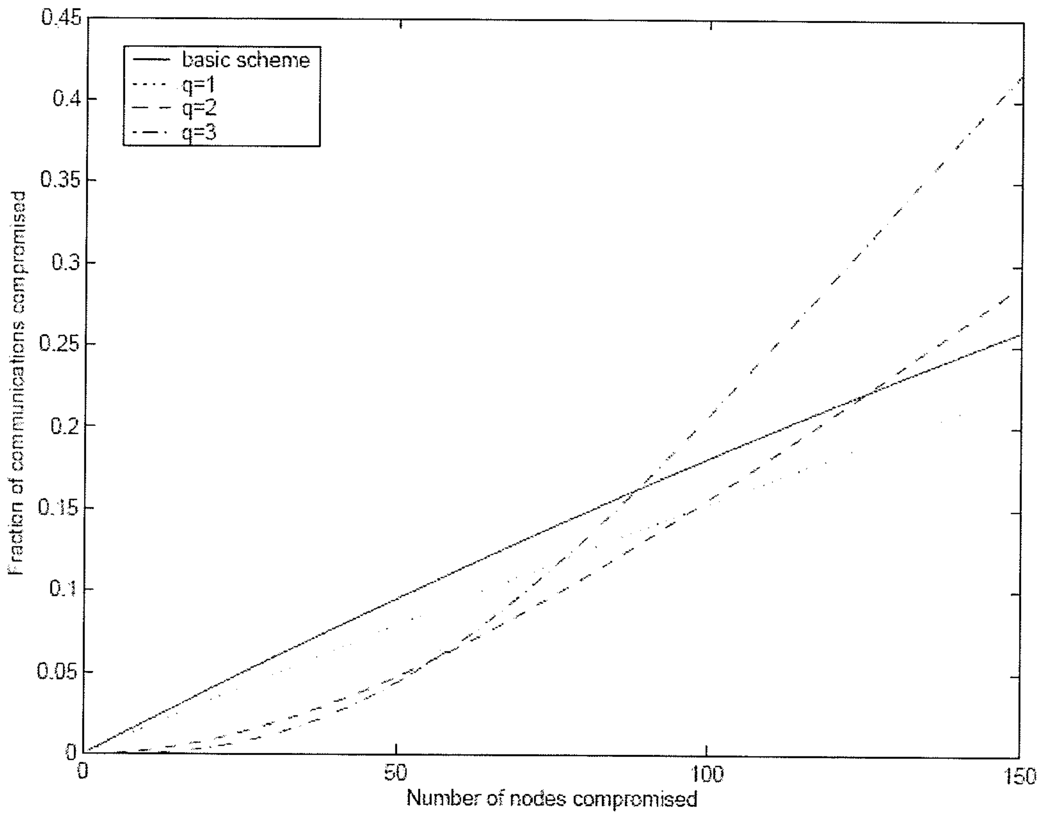


Figure 3.1: The probability that any link between two non-captured nodes can be decrypted by x captured nodes, where $k = 200$ and local connectivity $p_l = 0.33$ [Pietro et al., 2003]

Figure 3.1 (taken from [Chan et al., 2003]) shows that the q-composite scheme achieves better resiliency against node capture than the basic random scheme when the number of captured nodes is small. Chen et al. gives the following numerical example: Given $q = 2$, $k = 200$ and $p_l = 0.33$. When 50 nodes are captured, the probability that an additional communication link is compromised is 4.47% in the q-composite scheme and 9.52% in the basic scheme. However, when the number of captured nodes increases, additional links in the network with q-composite scheme deployed have higher probability to be compromised than the basic random scheme. Thus, the q-composite scheme should not be implemented in the case that the adversary may have chance to access large amount of nodes. In the next section, I describe another KPS that uses a pseudo-random method to generate key rings.

3.3 Pseudo-random Key Pre-distribution

In the basic random scheme, in order to discover neighbors of a node after the deployment, [Eschenauer and Gligor, 2002] set up some "trusted controllers" to temporarily store ID of each node and the key indexes associated to each node. Thus a node could determine its neighbor's ID(s) and the common key used to establish the link with that neighbor based on the information obtained from controllers. Unfortunately, some networks may not have such controllers. In addition, the existence of these controllers maybe too risky because the capture of a controller may allow an adversary to get detail information about the network. [Pietro et al., 2003] realized this problem and proposed a new KPS without using trusted controllers.

3.3.1 The Direct Protocol

Pietro et al. named their first KPS as *the direct protocol*, which is a probabilistic approach using *pseudo-random* key pre-distribution. The term *pseudo-random key pre-distribution* means that key indexes of a node are generated by an pseudo-random number generator, instead of truly randomly in the basic random scheme. In the basic random scheme, a node's key indexes are not computable by other nodes and can be regarded as truly random to all other nodes. Unlike the basic random scheme, the direct protocol generates keys indexes of a node based on a pseudo-random number generator, where a node's ID could be used as a seed. An important feature of this kind of pseudo-random number generator is that the generator will output the same sequence of numbers with the same input seed. For each node, a sequence of numbers are generated without repetition within the size of the key pool. Since the pseudo-random number generator and a node's ID is publicly known, other nodes could compute key indexes of the node locally. For example, based on a certain pseudo-random number generator, assume the key pool is large enough, node A has a sequence of key indexes as: $\{1, 7, 23, 55\}$. Then, all other nodes can compute the key indexes of node A. If another node B has the key indexes as: $\{3, 4, 7, 70\}$ and nodes A, B are distributed within each other's communication range, node B will regard node A as its neighbor. At the other end, node A could compute the key indexes of node B and find out the common key of index 7 between them. Note that nodes A and B only know the key indexes of each other, not the content of keys, except the common key(s). Messages between nodes A and B are encrypted using the key marked as 7, but the key itself would never be sent out.

Instead of using controller nodes, the direct protocol uses broadcast to perform the link establishment. Moreover, it uses all the common keys shared between two nodes to encrypt the link. The differences between the direct protocol and the basic random scheme are shown as follow:

1. Instead of randomly drawing keys from the key pool in the basic random scheme, every node generates a set of numbers pseudo-randomly, using the node's unique ID as the seed. These pseudo-randomly generated numbers are used as key indexes of that node. Key indexes are used to select keys from the key pool. The seed (ID) of each node and the pseudo-random number generating algorithm are publicly known.

2. Assume that the key ring size of each node is k and nodes A and B have $t(1 \leq t \leq k)$ keys in common. Let K_i be the ith common key between A and B, then the key $K_{a,b}$ that used to secure the link between nodes a and b is computed by:

$$K_{a,b} = \bigoplus_{i=1}^{t} K_i = K_{b,a},$$

,

where operator $\bigoplus$ stands for XOR. Note that in the construction of $k_{a,b}$, nodes

a and b use all the common keys between them. So, the direct protocol provides a solution when multiple keys are shared by two nodes. Note that a can calculate $K_{a,b}$ because the ID of b is publicly known. Thus, a can use b's ID as seed to generate key indexes used for node b. In addition, a only knows key indexes used for node b, but not the content of keys, except those common keys also stored in a. Node a computes $K_{a,b}$ then send an encrypted message to b. On the other side, b computes $K_{a,b}$ in the same manner to decrypt the message.

I will focus on the resiliency in the direct protocol. I would like to study how x captured nodes affect the rest of the network. In Section 3.2.2, I presented the the probability that a link between any two nodes A and B are compromised when x nodes are captured. Although the direct protocol uses pseudo-random key generator to generate key indexes, it is still a probabilistic approach for key pre-distribution. In addition, the direct protocol uses all common keys between two nodes to secure the link. Thus, it can be regarded as a special case of the q-composite scheme. It follows that the corruption probability of the link between any two uncaptured nodes A and B is given by:

$$\sum_{i=1}^{k}(1-(1-\frac{k}{P})^{x})\frac{p(i)}{p}.$$

Figure 3.2 (taken from [Pietro et al., 2003]) shows the probability that the link between two nodes are compromised as key ring size k increases. For a fixed key ring size k, the probability that an additional link is compromised is not satisfactory even when the number of captured nodes x is small. For example, when only $x = 16$ nodes are captured and $k = 100$, the probability is over 15%. Pietro et al. admit that the direct protocol does not suit the environment in which the security is highly desired. Therefore, they developed an advanced KPS: *the co-operative protocol.*

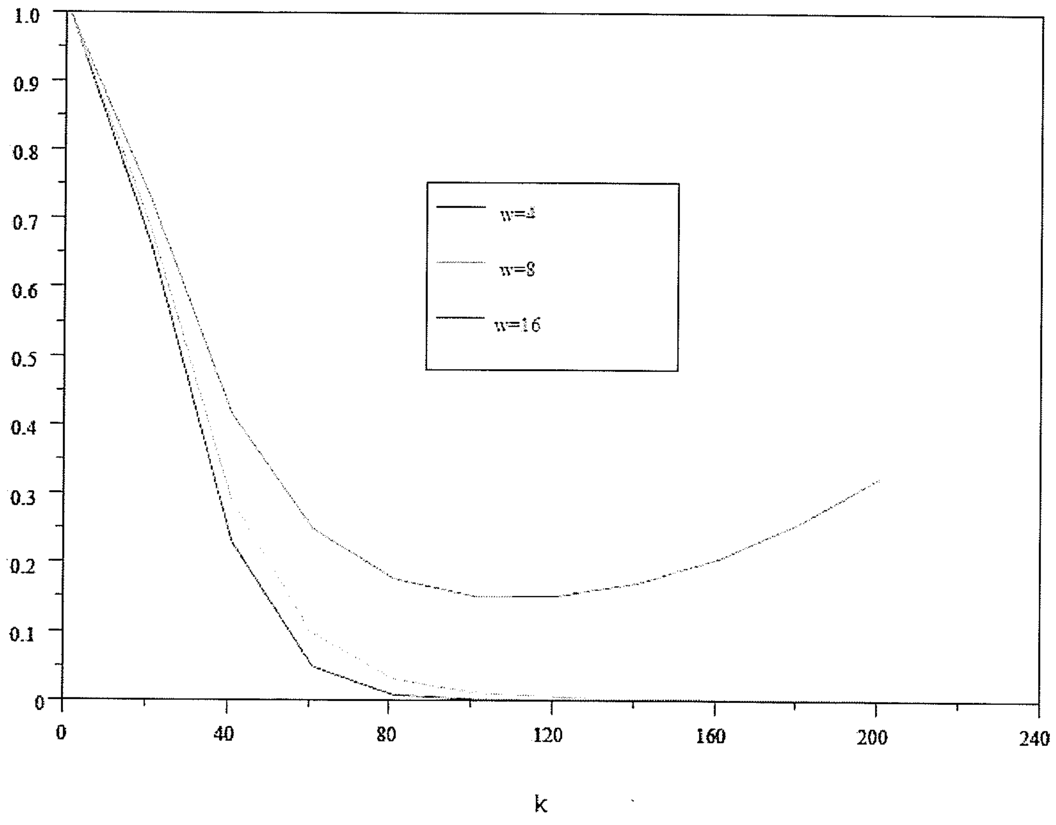


Figure 3.2: The probability that the link between two nodes are compromised when captured nodes $x = 4, 8, 16$ [Pietro et al., 2003]

3.3.2 The Co-operative Protocol

The *co-operative protocol* is built on top of the direct protocol to achieve enhanced security. The term *co-operative* means that a node chooses a set of other nodes to assist in encrypting messages. After the direct protocol is deployed, assume a sensor node a knows that it shares key(s) with another node b. In stead of sending request to b directly, node a selects $m(m \geq 0)$ additional nodes $C = \{C_1, C_2, ..., C_m\}$ that are directly connected to node b as co-operating nodes in its communication range. Sensor nodes in C may not be directly connected to node a, but by no more than two hops. First, node a sends requests to all nodes $C_i(i = 1, 2, ..., m)$ encrypted by using k_{a,C_i}, and the request carries information about the ID of the destination node b. Note that a and C_i establishes a secure link according to the direct protocol. After that, each C_i transforms its original key with b into a new hashed key, with the ID of a by:

$$hash(ID_a, k_{C_i,b}),$$

where $i = 1, 2, ..., m$ and $hash()$ is a one-way hash function. Then all the new $k_{C_i,b}$ are sent back to node a encrypted by using the symmetric key k_{a,C_i}. When a receives all the responses, a computes the new $k_{a,b}$ by

$$k_{a,b}^{new} = k_{a,b} \oplus (\bigoplus_{i=1}^{m} hash(ID_a, k_{C_i,b})).$$

Then a sends to b the list of sensor nodes C_i, such that b knows all the IDs of cooperative nodes between a and b. Moreover, b knows all the $k_{C_i,b}$. Therefore, node b has all the information to compute $k_{a,b}^{new}$. Both nodes a and b will use this new key for later communications. The direct protocol can be regarded as a special case of co-operative protocol when $m = 0$.

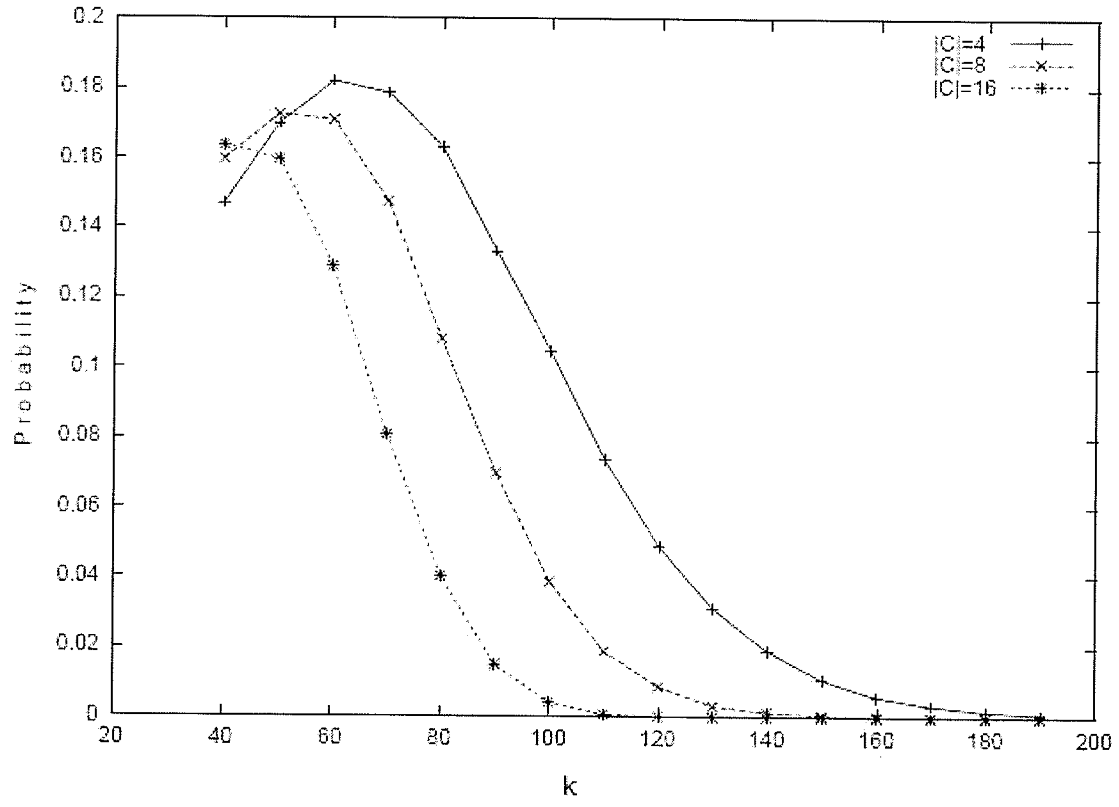


Figure 3.3: The probability that a link is compromised for different number of co-operating sensor nodes, $P = 10000$, captured nodes $s = 32$ [Pietro et al., 2003]

Pietro et al. [Chan et al., 2003] showed some simulation results of the new scheme. We review an example for a large-scaled network. With the pool size of $P = 10000$ and captured nodes $s = 32$, Figure 3.3 shows how different number of co-operating nodes affect the resiliency. With more cooperative nodes, the co-operative protocol achieves better resiliency when the same number of nodes are captured. When $|C| = 16$ and key ring size 80, the probability that a link is compromised is 0.04; when $|C| = 4$ and the same key ring size, the probability increases to 0.16.

Compared to the direct protocol in Figure 3.2, even with a small number of co-operative nodes ($|C| = 8$), the co-operative protocol achieves better resiliency against node capture. When $k = 100$ and 32 nodes are captured, the probability that a link is compromised is only 0.04, while in the direct protocol, 16 captured nodes make the probability of link corruption reach 0.15 (see Figure 3.1). With 32 captured nodes, the probability should be higher.

There is a price for the high resiliency in the co-operative protocol. The co-operative protocol generates more traffic than the direct protocol. As suggested by Pietro et al., improvement in reducing required messages for new key updating is desired for future co-operative protocols.

Chapter 4

Deterministic Key Pre-distribution Schemes

In previous chapters, I introduced several probabilistic key pre-distribution schemes. Probabilistic approaches have dominated the research of KPS designs since Eschenauer and Gligor introduced the first basic random scheme. Different probabilistic approaches emphasize on either less memory usage [Eschenauer and Gligor, 2002], or enhanced security [Chan et al., 2003; Pietro et al., 2003]. One problem of probabilistic approaches is that performance of random schemes varies in each deployment. In an application where a random scheme is applied, keys are randomly drawn from a key pool then loaded into each node, such that if we use a random scheme many times, it may generate totally different sets of key rings in each deployment. This may lead to different results in terms of of connectivity and resiliency. In some scenarios where security is highly desired, such variations are not tolerated.

To avoid the disadvantage of probabilistic schemes, another approach for studying the KPS problem was initiated in [Çamtepe and Yener, 2004], employing deterministic methods. Similar to probabilistic schemes, deterministic schemes also require a predefined key pool. The difference is that key rings are not selected randomly but in a deterministic manner. The basic idea is to use some sort of set system to represent the keys and key rings. A feature of deterministic schemes is that they produce same collection of key rings no matter how many times they are deployed, as long as the same underlying set system is used. Currently, the schemes that have been studied include using *Strongly Regular Graphs (SRG) [Lee and Stinson, 2004], Generalized Quadrangles(GQ) [Çamtepe and Yener, 2004], μ-Common Intersection Designs (μ-CID) [Lee and Stinson, 2005a]*, and *difference families [Wei and Wu, 2004]*. The first three schemes will be discussed in this chapter. The difference families scheme is generated by a product construction, which will be discussed in Chapter 5.

4.1 Preliminaries of Deterministic Schemes

To better understand deterministic schemes, I present some terminology from combinatorics. I begin by defining a set system and some combinatorial designs.

Definition 4.1.1. *A* set system *is a pair (X, B), where X is a set of points (or elements) and B is a collection of subsets (called blocks) of X.*

Definition 4.1.2. *In a set system (X, B), the* degree *of a point $x \in X$ is the number of blocks that point x occurs. If all points in X have the same degree r, (X, B) is* regular *of degree r. For any block $B_i \in B$, the size of a block B_i is the number of points in B_i. The* rank *of (X, B) is the size of the largest block. (X, B) is called* uniform *(of rank k, if k is the block size) if all blocks have the same size.*

The basic idea of deterministic schemes is to use some sort of set system to represent keys and key rings. In deterministic approaches, I only concentrate on uniform set systems. The reason is similar to random schemes: all nodes in a DSN are physically equivalent.

In a deterministic approach, we use a set system (X, B) to represent a KPS. First, let a set of points (X) represent the key pool. A key pool in a deterministic scheme plays the same role as in a random scheme. Secondly, let a collection of subsets of X correspond to key rings, each subset representing a key ring for a node. If the same rule of secure-link establishment is chosen as in the basic random scheme, then two nodes share a common key and can establish a secure link if their corresponding blocks intersect.

In a KPS using a set system, if each block is regarded as a vertex of a graph, and a link exists if two blocks have non-empty intersection, then the graph is called the *intersection graph* of the set system. In KPS designs, this is also known as the network graph of a scheme.

I now review a scheme with a complete network graph. In the $(n-1)$-key scheme where the network size is n, any node can establish secure channels with all other nodes. In deterministic approaches, the same complete network graph can be achieved by using *BIBDs*.

Definition 4.1.3. *A* (v, k, λ)*-BIBD (*Balanced Incomplete Block Design*) is a pair* (X, A)*, where* X *is a finite set and* A *is a set of* k*-subsets of* X *called blocks, such that* $|X| = v$ *and every pair of points occurs in exactly* λ *blocks.*

Definition 4.1.4. *A BIBD where the number of points equal to the number of blocks is called a* Symmetric BIBD.

Definition 4.1.5. *A* $(n^2+n+1, n+1, 1)$*-BIBD is a symmetric BIBD and also called* finite projective plane *of order* n*, where any pair of blocks have one common point.*

It is well known that, for a prime power n, there exists a finite projective plane of order n. Notice that a finite projective plane has the property that any two blocks intersect at exact one point. Therefore, a KPS using a finite projective plane will construct a fully connected network graph (complete graph). We now give an example with small parameters. A $(7, 3, 1)$-BIBD is a symmetric BIBD and also a finite projective plane of order 2. The blocks are $\{\{1, 2, 3\}, \{1, 4, 5\}, \{1, 6, 7\}, \{2, 4, 6\}, \{2, 5, 7\}, \{3, 4, 7\}, \{3, 5, 6\}\}$. In this $(7, 3, 1)$-BIBD, any two blocks have exactly one common point. Consider applying this BIBD to a KPS, we will have a complete intersection graph.

Compared to the scheme with $n - 1$ keys stored in each node, a finite projective plane scheme needs less memory space in each node. For example, If $n = 64 = 2^6$ is chosen, it can accommodate 4161 nodes in the target network. In the finite projective plane case, each node stores 65 keys while in the $(n-1)$-key scheme each node needs to store 4160 keys. However, the basic random scheme shows that the network graph does not have to be a complete graph. Practically, less memory usage is highly desired property and a connected network (not necessarily fully connected) is enough for message delivery. Various methods are suitable to reduce the memory usage, including using *Strongly Regular Graph, Generalized Quadrangles Design* and *μ-Common Intersection Design.*

4.2 Design KPSs using Strongly Regular Graphs

Unlike in the previous section, the *Strongly Regular Graph (SRG)* scheme [Lee and Stinson, 2004] is a deterministic scheme without using set systems. However, we can always map a set system to a network graph in KPS designs. Normally, we would introduce how key rings are generated first, then show the network graph and the physical graph. The reason that we introduce the *SRG* scheme first in this section is to give a graphical view of deterministic approaches, which will later make deterministic schemes more understandable. In fact, schemes using *SRG* are developed into a μ-CID scheme, in which set systems are adopted, by Lee and Stinson, same authors of *SRG* schemes. I introduce the μ-CID scheme in Section 4.4. I now describe strongly regular graphs.

Definition 4.2.1. *A* strongly regular graph, *denoted as $SRG(n, r, \lambda, \mu)$, is a regular graph where n is the number of vertices, r is the degree of each vertex, λ is the number of common neighbors of any two adjacent vertices, and μ is the number of common neighbors of any two non-adjacent vertices.*

If we consider a (n, r, λ, μ)-SRG as the network graph of a KPS, then any two nodes are either connected directly or via two-hop links (connected by μ common neighbors), which guarantees the network graph is a connected graph. However, after nodes are distributed into the target area, the connectivity becomes uncertain because the communication range of each node is limited such that some links may not exist any more. Thus, we should discuss the connectivity of a SRG in the physical graph. It is reasonable to consider the communication range of a sensor node as a circle, called a *neighborhood area.* Note that if in the network graph the degree of a node is r, then in a neighborhood, the probability that one node is connected to another node is $p = r/(n-1)$. Now consider two nodes u and v that are not connected directly in a neighborhood and find out the probability $p_{u,v}$ that they are connected within two hops. Let d be the number of nodes in a neighborhood, or density of the network. Let d' denote the number of nodes in the intersecting neighborhoods of u and v. Since u and v are located in the same neighborhood (either the circle centered at u or v), the distance between u and v is less than the radius of a neighborhood. It is well-known that the two circles intersect at least 1/3 the area of a neighborhood. Thus, we have $d' > \frac{d}{3}$. Furthermore, the probability that u and v are not connected via two hop is given by:

$$\binom{n-\mu}{d'} / \binom{n}{d'},$$

because nodes that *do not* connect u, v have to be selected from the intersection of two neighborhood. Now we generalize u, v to any two nodes in a neighborhood. The probability that they are connected within two hops (directly or via exact tow hops) is:

$$p^2(u,v) = p + (1-p)\left(1 - \binom{n-\mu}{d'} / \binom{n}{d'}\right),$$

which is

$$\approx p + (1-p)\left(1 - (1 - \frac{\mu}{n})^{d'}\right).$$

$$\geq p + (1-p)\left(1 - (1 - \frac{\mu}{n})^{\frac{d}{3}}\right).$$

Lee and Stinson give an example network of size 1024. With the network density $d = 40$, which is a reasonable neighborhood size, they construct a $(1024, 434, 186, 182)$-SRG, and achieve $p^2(u,v) \geq 0.9547$. That is, in the network, for any two nodes u and v in a neighborhood, very likely (with the probability 0.9547), they will be connected within two hops. This result set up a standard for later deterministic schemes: two-hop is enough. It seems that, compared to the random scheme, the connectivity in deterministic schemes is a little lower (0.9547 vs. 0.9999), but connectivity result in a SRG scheme is only obtained based on at most two-hop links. In fact, based on Lee and Stinson's illustration, such a connectivity is enough for most sensor networks. We now introduce schemes using SRGs.

4.2.1 The Basic ID-based one-way function schemes

The previous analysis shows the connectivity of a SRG. In this subsection we will introduce how Lee and Stinson construct a KPS using a SRG as the network graph. In the new scheme, each sensor node is given a unique ID and this is the reason that the scheme is called ID-based KPS. In addition, Lee and stinson used a public one-way hash-function h to reduce the memory usage, thus the scheme is called ID-based one-way function scheme. Simply, we call it IOS in this thesis.

SRG has been thoroughly studied. It is known, and very important that,

Theorem 4.2.1. *A regular graph G of order n and even degree r has an edge decomposition into star-like subgraphs such that each vertex acts as a center in one star-like subgraph and a leaf in $r/2$ distinct subgraphs.*

It is well known that in a regular graph, the total number of edges $E = \frac{1}{2}nr$, where n is the number of nodes and r is the degree of each node. Let r be even, we could take any node A and $\frac{1}{2}r$ edges connected to A as a subgraph. Since a node has r edges connected to it in a regular graph, there are other $\frac{1}{2}r$ edges that involve node

A. Each of these edges will be used in the subgraph centered at an adjacent node of A.

A SRG is a regular graph, and if we choose a SRG with even degree r, we could always decompose the SRG into star-like subgraphs. For example, a (1024, 434, 186, 182)-SRG is a regular graph with degree $r = 434$ that could be decomposed into star-like subgraphs where each vertex acts as the center of one subgraphs and a leaf of 217 other subgraphs.

Now consider the KPS using a SRG as the network graph. The SRG was decomposed into star-like subgraphs. Each node stores its own ID. Each node u in a star-like subgraph centered at node v is allocated K_u, its unique secret key, and the hashed key $h(K_v||ID(u))$, using a hash function h. When a node u attempts to communicate with centered node v, it uses the key $h(K_v||ID(u))$ to encrypt messages, while at the other end, node v is able to compute $h(K_v||ID(u))$ since it knows both K_v and $ID(u)$. Thus, a secure link is established.

In a IOS, each node stores $r/2 + 1$ keys, which is only half number of keys as in a fully connected projective plan scheme. Since one-way hash function is used, the IOSs has the perfect resiliency against node capture. When a node u is compromised, the adversary only obtains K_u and $h(K_{v_i}||ID(u))$ for $r/2$ $v_i s$. The rest of the communication links will remain secure because h is a one-way function and it is not feasible to decrypt K_{v_i} from $h(K_{v_i}||ID(u))$.

One issue in IOSs is that the largest supported network size is limited. It is constrained by the construction of SRGs. Since the construction of a very large size of SRG is not easy, the basic IOS scheme is only suitable for small sized network. In order to extend the IOSs to large-scaled networks, Lee and Stinson also presented a multiple SRG scheme, using multiple copies of a single SRG.

4.2.2 The Multiple ID-based One-way Function Scheme

In the multiple SRG scheme, nodes in the network are divided into m groups, either randomly or by node indexes, each consisting of l nodes. That is, network size is $n = ml$. For a group of nodes u, nodes are denoted as $u_1, u_2, ..., u_l$. If we regard a group as a single node, the multiple SRG scheme is as same as the basic IOS scheme. All the nodes in a group u is allocated the group common key k_u. Now regard a group of nodes as one vertex in the network graph and the graph is decomposed into star-like subgraphs. If a group u is in a leaf of a star-like subgraph centered at a vertex v, which is also a group of nodes, each node in group u will receive a hashed key $h(k_v||ID(u_i))$. On the other end, any node in a group v can compute $h(k_v||ID(u_i))$ since u_i is a publicly known ID. Thus, secure links are established between group u and group v. In the multiple SRG scheme, communications within a group is not allowed, which means $u_1, u_2, ..., u_l$ are regarded as sharing no common key.

The memory usage of the multiple SRG scheme is $k = r/2 + 1$, only roughly $1/l$ times smaller than using a IOS scheme. The multiple IOS scheme achieves much less memory usage than the basic IOS scheme. However, it sacrifices the perfect resiliency.

In order to decrypt a specific link between group u and v where v is at the center of star-like subgraph, say the link between node u_i to v_j, an adversary has to capture a node in group u other than u_i or v_j. It follows that the probability that a link is compromised after s nodes have been captured is computed as:

$$p = 1 - \frac{\binom{n-l-1}{s}}{\binom{n-2}{s}}$$

The basic IOS scheme achieves perfect resiliency against node capture, however, it can not be used to construct KPSs when a SRG of a certain number of nodes is unknown or too difficult to construct. Multiple IOS is introduced to solve this issue. It provide a trade-off between resiliency and memory usage. Moreover, Lee and Stinson combined the IOS scheme and Blom's scheme to construct an advanced scheme, which will be introduced in Chapter 5.

4.3 Design KPSs Using Combinatorial Designs

The previous section gives a graphical view on how deterministic approaches work. In this section, we will present how deterministic approaches work by using set systems, specifically, *combinatorial designs*. We will introduce two combinatorial designs that are capable of being deployed as KPSs. First I discuss the *Generalized Quadrangles Design*.

4.3.1 Generalized Quadrangles Designs

In Section 4.1, I introduced a mapping from a *finite projective plane* to a KPS, in which the network graph is a complete graph. However, a design in which the intersection graph is a connected graph is enough to construct a KPS. [Çamtepe and Yener, 2004] realized this point and introduced the first deterministic scheme (even earlier than the two SRG schemes) using *Finite Generalized Quadrangles (GQ)* in 2004.

Definition 4.3.1. *A* Finite Generalized Quadrangles (GQ) *is an incidence relationship $S = (P, B, I)$ where P is nonempty sets of points (or elements) and B is nonempty sets of lines (or blocks), and I is a point-line relationship such that:*

1. *There exists a $t(t \geq 1)$, such that each point occurs in exact $t + 1$ lines and each pair of points appear in at most one line.*

2. *There exists a $s(s \geq 1)$, such that each line contains $s + 1$ points and two lines have at most one point in common.*

3. *if a line L does not contain point x, then there exists another line contains both point x and another point y, such that point y is in line L.*

Thus a GQ is denoted as $GQ(s, t)$. Consider mapping from a $GQ(s, t)$ to a set system (X, B), that is, lines are mapped to blocks. Suppose $|X| = v$ and $B = b$. We have $v = (st + 1)(s + 1)$ and $b = (st + 1)(t + 1)$. [Çamtepe and Yener, 2004] choose three known GQs to construct KPSs: GQ(q, q), GQ(q, q^2) and GQ(q^2, q^3). For example, when mapping a GQ(q,q) to a set system, we have $s = t = q$ and $v = b = (q + 1)(q^2 + 1)$.

[Çamtepe and Yener, 2004] shows that these GQs have connected intersection graphs. The detail of applying a GQ to a KPS will be shown in the next subsection.

4.3.2 From Generalized Quadrangles to Key Distribution

In this section, I discuss how to applying GQs, t: $GQ(q, q), GQ(q, q^2), GQ(q^2, q^3)$ to KPSs in [Çamtepe and Yener, 2004].

When mapping from a set system to a KPS, we regard a point as a key. In a GQ, we regard a block (line) as a key ring of a sensor node. Note that in a GQ each point occurs in $t + 1$ blocks and each block has $s + 1$ points, which means a block intersects with exactly $t(s + 1)$ other blocks. That is, in the network graph a node directly connects to other $t(s + 1)$ nodes. In addition, if two blocks A and B do not share any common point, there exists another $s + 1$ blocks which intersect both of them. In the network graph, these $s + 1$ intermediate nodes (blocks) connect nodes A and B. Table 4.1 from [Çamtepe and Yener, 2004] shows the parameters of $GQ(q, q), GQ(q, q^2)$ and $GQ(q^2, q^3)$. In additional, the probability that any two blocks share at least one points, or the probability that any two nodes share at least one common key, is:

Designs	v	b	r	k
Symmetric BIBD	n^2+n+1	n^2+n+1	$n+1$	$n+1$
$GQ(n,n)$	n^3+n^2+n+1	n^3+n^2+n+1	$n+1$	$n+1$
$GQ(n,n^2)$	n^4+n^3+n+1	$n^5+n^3+n^2+1$	n^2+1	$n+1$
$GQ(n^2,n^3)$	$n^7+n^5+n^2+1$	$n^8+n^5+n^3+1$	n^3+1	n^2+1

Table 4.1: Parameters v, b, r, k for Symmetric BIBD and GQs [Çamtepe and Yener, 2004]

$$P_{GQ} = \frac{t(s+1)}{b} = \frac{t(s+1)}{(t+1)(st+1)}.$$

With the same memory usage, the *GQ scheme* achieves better connectivity than the basic random scheme. *GQ* is a thoroughly researched topic in combinatorics, but similar to the *SRG scheme*, it is also difficult to construct large-parametered *GQ*s for some desired number of nodes. The GQ scheme has been developed into a hybrid design (but not a product construction) for large sized network. We will discuss more about GQ and its properties in the next subsection.

4.3.3 Hybrid Designs for Large-scaled Networks

Çamtepe and Yener also realized that the non-existence of designs with some certain number of nodes so that they considered a *hybrid scheme* to solve the problem by combining deterministic approaches and probabilistic approaches. They constructed a *GQ* and its *Complementary Design.*

Definition 4.3.2. *Given a design D with a set X of $|X| = v$ points and blocks $B = \{B_1, B_2, ..., B_b\}$, the* Complementary Design $\overline{D}$ *contains all the complement*

Designs	v	b	r	k
$Complementary\ Symmetric$	n^2+n+1	n^2+n+1	n^2	n^2
$ComplementaryGQ(n,n)$	n^3+n^2+n+1	n^3+n^2+n+1	n^3+n^2	n^3+n^2
$ComplementaryGQ(n,n^2)$	n^4+n^3+n+1	$n^5+n^3+n^2+1$	n^5+n^3	n^4+n^3
$ComplementaryGQ(n^2,n^3)$	$n^7+n^5+n^2+1$	$n^8+n^5+n^3+1$	n^8+n^5	n^7+n^5

Table 4.2: Parameters v, b, r, k for Symmetric BIBD and GQs' complementary designs [Çamtepe and Yener, 2004]

blocks $\overline{B_i} = X - B_i$.

Çamtepe and Yener use a GQ plus some randomly selected blocks from its complementary design when the desired GQ does not exist with a particular b. Table 4.2 shows the parameters for some GQ's complementary designs. Suppose we need a KPS to accommodate a network with size n, buy the key ring's size is limited meaning that we do not have enough blocks in our design. The hybrid scheme first generates a GQ consisting $b(b < n)$ blocks. Then we still need extra $n - b$ blocks for the implementation. Çamtepe and Yener considered to use subsets of blocks in the complementary design to create key rings. They randomly choose $n - b$ blocks from the complementary design of the base GQ. However, for a GQ, $v - k > k$, thus the blocks of blocks in the complementary design can not be used as the key ring directly (exceed the key ring size). Çamtepe and Yener then selects $n-b$ k-subsets along with the b blocks in the base design to form all n blocks for the hybrid scheme. Note that this hybrid scheme can also use a symmetric BIBD as the base design and add extra blocks from k-subsets of its complementary design.

A simple example is given in [Çamtepe and Yener, 2004]. Suppose we need a network of size 10, key ring size 3, and we use a symmetric design $(v, b, r, k) = (7, 7, 3, 1)$ as base design in a hybrid scheme. Thus we set of blocks $B = \{\{1, 2, 3\}, \{1, 4, 5\}, \{1, 6, 7\}, \{2, 4, 6\},\{2, 5, 7\},\{3, 4, 7\},\{3, 5, 6\}\}$. We still need 3 more blocks. Then we construct a complementary symmetric design $\overline{B} = \{\{4, 5, 6, 7\},\{2, 3, 6, 7\},\{2, 3, 4, 5\},\{1, 3, 5, 7\}, \{1, 3, 4, 6\},\{1, 2, 5, 6\}, \{1, 2, 4, 7\}\}$. Assume the 3 randomly selected 3-subsets are $\{4, 5, 7\}$ from $\{4, 5, 6, 7\}$,$\{2, 3, 7\}$ from $\{2, 3, 6, 7\}$, and $\{1, 3, 5\}$ from $\{1, 3, 5, 7\}$. Let $H = \{4, 5, 7\}, \{2, 3, 7\}, \{1, 3, 5\}$, then the final hybrid design is $B \cup H = \{\{1, 2, 3\}, \{1, 4, 5\}, \{1, 6, 7\}, \{2, 4, 6\}, \{2, 5, 7\}, \{3, 4, 7\}, \{3, 5, 6\}, \{4, 5, 7\}, \{2, 3, 7\}, \{1, 3, 5\}\}$.

As far as I am aware, this is the only research in which the problem that desired designs are unknown were considered. However, such a hybrid scheme sacrifices the stable performance of deterministic schemes such that connectivity and resiliency may vary in each deployment.

Çamtepe and Yener give some numerical examples comparing the GQ scheme and the basic random scheme. For the same network size, when key ring size $k = 18$ and key pool size $P = 5220$, the probability that a link exists between any two nodes is 0.060 in the basic random scheme and 0.058 in the GQ scheme. With other numbers of k and P, the GQ scheme and the basic random scheme also achieve very close connectivity. Çamtepe and Yener does not provide resiliency evaluations for the GQ based schemes because they emphasize the providing of stable performance. Similar approaches will be introduced in the next section with resiliency analysis.

GQ is thoroughly studied in combinatorics, however, it is not specifically designed for the KPS problem. Since it is the very first deterministic scheme for the KPS problem, some detail properties of applying GQ to a KPS are not clearly illustrated, such as: If a line L does not contain point x, then there exists another line contains both point x and another point y, such that point y is in line L. This indicates that if two blocks do not have common points, they will be connected in the intersection graph via intermediate node(s). But how many intermediate nodes, and how these nodes affect the connectivity remain unclear.

[Lee and Stinson, 2005a] proposed another deterministic scheme, using *combinatorial designs*, and this time, the design is specially developed to serve the key pre-distribution problem.

4.4 μ-Common Intersection Designs

For the development of KPSs, especially deterministic schemes, some features are clearly desired: a node needs to directly connect to some other nodes by sharing a common key, and if two nodes do not share a common key, there should exist two-hop path(s) to connect them. Direct and two-hop links are sufficient to form a connected network graph [Lee and Stinson, 2005a]. Motivated by such an observation, Lee and Stinson developed a *μ-Common Intersection Design (μ-CID)* to address the KPS issue. We now introduce more concepts and terms related combinatorial designs.

Definition 4.4.1. *A (X, B) set system is called a $(v, b, r, k) - 1 - design$ if it is regular of degree r and uniform of a fixed block size k, where $|X| = v$ and $|B| = b$.*

Definition 4.4.2. *A $(v, b, r, k) - 1 - design$ is called a $(v, b, r, k) - configuration$ if and only if any two distinct blocks have at most one common point.*

μ-Common Intersection Designs (μ-CID) are specially designed for the KPS problem. Lee and Stinson [Lee and Stinson, 2005a] defined the following.

Definition 4.4.3. *a (v, b, r, k)-μ-*common intersection design (μ-CID) *is a $(v, b, r, k)-$ $configuration$, where $B = \{B_1, B_2, ..., B_b\}$, and*

$$|\{B_h \in B : B_i \cap B_h \neq \emptyset \text{ and } B_j \cap B_h \neq \emptyset\}| \geq \mu$$

when $B_i \cap B_j = \emptyset$ for all i, j.

Lee and Stinson showed that μ-CID has all desired features to construct a KPS. Any two nodes N_i and N_j communicate via either a directly connected secure link or via μ intermediate nodes if they do not have any common keys. A example of μ-CID with small parameters is shown in [Lee and Stinson, 2005b]. It is a $(12, 12, 3, 3)$ μ-CID, where $\mu = 3$. In the 3-CID, $X = \{1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c\}$ and $B = \{\{1, 5, 9\}, \{1, 7, c\}, \{1, 8, a\}, \{2, 5, c\}, \{2, 6, a\}, \{2, 8, b\}, \{3, 6, c\}, \{3, 7, b\}, \{3, 8, 9\}, \{4, 5, a\}, \{4, 6, b\}, \{4, 7, 9\}\}$. In this set system, any two blocks intersect in at most one point. Most important, if two blocks have no common point, such as block $\{1, 5, 9\}$

and block$\{2, 6, a\}$, the there exist three blocks, $\{4, 5, a\}, \{1, 8, a\}, \{2, 5, c\}$, intersect both of them.

Let us consider the connectivity in the network applying a μ-CID. Each block (node) contains k keys, and each key appears r times in the network, such that each node connects to $k(r-1)$ other nodes. There are $b-1$ other nodes in the network. Therefore for any two nodes N_i and N_j that in the same neighborhood in the physical graph, the probability that they connect directly is given by:

$$p_1 = \frac{k(r-1)}{b-1}$$

.

If s nodes are located in the intersection of the neighborhood of N_i and N_j and if they do not share a common key, then there are at least μ intermediate nodes that would connect them. The probability that none of the μ nodes are in the common neighborhood is estimated as:

$$(1 - \frac{\mu}{b-2})^s$$

.

It follows that N_i and N_j are connected either directly or via μ other nodes with the probability:

$$p \approx p_1 + (1 - p_1) \times (1 - (1 - \frac{\mu}{b-2})^s)$$

.

For example, in a $\mu - CID$ with $v = 1470$ and $k = 30$, Lee and Stinson [Lee and Stinson, 2005a] achieved the global connectivity of 0.99995 within two hops. Compared to the SRG scheme (0.9547), the global connectivity in the μ-CID scheme is greatly improved.

Let us consider the resiliency. The probability that any link is affected by some other random compromised nodes is $(r-2)/(b-2)$ because there are exactly $r-2$ other nodes that contain the communication key of this link. Therefore x randomly captured nodes will affect a given link with probability:

$$p(s) = 1 - (1 - \frac{r-2}{b-2})^x$$

.

It follows that,in the above network when 10 nodes are captured, the adversary has around 18% probability to compromise any other link. There are two problems in the current μ-CID scheme, the first one is the applications that could have μ-CID scheme deployed is very limited, because it is very hard, or impossible to construct a μ-CID with desired number of blocks. The second problem is that, as shown, the resiliency of μ-CID is not satisfactory for applications that need enhanced security.

In Chapter 6, we attempt to provide an enhanced μ-CID scheme, in which the memory usage is slightly increased while the resiliency is greatly improved. Before that, we would like to introduce a methodology first.

Chapter 5

Product Construction of Key Pre-distribution Schemes

In previous chapters, I introduced both probabilistic and deterministic approaches in designing KPSs for DSNs. I will present some new KPSs in Chapter 6. Before that, I would like to introduce a methodology that would lead to new schemes. Previous schemes in Chapter 1 to 4 covered many areas, such as probability theory, graph theory, combinatorial designs, networking, and cryptography, etc. However, these schemes do not provide any general method for designing a KPS. [Wei and Wu, 2004] generalized the KPS problem and contributed a method, known as the *product construction*, to derive new KPS from existing ones. Wei and Wu viewed *product construction* as a method of using different copies of schemes to produce a new scheme, such that *Du's scheme* [Du et al., 2003] may be regarded as a *product construction* of the *basic random scheme* [Eschenauer and Gligor, 2002] and the *Blom's scheme* [Blom, 1985]. The reason that Wei and Wu introduced the new method is that, every scheme involved in a the product construction has certain desired properties, such that the combination of these schemes (two or more schemes) will take advantages of all of them (drawbacks as well).

I will introduce some existing schemes that can be regarded as using product construction in this chapter, including *pairwise key pre-distribution scheme* (Du's scheme) in [Du et al., 2003], *pairwise key scheme* in [Liu et al., 2005], *difference family scheme* in [Wei and Wu, 2004] and the *deterministic multiple space Blom's scheme* in [Lee and Stinson, 2004]. Although some schemes had been presented before Wei and Wu generalized the method of product construction, so that they were not formally mentioned as using product constructions. I would like to classify them this way in this chapter because the construction of this class of schemes follow the similar methodology.

I would like to introduce how Wei and Wu construct their scheme using a product construction, However, they use Du's scheme [Du et al., 2003] as one of the original schemes. Thus, we will introduce Du's scheme first, which is a product construction of Blom's scheme and the basic random scheme. Since the basic random scheme has been introduced in detail in Chapter 2, I focus on describing Blom's scheme, which will be used in several later product constructions. After that, I show the generalized product construction, produced by Wei and Wu, and their new scheme using a product construction. Some background knowledge about set system and combinatorial designs have already been introduced in Chapter 4. Thus, I only introduce new terms and concepts related to these new schemes.

5.1 Blom's Scheme in Product Constructions

Several papers involve Blom's scheme for product constructions. In this section, we will briefly introduce the Blom's scheme.

In 1985, Blom proposed a key pre-distribution scheme that can establish a secure channel between any pair of nodes in the network, in which each channel uses its individual secret key. The idea of Blom's scheme is some sort of matrix manipulations. Blom's scheme is considered to be λ-*secure*, which means the network has perfect resiliency if no more than λ nodes are captured.

Several matrices are used in Blom's scheme. The first matrix is a $(\lambda + 1) \times N$ matrix G over a finite field $GF(q)$, where N is the number of nodes in the network, λ is the desired security factor and $q(q > N)$ is a prime power. A finite field $GF(q)$ is a finite set of q elements denoted as $0, 1, 2, ..., q - 1$. Moreover, the matrix G is publicly known and may be shared by different systems and users, even adversaries. The second matrix is a randomly generated $(\lambda + 1) \times (\lambda + 1)$ symmetric matrix D over $GF(q)$. Matrix D is the secure information of the scheme and should not be exposed to any sensor nodes or adversaries. The third matrix A is constructed from $A = (D \cdot G)^T$, where $(D \cdot G)^T$ is the transpose of $D \cdot G$. Since D is symmetric, it is easy to prove that

$$A \cdot G = (D \cdot G)^T \cdot G = G^T \cdot D^T \cdot G = G^T \cdot D \cdot G = (A \cdot G)^T.$$

If weset $K = A \cdot G$, then K is also a symmetric matrix. Since K is a $N \cdot N$ matrix, where N is the number of nodes in the network, we can assign each node a row and a column of K and let $K_{ij} = K_{ji}$ (symmetric matrix) represents the key used between nodes i and j. Motivated by this analysis, node k stores the kth row of matrix A and the kth column of matrix G. Since matrix A has N rows and matrix G has N columns, both assignments are feasible. Therefore, any two nodes are able to establish a secure channel by exchange their own columns of G and compute K_{ij} or K_{ji}.

To achieve the desired λ-secure feature in Blom's scheme, any $\lambda + 1$ columns of G have to be linearly independent. Such a matrix was designed before Blom's scheme in 1977 in [MacWilliams and Sloane, 1977]. First, Macwilliams and Sloane choose a finite field $GF(q)$. As required, q has to be larger than the network size N. In addition, q depends on the size of a key. For example, if each key is designed to occupy up to 64 bits memory, then some prime number $q \geq 2^{64}$ is chosen. This value is large enough to accommodate most network size. Then every element in matrix G is designed to be $s(s < q)$ in [MacWilliams and Sloane, 1977] as follow:

$$\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ s & s^2 & s^3 & \dots & s^N \\ s^2 & (s^2)^2 & (s^3)^2 & \dots & (s^N)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s^\lambda & (s^2)^\lambda & (s^3)^\lambda & \dots & (s^N)^\lambda \end{bmatrix}$$

This matrix G has the desired properties for resource limited environments. Since every element is a powers of s, each node only needs to store the seed s^k to compute the entire column k.

The network graph of Blom's scheme is a complete graph, where each pair of nodes have their unique pairwise key. We now introduce how Du et al, combine Blom's scheme and the basic random scheme.

5.1.1 Blom's Scheme and the Basic Random Scheme

Blom's scheme achieves good resiliency (λ-secure) against node capture. However, Blom's scheme was designed for establishing pairwise links between any two nodes in the network. Thus, the network graph is a complete graph. Although Blom's scheme uses less memory than other pairwise schemes, such as the $(n-1)$-key scheme, a complete network graph is not necessary in a KPS for DSNs. It was proved in the basic random scheme, a connected graph is sufficient for DSNs [Eschenauer and Gligor, 2002]. It means Blom's scheme could be optimized to save memory usage by changing the network graph from a complete graph to a connected graph. Du et al. [Du et al., 2003] adopted the random selection from the basic random scheme [Eschenauer and Gligor, 2002] and proposed a *multiple-space key pre-distribution scheme.* As in many other papers, I call it Du's scheme from now on. When introducing Du's scheme, I use $\omega = P$ to represent the key pool size and $\tau = k$ to represent the key ring size. In addition, I call a matrix $A \cdot G$ in Blom's scheme as a key space, which is a key in the key pool of Du' scheme and contains a matrix. The procedure of constructing a Du's scheme is shown as follows:

First, a $(\lambda + 1) \times N$ matrix G is generated by using the method introduced in the previous section. Note that only seed s^k needs to be stored in the memory of a node k, and s^k is unique for each node, such that it can also be used as the identity (ID) of a node.

Second, generate ω (like the key pool in the basic scheme) different $(\lambda+1) \times (\lambda+1)$ random, symmetric matrices $D_1, D_2, ..., D_\omega$. Note that in Blom's scheme, there is only one random generated matrix D. Du et al. modified the Blom's scheme here. Now it is feasible to compute corresponding ω different $A_i = (D_i \cdot G)^T, i = 1, 2, ..., \omega$. Let $A_i(j)$ represent the jth row of A_i.

Lastly, for each node, randomly select τ $(2 \leq \tau \leq \omega)$ matrices from $\{A_1, A_2, ..., A_\omega\}$. If A_i is selected for node j, the jth row of A_i will be stored in node j. This is the secret information for this node and would not be sent to any other node. Now if two nodes selected rows from the same A_i, they can establish a secure link. Since they both have one row of A_m (assume A_m is their common matrix) and one column of matrix G, they can exchange their column of G then compute their security key.

The memory usage for the new Du's scheme is less than Blom's scheme. Since A_i has $\lambda + 1$ columns, each node needs to store $(\lambda + 1)\tau$ elements in memory. The actual memory cost should be $(\lambda + 1)\tau$ times the length of each key. If fact, there is another one element memory cost which is the seed of a column of matrix G. Since the seed also acts as key identity, Du et al. ignore it when calculating the memory cost.

Since Du's scheme is the product construction of Blom's scheme and the basic random scheme, and the the Blom's scheme is only used as the key space (a scheme provides key pool), the computing of the pool size (ω) and the key ring size (τ) is the same as the basic random scheme.

5.1.2 The Security Analysis of Du's scheme

Du et al. analyze the new product scheme using the following evaluations: *the probability of at least one key space being broken* and *the fraction of compromised network communication.* We now explain both.

The first evaluation is the probability that at least one key space is broken after x nodes are captured. Since Blom's scheme is λ-secure, then an adversary needs to compromise at least $\lambda + 1$ nodes to break one key space. Du et al. showed the following analysis in case that x nodes are captured.

Let S_i denote the event that the ith ($i = 1, 2, ...\omega$) key space is broken. Let C_x

represent the event that x nodes are captured by an adversary. It follows that,

$$Pr(\text{at least one space is broken}|C_x) = Pr(S_1 \cup S_2 \cup ... \cup S_\omega|C_x).$$

Since all event S_i for $i = 1, 2, ..., \omega$ are equally likely, we apply the union bound and have:

$$Pr(S_1 \cup S_2 \cup ... \cup S_\omega|C_x) \leq \sum_{i=1}^{\omega} Pr(S_i|C_x).$$

Because the probability that any key space is broken is same, it follows that:

$$Pr(\text{at least one space is broken}|C_x) \leq \omega Pr(S_1|C_x).$$

Note that $Pr(S_1|C_x)$ is the probability that the first key space A_1 is broken when x nodes are compromised. In Du's scheme, each node stores τ keys. Thus, the probability that a node knows the information about A_1 is $\theta = \frac{\tau}{\omega}$. Suppose among x nodes, there are exact j nodes contain the information about A_1. In order to break the λ-secure feature, j has to be larger than λ. The probability that these j nodes contains information about A_1 is $\binom{x}{j}\theta^j(1-\theta)^{x-j}$. Then we have the result as following:

$$Pr(S_1|C_x) = \sum_{j=\lambda+1}^{x} \binom{x}{j}\theta^j(1-\theta)^{x-j}.$$

Finally we have:

$$Pr(\text{at least one space is broken}|C_x) \leq \omega \sum_{j=\lambda+1}^{x} \binom{x}{j} (\frac{\tau}{\omega})^j (1-\frac{\tau}{\omega})^{x-j}.$$

The second evaluation is the fraction of compromised network communication when x nodes are captured. This evaluation indicates how x captured nodes effect the remaining part of the network. We first consider one additional link that is not included in the captured nodes, denoted as c. Assume link c uses K as a communication key. Let event B_i denote the event that k is in a compromised key space $S_i, i = 1, 2, ..., \omega$. Then given x compromised nodes, the probability that link c is also compromised is:

$$Pr(c \text{ is broken}|C_x) = Pr(B_1 \cup B_2 \cup ... \cup B_\omega|C_x).$$

For mutually exclusive events $B_1, B_2, ..., B_\omega$, we have:

$$Pr(\text{c is broken})|C_x = \sum_{i=1}^{\omega} Pr(B_i|C_x) = \omega \cdot Pr(B_1|C_x).$$

Now the desired probability equals ω times the probability that B_1 occurs. Note that,

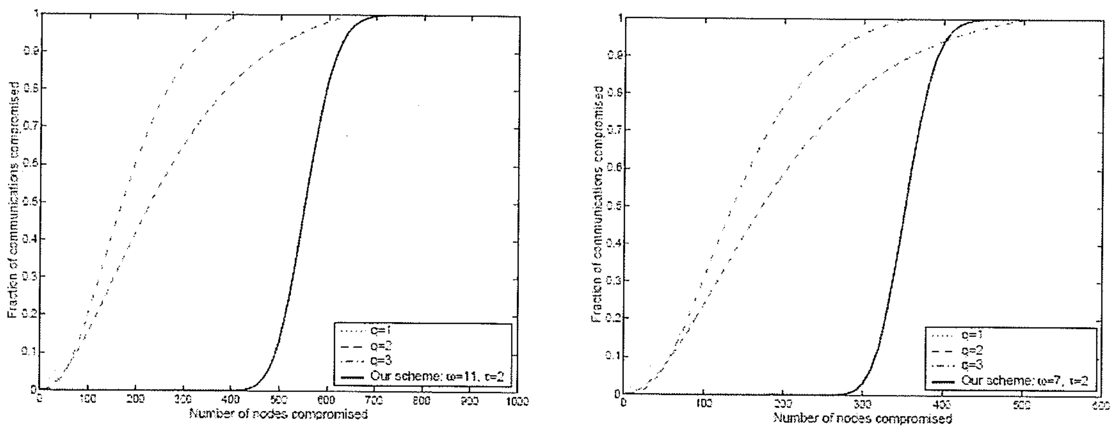
$$Pr(B_1|C_x) = \frac{Pr((K \in S_1) \cap (S_1 \text{ is compromised} \cap C_x)}{Pr(C_x)},$$

and $k \in S_1$ do not depend on the event C_x, then

$$Pr(B_1|C_x) = Pr(K \in S_1) \cdot Pr(S_1 \text{ is compromised}|C_x).$$

Since keys are randomly selected from ω possible key spaces, then $Pr(K \in S_1) = 1/w$. Therefore,

$$Pr(c \text{ is broken}|C_x) = \omega \cdot \frac{1}{\omega} \cdot Pr(S_1 \text{ is compromised}|C_x) = \sum_{j=\lambda+1}^{x} \binom{x}{j} (\frac{\tau}{\omega})^j (1 - \frac{\tau}{\omega})^{x-j}.$$



(a) $m = 200$. $p_{actual} = 0.33$

(b) $m = 200$. $p_{actual} = 0.5$

Figure 5.1: The probability that a link can be compromised after x nodes are captured. m is the memory usage and P_{actual} is the probability that any two nodes in a neighborhood are able to establish a secure link. Others are the probability for the q-composite scheme, when $q = 2, 3$ and $q = 1$(roughly equals to the basic scheme) [Du et al., 2003]

Figure 5.1 shows the probability that a link is compromised when x nodes are captured. Figure 5.1(a) shows that, in the basic random scheme and the q-composite scheme, the probability reaches about 0.2 only after 100 nodes are captured; in the

Du's scheme, with $\omega = 11$ and $\tau = 2$, the probability does not reach 0.2 until about 500 nodes are captured. Figure 5.2 shows the same trend under a different pre-defined connectivity. This result is achieved with the same memory usage. Thus, we can say, Du's scheme is a more secure than the basic random scheme and the q-composite scheme.

Although it is not formally mentioned, Du et al. used a method that later is generalized as product constructions. Before we introduce product constructions, there is another scheme that can be regarded as product construct based on Blom's scheme and the basic random scheme.

5.1.3 Another Blom's Scheme Based KPS

Liu and Ning [Liu et al., 2005] illustrated a new scheme that is constructed on top of the basic random scheme and the traditional *polynomial-based scheme* [Blundo et al., 1993], which is actually a special case of Blom's scheme with a certain parameter. Since this scheme is similar to Du's scheme, we will only give a brief description.

Liu and Ning first generate a t-degree polynomial

$$f(x, y) = \sum_{i,j=0}^{t} a_{i,j} x^i y^i,$$

over a finite field $GF(q)$, where q, as same as in Du's scheme, is a prime number that large enough to accommodate a key. This bivariate polynomial has the property that $f(x, y) = f(y, x)$. In the network, every node is allocated a unique ID. After that, each sensor node i will be allocated a polynomial of $f(i, y)$, which will occupy $(t+1)\log q$ memory space of a node. When two nodes i and j attempt to communicate, node i can compute the common key between nodes i and j by evaluating $f(i, y)$ at point $y = j$. In the same manner, node j can compute $f(j, i) = f(i, j)$ at the same time. Thus, nodes i and j share a common key and are feasible to establish a secure link. Note that $f(x, y)$ has different value with different pair of x, y. Therefore, $f(x, y)$ is the unique key between nodes i and j and nowhere else in the network uses the same key.

Now consider the polynomial of $f(i, j)$, for $i = 1, 2, ..., t$ and $j = 1, 2, ..., t$. Blundo et al [Blundo et al., 1993] designed polynomial schemes using more than two variables, but Liu and Ning only use two in their new scheme and it actually becomes a special case of Blom's scheme. If we consider using is as indexes of rows and js as indexes of columns, then the values of $f(i, j)$ can be considered to be a symmetric matrix, which is as same as in Du's scheme. Furthermore, Liu and Ning's polynomial based scheme is t-secure, corresponding to the λ-secure in Du's scheme. They are same schemes with different expressions.

As shown in Figure 5.2, the probability that a link is compromised in the new polynomial scheme is almost as same as that in Figure 5.1. About 500 captured nodes will result in 20% chance of any other link's corruption.

Until now, we have presented two schemes that could be regarded as product constructions. In next section, we will describe product construction in general, and

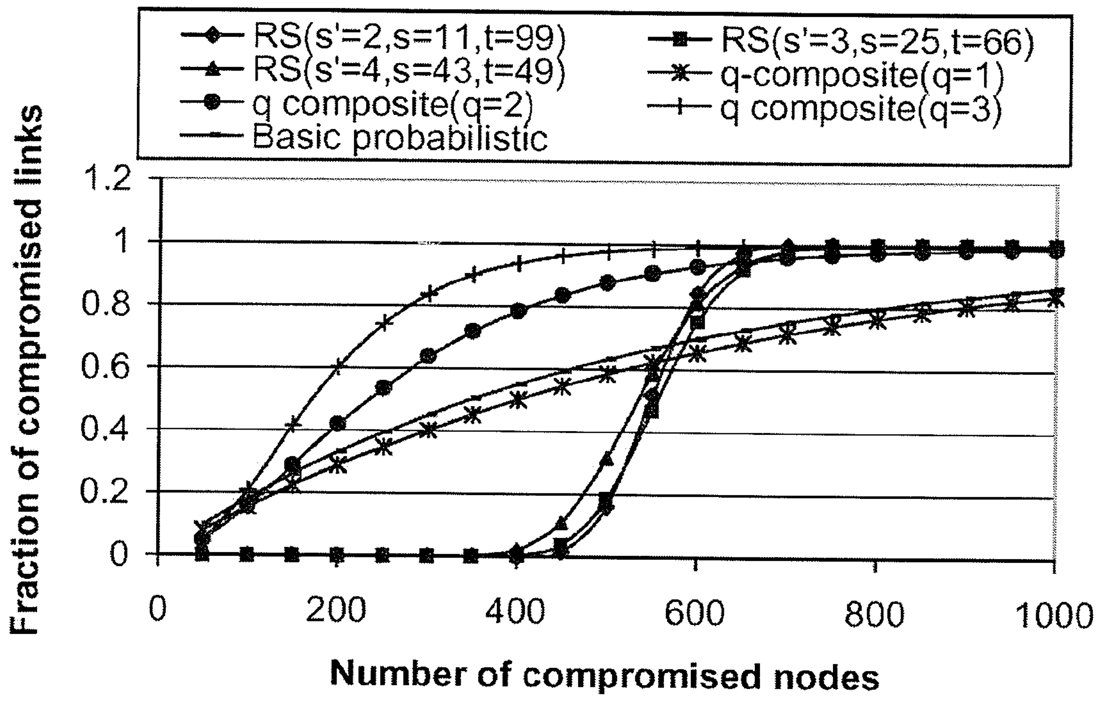


Figure 5.2: The probability that a link can be compromised after some nodes are captured. p is the probability that any two nodes in a neighborhood is able to establish a secure link. RS is the polynomial scheme with different parameters. Other are the probability for the q-composite scheme, when $q = 1, 2, 3$ and the basic random scheme [Liu et al., 2005]

present a scheme developed by Wei and Wu, using a product construction.

5.2 Difference Family and Product Constructions

Product construction is a method that inherits features from all (normally two) original schemes. However, schemes that involve in a product construction may not all necessary be KPSs for DSNs. They may only be a key distribution scheme in a traditional network, or a combinatorial design dealing with key distribution problems, although the combination of these schemes must have all desired features as a KPS for DSNs.

[Wei and Wu, 2004] first generalized and defined an ideal KPS scheme.

Definition 5.2.1. *A pairwise key pre-distribution scheme S in sensor networks is a triple (B, F, K), where B is the set of sensor nodes, F is the set of key distribution algorithms and K is the key pool, which satisfy following requirements:*

1. For every $b \in B$, an $f(b_i) \in F$ is assigned to b.

2. For any two distinct nodes $b_1, b_2 \in B$, there is a unique key K_{b_1,b_2} shared between b_1 and b_2, each obtained from $f(b_1)$ and $f(b_2)$.

3. For other $b_i \in B$, no information about K_{b_1,b_2} is obtained from $f(b_i)$.

This is an ideal KPS. However, it is very difficult to produce such an ideal KPS with an acceptable memory cost. Practically, we are looking for a balance between the memory usage and the security. If a KPS has the property that after λ nodes are compromised the rest links remain secure, then that scheme is called *λ-secure*, or the scheme is *λ-resilient*. Formally, for a scheme triple (B, F, K), a λ-secure scheme is defined as:

Definition 5.2.2. *In a λ-secure, or λ-resilient scheme, for any distinct u, v different from $b_1, b_2, ..., b_\lambda \in B$, $K_{u,v}$ can not be computed by $f(b_1), f(b_2), ..., f(b_\lambda)$.*

Based on the description of KPSs, a product construction can be defined as:

Definition 5.2.3. *Suppose $S = (B, F, K)$ is a KPS and (X, A) is a set system, where*

$|A| \geq |B|$. *Assume for each* $b_i \in B$ *there is a corresponding* $A_i \in A$. *Then a product construction is a triple* $(B, F \times A, K \times X)$ *such that the algorithm corresponding to* b_i *is* $f(u_i) \times A_i$.

Simply, D is a KPS, either designed for traditional networks or for DSNs. D will provide a key pool for the product construction. Set system (X, A), which is specially designed for DSNs, is the manner we choose keys from the key pool for sensor nodes. In this case, D is called a *key space.* I will use Blom's scheme as one of the product schemes to show this procedure in detail.

Wei and Wu considered several factors in their product scheme: the resiliency, the connectivity and the memory usage. In fact, they combined three schemes to produce a new one. They considered a combination of schemes satisfying the requirement of the resiliency, the connectivity and the λ-secured feature.

5.2.1 Analysis of Resiliency

Suppose the original KPS D satisfies the property of being λ-secure. In the product scheme $D \times S$ the resiliency is measured by the probability that one key is broken after s nodes are compromised. Let us consider the probability that any key K_i is broken. Let p_j^i represent exact j nodes in s compromised nodes containing information of K_i. Let C_s denote the event that s nodes were compromised. The probability that a key is broken after s captured nodes is computed as:

$$P_r(K_i|C_s) = \sum_{j=\lambda+1}^{s} p_j^i.$$

Theorem 5.2.1. *In a product scheme $D \times S$, every points in X should occur same times to keep the best possible resiliency of the scheme.*

Proof. In a $D \times S$ product construction scheme, let $b = |B|$, $X = 1, 2, ..., v$ thus $v = |X|$, k be the size of a block, and $i \in X$ and occurs in t_i blocks. Consider the probability that j of the s blocks contain i is:

$$p_j^i = \frac{\binom{t_i}{j}\binom{b-t_i}{s-j}}{\binom{b}{s}}.$$

The value of p_j^i depends on the value of t_i. To calculate the resiliency, we should count the key which is easiest to be broken. When t_i has same value, namely t, for all is, all $P_r(K_i|C_s) = \sum_{j=\lambda+1}^{s} p_j^i$ should have same value too. We assume this is the case, $P_r(K_i|C_s) = q$. Now, if some key, occurs less than others, it will result in for this key i, the probability that this key is broken appears smaller than others. But for all the $i \in X$, $\sum_{i=1}^{v} t_i = kb$ is fixed, which means, when some key(s), t_n, occurs less, some other key(s) appears in more nodes. Assume it is key m, then $t_m > t > t_n$. It will lead to $P_r(K_m|C_s) > q > P_r(K_n|C_s)$, which indicates that the scheme is easier to be broken compared to the case where keys occur in same nodes. Thus we want the points distributed evenly. □

When each point occurs in exact r blocks, the set system is regular. It follows that:

$$p_j^i = \frac{\binom{r}{j}\binom{r}{s-j}}{\binom{b}{s}}.$$

If we only consider the resiliency, a 1-design is an ideal choice applying from combinatorial designs to KPS. However, only the resiliency is not enough to construct a KPS for DSNs. We now describe how Wei and Wu consider the connectivity of the new scheme.

5.2.2 Analysis of Connectivity

Another critical requirement for a product scheme is the connectivity of the network. Like in other KPSs, the network graph needs to be at least connected. To analyze the connectivity, again, we need the knowledge from Theorem 2.2.1. The connectivity in a product scheme is also a monotone property. That is, in a network with fixed number of nodes, if the number of edges increases, the connectivity increases too. From the Theorem 2.2.1, the average expected degree of a node is calculated as:

$$d = \frac{b-1}{b}(\ln b - \ln(-\ln P_c)),$$

where $b = |B|$ and P_c is the desired globe connectivity. The connectivity of the network is determined by the degree d of each node, when the number of nodes are fixed. In the new product scheme, if two blocks share at least one common key, they can establish a secure link. Assume, in the network graph, each node has t neighbors, and the expected average number of nodes in a neighborhood is q, then $d = \frac{qt}{b}$. Since b and q are fixed, we have:

Theorem 5.2.2. *The connectivity of a product scheme depends on the number of blocks each block intersects.*

Wei and Wu showed that, to achieve both good resiliency and connectivity, the result from 5.2.1 should satisfy the following optimization.

Lemma 5.2.1. *The set system in Theorem 5.2.1 achieves largest number of intersections for each block, when $b = \binom{v}{k}$ and $r = \binom{v-1}{k-1}$.*

Since in a $(v, k, 1)$ 1-design, each block intersects at most $k(r-1)$ blocks. If a pair of points appear in more than one block, one of the block that contains the pair will intersect less than $k(r-1)$ blocks. Therefore the number of blocks containing the same pair of points should be as small as possible to maximize the number of blocks that intersect a fixed block.

Definition 5.2.4. *A* set system (X, B) *is called a* (v, k, λ) difference family *if every possible difference between any two elements appears in exactly λ blocks.*

Wei and Wu find that when $\lambda = 1$, a (v,k,1) difference family satisfies the requirements of both resiliency and connectivity. Now they consider using the Du's scheme as the key space. In their $(v, k, 1)$ *difference family scheme*, Wei and Wu combined a *difference family* and *Du's scheme*(the product of two schemes). They use *Du' scheme* as the key space (the content of keys stored in the key pool) and *difference family* to select key rings in a deterministic manner. For example, matrices are generated as same as in the Du's scheme, with $\omega = v, \tau = k$. If a block $\{4, 5, 6\}$ was chosen as the key ring for node 5, the 4^{th}, 5^{th}, and 6^{th} rows of the 5^{th} key matrix A_5 in the Du's scheme will be assigned to the 5^{th} node.

Note that Wei and Wu attempt to introduce a general method in constructing KPS and provide a deterministic replacement of the basic random scheme. As a result, the difference family scheme only achieves the same resiliency against node capture as the basic random scheme. However, their major contribution is that they provided a general method for later KPS constructions. I introduce two new schemes using product constructions in Chapter 6. Before that, I review one more scheme that can be regarded as using product constructions.

5.3 Blom's Scheme and Strongly Regular Graphs

Lee and Stinson attempted to combine a SRG scheme with Blom's scheme. They named the new scheme as *deterministic multiple space Blom's scheme.* We now introduce this scheme.

In Blom's scheme, given a public matrix G and a secret symmetric matrix D, it is feasible to computer $A = (D \cdot G)^T$. Then the kth row of A and kth column of G are assigned to node k. Lee and Stinson slightly modified Blom's scheme by first changing the secret matrix D. The new D is not necessarily symmetric.

Lee and Stinson defined that nodes in the network are divided into two groups u and v. For group u, compute $A = G^T \cdot D$ and assign a column of G (denoted as x_u) and a row of A to each node in u; for group v, compute $A = D \cdot G$ and assign a column of G (denoted as x_v) and a row of A to each node in v. Now nodes in both groups can to compute $x_u^T D x_v$ and can use this as the pairwise key between them.

The deterministic multiple space Blom's scheme involves more modifications. It can be considered as the product construction of multiple IOS scheme and Blom's scheme. In the multiple IOS scheme, Lee and Stinson use l copies of an (n, r, λ, μ)-SRG to accommodate nodes in the network of size $n = ml$. As introduced in Chapter 4, a $(m, r, \lambda, \mu) - SRG$ has an edge decomposition into star-like subgraphs such that each vertex acts as a center in one star-like subgraph and a leaf in $r/2$ distinct subgraphs. Lee and Stinson defined that the direction of a edge is: it starts from the center node and end at a leaf node.

In the SRG, every vertex represents l nodes. We use u to represent any vertex of the SRG, and u_i $(i = 1, 2, ..., l)$ to represent every node associated to that vertex. In the new scheme, for every edge of the SRG, a unique random matrix D_e of size $(t + 1) \times (t + 1)$ is generated, where t is the requirement of t-secure. As same as in Blom's scheme, we have a publicly known matrix G of size $(t + 1) \times n$. Note that here n is the network size $n = ml$, not the number of vertexes in the SRG. Let $G(i)$ represents the ith column of the matrix G, such that for every G_{u_i}, there is a unique column of G corresponding to it.

If a D_e starts from u_i, then node u_i is allocated $G(u_i)^T \cdot D_e$; if a D_e ends at u_i, then node u_i is allocated $D_e \cdot G(u_i)$. Now for any two directly connected vertices (not nodes) u and v in the SRG, assume u and v are connected by an edge e and e starts at u, any node u_i associated to vertex u and any node v_j associated to vertex v could use $G(u_i^T) \cdot D \cdot G(v_j)$ as the key to encrypt the link between them, because G is publicly known.

Similar to the analysis in Du's scheme, Lee and Stinson also showed that in the case that s nodes are captured, the probability that an additional link is broken is given by:

$$P(s) = 1 - \frac{\sum_{i=0}^{t} \sum_{j=0}^{t} \binom{l-1}{i} \binom{l-1}{j} \binom{n-2l}{s-i-j}}{\binom{n-2}{s}}.$$

Figure 5.3 shows that, similar to the Du's scheme, the multiple space Blom's scheme achieves better resiliency against node capture. The probability reaches 0.2 after around 350 nodes are captured, similar to Du's scheme, but in other schemes, such as q-composite, and basic random scheme, the probability reaches 0.2 very early, around 150 or even less.

Until now, we have presented both probabilistic and deterministic approaches in designing KPSs for DSNs, and the product construction, a method could lead to new KPSs. We would like to present two attempts in designing KPSs using product constructions.

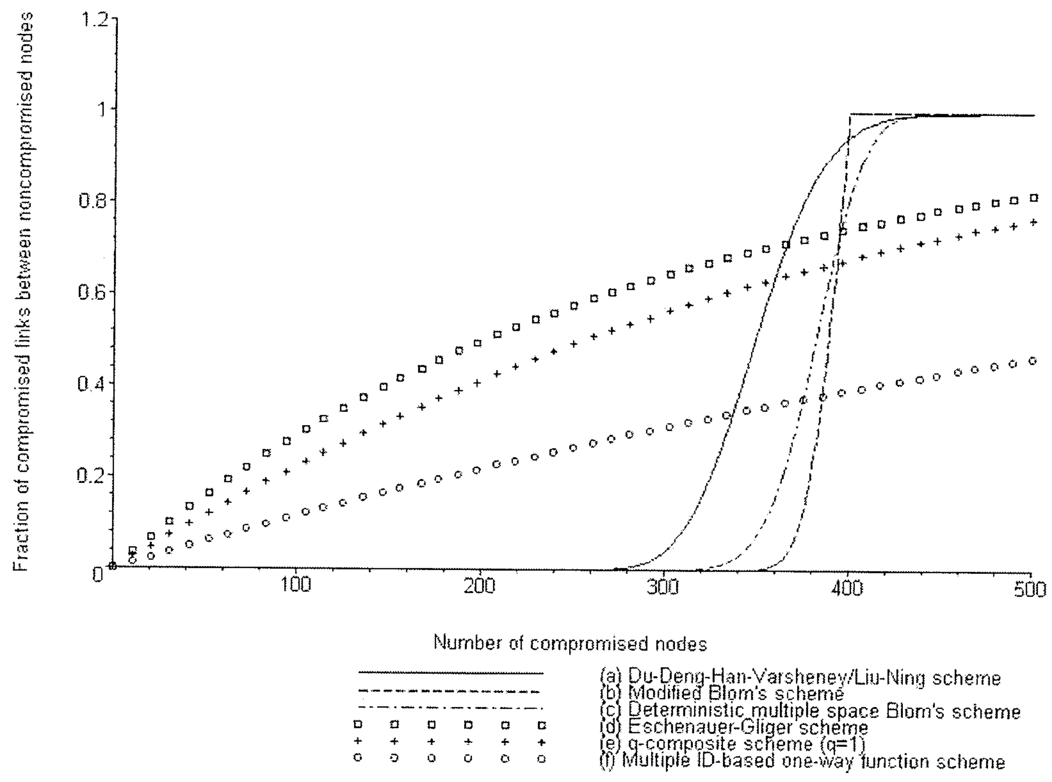


Figure 5.3: The probability that a link can be compromised after some nodes are captured. [Lee and Stinson, 2004]

Chapter 6

New Product Constructions of Key Pre-distribution Schemes

In this chapter, I will introduce two new product constructions for KPSs in DSNs. To better understand the new schemes, I would like to give a brief review about KPS designs presented in previous chapters.

In Chapter 2, the basic random scheme [Eschenauer and Gligor, 2002], the first KPS for DSNs, is introduced as the basis of later schemes. The basic random scheme explored a new area for later KPS designs. In chapter 3, I introduced some other probabilistic schemes [Chan et al., 2003; Pietro et al., 2003] that can be derived from the basic random scheme. These schemes achieve enhanced security by modifying the rule of secure-link establishment in the basic random scheme. After that, we introduced a brand new approach, the deterministic approach, for designing KPSs for DSNs. Deterministic schemes use some sort of set system to present the key pool and key rings, such that a network could use the same set of key rings in every deployment and the performance of the scheme would not be affected by the random number generators. Until then, we had introduced two existing types of KPSs for DSNs and need a methodology for producing new schemes. Therefore, in Chapter 5, we introduced the product construction, a method that can be regarded as producing new schemes using a combination of multiple existing schemes. Among those schemes that can lead to new constructions, Blom's scheme [Blom, 1985], one of the KPSs for traditional networks, is efficient to construct key spaces for producing new schemes. Several schemes [Wei and Wu, 2004; Liu et al., 2005; Du et al., 2003] have been successfully constructed by combining and Blom's scheme and other schemes. Motivated by these researches, I attempt to follow their ideas to use the product construction to produce new KPSs for DSNs. Specifically, I will study the combination of the "$\mu - CID$ scheme" and Blom's scheme.

6.1 The μ-CID Blom Scheme

In Chapter 4, I introduced the μ-CID KPS. The μ-CID scheme is a deterministic method using set system and combinatorial designs. The μ-CID scheme takes advantages of both direct links and two-hop links to form a connected network. The knowledge about combinatorial designs behind the scheme is as follow:

A $(v, b, r, k) - 1 - design$ is called a $(v, b, r, k) - configuration$ if any two distinct blocks have at most one common point, Lee and Stinson [Lee and Stinson, 2005a] defined that, a (v,b,r,k)-*μ-common intersection design (μ-CID)* is a $(v,b,r,k)-$ *configuration*, where $B = \{B_i, B_2, ..., B_b\}$, and

$$|\{B_h \in B : B_i \cap B_h \neq \emptyset \text{ and } B_j \cap B_h \neq \emptyset\}| \geq \mu$$

when $B_i \cap B_j = \emptyset$ for all i, j.

The μ-CID scheme creates key rings in a deterministic manner. In this section, we attempt to produce a new product construction by combining the Blom's scheme and the μ-CID scheme. We name it as μ-CID Blom's scheme.

The procedure of key pre-distribution in the new product construction is shown as follow:

Step 1: Generate a $(\lambda + 1) \times b$ matrix G, where b is the network size and λ is the security factor. The matrix G is the well-known Vandermonde matrix (the same as Du's scheme and the 2-composite Blom's scheme).

Step 2: Generate v key spaces, where the value of v is derived from the selected μ-CID. We generate v random, symmetric matrices $D_1, ..., D_v$ of size $(\lambda + 1)(\lambda + 1)$, then compute the matrix $A_i = (D_i \cdot G)^T$. Use $A_i(j)$ to mark the jth row of A_i.

Step 3: Select k spaces for each node, where k is the block size of the μ-CID. For each node, select k distinct key spaces based on the block corresponding to this node. For example, If a block j (correspond to node j) is (2,3,5,6) in the μ-CID, key spaces

A_2, A_3, A_5 and A_6 are selected for node j. Node j stores the jth row of each matrix.

In the μ-CID product scheme, two nodes may communicate directly if they contain same elements (different rows of same matrix), otherwise they communicate via two-hop links.

6.1.1 Connectivity and Memory Usage

The connectivity of the new product scheme remains unchanged compared to the μ-CID scheme. Any two nodes who choose a common matrix (common element in μ-CID) form a link in the network graph in the new product scheme. Lee and Stinson proposed the connectivity as:

$$p_{connect} = p_1 + p_2,$$

where

$$p_1 = \frac{k(r-1)}{b-1},$$

and

$$p_2 = \left(1 - \frac{k(r-1)}{b-1}\right) \times \left(1 - (1 - \frac{\mu}{b-2})^{\eta}\right).$$

where η denotes the number of nodes in the intersection of the neighborhoods of two nodes.

In a (v,k,r,b) μ-CID scheme, the memory cost depends on the block size k. The memory usage increases to $k(\lambda+1)$ in the μ-CID Blom's scheme, since a node needs to carry $\lambda+1$ elements for every selected matrix in Blom's scheme, in which the elements is a security unit of a fixed size (64bits,128bits,etc.). As known, the Blom's scheme is λ-secure, which means communication links remain secure if less than $\lambda+1$ nodes are compromised. In a new product scheme with $r \leq \lambda$, we achieve perfect security because the capture of one key space (a matrix) will never reach $\lambda+1$. This feature does not appeared in Du's scheme, because each key space is randomly selected. The price of this perfect resiliency is the memory cost, which is unacceptable in some applications.

Now let's review a scheme with perfect resiliency. In the $(n-1)$-key scheme where n is the network size, each pair of nodes have a unique common key such that the capture of other nodes will not compromise the link between them. If we can construct a μ-CID with $k(\lambda+1) < n-1$, it will worth that memory cost in data extremely sensitive scenarios. However, this is a extreme case with very high memory cost, in next section, we will show a general analysis of the resiliency against node capture in the u-CID Blom's scheme.

6.1.2 Resiliency Analysis

As shown in Chapter 4, the resiliency in the original μ-CID scheme is not good enough. In the μ-CID Blom's scheme, we try to achieve enhanced security. In order to make $k(\lambda+1)$ small, λ should be relatively small. However, the μ-CID Blom's scheme lost the perfect resiliency when more than λ compromised nodes contain the same key space in the network.

Du et al. used two ideas to evaluate the security of their product scheme: 1)The probability that at least one key space is broken when x nodes are captured. 2)The fraction of the additional communication also becomes compromised when x nodes are compromised. To compare the μ-CID Blom's scheme with Du's scheme, I will follow these two ideas.

Du et al. give the following upper bound:

$$Pr(\text{at least one space is broken} \mid C_x) \leq \omega \sum_{j=\lambda+1}^{x} \tbinom{x}{j}\theta^j(1-\theta)^{x-j}.$$

This inequality shows that in Du's scheme, the probability (Pr) that at least one key space is broken when x nodes are captured. θ is the probability that each compromised node carries the key of the first key space (or any key space). We give the following mapping table to illustrate the relationship between the parameters of the two schemes, including those used for security analysis.

To compare the two scheme, we need to choose the same value for matching parameters, such as $v=\omega$ and $k=\tau$.

We first analyze the probability that at least one key space is broken when x nodes are captured in the μ-CID Blom's scheme. The analysis given by Du et al. has been shown in Chapter 5. The result is:

Parameters	**Du's scheme**	**μ-CID Product**
Pool Size	ω	v
Network Size	NA	b
Frequency of each key	NA	r
Key ring length	τ	k
Compromised nodes	x	x
Matrix size	λ	λ

Table 6.1: Different Symbols used in Du's scheme and the μ-CID scheme

$$Pr(\text{at least one space is broken}|C_x) \leq \omega \cdot Pr(S_1|C_x).$$

Although the μ-CID Blom's scheme uses a deterministic manner to generate key rings, it still satisfies this inequation. Thus this statement is also true for μ-CID scheme with $\omega = v$. In Du's scheme, we have:

$$Pr(\text{at least one space is broken}|C_x) = \sum_{j=\lambda+1}^{x} \tbinom{x}{j}\theta^j(1-\theta)^{x-j}$$

where $\theta = \frac{\tau}{\omega}$ and j represents the possible number of compromised nodes in which S_1 are stored that may lead to S_1 is broken, when x nodes are compromised by an adversary. However this result is only obtained in Du's scheme. In the μ-CID Blom's scheme, the analysis of $Pr(S_1|C_x)$ is different.

In the μ-CID Blom's scheme, when x nodes are compromised, there are $\binom{b}{x}$ possible ways to choose x captured nodes. Let j denote the number of nodes contain S_1, among all these choices, S_1 can be broken only when $j > \lambda$, that is, in order to break S_1, j

should at least be $\lambda + 1$. In addition, in a μ-CID, exact r blocks contain key space S_1. There are exactly $\binom{r}{j}\binom{b-r}{x-j}$ blocks that contain same fixed element x, exactly j times. Then we have the following result:

$$Pr(\text{at least one space is broken}|C_x) \leq v \cdot \sum_{j=\lambda+1}^{r} \frac{\binom{r}{j}\binom{b-r}{x-j}}{\binom{b}{x}}.$$

Now we analyze the fraction of the additional communication links being compromised when x nodes are compromised. When an adversary compromise x nodes, the rest of the network may also be compromised because of the information derived from the x compromised nodes. This issue can be regarded as: when x nodes are compromised, what's the probability that any non-compromised link in the network is also compromised by the adversary, based on the information they obtained from x captured nodes. Du et al showed the following analysis.

Let c denote a link between any two nodes in the network graph, not involving those x compromised nodes. Assume link c uses key K to secure the communication. Then $K \in S_i$ means "key K was derived using S_i". B_i represents the joint event that both $K \in S_i$ and S_i is compromised. The probability that c is compromised given the compromise of x other nodes is:

$$Pr(c \text{ is broken}|C_x) = Pr(B_1 \cup B_2 \cup ... \cup B_\omega|C_x).$$

Since only one key is used to secure c, $B_1, ..., B_\omega$ are mutually exclusive. Because all B_i occur in the same probability we have

$$Pr(c \text{ is broken}|C_x) = v \cdot Pr(B_1|C_x).$$

Note that the event $(K \in S_1)$ is not depend on the event C_x and the event $(S_1 \text{ is compromised})$

$$Pr(B_1|C_x) = \frac{Pr((K \in S_1) \cap (S_1 \text{ is compromised}) \cap C_x)}{Pr(C_x)}$$

$$= Pr(K \in S_1) \cdot Pr(S_1 \text{ is compromised}|C_x).$$

$Pr(S_1 \text{ is compromised}|C_x)$ is computed. As illustrated, $Pr(K \in S_1)$ represents the probability that the key used to secure link c are derived from key space S_1. In Du's scheme, they showed that since key spaces are selected uniformly from ω possibilities, then

$$Pr(K \in S_1) = \frac{1}{\omega}$$

In the μ-CID Blom's scheme, to compute $Pr(k \in S_1)$, we randomly select a node A_i from b possibilities. Then the probability that node A_i contains information about S_1 is $\frac{r}{b}$. Among all the $k(r-1)$ possible links involving node A_i, $(r-1)$ links use S_1 as encryption key. Then in the new μ-CID scheme, we have

$$Pr(k \in S_1) = \frac{r}{b} \cdot \frac{r-1}{k(r-1)} = \frac{r}{bk} = \frac{1}{v},$$

since $vr = bk$ in a μ-CID. Therefore,

$$Pr(c \text{ is broken}|C_x) = Pr(S_1 \text{ is compromised}|C_x) = \sum_{j=\lambda+1}^{r} \frac{\binom{r}{j}\binom{b-r}{x-j}}{\binom{b}{x}}.$$

Based on this analysis, we will compare the μ-CID Blom's scheme with Du's scheme in next subsection.

6.1.3 Comparison to Previous Work

Since we will choose same values for matching parameters in both μ-CID Blom's scheme and Du's scheme, the two schemes will have the same memory usage.

We now start to compare the resiliency of the new μ-CID Blom's scheme with Du's scheme. Since the fraction of the additional compromised links is the probability that one key space been broken multiples a constant, we will only compare the probability of the key space being broken. We choose the same parameters introduced in [Lee and Stinson, 2005a], which forms a $(1470, 2401, 49, 30)$ μ-CID. Figure 5.4 shows the resiliency of the μ-CID Blom's scheme and the Du's scheme, when x nodes are captured.

When $\lambda = 15$, compared to Du's scheme, our μ-CID Blom's scheme allows extra 100 captured nodes before the network starts to loss the perfect resiliency. With

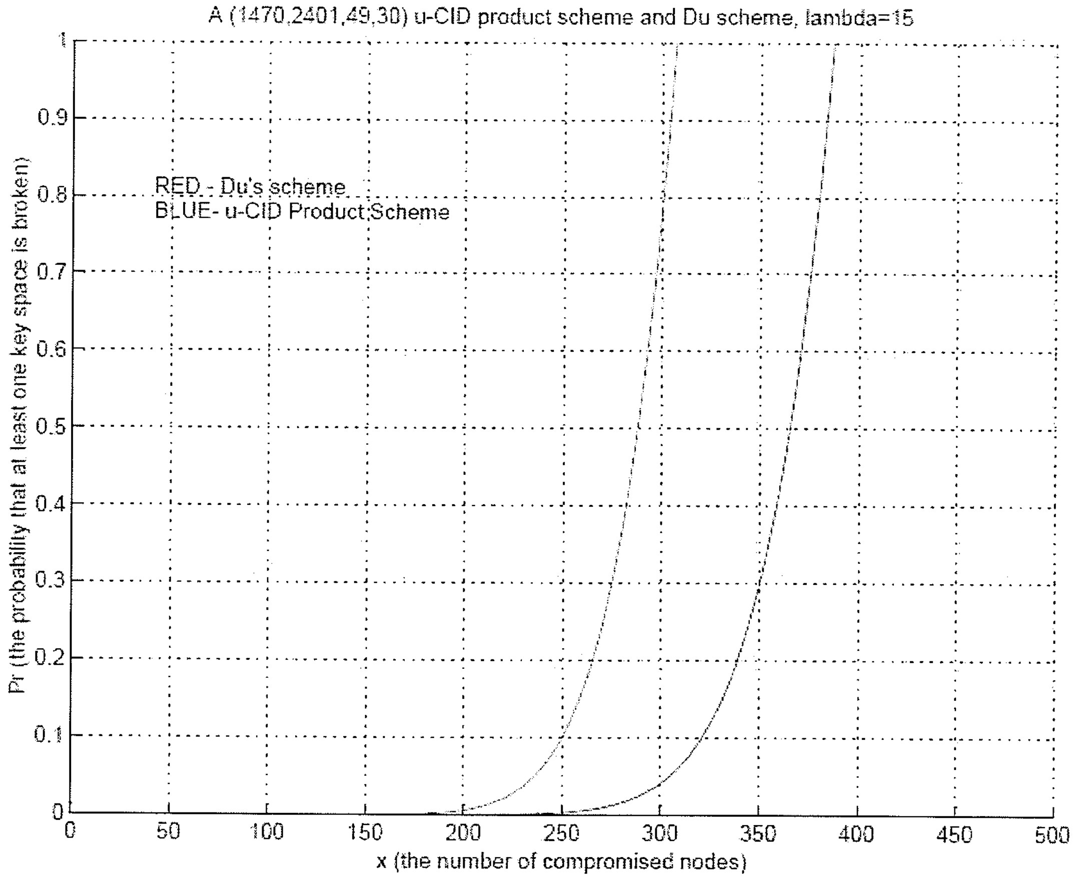


Figure 6.1: Resiliency of the μ-CID Blom's Scheme and Du's Scheme

the same memory usage, the analytical global connectivity of the new scheme is approximately 0.99995, which is 0.00004 less than Du's scheme. In an application, such little difference of connectivity could be ignored. However, the current μ-CID Blom's scheme is not perfect. The problem of the μ-CID Blom's scheme is that, the construction of a μ-CID with large parameters is very difficult, sometimes even beyond the computational capability of existing computer systems. Moveover, a μ-CID with the needed parameters may not exist, so for a network of certain size, the current μ-CID Blom's scheme is not suitable.

Chapter 7

Conclusion and Future Work

In this thesis, we have introduced most existing key pre-distribution schemes for distributed sensor networks. Sensor nodes are resource-limited devices in terms of memory capacity and computational capability. The constraints of sensor nodes make key pre-distribution scheme the only feasible solution for the key agreement in distributed sensor networks. In this thesis, we divided key pre-distribution schemes into three classes: probabilistic schemes, deterministic schemes, and schemes using product constructions.

The basic random scheme is the first key pre-distribution scheme and became the foundation of later researches in this area. The most important contribution in the basic random scheme is to give out guidelines for later researches, such as the concept of key pool and key rings. In addition, the basic random scheme brings an important result from random graph theory to compute some essential parameters for itself, and later on for all probabilistic schemes and related product constructions. After the basic random scheme, other probabilistic schemes are introduced, such as the q-composite scheme and the pseudo-random scheme. In these schemes, multiple possible keys are considered to secure one link between two nodes. These probabilistic scheme also provide enhanced resiliency against node capture.

One problem in probabilistic schemes is that the key ring generating depends on a random number generator so that the performance (connectivity, resiliency, etc.) of a scheme may vary in each deployment. Another type of scheme, deterministic schemes, are introduced to avoid the variety of performance in probabilistic schemes. I first discussed a scheme applying strongly regular graph to give a graphical view about deterministic approaches. Then, I introduced the first deterministic scheme using set systems, the scheme using a generalized quadrangle design, which is an intensively studied topic in combinatorial designs. The μ-common intersection design scheme is a set system specially designed for the key pre-distribution scheme problem. Personally, this is my favorite scheme, although it has the common problem for all schemes using combinatorial designs: For certain parameters, the desired design is unknown or even worse, does not exist.

One methodology for designing key pre-distribution schemes is the product construction. It can be regarded as combining two or more existing schemes to construct new ones. I introduced several schemes using product constructions: Du's scheme, the difference family scheme, the polynomial scheme, etc. The product construction is practical in constructing new schemes.

I followed the product construction and constructed a new key pre-distribution scheme for distributed sensor networks. The new scheme is constructed by combining the μ-CID scheme and Blom's scheme. It improved the weak resiliency in the original μ-CID scheme.

There are several directions that could be considered as future work. One is dealing with the situation when desired designs do not exist for those deterministic schemes using combinatorial designs. It could possibly be achieved by selecting blocks from an existing design, randomly or in a certain manner. A problem in this approach is how to keep the desired properties of the design.

Another approach worth trying is to combine (not as easy as existing product constructions) the q-composite feature and the μ-CID scheme. This may lead to new combinatorial designs. For $q = 2$, the desired properties in the new design should be: If two blocks intersect at less than 2 points, there are μ common other blocks intersect both of that at at least 2 points.

Bibliography

Wireless integrated network sensors. http://www.janet.ucla.edu/WINS/.

Ian F. Akyildiz, Weilian Su, Yogis Sankarasubramaniam, and Erdal Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, pages 102–114, August 2002.

Rolf Blom. An optimal class of symmetric key generation systems. In *Proceedings of EUROCRYPT 84*, pages 335–338, 1985.

Carlo Blundo, Alfredo De Santis, Amir Herzberg, Shay Kutten, Ugo Vaccaro, and Moti Yung. Perfectly-secure key distribution for dynamic confrerences. In *Advances in Cryptology - CRYPTO'92*, pages 471–489, 1993.

David W. Carman, Peter S. Kruus, and Brian J. Matt. Constraints and approaches for distributed sensor network security. Technical Report TR 00-010, NAI Lab, 2000.

Seyit A. Çamtepe and Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. Technical Report TR 04-10, RPI Dept. of Computer Science, 2004.

Seyit A. Çamtepe and Bülent Yener. Key distribution mechanisms for wireless sensor networks: a survey. Technical Report TR-05-07, Rensselaer Polytechnic Institute, 2005.

Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy(SP'03)*, pages 197–213, 2003.

Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 42–51, New York, NY, USA, 2003. ACM Press. ISBN 1-58113-738-9.

Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47, New York, NY, USA, 2002. ACM Press. ISBN 1-58113-612-9.

Joseph M. Kahn, Randy H. Katz, and Kristofer S. J. Pister. Next century challenges: Mobile networking for smart dust. In *Proceedings of the 5th Annual ACM/IEEE Internation Conference on Mobile Computing and Networking (MobiCom)*, pages 483–492, 2003.

Jooyoung Lee and Douglas R. Stinson. Deterministic key predistribution schemes for distributed sensor networks. In *Lecture Notes in Computer Science (SAC 2004 Proceedings)*, 2004.

Jooyoung Lee and Douglas R. Stinson. A combinatorial approach to key predistribution for distributed sensor networks. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 2, pages 1200–1205, May 2005a.

Jooyoung Lee and Douglas R. Stinson. Common intersection designs. *Journal of Combinatorial Designs*, 14(4):251–269, 2005b.

Donggang Liu, Peng Ning, and Rongfang Li. Establishing pairwise keys in distributed sensor networks. volume 8, pages 41–77, New York, NY, USA, 2005. ACM Press.

F. Jessie MacWilliams and Neil Sloane. *The Theory of Error-Correcting Codes*. Elsevier Science, New York, 1977.

Paul Erdös and Alfred Rényi. *On ramdom graphs*. Publ. Math. Debrecen, 1959.

Adrian Perrig, John Stankovic, and David Wagner. Security in wireless sensor networks. *Communications of the ACM*, 47(6):53–57, 2004.

Roberto Di Pietro, Luigi V. Mancini, and Alessandro Mei. Random key-assignment for secure wireless sensor networks. In *SASN '03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 62–71, New York, NY, USA, 2003. ACM Press. ISBN 1-58113-783-4.

Jan M. Rabaey, M. Josie Ammer, Julio L. da Silva, Danny Patel, and Shad Roundy. Picoradio supports ad hoc ultra-low power wireless networking. *Computer*, 33(7): 42–48, 2000. ISSN 0018-9162.

Douglas R. Stinson. *Gryptography: Theory and Practice*, pages 155–157. CRC Press, 2002.

Ruizhong Wei and Jiang Wu. Product construction of key distribution schemes for sensor networks. In *Lecture Notes in Computer Science (SAC 2004 Proceedings)*, 2004.

Ossama Younis and Sonia Fahmy. Distributed clustering for scalable, long-lived sensor networks. Technical Report TR-03-026, Department of Computer Science, Purdue University, 2003. URL `citeseer.ist.psu.edu/younis03distributed.html`.