1999

Society-oriented cryptographic techniques for information protection

Hossein Ghodosi
University of Wollongong

Society-Oriented Cryptographic Techniques for Information Protection

A thesis submitted in fulfillment of the
requirements for the award of the degree

Doctor of Philosophy

from

UNIVERSITY OF WOLLONGONG

by

Hossein Ghodosi, MSc (Hons), University of Wollongong

School of Information Technology and Computer Science
January 1999

Dedicated to

my parents

&

my wife and children

Declaration

This is to certify that the work reported in this thesis was done by the author, unless specified otherwise, and that no part of it has been submitted in a thesis to any other university or similar institution.

H. Ghodosi

Hossein Ghodosi, MSc (Hons), University of Wollongong
January 25, 1999

Abstract

Groups play an important role in our modern world. They are more reliable and more trustworthy than individuals. This is the reason why, in an organisation, crucial decisions are left to a group of people rather than to an individual. Cryptography supports group activity by offering a wide range of cryptographic operations which can only be successfully executed if a well-defined group of people agrees to co-operate.

This thesis looks at two fundamental cryptographic tools that are useful for the management of secret information. The first part looks in detail at secret sharing schemes. The second part focuses on society-oriented cryptographic systems, which are the application of secret sharing schemes in cryptography. The outline of thesis is as follows.

Chapter 1 contains a survey of both cryptographic systems and secret sharing schemes. It also provides terminology that is used throughout this thesis. In Chapter 2 a basic model for secret sharing scheme is studied. This model is then compared with existing secret sharing models formulated in the literature. Chapter 3 deals with a particular class of secret sharing schemes, the so-called threshold schemes. Several approaches to the construction of threshold secret sharing are reviewed in this chapter. Misconceptions about some constructions are pointed out and different constructions are compared.

Chapter 4 studies the generalisation of secret sharing schemes. Extended capabilities required in secret sharing schemes are discussed. An approach to computationally secure secret sharing schemes is reviewed. We show how to prevent the cheating problem at the secret reconstruction phase in the studied scheme. We also present efficient solutions for constructing secret sharing schemes in both multilevel and compartmented access structures.

The second part of the thesis concerns cryptographic algorithms. In contrast to existing cryptographic systems that use integers only, we show how floating-point arithmetic can be used in the construction of cryptographic algorithms. In Chapter 5, two classes of transcendentals are applied to construct novel encryption algorithms.

Chapters 6 and 7 concern society-oriented cryptographic systems. In Chapter 6 threshold cryptography is studied. A cryptographic system that can control the flow of information in hierarchical organisations is presented in this chapter. Chapter 7 considers a particular class of society-oriented cryptographic systems, the so-called group-oriented cryptographic systems. A model for the construction of group-oriented cryptosystems is discussed and some group-oriented cryptosystyems are presented.

Finally, in Chapter 8 we high light some directions for future research in the areas of secret sharing and cryptographic systems.

Thesis Related Publications

1. Hossein Ghodosi, Josef Pieprzyk and Reihaneh Safavi-Naini, A Flexible Threshold Cryptosystem, In *Proceedings of ISITA '96 - The 1996 IEEE International Symposium on Information Theory and its Applications*, pp. 75-77, Victoria, B.C., Canada, 1996.

2. Hossein Ghodosi, Josef Pieprzyk, Chris Charnes and Reihaneh Safavi-Naini, Cryptosystems for Hierarchical Groups, In *Proceedings of ACISP '96 - Australasian Conference on Information Security and Privacy*, Vol. 1172 of Lecture Notes in Computer Science, pp. 275-286, Springer-Verlag (Berlin), 1996.

3. Josef Pieprzyk, Hossein Ghodosi, Chris Charnes and Reihaneh Safavi-Naini, Cryptography Based on Transcendental Numbers, In *Proceedings of ACISP '96 - Australasian Conference on Information Security and Privacy*, Vol. 1172 of Lecture Notes in Computer Science, pp. 96-107, Springer-Verlag (Berlin), 1996.

4. Hossein Ghodosi, Josef Pieprzyk and Reihaneh Safavi-Naini, Dynamic Threshold Cryptosystems: A New Scheme in Group-Oriented Cryptography, In *Proceedings of PARAGOCRYPT '96 - International Conference on the Theory and Applications of Cryptology*, J. Přibl, ed., Prague, Czech Republic, pp. 370-379, CTU Publishing House, ISBN: 80-81-01502-5, 1996.

5. Hossein Ghodosi, Ghulam R. Chaudhry, Josef Pieprzyk and Jennifer Seberry, How to prevent cheating in Pinch's scheme, in *Electronics Letters*, Vol. 33, pp. 1453-1454, Aug. 1997.

6. Hossein Ghodosi, Josef Pieprzyk and Reihaneh Safavi-Naini, Remarks on the Multiple Assignment Secret Sharing Scheme, In *Proceedings of ICICS '97 - International Conference on Information and Communications Security*, Beijing, China (Y. Han, T. Okamoto and S. Qing, eds.), Vol. 1334 of Lecture Notes in Computer Science, pp. 72-80, Springer-Verlag (Berlin), 1997.

7. Ghulam-Rasool Chaudhry, Hossein Ghodosi and Jennifer Seberry, Perfect Secret Sharing Schemes from Room Squares, in *Journal of Combinatorial Mathematics and Combinatorial Computing*, vol. 28, pp. 55-61, 1998.

8. Hossein Ghodosi, Josef Pieprzyk and Reihaneh Safavi-Naini, Secret Sharing in Multilevel and Compartmented Groups, In *Proceedings of ACISP '98 - Australasian Conference on Information Security and Privacy*, Vol. 1438 of Lecture Notes in Computer Science, pp. 367-378, Springer-Verlag (Berlin), 1998.

9. Hossein Ghodosi, Josef Pieprzyk, Reihaneh Safavi-Naini and Huaxiong Wang, On Construction of Cumulative Secret Sharing Schemes, In *Proceedings of ACISP '98 - Australasian Conference on Information Security and Privacy*, Vol. 1438 of Lecture Notes in Computer Science, pp. 379-390, Springer-Verlag (Berlin), 1998.

10. Shahrokh Saeednia and Hossein Ghodosi, A Self-Certified Group-Oriented Cryptosystem Without a Combiner, In *Proceedings of ACISP '99 - Australasian Conference on Information Security and Privacy*, Lecture Notes in Computer Science, Springer-Verlag (To Appear).

Acknowledgements

"Man who does not acknowledge creatures, has not acknowledged creator."

I would like to express my appreciation to the University of Tehran, for supporting my application to the Ministry of Culture and Higher Education who provided financial support.

Also, many people deserve acknowledgement for their support and assistance during the development of this thesis.

Firstly I would like to extend my regards and thanks to my supervisor, A/Prof. Josef Pieprzyk, without whose invaluable assistance this study would not have been possible. Secondly, I wish to thank A/Prof. Reihaneh Safavi-Naini (my co-supervisor) for the help she gave me whenever I needed it.

Very warm and grateful thanks to Prof. Jennifer Seberry for her generous financial support during the (long) period in which my scholarship was not approved.

I am also grateful for the help and assistance that the staff, visitors and students in the Centre for Computer Security Research provided to me. In particular, I would like to thank Prof. Yvo Desmedt, from whom I learned so much during the short period that he was visiting the Centre.

Many thanks to Mrs. Margot Koorey and Mr. Terry Rudkin for their helpful attitude and proofreading of this thesis.

Finally, I would like to express my gratitude to my parents, wife and children for their patience throughout this study.

Basic Notation

Most of the notation used in this thesis is defined in the text. Here we list notation for which this is not done.

$\oplus$	Exclusive-or (of Booleans)
$\vee$	Or (of Booleans)
$\wedge$	And (of Booleans)
$/$	Not (e.g., $\neq$ denotes "not equal")
$\cup$	Set union
$\cap$	Set intersection
$\in$	Set membership
$\mathcal{P} \setminus \mathcal{A}$	The set of elements in $\mathcal{P}$ but not in $\mathcal{A}$
$\mathcal{A} \subset \mathcal{P}$	$\mathcal{A}$ is a subset of $\mathcal{P}$, $\mathcal{A} \neq \mathcal{P}$
$\mathcal{A} \subseteq \mathcal{P}$	$\mathcal{A}$ is a subset of $\mathcal{P}$
$\mid$	Such that (set notation)
$a \mid b$	a divides b (a, $b \in \mathbb{N}$)
$\lvert\mathcal{A}\rvert$	The cardinality of set $\mathcal{A}$
$\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$	The set of natural numbers, integers and reals, respectively
$\|\|$	Concatenation
$2^{\mathcal{A}}$	The set of all subsets of set $\mathcal{A}$
2^x	Raising 2 to power x
$\lceil x \rceil$	Smallest integer greater than x
$\lfloor x \rfloor$	Greatest integer smaller than x
$[a]$	A reference (used in bibliography)
$[x, y]$	An interval (a subset of set $\mathbb{R}$)
$\mathbb{Z}_a$	The set of integers modulo a
$\log_a$	Logarithm to base a
$\sum$	Summation
$\prod$	Multiplication

$\rightarrow$	Mapping
$\begin{pmatrix} n \\ t \end{pmatrix}$	The number of subsets of cardinality t of a set of cardinality n
$\equiv$	Congruence
!	Factorial (e.g., $n! = 1 \times 2 \times \cdots \times n$)
$GF(p)$	The Galois field with p elements

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Information protection covers not only secrecy (a traditional protection against eavesdropping) but also authentication, integrity, verifiability and other more specific security countermeasures. Cryptography is the science that deals with the design of algorithms, protocols and systems for solving two kinds of security problems: *privacy* and *authentication.* More precisely, cryptography is the use of transformations of data intended to make the data useless to opponents, but meaningful to legitimate receivers. The only secret part of almost all modern cryptographic systems, however, is the *key* – the parameter that selects the particular transformation to be employed. So there is a clear need for providing for the secrecy of such sensitive information.

Secret sharing schemes provide for the secrecy of sensitive information by partitioning it into several parts in such a way that a specified number of the parts must be combined in order to recover the original information. So, losing a piece does not compromise the secret, that is, the opponent cannot learn the secret as long as he does not have access to a predetermined number of pieces.

Both secret sharing schemes and cryptographic systems are used in *society-oriented cryptographic systems.* A trivial implementation of a society-oriented cryptographic system would involve the concatenation of a secret sharing scheme with single user cryptography. This arrangement is usually unacceptable as the cooperating subgroup must first recover the cryptographic key. The access to the key, however, may compromise the system as it can be used for more than the requested operation.

This thesis looks at two systems for the management of secret information. The first part looks in detail at secret sharing schemes. The second part focuses on society-oriented cryptographic systems, which are the application of secret sharing schemes in cryptography. In this chapter, we present the basic concepts of cryptographic systems and secret sharing schemes, which are necessary to study the rest of this thesis.

1.1 Introductory Concepts of Cryptography

Cryptography is the science that deals with the design of algorithms, protocols and systems for solving two kinds of security problems: *privacy* and *authentication.* More precisely, cryptography is the use of transformations of data intended to make the data useless to opponents, but meaningful to legitimate receivers. Thus cryptographic techniques can be applied to protect communication channels.

The primitive operation of cryptography is *encryption.* It is an invertible operation, E_K, that converts the message M into a representation $C = E_K(M)$, such that it is meaningless to all parties other than the intended receiver. The legitimate receiver can apply the inverse transformation $D_K = E_K^{-1}$ (the *decryption*) to retrieve the message as,

$$D_K(C) = E_K^{-1}(E_K(M)) = M.$$

The parameter K, that selects the particular transformation to be employed, is called the *key.* Consider a simple transformation to construct a secure communication channel between Alice and Bob. The system is set up by Alice (the sender) who agrees with Bob (the receiver) on a secret key K to communicate a message M with the same length as the binary representation of the secret key, K.

1. The sender, Alice, generates the cryptogram $C = M \oplus K$.
2. Alice transmits the cryptogram to Bob.

To recover the message, Bob uses his secret key K as $M = C \oplus K$.

This technique, if done properly (K is kept secret and used only once) is unconditionally secure. In fact, the protocol uses the well known *one-time pad* encryption method, which was developed in 1917 for telegraph communications [91]. Shannon [151] has shown that such a cryptographic system provides *unconditional secrecy*, that is, no matter how much computing power is available to an opponent, the scheme is unbreakable unless the opponent can guess K.

In spite of offering unconditional secrecy, the one-time pad cryptosystem suffers from the following weaknesses.

- In order to maintain unconditional secrecy a separate random key K, for each message must be generated; thus an unlimited number of keys are needed.
- Since the keystream cannot be reproduced (because of randomness) both the sender and the receiver of the message need to know the keys, that is, a secure channel is required in order to transmit the key.

The operational disadvantages of the one-time pad have led to the development of conditionally secure stream ciphers which are able to retain the positive characteristics of one-time pads while avoiding most of their negative aspects. The plaintext is encrypted in much the same way as the one-time pad, but with deterministically generated pseudo-random sequences.

1.1.1 History

As early as the fifth century B.C. the Spartans established the first system of military cryptography [91]. Cryptography in its early years resembled very much secret writing. The well known Caesar cipher [91], which was used to encrypt military orders, is an example of this generation. In this system characters were transformed using a very simple substitution. It was reasonable to assume that the cryptogram was strong enough as most of the potential attackers were illiterate and hopefully others would think that the document was written in an unknown foreign language.

It was quickly realized that the assumption of an ignorant attacker was not realistic. Most early European cryptosystems were designed to withstand the attacks of educated opponents who knew the encryption process, but did not know the cryptographic key. Additionally, it was requested that the encryption and decryption processes could be done quickly, usually by hand, or with the aid of mechanical devices such as the cipher disk invented by Leon Battisa Alberti [91]. At the beginning of the nineteenth century, the first mechanical-electrical machines were introduced for "fast" encryption. This was the first breakthrough in cryptography.

1.1.2 Modern Cryptography

Shannon [151], in his seminal work, laid the theoretical foundations of modern cryptography. He used information theory to analyse ciphers and considered the so-called *product ciphers*, which use small substitution boxes connected by larger permutation boxes. Substitution boxes, also called S-boxes, are controlled by a relatively short cryptographic key to provide confusion (because of the unknown secret key). The permutation boxes (P-boxes), however, have no key. Their structure is fixed and they provide diffusion. Product ciphers are also termed S-P networks (for more detail, see [117]).

Feistel [57] used the concept of the S-P network to design the Lucifer encryption algorithm. It encrypts 128-bit messages into 128-bit cryptograms using a 128-bit cryptographic key. The designer of the Lucifer algorithm was able to modify the S-P network

in such a way that both the encryption and decryption algorithms could be implemented by a single program or a piece of hardware. Encryption/decryption is done in sixteen rounds. Each round acts on 128-bit input (L_i, R_i) and generates 128-bit output (L_{i+1}, R_{i+1}) using a 64-bit partial key K_i.

The Data Encryption Standard (DES) [122] was the first commercial-grade modern cryptographic algorithm with openly and fully specified implementation details. It was developed from Lucifer and very soon became a standard for encryption in banking and other non-military applications. It uses the same Feistel structure with shorter 64-bit data blocks and a shorter 64-bit key. As a matter of fact, the key contains 56 independent and 8 parity-check bits. Due to its wide utilisation, the DES was extensively investigated and analysed. The experience with the analysis of the DES gave valuable insights into the design properties of cryptographic algorithms. Amongst the many descendants of the DES, whose structure was based on Feistel permutation, are the Japanese Fast Encryption Algorithm (FEAL) [152] and the Australian LOKI [27, 26] algorithm.

Note that all the above mentioned cryptographic systems are *private-key* systems. In private-key cryptosystems, both the encryption and the decryption keys are secret and either the same, or the knowledge of one of them is sufficient to determine the other (this is why private-key systems are also called symmetric systems).

In 1976 Diffie and Hellman [53] introduced the concept of public-key cryptosystems. Public-key cryptosystems (also called asymmetric systems) use two different keys; one is public while the other is kept secret. Clearly, it is required that computing the secret key from the public one has to be intractable. In 1978 three designs based on the notion of public-key systems were published. Rivest, Shamir and Adleman [136] showed how the factorisation problem could be used to construct a public-key cryptosystem (this is the well-known RSA cryptosystem). Merkle and Hellman [119] used the knapsack problem in their construction. McEliece [113] built a system which applied error correcting codes. Later in 1985, ElGamal [55] designed a public-key cryptosystem using the discrete logarithm problem. Miller [121] and Koblitz [95] suggested using elliptic curves to design public-key cryptosystems.

1.1.3 Terminology

Cryptography has quite an extensive vocabulary. More complex terms will be introduced gradually throughout this thesis. There is however a collection of basic terms that are discussed briefly now. These definitions are mainly from Menezes, Oorschot and Vanstone [117], however the reader is also referred to Stinson [167] and Seberry and

Pieprzyk [144].

A *party* is someone or something which sends, receives, or manipulates information.

The *sender* is the party in a communication system that is the legitimate transmitter of information.

The *receiver* is the party in a communication system that is the intended recipient of information.

An *adversary* is a party in a communication system that is neither the sender nor receiver, and which tries to defeat the information security service being provided between the sender and receiver.

Secrecy ensures that information flow between the sender and the receiver is unintelligible to outsiders. It protects information against threats based on eavesdropping.

Integrity enables the receiver to verify whether the message has been tampered with by outsiders whilst in transit via an unsecured channel. It ensures that any modification of the stream of messages will be detected.

An *identification* or *party authentication* assures the parties of their identity.

Message authentication provides to the party which receives a message evidence of the identity of the sender.

A *channel* is a means of conveying information from one party to another.

A *secure channel* (or a *private channel*) is one from which an adversary does not have the ability to reorder, delete, insert, or read.

An *unsecured channel* (or a *public channel*) is one from which parties other than those for which the information is intended can reorder, delete, insert, or read.

Encryption is the primitive cryptographic operation used to ensure secrecy or confidentiality of information transmitted across an unsecured communication channel. The encryption operation takes a piece of information, also called *message* or *plaintext*, and transforms it into a *cryptogram* or *ciphertext* using a secret cryptographic key.

Decryption is the reverse operation to encryption. The receiver who holds the correct secret key can recover the message (plaintext) from the cryptogram (ciphertext).

The *encryption algorithm* (or *decryption algorithm*) is the procedure that describes the step-by-step description of the encryption (or decryption) process. If there is no need to distinguish encryption from decryption, we call them collectively *ciphers* or *cryptosystems.*

Private-key (also called *symmetric*) cryptosystems use the same secret key for encryption and decryption. Although the encryption and decryption keys do not need to be identical, the knowledge of one of them suffices to obtain the other.

Public-key (also called *asymmetric*) cryptosystems use a different key for encryption and decryption. The knowledge of one key, however, does not allow the other to be determined.

A *one-way function* is a function for which it is "easy" to compute its value from its argument(s), but it is "difficult" to reverse it, that is, to find its argument(s) knowing its value.

Cryptanalysis is the study of mathematical techniques for attempting to defeat information security services.

A *cryptanalyst* is someone who engages in cryptanalysis.

Cryptology is the study of cryptography and cryptanalysis.

A *cryptosystem* or a *cryptographic system* is a general term referring to a set of cryptographic primitives used to provide information security services. Most often this term is used in conjunction with primitives providing confidentiality, that is, encryption.

An encryption system is said to be *breakable* if a third party, without prior knowledge of the key, can systematically recover plaintext from corresponding ciphertext within some appropriate time frame. The objective of the following attacks is to systematically recover plaintext from ciphertext, or even more drastically, to deduce the decryption key.

- A *ciphertext-only attack* is one whether the adversary (or cryptanalyst) tries to deduce the decryption key or plaintext by observing ciphertext only. Any encryption scheme vulnerable to this type of attack is considered to be completely insecure.
- A *known-plaintext attack* is one where the adversary has a quantity of plaintext and corresponding ciphertext.
- A *chosen-plaintext attack* is one where the adversary chooses plaintext and is then given corresponding ciphertext. Subsequently, the adversary uses the information deduced in order to recover plaintext corresponding to previously unseen ciphertext or to find the key applied.
- An *adaptive chosen-plaintext attack* is a chosen-plaintext attack wherein the choice of plaintext may depend on the ciphertext received from previous requests.
- A *chosen-ciphertext attack* is one where the adversary selects the ciphertext and is then given the corresponding plaintext. The objective is then to be able to deduce the plaintext from "different" ciphertext.
- An *adaptive chosen-ciphertext attack* is a chosen-ciphertext attack where the choice of ciphertext may depend on the plaintext received from previous requests.

1.1.4 Assessment of a Cryptographic System

The most important criterion to assess a cryptographic system is the security of the system. There are different ways in which a cryptosystem may be secure. Among them we consider the following models for evaluating the security of a cryptosystem.

Unconditional Security

A cryptosystem is said to be unconditionally secure if an adversary having unlimited computational resources cannot defeat the system. Unconditional security for an encryption system is called *perfect secrecy.* For perfect secrecy, observation of the ciphertext provides no information whatsoever to an adversary.

A necessary condition for a symmetric-key encryption system to be unconditionally secure is that the key be at least as long as the message. The one-time pad is an example of an unconditionally secure encryption algorithm. Public-key encryption schemes, however, cannot be unconditionally secure.

Computational Security

This measures the amount of computational effort required, by the best currently-known methods, to defeat a system. A proposed algorithm is said to be computationally secure if the perceived level of computation required to defeat it exceeds the computational resources of the hypothesised adversary.

Most of the best known cryptosystems in current use are computationally secure. The members of this class are sometimes also called *practically secure.*

Provable Security

A cryptosystem is said to be provably secure if the difficulty of defeating it can be shown to be essentially as difficult as solving a well-known difficult problem, such as integer factorisation [64] or the computation of discrete logarithms [127].

Provable security is considered to be adequate for most practical applications. Computational security includes provable security as a proper subset.

1.2 Hashing

There are several cryptographic applications which require the production of a short fingerprint (or a *digest*) of a much longer document/message. Cryptographic applications

of hashing include, amongst others, the generation of digital signatures and message authentication codes[1].

A hashing function h, in general, is a procedure that takes as input a message, M, of arbitrary length and produces a digest, $h(M)$, of a fixed length. In order to assess the security of a hash function, a commonly used criteria is the collision freeness property. A hash function h is called *collision free*, if finding messages M_1 and M_2 with $h(M_1) = h(M_2)$ is a hard problem [43]. A formal definition of a collision free, also called strong one-way hash function, h, is given as follows:

1. h can be applied on any message or document, M, of any size.
2. h produces a fixed size digest $h(M)$.
3. Given h and M, it is easy to compute $h(M)$, but it is computationally intractable to find the message M for the given digest $h(M)$, that is, h is one-way.
4. Given the description of the hash function h, it is computationally infeasible to find two distinct messages M_1 and M_2 which collide, i.e., $h(M_1) = h(M_2)$. That is, h is collision free[2].

Several constructions of hash functions (for different purposes and with different levels of security) have been proposed in the literature (see, for example, [142], [174] and [133]).

1.3 Digital Signatures

Hand-written signatures have been used in everyday situations such as writing a letter, signing a contract, withdrawing money from a bank, and so on. Since a copy of a hand-written signature can usually be distinguished from an original, the signer cannot deny the original signature. This is why the signature is used to take the responsibility of the signer for signed messages.

One of the greatest achievements of modern cryptography is the digital signature. Digital signatures should be in a sense similar to hand-written signatures. Since a copy of electronic documents is identical to the original, digital signatures have to create some sort of digital encapsulation for the document so any interference with either its contents or the signature will be detected with a very high probability. In order to

[1] We will briefly discuss the generation of digital signatures, but readers who are interested in message authentication codes are referred to see the book by Pieprzyk and Sadeghiyan [133].

[2] Obviously there are infinitely many collisions for a hash function h, since the message source is much larger than the digest source.

achieve this requirement, a digital signature on a message is a special encryption of the message that can be applied only by the legitimate signer. That is, in contrast to hand-written signatures, which are independent from the messages, the digital signatures must somehow bind to the message.

Of course, in both hand-written and digital signature schemes a third party (the receiver of the signature) must be able to verify the signature. A hand-written signature is verified by comparing it to other, authentic signatures. For example, in order to withdraw money from a bank, the bank compares the signature with one which is provided at account opening time. A verification of a digital signature, however, needs to apply a particular (in general, a publicly known) algorithm. So, a digital signature scheme is a collection of two algorithms and must have the following properties:

1. The signing algorithm $Sig_K : \mathcal{K} \times \mathcal{M} \rightarrow \Sigma$ assigns a signature $\sigma = Sig_K(M)$, where $M \in \mathcal{M}$ is a message, $K \in \mathcal{K}$ is the secret key of the signer and Σ is the set of all possible values of the signatures.

2. The signing algorithm executes in polynomial time when the secret key K is known. For an opponent, who does not know the secret key, it should be computationally intractable to forge a signature, that is, to find a valid signature for a given message.

3. The verification algorithm $V_k : k \times \mathcal{M} \times \Sigma \rightarrow \{\text{yes, no}\}$ takes a public information $k \in \mathcal{K}$ of the signer, a message $M \in \mathcal{M}$ and a given signature $\sigma \in \Sigma$ of the message M. It returns "yes" if σ is the signature of the message M, otherwise it returns "no".

4. The verification algorithm, in general, is a publicly known (polynomial time) algorithm. So, anyone can use it to check whether a message M matches the signature σ or not.

There are two main classes of digital signature schemes.

Digital Signature Schemes with Appendix

This class of digital signatures require the original message as input to the verification algorithm. They rely on cryptographic hash functions and are the most commonly used in practice. An example of digital signature with appendix is the ElGamal [55] signature scheme (see section 1.5.2).

Digital Signature Schemes with Message Recovery

Digital signature schemes with message recovery have the feature that the message signed can be recovered from the signature itself. In practice, this feature is of use for short messages. This class of digital signatures does not require the knowledge of the message for the verification algorithm. An example of digital signature with message recovery is the RSA [136] signature scheme (see section 1.5.1).

Note. Most digital signatures with message recovery are applied to a message of fixed length, while digital signatures with appendix are applied to messages of arbitrary length.

1.4 Private-key Cryptosystems

A private-key cryptosystem enables two parties, the *sender* and the *receiver*, to communicate in secrecy via an insecure channel (Figure 1.1 illustrates this scenario). Before any communication of messages takes place, both the sender and receiver must exchange the secret key $K \in \mathcal{K}$ via a secure channel. The secure channel can be implemented using a messenger or a registered mail. After exchanging the key, the sender can select a message $M \in \mathcal{M}$, apply the encryption algorithm $E : \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$, and dispatch the cryptogram $C = E_K(M)$ through the insecure channel. The receiver, who knows the secret key K (in modern cryptographic systems the encryption/decryption algorithms are publicly known) recreates the message from the cryptogram using $D : \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$, that is, $M = D_K(C)$.

Clearly, the cryptosystem works correctly if

$$D_K(E_K(M)) = M$$

for all keys $K \in \mathcal{K}$. The well-known private-key cryptosystem which is used in today's cryptographic world is the Data Encryption Standard (DES) algorithm [122]. It was developed at IBM in the mid 70s and was the successor of Lucifer.

1.5 Public-key Cryptosystems

In private-key cryptosystems, both encryption and decryption keys are secret and either the same or the knowledge of one of them is sufficient to determine the other (this is why private-key cryptosystems are also called *symmetric*). For example, DES decryption is identical to DES encryption, but the key schedule is reversed. The main drawback

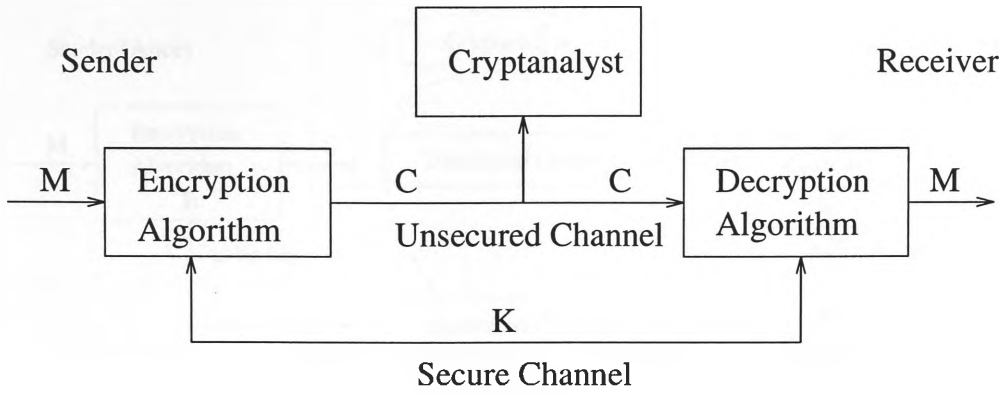

Figure 1.1: Scenario of a private-key cryptosystem

of applying a private-key cryptosystem is that it requires the prior communication of the key K between sender (Alice) and receiver (Bob) using a secure channel, before any cryptogram is transmitted.

The idea behind a public-key cryptosystems is that it might be possible to design a system that uses two different keys, k and K, for encryption and decryption, respectively. The knowledge of one of these keys, however, must not be sufficient (computationally) to determine the other. Hence, one of the keys can be made public by publishing it in a directory (that is where the term public-key comes from). The advantage of a public-key cryptosystem is that, if the encryption key, k, is made public then Alice (or anyone else) can use the public key to send an encrypted message (without a prior communication of the other key) to Bob. Bob is the only person that can decrypt the cryptogram, using the secret key K. Figure 1.2 shows a scenario in which the encryption key of a public-key system is made public.

Although applying a public-key cryptosystem does not require prior communication (to transmit a secret key) between two parties of the system, there must be a trusted public registry (e.g., *White Pages*) that keeps an up-to-date list of all active public-key systems. An entry on the list has to include the name of the receivers along with their original public-keys. The lack of a registry allows an attacker, instead of breaking the public-key cryptosystem, to set up his own system and try to convince senders that the system is someone else's (masquerading attack).

The notion of public-key cryptography was introduced by Diffie and Hellman [53] in 1976. They discussed the shortcomings of private-key cryptosystems and introduced two novel approaches (called public-key cryptosystems and public-key distribution systems) to solve related problems.

Definition 1.1 [53] *A public-key cryptosystem is a pair of families* E_k, $k \in \mathcal{K}$ *and* D_K,

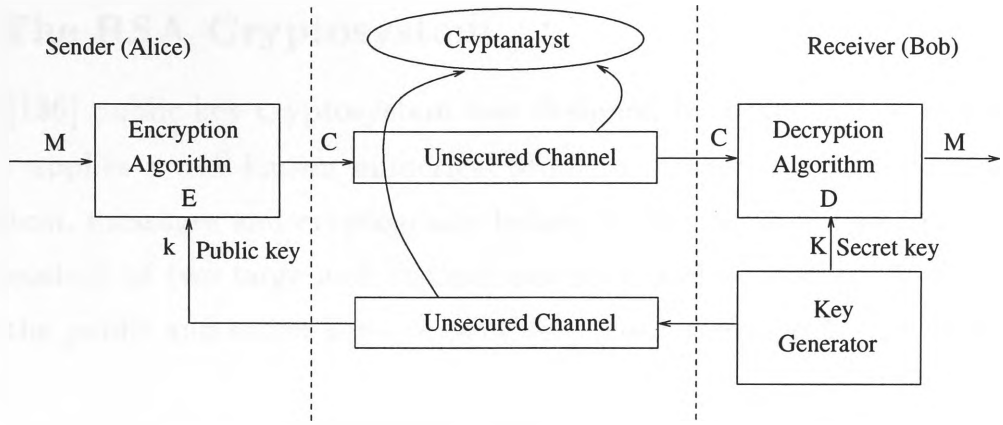

Figure 1.2: Scenario of a public-key cryptosystem with a public encryption key

$K \in \mathcal{K}$ of algorithms representing invertible transformations,

$$E_k \ : \ \mathcal{M} \to \mathcal{M},$$

$$D_K \ : \ \mathcal{M} \to \mathcal{M},$$

such that the following are satisfied (k is the public and K is the secret keys):

1. *For every $k \in \mathcal{K}$, E_k is the inverse of D_K.*
2. *For every $k \in \mathcal{K}$ and $M \in \mathcal{M}$, the algorithms E_k and D_K are easy to compute.*
3. *For almost every $k \in \mathcal{K}$, each easily computed algorithm equivalent to D_K is computationally infeasible to derive from E_k.*
4. *For every $k \in \mathcal{K}$, it is feasible to compute inverse pairs E_k and D_K from k.*

The third property indicates that the user's encryption key can be made public without compromising the security of his secret decryption key. Of course, the public file of encryption keys must be protected from unauthorised modification.

In 1978 three designs based on the notion of public-key systems were published. Rivest, Shamir and Adleman [136] showed how the factorisation problem could be used to construct a public-key cryptosystem (this is the well-known RSA cryptosystem). Merkle and Hellman [119] used the knapsack problem in their construction. McEliece [113] built a system which applied error correcting codes. Later in 1985, ElGamal [55] designed a public-key cryptosystem using the discrete logarithm problem. Miller [121] and Koblitz [95] suggested using elliptic curves to design public-key cryptosystems.

1.5.1 The RSA Cryptosystem

The RSA [136] public-key cryptosystem was designed by Rivest, Shamir and Adleman in 1978. It applies a well-known numerical problem, namely the factorisation problem. In this system, messages and cryptograms belong to the set $\mathbb{Z}_N$. The composite integer N is the product of two large and distinct primes p and q, that is, $N = p \times q$. Let k and K be the public and secret keys, respectively. For a message M the RSA encryption function is

$$C = E_k(M) = M^k \pmod{N}.$$

The RSA decryption function applies the secret key K on the cryptogram $C \in \mathbb{Z}_N$ as follows:

$$M = D_K(C) = C^K \pmod{N}.$$

Since the decryption of an encrypted message M, should provide the original message M, it requires

$$D_K\left(E_k(M)\right) = (M^k)^K = M \pmod{N}. \tag{1.1}$$

The equation (1.1) would have a solution if and only if,

$$k \times K \equiv 1 \pmod{\varphi(N)} \tag{1.2}$$

where $\varphi(N) = (p-1)(q-1)$ is the *Euler's totient function.* Equation (1.2) has a solution if k is coprime to $\varphi(N)$.

RSA Signatures

In the RSA system, the signature algorithm is identical to the decryption. That is, to sign a message M $(0 \leq M < N)$ the owner of the secret key generates the signature using

$$\sigma = M^K \bmod N.$$

The Verification of the signature, however, is similar to the encryption. That is every one who knows the public key k can check the validity of (M, σ) using

$$(\sigma)^k \stackrel{?}{=} M \pmod{N}.$$

If the above equation is satisfied, then the signature is accepted as a valid signature; otherwise the signature is rejected as a forged one. Since the verification recovers the message signed, this sort of RSA signature is a *digital signature with message recovery.*

Note. As is seen, the signing algorithm produces a signature with the same length as the message. This is an expensive and unsatisfactory scheme as it needs double

space for storage and double bandwidth for transmission. Moreover, signing individual blocks has a disadvantage, as blocks may be fraudulently interchanged. For example, RSA signatures are subject to attacks which explore the homomorphic structure of exponentiation. Let $\sigma_1 = M_1^K \bmod N$ and $\sigma_2 = M_2^K \bmod N$ be two valid signatures for messages M_1 and M_2, respectively. It is possible to forge the signature for the message $M_3 = M_1 \cdot M_2 \bmod N$ as $\sigma_3 = \sigma_1 \cdot \sigma_2 = (M_1 \cdot M_2)^K = M_3^d \bmod N$ (for more details, see [89, 90]).

In order to avoid these problems, we shall assume throughout that the document (a message of an arbitrary length) is first hashed and then the signature is produced for its digest. Obviously, the hashing employed has to be collision free and avoid attacks which exploit existing algebraic structures in both the signing algorithm and the hashing function (see [44]).

RSA-type signatures have been the subject of investigation by several authors. Recently, Cramer and Damgård [41] have shown how to generate secure and practical RSA-based signatures.

Implementing The RSA

One obvious attack on the RSA cryptosystem is for a cryptanalyst to factorise the integer N (since knowing the factors p and q provides $\varphi(N)$ and hence the secret key K). So, if the RSA cryptosystem is to be secure, it is necessary that N must be large enough so that factoring it will be computationally infeasible.

The RSA system is set up by the receiver, Bob, who

- chooses two large and distinct primes p and q,
- computes $N = p \times q$ and $\varphi(N) = (p-1)(q-1)$,
- selects at random the key $0 \leq k \leq \varphi(N)$, such that $\gcd(k,\ \varphi(N)) = 1$,
- computes the secret key K using the equation (1.2),
- publishes N and k in a public directory as the public parameters of his RSA system.

Security of RSA

Rivest, Shamir and Adleman [136] considered methods a cryptanalyst may use to break the system. As they have pointed out, all attacks seem to be as difficult as factorisation. That is, the security of the RSA cryptosystem depends on the difficulty of the factorisation problem (the conjecture, however, has not been proved).

Simmons and Norris [161] have shown that the secret key K can be computed using their iteration attack (without factoring N). However, Rivest [135] has shown that if the factors of the RSA modulus are *safe* primes (that is, $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are distinct and large primes) then the probability of a successful iteration attack is negligible for large N.

1.5.2 The ElGamal Cryptosystem

Another notable public-key cryptosystem is the ElGamal [55] cryptosystem, which was designed in 1985 by ElGamal. This cryptosystem applies the discrete logarithm problem. In this system, messages, cryptograms and keys (public and secret) belong to a finite field $GF(p)$. The integer p is prime and for security reasons $p-1$ must have at least one large prime factor. The secret key $K \in GF(p)$, in this system is selected randomly. The public key is obtained from the secret key using

$$k = g^K \pmod{p}$$

where g is a primitive element in $GF(p)$.

Let k and K be the public and secret keys, respectively. For a message M the ElGamal encryption function is

$$C = (c_1, c_2) = (g^r, Mk^r)$$

where $r \in GF(p)$ randomly and uniformly selected by the sender (Alice or anyone else) and all computations are done in $GF(p)$. The ElGamal cryptosystem is *non-deterministic*, since the cryptogram depends on both the message and on the random value r chosen by the sender. That is, there will be many ciphertexts, which are the encryptions of the same plaintext.

The ElGamal decryption function applies the secret key K for the cryptogram C as follows. First, Bob (the receiver) uses his secret key K and the first component of the cryptogram to compute,

$$(c_1)^K = (g^r)^K = \left(g^K\right)^r = k^r.$$

Then he uses the multiplicative inverse of k^r and the second component of the cryptogram to retrieve the message M as,

$$c_2 \times \frac{1}{k^r} = M \times k^r \times \frac{1}{k^r} = M.$$

ElGamal Signatures

The ElGamal signature scheme has been the subject of investigation by several authors and several variants of this signature scheme have been introduced in the literature. In fact, the ElGamal system is designed specifically for the purpose of signatures, as opposed to the RSA system, which is suitable for both signature and cryptographic purposes.

The ElGamal signature scheme is a non-deterministic signature scheme (like the ElGamal encryption scheme), that is, there are many valid signatures for a given message. Thus, the verification algorithm must be able to accept any of the valid signatures as authentic. The description of the ElGamal signature scheme is as follows. As in the ElGamal cryptosystem, let K be the secret key and $k = g^K \pmod p$ be the public key (note that g and p are also public). Assume that Bob, the owner of the system, wishes to sign a message M, $M \in \mathbb{Z}_p$. First, Bob selects a random integer r, $r \in \mathbb{Z}_p$ such that $\gcd(r, p-1) = 1$ and calculates

$$x = g^r \pmod p.$$

Then, he solves the following congruence

$$M \equiv K \cdot x + r \cdot y \pmod{p-1}$$

for variable y. The signature of the message is

$$\sigma = Sig_K(M) = (x, y).$$

Upon reception of M and $\sigma = (x, y)$, Alice (or anyone else, who knows the public parameters of Bob's ElGamal cryptosystems) can verify Bob's signature using,

$$g^M \stackrel{?}{\equiv} k^x \times x^y.$$

Note that possessing the pair (x, y) does not allow the message M to be recreated, that is, the ElGamal signature is a *digital signature with appendix.* In fact, there are many pairs which match the message – for every random value r there is a pair (x, y). However, Nyberg and Rueppel [124] have proposed ElGamal-type signature schemes with message recovery.

The Digital Signature Standard (DSS)

In 1991, the U.S.National Institute of Standards and Technology (NIST) proposed a Digital Signature Algorithm (DSA). The DSA has become a U.S. Federal Information

Processing Standards (FIPS) called the *Digital Signature Standard* (DSS [56]), and is the first digital signature scheme recognised by any government. The algorithm is a digital signature scheme with appendix. Since DSS is a modified version of the ElGamal signature scheme, for the sake of completeness we briefly describe the scheme. The parameters of this system are:

- p, a prime modulus, L bits long, where L is a multiple of 64 ($512 \leq L \leq 1024$);
- q, a 160-bit prime, such that $q|p-1$;
- $g = h^{(p-1)/q} \pmod{p}$, where $h \in GF(p)$ is any integer such that $g > 1$ (g is an element of order q in $GF(p)$);
- K, an integer (the secret key) with $0 < K < q$;
- k, an integer (the public key), where $k = g^K \pmod{p}$;
- r, a random integer with $0 < r < q$.

The system parameters p, q, g and k are public. To generate a signature on message M ($0 \leq M < q$), the user chooses a *one-time* random integer r, computes $r^{-1} \bmod q$, and calculates,

$$z = \left(g^{r^{-1}} \pmod{p}\right) \bmod q,$$

and

$$\sigma = r\,(M + Kz) \bmod q.$$

The pair (z, σ) is the desired signature.

To verify a signature, the recipient uses,

$$z \stackrel{?}{=} g^{M\sigma^{-1}} k^{z\sigma^{-1}} \pmod{p}.$$

Implementing the ElGamal

One obvious attack on the ElGamal cryptosystem is for a cryptanalyst to obtain the random value r, using the first component of the cryptogram and solve the discrete logarithm for g^r. If this can be done, it is simple to obtain the message, M, by computing k^r and applying its multiplicative inverse on the second component of the cryptogram. Hence, if the ElGamal cryptosystem is to be secure, it is necessary that p must be large enough that solving the discrete logarithm over $GF(p)$ is computationally infeasible.

Let p be a prime such that the discrete logarithm problem in $\mathbb{Z}_p$ is intractable, and let $0 < g < p-1$ be a primitive element. The following procedure shows how an ElGamal system can be set up by the receiver (Bob).

- Bob chooses (uniformly at random) the secret key K, $0 < K < p-1$,
- he computes the public key, $k = g^K \pmod{p}$,
- Bob publishes g, p and k in a public directory, as the parameters of his ElGamal cryptosystem.

Security of ElGamal

ElGamal considered methods a cryptanalyst may use to break his system [55]. As he pointed out, all attacks seem to be as difficult as the discrete logarithm problem. That is, the security of the ElGamal cryptosystem hinges on the difficulty of the discrete logarithm problem. Note that these conjectures are based on the assumption that the public parameters of the system are chosen properly. Bleichenbacher [15] has shown if the public parameters of the system are not chosen properly, a particular attack is effective in forging an ElGamal signature. Since the secret key is not found in this attack, the difficulty of forging an ElGamal signature is sometimes weaker than the difficulty of the underlying discrete logarithm problem. Anderson and Vaudenay [1] have also discussed the effect of choosing improper public parameters.

Note. The ElGamal system is recommended to be used once only for any single integer r, $0 < r < p-1$. Every time Alice (or anyone else) wants to send a message, she has to generate at random a new r. The violation of this requirement can be exploited by an opponent, say Oscar, who wants to decrypt a cryptogram knowing the message corresponding to another cryptogram. To illustrate the point, assume that Alice was careless and sent two cryptograms using the same r. Let them be: $C = (c_1, c_2) = (g^r, M_1 k^r)$ and $\hat{C} = (\hat{c}_1, \hat{c}_2) = (g^r, M_2 k^r)$. Oscar computes,

$$\frac{c_2}{\hat{c}_2} = \frac{M_1}{M_2}$$

which provides the opportunity to learn the message M_1 (or M_2) knowing the message M_2 (or M_1).

Efficiency

The ElGamal encryption algorithm requires two exponentiations, namely $g^r \bmod p$ and $k^r \bmod p$ (which is about two times of the RSA system). Although these exponentiations can be sped up by selecting random exponent r having some particular structure (e.g.,

having low Hamming weights[3]), care must be taken that this does not make the system prone for any possible attack.

Another disadvantage of ElGamal encryption is that there is a message expansion by a factor of 2. That is, the ciphertext is about twice as long as the corresponding plaintext.

1.5.3 Some Other Public-key Cryptosystems

Almost all existing society-oriented cryptosystems, which are the subject of investigation for this thesis, utilise either the RSA or the ElGamal public-key cryptosystems. For the sake of completeness, we give an overview of some other public-key cryptosystems that are introduced in the literature of public-key cryptography.

The Merkle-Hellman Cryptosystem

The Merkle-Hellman cryptosystem [119] was first described by Merkle and Hellman in 1978. Although this cryptosystem utilises the knapsack problem, which is a very difficult problem in number theory, it was broken by Shamir [149].

There is also a version of the Merkle-Hellman system, called the iterated Merkle-Hellman system. All variants of this cryptosystem were broken in the early 1980's. Readers interested in details of breaking the Merkle-Hellman system are referred to the book by O'Connor and Seberry [126].

McEliece Cryptosystem

McEliece [113] suggested utilising error-correcting codes in the design of a public-key cryptosystem. The purpose of an error-correcting code is to correct random errors that occur in the transmission of binary data through a public channel. This system, however, has not been studied well since error-correcting codes require data expansion that is not desirable in cryptographic systems. Another problem with this cryptosystem is that it is not suitable for producing signatures.

Elliptic Curve Cryptosystems

The RSA and ElGamal cryptosystems utilise cyclic groups which exist in their underlying algebraic structures. Since elliptic curves can be applied in cyclic groups, they can be used in cryptographic applications. The idea of applying elliptic curves in cryptography

[3]The Hamming weight of an integer is the number of ones in its binary representation.

is due to Koblitz [95] and Miller [121]. Koyama, Maurer, Okamoto and Vanstone [97] proposed an elliptic curve variant of the RSA system. Menezes, Okamoto and Vanstone [118] proposed an elliptic curve variant of the ElGamal system.

1.6 Introductory Concepts of Secret Sharing Schemes

There is a clear need for providing the secrecy of sensitive information. Examples of such information include the code to activate a nuclear weapon, cryptographic master keys in a Key Distribution Centre (KDC), proprietary trade-secret formulae, and so on. The sensitive secret information is collectively called *secret* or *key*. Clearly, assigning such sensitive information to an individual is not a good solution (what happens if the legitimate owner of the secret information loses the key or is himself incapacitated?). An alternative solution could be to provide copies of the secret and assign each legitimate member one copy of the key. This scheme, however, increases the threat of theft, loss, or abuse of the secret.

Since a group is more reliable and more trustworthy than an individual (this is the reason why, in an organisation, crucial decisions are left to a group of people rather than to an individual), a reasonable solution could be to partition the secret into several pieces in such a way that the original secret can be reconstructed from those pieces, but each piece by itself provides no information about the secret. So, losing a piece does not compromise the secret, that is, an opponent cannot learn the secret as long as he does not have access to all partial information. Consider a simple scheme for sharing secret information K between Alice and Bob. The scheme is established by a dealer Tom, who generates the pieces of the secret (also called shares). The steps are:

1. Tom, who knows the secret K, generates a random bit string R (with the same length as the binary representation of K).
2. Tom calculates $T = R \oplus K$.
3. Tom gives T to Alice and R to Bob.

To reconstruct the secret, Alice and Bob pool their shares together and find the secret as, $K = R \oplus T$.

This technique, if done properly and the shares R and T are kept secret, is secure. That is, each piece by itself provides no information about K (Alice and Bob only can learn the length of the secret), however, the secret uniquely can be reconstructed from their information.

It is not difficult to extend this scheme to distribute the secret among many participants. However, there is a problem with this scheme; if any of the pieces gets lost, so does the secret. In other words, if any of the shareholders refuse to cooperate in the secret reconstructing phase then the secret remains undetermined. The goal of designing secret sharing schemes is to solve this and many other related problems.

1.6.1 History

In 1979 Shamir [147] introduced his easy and elegant secret sharing scheme. His motivation was to solve the following sort of problem:

> "Eleven scientists are working on a secret project. They wish to lock up the documents in a cabinet so that the cabinet can be opened if and only if six or more of the scientists are present. What is the smallest number of locks needed? What is the smallest number of keys to the locks each scientist must carry?"

Shamir's answer to the above mentioned problem was that, the minimal solution uses 462 locks and 252 keys per scientist. However, it was noted that these numbers are clearly impractical and they become exponentially worse when the number of scientists increases. So Shamir proposed a non-mechanical solution to the above and many other related problems. More precisely, he showed how to divide a data K into n pieces in such a way that K can be reconstructed from any t ($t \leq n$) pieces, but even complete knowledge of $t-1$ pieces reveals absolutely no information about the K. Such secret sharing schemes are called (t, n) *threshold* schemes. In order to achieve this goal, Shamir suggests utilising polynomial interpolation or any other collection of functions which are easy to evaluate and to interpolate.

It is worth mentioning that in 1979, Blakley [12] and Chaum [36] have also, independently, introduced their secret sharing schemes. With contrast to the algebraic nature of the Shamir scheme, Blakley's secret sharing scheme has a geometric nature. That is, in Blakley's scheme the secret is a point in an n dimensional space. The shares of participants, however, are hyper-planes of this geometric space such that every t (*out-of-n*) of the shares uniquely determines the secret point. Although Shamir's construction has been widely applied in society-oriented cryptographic systems, the Blakley type geometrical secret sharing schemes have been applied to solve several problems regarding to sharing a secret (see, for example, [154, 157, 158]). Unfortunately, Chaum's scheme [36], with its mechanical nature, is not very practical, and therefore has not received much

attention in consideration of secret sharing schemes.

1.6.2 Terminology

Since the invention of secret sharing schemes in 1979, they have been studied by numerous authors. Properties of the proposed secret sharing schemes are discussed and several new structures have been proposed to build secret sharing schemes. The essential idea of all these schemes is to protect the secrecy and integrity of information by distributing the information over different locations. In order to achieve this goal, someone who knows the secret, called the *dealer*, distributes the secret among n members (or *participants*) in such a way that a specified sets of participants must cooperate in order to determine the secret.

The set of all participants is $\mathcal{P} = \{P_i \mid i = 1, \ldots, n\}$. The partial information s_i, which is given (in private) to participant P_i, is called the *share* of participant P_i from the secret.

Every set of participants that is designated to be able to recover the secret is called an *access set* or an *authorised set.* Similarly, every set of participants which is not designated to be able to recover the secret is called an *unauthorised set.* So, the family of all sets $2^{\mathcal{P}}$ can be partitioned into two classes:

1. the class of authorised sets Γ, the so called *access structure*,
2. the class of unauthorised sets $\Gamma^c = 2^{\mathcal{P}} \setminus \Gamma$.

The set $\mathcal{K}$ of all possible values of the secret is called the *secret set* and the set $\mathcal{S}$ of all possible values of the shares is known as the *share set.*

We shall assume throughout that $\mathcal{P}$, $\mathcal{K}$ and $\mathcal{S}$ are all finite sets. Secret sharing schemes with an infinite set of secrets have been discussed in [40].

In the secret reconstruction phase, participants of an access set pool their shares and recover the secret. Alternatively, participants could give their shares to a trusted authority, called the *combiner*, to perform the computation for them. Thus a secret sharing scheme for the access structure Γ is the collection of two algorithms:

1. **the dealer** – this algorithm has to be run in a secure environment by a trustworthy party. The algorithm uses the function

$$f : \mathcal{K} \times \mathcal{P} \to 2^{\mathcal{S}}$$

which for a given secret $K \in \mathcal{K}$ and a participant $P_i \in \mathcal{P}$, assigns a set of shares from the set $\mathcal{S}$, that is, $f(K, P_i) = s_i \subseteq \mathcal{S}$ for $i = 1, \ldots, n$,

2. **the combiner** – this algorithm has to be executed collectively by cooperating participants. It is enough to assume that the combiner is embedded in a tamper-proof module and all participants have access to it. Also the combiner outputs the result via secure channels to cooperating participants. The combiner applies the function

$$g : \mathcal{S}_{\mathcal{A}} \rightarrow \mathcal{K}$$

to calculate the secret. If $\mathcal{A} \in \Gamma$ then the combiner is able to recover the secret. If the group of the cooperating participants does not belong to the access structure, either the combiner fails to compute the secret (e.g., in a geometrical secret sharing scheme the contributed shares may not hit the publicly known object which contains the secret) or there is no guarantee that the computed value is the original secret.

Not all secret sharing schemes require secure communication channels among the authorised set of participants (or participants and the combiner) in order to reconstruct the secret. Beimel and Chor [6] proposed a secret sharing scheme which performs the secret reconstruction protocol over public communication channels.

We also employ the following definition used in [83]:

Definition 1.2 *For any access set $\mathcal{A} \in \Gamma$ $(\mathcal{A} \subset \mathcal{P})$, any superset $\mathcal{A}'$ of $\mathcal{A}$ $(\mathcal{A} \subset \mathcal{A}')$ must be an access set as well.*

This is the well-known *monotone* property [8]. Thus we have:

$$\mathcal{A} \in \Gamma \text{ and } \mathcal{A} \subset \mathcal{A}' \subset \mathcal{P} \text{ imply that } \mathcal{A}' \in \Gamma$$

It is in fact very difficult to imagine a meaningful secret sharing scheme which does not satisfy this property. Beutelspacher [11] attempted to model a secret sharing scheme with a non-monotone access structure. In this scheme, there exist two types of shares (negative and positive) and a trustworthy machine is needed to operate on the secret reconstruction protocol. However, Obana and Kurosawa [125] have shown that there exists no such scheme, if one does not assume that the reconstruction machine is trustworthy.

An immediate consequence of the monotone property is that an access structure can be defined uniquely by its *minimal access sets.* An access set $\mathcal{A} \in \Gamma$ is minimal if

$$\mathcal{A}' \in \Gamma, \ \mathcal{A}' \subseteq \mathcal{A} \text{ implies } \mathcal{A}' = \mathcal{A}.$$

We use Γ^- to denote the representation of Γ in terms of minimal access sets.

On the other hand, for any access structure Γ, the family of unauthorised sets $\Gamma^c = 2^{\mathcal{P}} \setminus \Gamma$ has the property in which, given any unauthorised set $\mathcal{B} \in \Gamma^c$, then any set $\mathcal{B}' \subset \mathcal{B}$ must be an unauthorised set as well [83]. An immediate consequence of this property is that for any access structure Γ the set of unauthorised sets can be determined uniquely by its *maximal* sets. We use Γ^{c+} to denote the representation of Γ^c in terms of maximal sets.

We require that for any participant P_i $(1 \leq i \leq n)$ there exists an unauthorised set of participants who can recover the secret once P_i contributes with his share. That is,

$$\text{for all } P_i \in \mathcal{P} \text{ there exists } \mathcal{A} \in \Gamma^- \text{ such that } P_i \in \mathcal{A}.$$

Schemes which satisfy this property have been termed *connected* [23].

A useful way of representing an access structure is by using Boolean expressions. We introduce the concept through an example and refer the reader to [157] for a more precise treatment.

Example 1.1 *Let $\mathcal{P} = \{P_1, P_2, P_3\}$. Then $\Gamma^- = P_1P_2 + P_2P_3$ is an access structure consisting of two access sets $\{P_1, P_2\}$ and $\{P_2, P_3\}$ denoting that participants P_1 and P_2 (or participants P_2 and P_3) cooperatively can recover the secret. That is, for every minimal access set one product term consisting of the participants in the access set is included. The set of unauthorised sets Γ^c, however, can be represented as $\Gamma^c = \{P_1, P_2, P_3, P_1P_3\}$ and thus $\Gamma^{c+} = \{P_2, P_1P_3\}$.*

To represent an access structure, Γ, in terms of its access sets, $\mathcal{A}_i$ $(i = 1, \ldots, \ell)$, we use the two representations $\Gamma = \mathcal{A}_1 + \cdots + \mathcal{A}_\ell$ and $\Gamma = \{A_1, \ldots, \mathcal{A}_\ell\}$ equivalently.

Note. Almost all secret sharing schemes are *one-time* schemes. That is, once an authorised set of participants reconstruct the secret (by pooling their shares or with the help of the combiner) both the secret and all shares become known to everyone within the group, and there is no further secret. In many applications of secret sharing schemes, e.g., shared decryption and shared generation of signatures (see the second part of this thesis) participants apply a function of their shares, without revealing their shares or recovering the secret, and thus after performing the task neither the combiner nor the participants learn about the shares and/or the secret itself. In order to achieve this capability, in general, a *one-way function* is applied (informally, a function $f(x)$ is one-way if computing $y = f(x)$ for a known value of x is easy, but knowing y it is difficult to find x). Some authors used this technique to propose secret sharing schemes with desirable capabilities. For example, He and Dawson [78] used this technique to propose *multistage*

secret sharing schemes in which participants can use their shares to recover different secrets, stage by stage in a specified order. Cachin [29] applied one-way functions to construct *on-line* secret sharing schemes and Pinch [134] has extended it to construct *on-line multiple* secret sharing schemes.

1.6.3 Assessment of a Secret Sharing Scheme

To assess the quality of a secret sharing scheme two kinds of measures are used: *security* measures and *efficiency* ones.

Security

The security measure can be expressed by the number of intelligent guesses that an unauthorised set would have to make in order to have a guarantee of determining the secret. Two types of security have been discussed in the literature of secret sharing schemes:

1. Secret sharing schemes with *unconditional security.* This means that the security of the system is independent of the time and resources available to any opponent who is trying to interfere with the procedure.

2. Secret sharing scheme with *conditional security.* This means that the security of the system relies on the difficulty of computing some "difficult" problems (to study on difficult problems see [64]).

A secret sharing scheme is *perfect* [168] if the probability of an unauthorised set of participants being able to determine the secret is no better than that of an outsider, and therefore is no better than guessing the secret.

The basic tool in studying secrecy is the notion of *entropy*, a concept introduced by Shannon in 1948. Entropy can be thought of as a mathematical measure of information or uncertainty, and is computed as a function of a probability distribution.

Let a random variable K take on a finite set of values according to a probability distribution *Prob(K)*. The uncertainty about the information gained by an event which takes place according to distribution *Prob(K)* is called the entropy of K and is denoted by $H(K)$. The formal definition of entropy is as follows [167].

Definition 1.3 *Suppose K is a random variable which takes on a finite set of values*

according to a probability distribution Prob(K). Then the entropy of this probability distribution is defined by the quantity

$$H(K) = -\sum_{i=1}^{n} Prob(K = k_i) \log_2 (Prob(K = k_i))$$

where k_i are the possible values of K, $1 \leq i \leq n$.

Using the concept of entropy, in a perfect secret sharing scheme for every unauthorised set of participants the conditional entropy of the secret is

$$H(\bar{K} \mid \bar{s}_{i_1}, \ldots, \bar{s}_{i_m}) = \begin{cases} 0 & \text{if } \{P_{i_1}, \ldots, P_{i_m}\} \in \Gamma \\ H(\bar{K}) & \text{otherwise} \end{cases}$$

where $\bar{K}$ and $\bar{s}_{i_j}$ are corresponding random variables for K and s_{i_j}, respectively.

Efficiency

There are several issues in the implementation of an efficient secret sharing scheme. Among these parameters are the total size of shares generated in the system, the total size of shares stored by participants, the total size of communications in the system and the upper and lower bounds on the randomness required by the dealer to set up a secret sharing scheme (for more study on the efficiency of a secret sharing scheme see [87], [86] and [17]).

The most studied measure of efficiency for a secret sharing scheme is the amount of storage needed to store the shares (for example, [24], [109]). Hence, the efficiency of secret sharing schemes can be measured by their *information rate* [165] and *average information rate* [109].

The information rate of the secret sharing scheme is defined to be:

$$\rho = \min_{i=1,\ldots,n} \frac{\log_2 |\mathcal{K}|}{\log_2 |\mathcal{S}_i|}.$$

A secret sharing scheme is *ideal* [22] if $|\mathcal{K}| = |\mathcal{S}|$, that is, the length of the share assigned to each participant is the same as the length of the secret.

In [8] Benaloh and Leichter showed that it is not always possible to construct an ideal scheme for an arbitrary access structure. Subsequent to Benaloh-Leichter's result, several authors worked on improving the efficiency of such schemes and reducing the size of the shares given to participants (see, for example, [24], [157], [160], [108], [30], [10], [19], [18] and [166]).

The average information rate is defined as:

$$\tilde{\rho} = \frac{|\mathcal{P}| \log_2 |\mathcal{K}|}{\sum_{P_i \in \mathcal{P}} \log_2 |\mathcal{S}_i|}.$$

It is clear that the information rate can never be greater than the average information rate, that is, for any secret sharing scheme we have $\tilde{\rho} \geq \rho$.

Note. Every secret sharing scheme requires some public information in the form of the description of the system and its parameters. The amount of this public information is not used as a measure of the performance of the systems as yet. Some systems have utilised extra public information and/or broadcast messages to achieve other goals. For example, Laih et al [101] and Blundo et al [16] have used public information to construct *dynamic* secret sharing schemes, which are similar to Simmons [156] *prepositioned* secret sharing scheme. In a dynamic secret sharing scheme, the dealer has the feature of being able to activate the access structure by sending all participants the same broadcast message (note that, before sending this broadcast message, the participants' shares are not sufficient to recover the secret).

Part I

Secret Sharing Schemes

Contents

This part of thesis is concerned with secret sharing schemes. It consists of three chapters, Chapters 2, 3, and 4.

In Chapter 2 we consider an abstract model for secret sharing scheme. Equivalent secret sharing schemes, updating a secret sharing scheme and extended capabilities for secret sharing schemes will also be discussed in this chapter.

Chapter 3 considers threshold secret sharing schemes. After a brief introduction to the notion of threshold scheme, an abstract model for threshold schemes and some well known approaches to the construction of threshold secret sharing schemes, namely polynomial approach (due to Shamir [147]), geometrical approach (due to Blakley [12]), modular approach (due to Asmuth and Bloom [4]), and vector space approach (due to Brickell [22]) will be discussed. The final section of this chapter considers the concept of proactive secret sharing schemes.

In Chapter 4 secret sharing schemes with arbitrary access structures will be discussed. Several approaches to the construction of general schemes and the concept of computationally secure secret sharing schemes will be considered in this chapter. An online secret sharing scheme and the cheating problem in the secret reconstruction phase will be considered. Finally, extended capabilities for secret sharing schemes will be studied and two solutions for constructing secret sharing schemes in multilevel and compartmented access structures will be presented.

Chapter 2

Secret Sharing Schemes

This chapter introduces the concept of secret sharing. In Section 2.1 an abstract model for secret sharing will be described. The model is due to Martin [108] and is based on the model used by Brickell and Davenport [23][1]. Two important issues in secret sharing schemes, namely equivalent secret sharing schemes and updating a secret sharing scheme, will be discussed in Sections 2.2 and 2.3. The final Section of this chapter considers extended capabilities for secret sharing schemes.

2.1 An Abstract Model for Secret Sharing

Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be a set of n participants and let Γ be a monotone access structure over the set $\mathcal{P}$. Let X be a matrix with at least $n+1$ columns indexed from the set $W = \{D, 1, \ldots, n\}$ such that no two rows of matrix X are identical. Also let d be a mapping such that $d : \mathcal{P} \rightarrow 2^W$. For $\mathcal{A} \subseteq \mathcal{P}$, let $d(\mathcal{A}) = \{d(P_i) \mid P_i \in \mathcal{A}\}$.

In order to construct a *perfect* secret sharing scheme that realizes the access structure Γ, the dealer $\mathcal{D}$ chooses a matrix X such that:

- the column D contains entries from the set $\mathcal{K}$,
- the remaining columns contain entries from some finite set $\mathcal{S}$,
- if $\mathcal{A} \in \Gamma$ then $d(\mathcal{A})$ determines the secret K, that is, there is a unique row j whose entries match the shares of cooperating participants.
- if $\mathcal{A} \notin \Gamma$ then $d(\mathcal{A})$ determines nothing about the secret K, that is, the column D contains all possible values for the secret K for the rows corresponding to shares of $\mathcal{A}$.

[1] A similar model has also been discussed by Stinson [165].

To illustrate the model we employ the following example from Martin [108].

Example 2.1 *Let* $\Gamma = P_1P_2 + P_2P_3 + P_3P_4$ *be a monotone access structure over the set* $\mathcal{P} = \{P_1, P_2, P_3, P_4\}$ *and let* $\mathcal{K} = \mathcal{S} = \{0, 1\}$.

Set-up Phase:

1. *The dealer,* $\mathcal{D}$, *generates*

$$X = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \end{array} \begin{array}{c} \begin{array}{ccccc} D & 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

   where $W = \{D, 1, 2, 3, 4\}$ *and* $d : \mathcal{P} \to 2^W$ *is given by,*

$$d(P_1) = \{1\}, \quad d(P_2) = \{2\}, \quad d(P_3) = \{1, 3\}, \quad d(P_4) = \{4\}.$$

2. $\mathcal{D}$ *randomly chooses a row,* r_j $(1 \leq j \leq 8)$ *from the matrix* X.

3. *Each participant* P_i, $1 \leq i \leq 4$, *receives (in private) share(s)* s_i *as determined by* $d(P_i)$. *That is,* P_1 *receives the entry in column 1,* P_2 *receives the entry in column 2,* P_3 *receives the entries in both columns 1 and 3, and finally* P_4 *receives the entry in column 4 (the secret is the entry in column D).*

Secret Reconstruction Phase:

1. *An authorised set,* $\mathcal{A}$, *of participants recovers the value of the secret (the entry in column D) among the rows of* X *whose entries in the columns of set* $d(\mathcal{A})$ *match their shares.*

As can be seen, any authorised set of participants can determine the secret uniquely. On the other hand, for any unauthorised set, $\mathcal{B}$, uncertainty of the secret is the same as for an outsider – both values 0 and 1 with equal probability can be the unknown value of K. That is, the model provides a perfect secret sharing scheme. Moreover, no matter how much computing power has an opponent it cannot obtain anything about the secret. In other words, the scheme is unconditionally secure.

2.2 Equivalence of Secret Sharing Schemes

Since the invention of secret sharing scheme in 1979, several models have been introduced in the literature. A frequently raised question is that concerning when two different secret sharing schemes are in fact equivalent. Following Martin's definition [108], we say two connected secret sharing schemes are equivalent if:

1. their access structures are the same,
2. their secret spaces are of the same cardinality,
3. they have the same probability associated with each key for every subset of participants,
4. the size of the share held by a given participant is the same for each scheme.

Note. In perfect secret sharing schemes, for every subset $\mathcal{A} \subseteq \mathcal{P}$, either $\mathcal{A} \in \Gamma$ so $\mathcal{A}$ can determine the secret uniquely (with probability one) or $\mathcal{A} \notin \Gamma$ and therefore cannot obtain anything about the secret. That is, in case of perfect secret sharing schemes condition (3) is satisfied.

Example 2.2 *Let* $\Gamma = P_1P_2$, *where* $\mathcal{P} = \{P_1, P_2\}$. *Then*

$$X_1 = \begin{array}{c} \begin{matrix} D & 1 & 2 \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 2 & 2 \\ 0 & 3 & 3 \\ 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix} \end{array} \qquad \textit{and} \qquad X_2 = \begin{array}{c} \begin{matrix} D & 1 & 2 \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \end{array}$$

where $W = \{D, 1, 2\}$ *and* $d : \mathcal{P} \to 2^W$ *is given by,*

$$d(P_1) = \{1\}, \quad d(P_2) = \{2\},$$

are not equivalent secret sharing schemes, since shares in scheme X_1 *belong to the set* $\mathbb{Z}_4$ *while shares in scheme* X_2 *belong to the set* $\mathbb{Z}_2$.

However, a secret sharing scheme associated with the following matrix,

$$X = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \end{array} \begin{array}{c} \begin{array}{ccccc} D & 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 & 0 \\ 0 & 1 & 1 & 3 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 2 & 1 \\ 1 & 1 & 0 & 3 & 0 \end{pmatrix} \end{array}$$

where $W = \{D, 1, 2, 3, 4\}$ *and* $d : \mathcal{P} \to 2^W$ *is given by,*

$$d(P_1) = \{1\}, \quad d(P_2) = \{2\}, \quad d(P_3) = \{3\}, \quad d(P_4) = \{4\},$$

is equivalent to the scheme given in Example 2.1.

2.3 Updating a Secret Sharing Scheme

A desirable capability of every secret sharing scheme is that the scheme must be able to cope with situations when new members join the group (enrolment) or when members leave the group (disenrolment). Indeed, it is not very practical to modify the shares each time a member leaves the group, or when a new member joins the group. This is a common problem with long-lived and sensitive information (e.g., cryptographic master keys) in which the secret must be kept secure even if some shareholders enrol or disenrol.

An alternative scenario is that after the initial setups of a secret sharing scheme, one (or more) participant is no longer deemed to be trustworthy. Martin [110] studied the effect of untrustworthy participants on the security of the system when a dishonest participant reveals its share. Blakley, Blakley, Chan and Massey [13] constructed a disenrolment scheme which allows reorganisation of the system in the case that such an event occurs. Chaum [37], Simmons [159], Charnes, Pieprzyk and Safavi-Naini [34] have also discussed on this problem.

2.3.1 Extension of a Secret Sharing Scheme

In this section we consider the case that some trustworthy members are joining the group. More precisely, we are dealing with the *extension* of a secret sharing scheme, in such a

way that the old secret and old shares are still valid in the extended system, however, some new shareholders and new shares are included in the system.

Definition 2.1 [83] *Let a secret sharing scheme realise* $\Gamma_1 \subset 2^{\mathcal{P}_1}$ *on a set* $\mathcal{P}_1$. *Further assume that,* $\mathcal{P}_1 \subset \mathcal{P}_2$ *and a scheme realizes an access structure* $\Gamma_2 \subset 2^{\mathcal{P}_2}$. *The scheme which realizes* Γ_2 *is an extension of the scheme that realizes* Γ_1 *if:*

(a) both schemes allow to recover the same secret,

(b) the collection of shares defined in Γ_1 *is a subset of shares generated in* Γ_2.

(c) $\Gamma_1 \subset \Gamma_2$, *that is, any access set in* Γ_1 *is an access set in* Γ_2.

(d) $\Gamma_1^c \subset \Gamma_2^c$, *that is, any unauthorised set* $\mathcal{B}_i \subset \mathcal{P}_1$ *is still an unauthorised set.*

Note. The authors of [83] did not define explicitly the extension of a secret sharing scheme. The definition 2.1 reflects the assumptions and properties of the extension method used in [83]. In fact properties (b), (c) and (d) can be derived from Lemma 1 in [83]. The assumption (a) is derived from the proof in [83] in which Ito, Saito and Nishizeki selected the degree of a polynomial for the extended Shamir scheme in such a way that the secret in the extended scheme is the same as in the old scheme.

Example 2.3 *Let* $\Gamma_1 = P_1P_2$, *where* $\mathcal{P}_1 = \{P_1, P_2\}$. *Then*

$$X_1 = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{ccc} D & 1 & 2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \end{array}$$

where $W = \{D, 1, 2\}$ *and* $d : \mathcal{P} \to 2^W$ *is given by,*

$$d(P_1) = \{1\}, \quad d(P_2) = \{2\},$$

is a perfect secret sharing scheme. Assume that we want to include participants P_3 *and* P_4 *to the system in such a way that the extended scheme realizes the access structure* $\Gamma_2 = P_1P_2 + P_2P_3 + P_3P_4$ *on the set* $\mathcal{P}_2 = \{P_1, P_2, P_3, P_4\}$ *of participants. We observe*

that the matrix,

$$X_2 = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \end{array} \begin{array}{c} \begin{array}{ccccc} D & 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

where $W = \{D, 1, 2, 3, 4\}$ *and* $d : \mathcal{P} \rightarrow 2^W$ *is given by,*

$$d(P_1) = \{1\}, \quad d(P_2) = \{2\}, \quad d(P_3) = \{1, 3\}, \quad d(P_4) = \{4\}.$$

realizes Γ_2 *over the set* $\mathcal{P}_2$*. To construct the extended scheme, however, the randomness of the dealer is somehow decreased. That is, if in the basic scheme the secret and shares are associated with row* x_i, $1 \leq i \leq 4$*, then in the extended scheme the dealer must choose (randomly) one of the rows* r_{2i-1} *or* r_{2i}*. The dealer then distributes the shares according to mapping d only to new participants.*

Note that extending a threshold scheme such that the threshold parameter remains unchanged is an easy task since shares can be dynamically generated without affecting the other shares. However, it is not a trivial task for general schemes or for threshold schemes where the threshold parameter needs to be updated as well. In order to illustrate the problem, we shall show that both extended schemes proposed by Ito et al [83] to extend a multiple assignment scheme and a corresponding Shamir threshold scheme, compromise the security of the system. We shall also show how to securely extend the Shamir threshold scheme.

2.4 Extended Capabilities

So far, we have assumed that the dealer is trustworthy. In fact, distribution of the shares in a secret sharing scheme is based on the assumption that the dealer can transmit the shares *privately.* However, if a secure channel does not exist or there is a possibility that the dealer is dishonest, then there is a need to provide certainty about the distributed

shares. Chor, Goldwasser, Micali and Awerbuch [39] introduced the verifiability problem of secret sharing schemes and proposed an interactive protocol to solve this problem. Later, Feldman [58] proposed a non-interactive solution. Verifiable secret sharing schemes have been the subject of investigation by several authors (see, for example, [9], [130], [66]). Note that all these verifiable secret sharing schemes allow the honest participants to ensure that their shares are correct (related to the secret) and thus in the secret reconstruction phase they will recover the original secret. Stadler [164] proposed a *publicly verifiable* secret sharing scheme in which not only the participants but also an outsider can verify that the shares are correctly distributed.

Meadows [115] introduced the problem of setting up shared secret schemes in the absence of a trusted key distribution centre. One approach of her scheme, however, relies on the unconditional trustworthiness of a black box, which serves in place of the dealer. The other approach works with the help of a public-key encryption system at the secret distribution phase. Later, Ingemarsson and Simmons [82], and Jackson et al [87, 88] proposed a protocol to set up shared secret schemes without the help of a mutually trusted party.

It is worth mentioning that, even if the shares are distributed correctly not every secret sharing scheme can guarantee that the participants can recover the genuine secret. That is, in the secret reconstruction phase, a dishonest participant may cheat the system by contributing a fake share to force the group to recover an incorrect secret, while s/he can obtain the original secret. For example, Tompa and Woll [170] have shown that Shamir's scheme is not secure against certain forms of cheating and they have shown how to cope with this problem. Brickell and Stinson [25] have proposed a secret sharing scheme with the capability of detecting the cheaters. The cheating problem has been the subject of investigation by several others (see, for example, [107], [31], [98], [68]).

Another important capability of a secret sharing scheme is the *homomorphism* property. Benaloh [7] discussed the homomorphism property of secret sharing schemes. Informally, a secret sharing scheme has the homomorphism property if the composition of the shares (corresponding to different secrets) is the share of the composition of secrets. In general, the two compositions rule can be different. It has been shown [7] that Shamir's polynomial based secret sharing scheme is $(+,+)$-homomorphic. That is, summation of the shares is the share of the summation of their corresponding secrets. This property plays an important role in the implementation of society-oriented cryptographic systems, e.g., shared decryption and shared generation of signatures (see the second part of this thesis). The homomorphism property has also been discussed in [61], [60] and [50].

Chapter 3

Threshold Secret Sharing Schemes

This chapter is concerned with threshold secret sharing schemes. In Section 3.1 a brief introduction to threshold secret sharing schemes will be presented. An abstract model for threshold schemes is reviewed in Section 3.2. Then we shall consider some different approaches for threshold secret sharing constructions. A direction of research in the theory of secret sharing schemes will be discussed in Section 3.9.

3.1 Introduction

A particularly interesting class of secret sharing schemes includes threshold schemes with a group of n participants. Their access structure consist of all sets of t or more participants. Such schemes are called t out of n threshold schemes or simply (t, n) schemes. So, the access structure of a (t, n) threshold scheme can be expressed as:

$$\Gamma = \{\mathcal{A} \subseteq \mathcal{P} \mid |\mathcal{A}| \geq t\},$$

where t, so called the *threshold parameter*, is an integer, $t \leq n$. More precisely, in a (t, n) threshold scheme a secret K is divided into n pieces, $s_1, \ldots, s_n$ such that the following conditions are satisfied [147]:

(1) knowledge of any t or more s_i pieces makes K easily computable;

(2) knowledge of any $t-1$ or fewer s_i pieces leaves K completely undetermined (in the sense that all its possible values are equally likely).

Using the entropy concept, the above mentioned conditions can be expressed as;

$$H(K \mid \mathcal{A}) = \begin{cases} 0 & \text{for } |\mathcal{A}| \geq t \\ H(K) & \text{for } |\mathcal{A}| < t \end{cases}$$

Such threshold secret sharing schemes are called perfect. In a non-perfect threshold scheme, although an unauthorised set cannot recover the secret, they may obtain some partial information regarding the secret (see, for example, Blakley and Meadows [14], Ogata et al [128] and Kurosawa et al [99]).

Theorem 3.1 *In a perfect threshold secret sharing scheme, each share must be at least as large as the secret itself.*

Proof. (Sketch) By contradiction; let, in a (t, n) threshold scheme, the share s_i (corresponding to participant P_i) have less private information than does the secret. Also, let $\mathcal{A}$ ($|\mathcal{A}| = t$, $P_i \in \mathcal{A}$) be a minimal access set. Clearly, $\mathcal{B} = \mathcal{A} \backslash \{P_i\}$ is an unauthorised set. Since the system is perfect, the $t-1$ collaborating participants of the set $\mathcal{B}$ must have equal uncertainty about the secret as an outsider. On the other hand, if they know s_i they can recover the secret. However, for all possible values of s_i they can recover a unique secret. That is, their uncertainty about the secret is equal to their uncertainty about s_i; a contradiction. □

3.2 An Abstract Model for Threshold Schemes

An abstract model for threshold schemes has been considered by several authors (see, for example, [23], [108] and [165]). Considering the abstract model for secret sharing schemes (see Section 2.1), in [108] it is shown that *ideal* threshold schemes can be classified as a special type of *orthogonal array*[1]. That is, to construct a (t, n) ideal threshold scheme over the set $\mathbb{Z}_q$, in any t columns of matrix X, every ordered t-tuple must occur precisely once.

We employ the following results and refer the reader to [108] for a more precise treatment.

Lemma 3.2 [108] *There exists an ideal $(1, n)$ threshold scheme over $\mathbb{Z}_q$ for all $n \geq 1$ and $q \geq 1$.*

Example 3.1 *Let $|\mathcal{P}| = \{P_1, P_2, P_3, P_4\}$ and let Γ be a $(1, 4)$ threshold scheme. Then,*

$$X = \begin{array}{c} \\ r_1 \\ r_2 \end{array} \begin{array}{c} \begin{array}{ccccc} D & 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

[1] An orthogonal array $\mathrm{OA}(q, W, t; \lambda)$ on q symbols is a $\lambda q^t \times W$ array such that in any t columns, every ordered t-tuple occurs precisely λ times.

where $W = \{D, 1, 2, 3, 4\}$ *and* $d : \mathcal{P} \to 2^W$ *is given by,*

$$d(P_i) = \{i\}, \quad i = 1, 2, 3, 4$$

determines an ideal $(1, 4)$ *threshold scheme over* $K = \mathbb{Z}_2$.

Lemma 3.3 [108] *There exists an ideal* (n, n) *threshold scheme over* $\mathbb{Z}_q$ *for all* $n \geq 2$ *and* $q \geq 2$.

Example 3.2 *Let* $|\mathcal{P}| = \{P_1, P_2, P_3, P_4\}$ *and let* Γ *be a* $(4, 4)$ *threshold scheme. Then,*

$$X = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \\ r_{11} \\ r_{12} \\ r_{13} \\ r_{14} \\ r_{15} \\ r_{16} \end{array} \begin{array}{c} \begin{array}{ccccc} D & 1 & 2 & 3 & 4 \end{array} \\ \left(\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{array} \right) \end{array}$$

where $W = \{D, 1, 2, 3, 4\}$ *and* $d : \mathcal{P} \to 2^W$ *is given by,*

$$d(P_i) = \{i\}, \quad i = 1, 2, 3, 4$$

determines an ideal $(4, 4)$ *threshold scheme over* $K = \mathbb{Z}_2$.

Lemma 3.4 [108] *There exists an ideal* (t, n) *threshold scheme over* $\mathbb{Z}_q$ *for all* $n \geq t$ *and all prime powers* $q \geq t$.

Example 3.3 *Let* $|\mathcal{P}| = \{P_1, P_2, P_3\}$ *and let* Γ *be a* $(2,3)$ *threshold scheme. Then the following matrix,*

$$X = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \end{array} \begin{array}{c} \begin{array}{cccc} D & 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 1 & 0 & 1 & 2 \\ 1 & 1 & 2 & 0 \\ 1 & 2 & 0 & 1 \\ 2 & 0 & 2 & 1 \\ 2 & 1 & 0 & 2 \\ 2 & 2 & 1 & 0 \end{pmatrix} \end{array}$$

where $W = \{D, 1, 2, 3, 4\}$ *and* $d : \mathcal{P} \to 2^W$ *is given by,*

$$d(P_i) = \{i\}, \quad i = 1, 2, 3,$$

determines an ideal $(2,3)$ *threshold scheme over* $K = \mathbb{Z}_3$.

Threshold schemes were first, independently, introduced by Shamir [147] and Blakley [12]. In spite of the fact that threshold schemes can only handle a small fraction of the secret sharing schemes which we may wish to form, they have been widely investigated in the literature (see, for example, Asmuth and Bloom [4], Karnin et al [93], Mignotte [120], Kothari [96], Blakley and Meadows [14], Meadows [115], De Soete and Vedder [163], Stinson and Vanstone [168], Laih et al [101], Simmons [154, 155, 158] and Blakley et al [13]). Although several approaches have been proposed for constructing threshold secret sharing schemes, the essential notion of all these schemes is the same.

3.3 Polynomial Approach

The Shamir (t, n) threshold scheme is based on polynomial interpolation. Given t points in the two-dimensional plane $(x_1, y_1), \ldots, (x_t, y_t)$ with distinct x_i's, there is one and only one polynomial $f(x)$ of degree at most $t-1$ such that $y_i = f(x_i)$ for all i. The Lagrange interpolation formula is as follows:

$$f(x) = \sum_{i=1}^{t} y_i \prod_{\substack{j=1 \\ j \neq i}}^{t} \frac{x - x_j}{x_i - x_j}. \tag{3.1}$$

Let the secret be an element of a finite field, that is, $K \in GF(p)$, where p is a prime number. Since polynomial interpolation is possible over $GF(p)$, Shamir suggests the following algorithm for constructing a (t, n) threshold scheme.

Set-up Phase:

1. The dealer, $\mathcal{D}$, chooses n distinct and non-zero elements of $\mathbb{Z}_p$, denoted $x_1, \ldots, x_n$ and sends x_i to P_i via a public channel.

2. $\mathcal{D}$ secretly chooses (independently at random) $t-1$ elements of $\mathbb{Z}_p$, denoted $a_1, \ldots, a_{t-1}$ and forms the polynomial
$$f(x) = K + \sum_{i=1}^{t-1} a_i x^i.$$

3. For $1 \leq i \leq n$, the dealer computes s_i, where
$$s_i = f(x_i) \pmod{p}.$$

4. $\mathcal{D}$ gives (in private) share s_i to participant P_i.

Secret Reconstruction Phase:

1. Every set of at least t participants can apply the Lagrange interpolation formula to reconstruct the polynomial and hence to recover the secret.

Note. The participants do not need to reconstruct the polynomial $f(x)$. The secret is the constant term of the polynomial, that is, $K = f(0)$. So, they can recover the secret using:
$$K = \sum_{j=1}^{t} s_{ij} \prod_{\substack{k=1 \\ k \neq j}}^{t} \frac{x_{ik}}{x_{ik} - x_{ij}} \pmod{p}. \tag{3.2}$$

An alternative method of secret reconstruction is to solve linear equations in $\mathbb{Z}_p$. Every set of at least t participants can always form the following system of equations:
$$\begin{aligned}
K + a_1 x_{i1} + a_2 x_{i1}^2 + \cdots + a_{t-1} x_{i1}^{t-1} &= s_{i1} \\
K + a_1 x_{i2} + a_2 x_{i2}^2 + \cdots + a_{t-1} x_{i2}^{t-1} &= s_{i2} \\
&\vdots \qquad\qquad\qquad\qquad \vdots \\
K + a_1 x_{it} + a_2 x_{it}^2 + \cdots + a_{t-1} x_{it}^{t-1} &= s_{it}
\end{aligned}$$

This can be written as:
$$\begin{pmatrix}
1 & x_{i1} & x_{i1}^2 & \cdots & x_{i1}^{t-1} \\
1 & x_{i2} & x_{i2}^2 & \cdots & x_{i2}^{t-1} \\
\vdots & \vdots & \vdots & & \vdots \\
1 & x_{it} & x_{it}^2 & \cdots & x_{it}^{t-1}
\end{pmatrix}
\begin{pmatrix}
K \\ a_1 \\ \vdots \\ a_{t-1}
\end{pmatrix}
=
\begin{pmatrix}
s_{i1} \\ s_{i2} \\ \vdots \\ s_{it}
\end{pmatrix}.$$

The leftmost matrix is a so-called Vandermonde matrix and its determinant is given by the following formula:

$$\prod_{1 \le k < j \le t} (x_{ij} - x_{ik}).$$

Since all x_i's are distinct, no term $(x_{ij} - x_{ik})$ is zero. Thus the determinant of a Vandermonde matrix over a finite field is always non-zero and the above system of equations has a unique solution over $\mathbb{Z}_p$. That is, every set of at least t participants can uniquely reconstruct the polynomial and hence recover the secret.

Note. The size of $GF(p)$ must be large enough such that the selection of distinct and non-zero elements x_i's is possible. That is, the required condition for constructing a Shamir (t, n) threshold scheme is that the prime number p (the size of the field) must be greater than n (the number of participants in the system).

3.3.1 Characteristics of the Shamir Scheme

Shamir's secret sharing scheme has been the subject of investigation by several authors (see, for example, McEliece and Sarwate [114], Kothari [96], Stinson and Vanstone [168], Ito et al [83] and Stinson [165]). It is well-known that the Shamir scheme is perfect. As we shall show in a moment, the proof of perfectness, however, needs more clarification.

In order to prove that Shamir's scheme meets condition (2) of threshold schemes (i.e., the scheme is perfect) Shamir [147] argued that; if $t-1$ of shares are revealed to an opponent, for each candidate value K' in $[0, p)$ he can construct one and only one polynomial $g(x)$ of degree $t-1$ such that $g(0) = K'$ and $g(x_i) = s_i$ for the $t-1$ given arguments. However, by construction, these p possible polynomials are equally likely, and thus there is absolutely nothing the opponent can deduce about the real value of K.

Our observation to this proof is that:

1. Not all p polynomials, in which the opponent can generate for possible candidate values K', are of degree $t-1$;

2. The proof implicitly determines that in Shamir's (t, n) threshold scheme the degree of the associated polynomial is "exactly" $t-1$.

After considering Blakley's secret sharing schemes, the reader can find out why the majority of authors believe that in Shamir's scheme the associated polynomial is of degree "exactly" $t-1$ (which is not the case). In this section we present a complete proof of security and a precise method for constructing the Shamir (t, n) threshold secret sharing scheme.

Proposition 3.5 *In the Shamir (t,n) threshold scheme, for any set of $t-1$ shares there exists a value K' and a polynomial $g(x)$ of degree less than $t-1$ such that $g(0)=K'$ and $g(x_i)=s_i$.*

Proof. According to the Lagrange interpolation formula, the set of $t-1$ points determines one and only one polynomial $g(x)$ of degree at most $t-2$ such that $g(x_i)=s_i$ for the $t-1$ given arguments. Considering $K'=g(0)$ completes the proof. □

Note that, the value K' is unique, since $g(x)$ is unique.

Theorem 3.6 *The Shamir (t,n) threshold secret sharing scheme is perfect.*

Proof. Let us now assume that in the Shamir (t,n) threshold scheme $t-1$ shares are revealed to an opponent. For p possible values $K' \in GF(p)$ the opponent can construct one polynomial of degree less than $t-1$ and $p-1$ polynomials of degree $t-1$ such that they satisfy the shares for the $t-1$ given arguments and the candidate values K'. Now, by construction these p polynomials are equally likely, since the dealer generates the coefficients a_i, $1 \leq i \leq t-1$, independently at random and thus with probability $1/p$ the coefficient a_{t-1} is zero and the constructed polynomial is of degree less than $t-1$. Thus the opponent can deduce no information about the real value of the secret. That is, the scheme is perfect (in the information theoretic sense). □

Corollary 3.7 *Since $s_i=f(x_i) \in GF(p)$, for all i and $K \in GF(p)$, the Shamir secret sharing scheme is ideal.*

Similarly to Simmons [158], we find it useful to use graphical representations to illustrate the Shamir scheme. Graphical representation of polynomials of degree zero, one, two and three are shown in Figure 3.1. In all these graphs, the secret K is the intersection point of the graph and the Y axis. Note that, over finite fields, the graphical representation of these functions consists of a collection of disjoint points, which does not have any resemblance to those shown in Figure 3.1.

Now, we discuss the construction of a Shamir type threshold scheme. Since a (t,n) threshold scheme requires that less than t shares must not be sufficient to determine the secret, several authors suggest the dealer choose a polynomial of degree $t-1$ and then distribute the shares as in the Shamir scheme. Among these authors, Ito, Saito and Nishizeki [83] suggest the following algorithm for constructing the Shamir (t,n) threshold scheme.

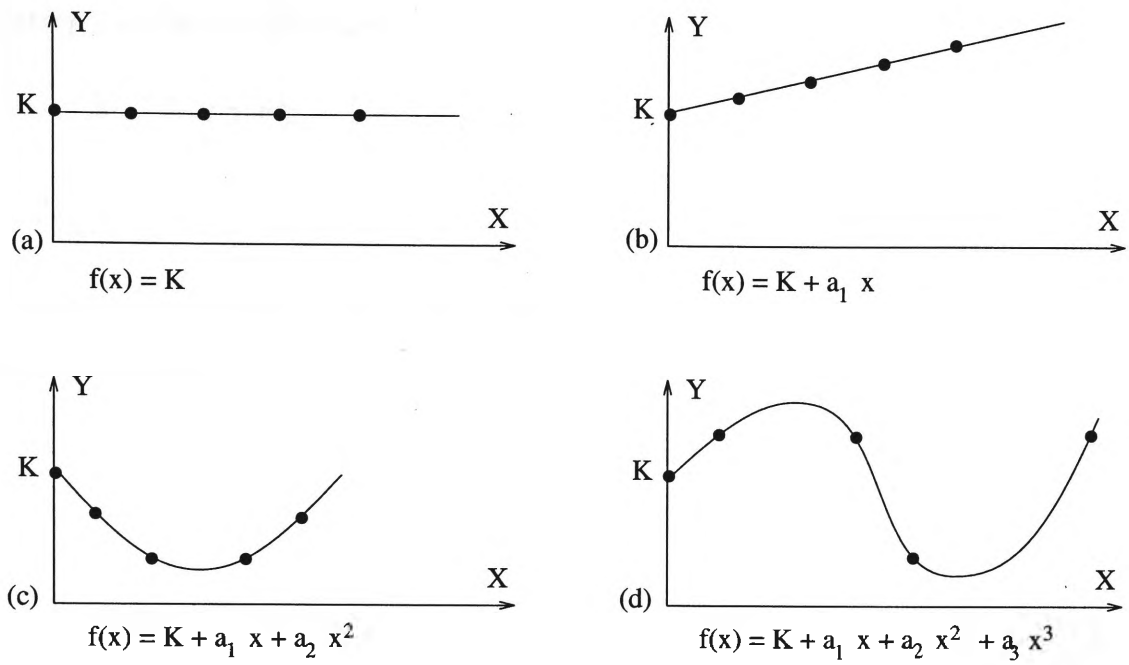

Figure 3.1: Graphical representation of the Shamir scheme

1. Take a prime power p such that $p > n$ and select distinct elements $x_1, \ldots, x_n$, $x_i \in \{GF(p) - \{0\}\}$ at random.
2. Choose $a_1, \ldots, a_{t-2} \in GF(p)$ and $a_{t-1} \in \{GF(p) - \{0\}\}$ randomly, where $t \leq n$
3. Let $f(x) = K + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}$.
4. Let $s_i = f(x_i)$ and assign (x_i, s_i) to P_i for each i, $1 \leq i \leq n$.

That is, in a Shamir (t, n) threshold scheme the associated polynomial is of degree $t-1$. The following theorem shows that such a Shamir (t, n) threshold scheme is not perfect.

Theorem 3.8 *Given a Shamir (t, n) threshold scheme. If the degree of the associated polynomial $f(x)$ is known to be $t-1$ then the scheme is not perfect.*

Proof. Let $P_1, \ldots, P_{t-1}$ be a set of $(t-1)$ collaborating participants that pooling their shares $s_1, \ldots, s_{t-1}$ in order to perform the Lagrange interpolation formula. Certainly, they can construct a unique polynomial $g(x) = K' + b_1 x + \cdots + b_{t-2} x^{t-2}$ of degree at most $t-2$ such that $s_i = g(x_i)$ for all $i = 1, \ldots, t-1$ (see Proposition 3.5). On the other hand, they know $s_i = f(x_i)$ for $i = 1, \ldots, t-1$, where $f(x) = K + a_1 x + \cdots + a_{t-1} x^{t-1}$ is the associated polynomial for the system. So, they have the following system of equations:

$$\begin{aligned} s_1 &= g(x_1) = f(x_1) \\ &\vdots \\ s_{t-1} &= g(x_{t-1}) = f(x_{t-1}) \end{aligned}$$

The system can be transformed to:

$$\begin{aligned}(K-K')+(a_1-b_1)x_1+\ldots+(a_{t-2}-b_{t-2})x_1^{t-2}+a_{t-1}x_1^{t-1} &= 0\\ &\vdots\\ (K-K')+(a_1-b_1)x_{t-1}+\ldots+(a_{t-2}-b_{t-2})x_{t-1}^{t-2}+a_{t-1}x_{t-1}^{t-1} &= 0\end{aligned}$$

Now, we show by contradiction that $K \neq K'$. Suppose that $K = K'$. This implies that the system becomes

$$\begin{aligned}(a_1-b_1)x_1+\ldots+(a_{t-2}-b_{t-2})x_1^{t-2}+a_{t-1}x_1^{t-1} &= 0\\ &\vdots\\ (a_1-b_1)x_{t-1}+\ldots+(a_{t-2}-b_{t-2})x_{t-1}^{t-2}+a_{t-1}x_{t-1}^{t-1} &= 0\end{aligned}$$

As the Vandermonde determinant of the system is different from zero, there is only one solution in which $a_{t-1} = 0$. This contradicts that $f(x)$ is of degree $t-1$ and proves that $K \neq K'$.

Since the $(t-1)$ participants have been successful in finding an integer K' which is not the secret, their uncertainty about the secret is not equal to the uncertainty of an outsider and therefore the scheme is not perfect. □

Note. Here we have analysed the Shamir threshold secret sharing scheme as a method for sharing a *one-time* secret. We showed that the scheme, if constructed carefully, is *unconditionally* perfect. That is, no matter how much computing power is available to an opponent he cannot learn anything about the secret. This is completely different from *long-time* secret sharing schemes in which participants utilise a function of their shares to perform a desirable task without revealing their shares and/or compromising the secret (e.g., society-oriented cryptographic systems). In this case, in general, every unauthorised set of participants can check whether a given value is the secret or not. However, the secret space is large enough such that exhaustive searching (in order to obtain useful information regarding the secret) is infeasible. Such secret sharing schemes are called *computationally* perfect. As a matter of fact, in a computationally perfect scheme the above mentioned construction of the Shamir scheme does not work. If the associated polynomial with a Shamir (t, n) threshold scheme is of degree less than $t-1$ then the system easily can be broken. Thus in computationally perfect systems, in order to construct a Shamir (t, n) threshold scheme, the dealer selects a polynomial of degree "exactly" $t-1$.

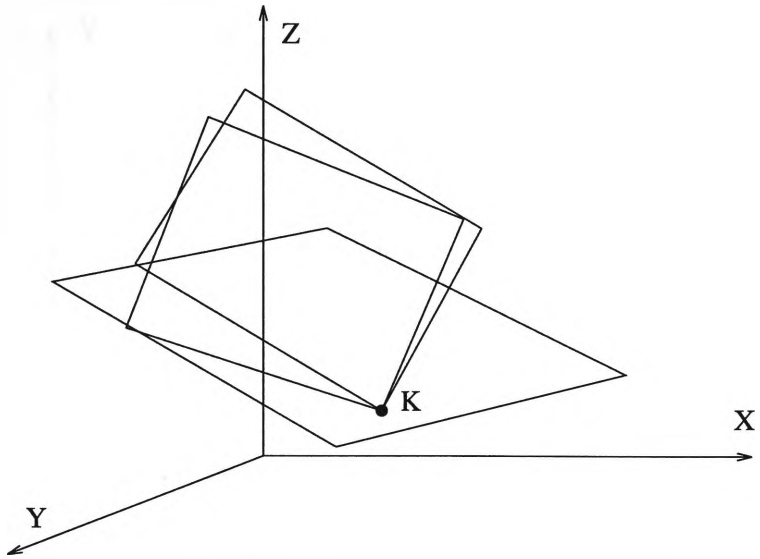

Figure 3.2: Blakley-type $(3, n)$ threshold scheme

3.4 Geometrical Approach

Blakley's construction for a (t, n) threshold scheme is based on *projective* geometry.

Definition 3.1 *[108]. Let $V = V(t+1, p)$ be the vector space of dimension $t+1$ over a finite field $GF(p)$. The set of subspaces of V together with the relation of incidence induced by subspace containment is called projective space $PG(t, p)$. The integer t is called the dimension of $PG(t, p)$.*

In geometric (t, n) threshold schemes, the secret K is a randomly chosen point in $PG(t, p)$. The share of each participant, however, is a subspace of projective dimension $t-1$ (a hyperplane), such that every t of these shares intersects at point K. For example, let $K \in PG(3, p)$, that is, the secret is a point in a three-dimension projective space. Each share is a hyperplane (plane) in $PG(3, p)$, such that intersection of every three of these shares uniquely determines the secret (Figure 3.2).

Consider an example of Blakley's scheme (see Figure 3.2). An outsider can guess the secret K being a point in $PG(3, p)$, while any participant can guess the secret K being a point on the plane that he knows. Furthermore, any set of two collaborating participants can obtain a line (the intersection of their planes) which the secret K lays on. So Blackley's scheme is not perfect.

3.4.1 Perfect Geometric Secret Sharing Schemes

Geometric secret sharing schemes have been the subject of investigation by several authors (see, for example, [157], [108], [158] and [85]). As Simmons [158] pointed out, a

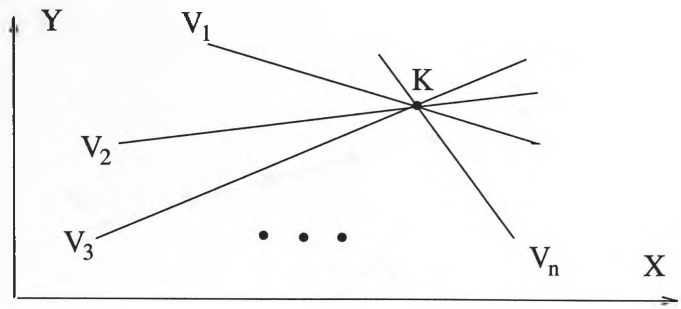

Figure 3.3: Blakley's $(2, n)$ threshold scheme

Blakley-type secret sharing scheme can never be perfect, because as the number of participants in a collusion increases, the uncertainty about the secret must decrease since the secret is in each of the privately held geometric objects and hence in their intersection. As a simple example, in Blakley's construction for a $(2, n)$ threshold scheme, the share of each participant is a line such that every two lines intersect at the secret point K (Figure 3.3). From an outsider point of view, every point in the plane is equally likely to be the point K, hence his uncertainty about K is

$$H(K) = \log p^2 = 2 \log p.$$

However, every participant knows that K is a point on the line he possess as his share from the secret. That is, his uncertainty about the secret is only

$$H(K) = \log p$$

and therefore, the scheme is not perfect. In order to construct perfect schemes, Simmons suggests making all coordinates except one public. Let the secret K be a point on the publicly known line V_K, which is embedded in the projective plane $PG(2, p)$. The uncertainty about the secret K is only $H(K) = \log p$. To construct a perfect $(2, n)$ threshold geometric secret sharing scheme, let $V_s \neq V_K$ be a randomly chosen line in the plane that intersects the line V_K at the secret point K (Figure 3.4). The private piece of information can be taken to be distinct points on V_s, none of which are the point K itself. Any pair of points on V_s determine the line and hence its intersection with V_K; the secret point K. However, knowing any one of the points, s_i, on line V_s leaves K completely undetermined since for each point, K', on line V_K there exists a unique line which could be (with equal probability) the unknown line V_s.

In order to make the scheme ideal, the size of the shares must be equal to the size of the secret. Since the secret has one-dimensional uncertainty, all coordinates of the shares (except one coordinate) can be made public (for this particular example, let the

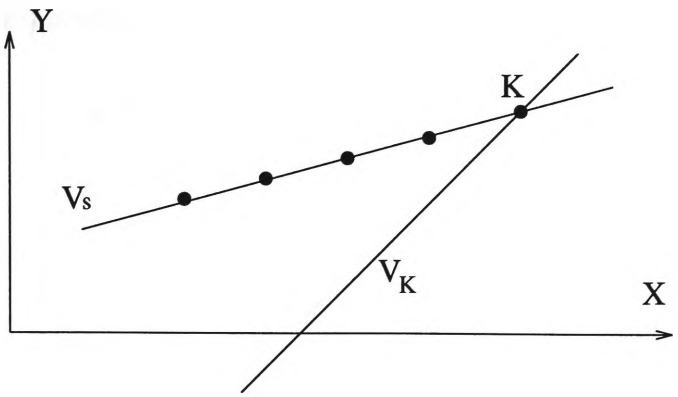

Figure 3.4: Simmons' $(2, n)$ threshold scheme

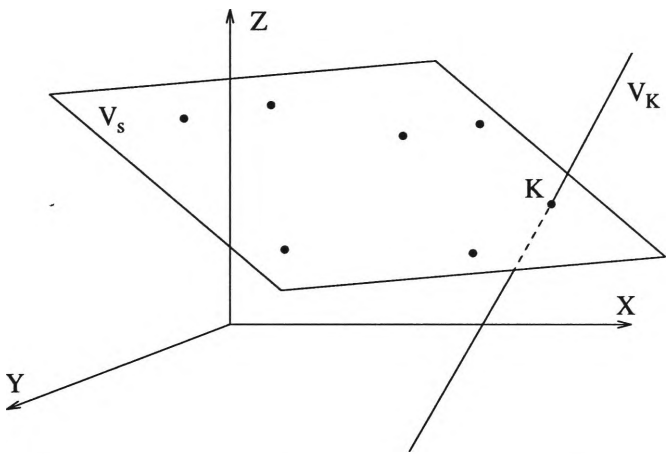

Figure 3.5: Simmons' $(3, n)$ threshold scheme

x coordinate be made public). Now, the reader can compare this scheme (Figure 3.4) with the Shamir construction (assuming that V_K is the Y axis).

Similarly, a perfect geometric $(3, n)$ threshold scheme can be constructed (Figure 3.5). The shares, in this scheme, are points in the plane V_s, which intersects the line V_K at point K. None of these points are the secret K and no two of them are collinear with K.

3.5 Modular Approach

In 1983 Asmuth and Bloom [4] introduced a modular approach to threshold secret sharing schemes. In their scheme, shares are congruence classes of a number associated with the secret. The scheme utilises the well known tool in modern cryptographic systems, the *Chinese Remainder Theorem.*

Theorem 3.9 – Chinese Remainder Theorem (CRT) [94]

Let $m_1, m_2, \ldots, m_n$ be positive integers that are relatively prime in pairs, that is,

$$\gcd(m_i, m_j) = 1 \quad \textit{for any } i \neq j.$$

Let $m = m_1 \cdot m_2 \cdots m_n$ and let $y_1, y_2, \cdots, y_n$ be integers. Then there is exactly one integer y that satisfies the conditions

$$0 \leq y < m \quad \textit{and} \quad y \equiv y_i \pmod{m_i} \ \textit{for} \ \ 1 \leq i \leq n.$$

3.5.1 The Scheme

In order to construct a (t, n) threshold secret sharing scheme for sharing a secret K, $0 \leq K < q$ (q is not necessarily a prime number), the modular approach suggests the following algorithm. Let integers $m_1 < m_2 < \cdots < m_n$ be chosen such that:

1. $\gcd(m_i, m_j) = 1$ for $i \neq j$,
2. $\gcd(q, m_i) = 1$ for all i,
3. $\prod_{i=1}^{t} m_i > q \cdot \prod_{j=1}^{t-1} m_{n-j+1}$.

Hence, the share distribution and the secret reconstruction phases will be as follows.

Set-up Phase:

Let $M = \prod_{i=1}^{t} m_i$. The dealer sends (via a public channel) m_i to participant P_i, $1 \leq i \leq n$ and selects an arbitrary integer r such that $0 \leq y = K + qr < M$. Then, $\mathcal{D}$ calculates the share s_i using:

$$s_i \equiv y \pmod{m_i} \text{ for all } i$$

and gives (in private) the share s_i to participant P_i.

Secret Reconstruction Phase:

Let $P_{i1}, P_{i2}, \ldots, P_{it}$ be the collaborating participants and let $M' = \prod_{j=1}^{t} m_{ij}$. Clearly, $M \leq M'$ and the collaborating participants can uniquely determine the value y, by applying the CRT, and hence the secret K.

It is not difficult to show that less than t participants cannot deduce their uncertainty about the secret K and thus the scheme is perfect. The scheme, however, is not ideal since $q < m_i$, $1 \leq i \leq n$ and therefore every share is selected from a bigger domain than that of the secret itself.

It is worth mentioning that a similar scheme has also been proposed by Mignotte [120].

3.6 Vector Space Approach

Brickell [22] introduced a vector space construction for certain ideal schemes, namely the *multilevel* and the *compartmented* schemes. Let V be a vector space of all d-tuples ($d \geq 2$) over a finite field $GF(p)$ and let e_i denote the ith d-dimensional unit coordinate vector, that is, $e_1 = (1, 0, \ldots, 0)$. Suppose there exists a function

$$\phi : \mathcal{P} \rightarrow (\mathbb{Z}_p)^d$$

which satisfies the property

$$e_1 \in \langle \phi(P_i) : P_i \in \mathcal{A} \rangle \tag{3.3}$$

if and only if $\mathcal{A}$ is an access set. Brickell's construction is as follows:

Set-up Phase:

1. The dealer chooses a vector $a = (K, a_1, \ldots, a_{d-1})$, where the secret K and a_i ($1 \leq i \leq d-1$) are elements of $GF(p)$.

2. For each participant P_i, the dealer selects a d-dimensional vector $v_i = \phi(P_i) \in (\mathbb{Z}_p)^d$. These vectors are public.

3. For $1 \leq i \leq n$, the dealer computes $s_i = a \cdot v_i$, where "$\cdot$" is the inner product modulo p.

4. $\mathcal{D}$ gives (in private) the share s_i to participant P_i ($1 \leq i \leq n$).

Secret Reconstruction Phase:

According to the condition (3.3), for an access set $\mathcal{A}$ the vector e_1 can be expressed as a linear combination of all v_i, that is,

$$e_1 = \sum_{P_i \in \mathcal{A}} w_i \cdot v_i$$

where $w_i \in GF(p)$ can be precomputed for every access set. On the other hand, $K = a \cdot e_1$. Thus,

$$K = a \cdot \sum_{P_i \in \mathcal{A}} w_i \cdot v_i = \sum_{P_i \in \mathcal{A}} w_i \cdot a \cdot v_i.$$

Considering $s_i = a \cdot v_i$, the secret can be recovered using,

$$K = \sum_{P_i \in \mathcal{A}} w_i \cdot s_i$$

Clearly, the scheme is ideal, since shares are elements of the same domain as the secret. To prove the perfectness of the scheme, we refer the reader to Brickell's original paper (see also, Stinson [166, 167]).

3.7 Karnin-Green-Hellman (n,n) Scheme

A desirable characteristic of a (t,n) threshold schemes is that if even $n-t$ pieces are destroyed the secret still can be reconstructed from the remaining pieces. Thus, one may think a $(1,n)$ threshold scheme provides an appropriate scheme; if even $n-1$ pieces are destroyed the secret accessible from the last remaining piece. A main disadvantage of this scheme, however, is that theft of even one piece compromises the secret. Hence, to protect the secret against the threat of theft, the threshold parameter should be large enough. That is, an appropriate scheme could be a (n,n) threshold scheme. But, the advantage of this scheme is also a disadvantage; if even one piece is destroyed, the secret cannot be reconstructed. Considering the fact that, in a group, the majority of participants are honest, a (t,n) threshold scheme with threshold parameter $t = \lfloor \frac{n}{2} \rfloor + 1$ provides a robust secret sharing scheme.

However, some applications require a tradeoff between security and convenience of use. For example, (n,n) threshold schemes are frequently used in the implementation of general secret sharing schemes (see Chapter 4). For this particular case of threshold schemes, the Karnin-Greene-Hellman [93] algorithm produces an efficient threshold scheme. Their scheme works as follows:

Set-up Phase:

1. The dealer selects (independently at random) $n-1$ elements of $\mathbb{Z}_q$, denoted by $s_1, \ldots, s_{n-1}$.
2. $\mathcal{D}$ computes
$$s_n = K - \sum_{i=1}^{n-1} s_i \pmod{q}.$$
3. For $1 \leq i \leq n$, the dealer gives (in private) the share s_i to participant P_i.

Secret Reconstruction Phase:

1. The secret can be recovered by adding all shares (computation is done over $\mathbb{Z}_q$).

This scheme is computationally more simple than the previously mentioned threshold schemes. Moreover, q is not necessarily a prime and can be even smaller than, n, the number of participants in the system. It is not difficult to show that the scheme is perfect, that is, a set of less than n participants obtains no information about the secret (for more detail and proof of security see [93] or [167]).

Example 3.4 *Let* $|\mathcal{P}| = \{P_1, P_2, P_3, P_4\}$ *and let* Γ *be a* $(4,4)$ *threshold scheme over* $\mathbb{Z}_2$. *The dealer,* $\mathcal{D}$, *selects (randomly) three elements of* $\mathbb{Z}_2$ *denoted by* s_1, s_2 *and* s_3 *and computes*

$$s_4 = K - (s_1 + s_2 + s_3) \pmod 2.$$

Then, $\mathcal{D}$ *distributes the shares among the participants.*

The secret can easily be recovered by adding all shares as:

$$K = s_1 + s_2 + s_3 + s_4 \pmod 2.$$

One can check that this scheme is equivalent to the scheme of Example 3.2.

Remark 1 *A final remark of this chapter is that, although the essential notion of all threshold schemes is the same, they do not necessarily produce equivalent schemes. For instance, there is no equivalent Shamir scheme corresponding to any of threshold schemes presented in Examples 3.1, 3.2 and 3.3. In fact, the common problem with the majority of threshold schemes is that their initial conditions for constructing ideal schemes are tighter than the conditions given in Lemmas 3.2, 3.3 and 3.4.*

3.8 Anonymous Secret Sharing Schemes

We have only investigated a few approaches for constructing threshold secret sharing schemes. The selection of these approaches is due to their relevance to the rest of this thesis. A common characteristics of all these schemes, however, is that in the secret reconstruction phase the shares of cooperative participants must be accompanied by some form of authentication. In other words, if participant P_i presents share s_i, this is useless for determining the secret unless the identity of shareholder is also known. For instance, in a Shamir scheme it is necessary to know that share s_i is being presented by P_i whose identity is x_i.

However, it may be desirable to keep the membership of a group secret. This implies keeping secret the identifications of participants along with their shares. Hence, construction of an anonymous scheme implies considerably increase the size of the share assigned

to each participant. Stinson and Vanstone [168] proposed a combinatorial approach to anonymous threshold secret sharing schemes (see also, [145] and [38]). For more details and study of anonymous secret sharing schemes we refer the reader to Martin [108].

3.9 New Directions in Secret Sharing Research

As a final section of this chapter we briefly consider the direction of research in the theory of secret sharing schemes. A survey regarding secret sharing schemes shows that some topics have received more attention than others. For example, information rate (since an important issue in the implementation of secret sharing schemes is that of how to reduce the size of shares) was the subject of investigation for several authors. Among several topics of research in the theory of secret sharing schemes we focus on a particular subject, so called *long-lived secrets* (e.g., cryptographic master keys), which is more relevant to the rest of this thesis.

3.9.1 Long-lived Secrets

The goal of implementing a secret sharing scheme is to protect sensitive information by distributing it among different locations. The idea behind this technique is that an adversary cannot obtain the secret as long as specific numbers of shares have not been compromised. Compromising the secret information, however, is a matter of time. That is, in a computationally secure secret sharing scheme, an adversary or a group of adversaries may have enough time to compromise sufficient numbers of shares and hence to obtain the secret. Or in an unconditionally secure secret sharing scheme, a sufficient number of shares may be stolen or destroyed[2] in a long enough period of time.

This indicates that conventional secret sharing schemes might be insufficient to provide the secrecy of long-lived sensitive information, such as a contract between two countries, proprietary trade-secret information, and so on. The implementation of secret sharing schemes that provide the secrecy of long-lived secrets and consideration of relevant problems are subjects of research in *proactive secret sharing schemes* [79],

[2]As mentioned in [93], even a safe deposit box, which is used to protect a punch card or similar data storage medium that contains a share of the secret, is vulnerable (e.g., to the "silverfish threat," named for an insect which eats punch cards).

3.9.2 Proactive Secret Sharing Schemes

Herzberg, Jarecki, Krawczyk and Yung [79] considered the problems regarding the security and integrity of some *long-lived* and sensitive secrets, such as cryptographic master keys (e.g., certification keys), data files (e.g., medical records), proprietary trade-secret information (e.g., Coca-Cola's formula), etc. They have argued that conventional secret sharing schemes may not be sufficient for the protection of such information. For example, in a (t, n) threshold secret sharing scheme, an adversary needs to compromise at least t shares in order to learn the secret. Since the adversary has the entire life-time of the secret to mount his attacks, he may be able to apply his attacks gradually over a long period of time. Therefore, for long-lived information, the protection provided by conventional secret sharing schemes may not be sufficient.

In order to overcome this problem, they suggest dividing the life-time of the secret into periods of time (e.g., a day, one week, etc.) such that the adversary cannot compromise t shares in one period of time. At the beginning of each time period, however, all shares are renewed (without changing the secret) and all old shares erased. The secret can be reconstructed only from a set of authorised participants' shares associated with a single period of time. Since the adversary cannot compromise enough shares in one period of time, the system provides the required protection.

The Basic Idea

Although the proactive secret sharing scheme [79] was proposed in 1995, the basic idea was given in Shamir's original paper, where it says (for consistency with notations used in this thesis, parameters are renamed)

> "It is easy to change the s_i – all we need is a new polynomial $f(x)$ with the same free term. A frequent change of this type can greatly enhance security since the pieces exposed by security breaches cannot be accumulated unless all of them are values of the same edition of the $f(x)$ polynomial."

In the proactive secret sharing scheme[79], however, it is assumed that the dealer exists only in the initialisation phase of the system. So in every period of time, the shareholders apply some verifiable secret sharing schemes (e.g., [58] or [130]) in order to distribute the value zero among other participants. That is, each participant selects a polynomial of degree at most $t-1$ with a constant term of zero and distributes the shares among other shareholders (similar to the Shamir scheme). According to the homomorphism property of the Shamir scheme, if the participants add their original shares to this new

share, then they will have a share relating to the summation of the original secret K and the new secret. However, the new secret is zero, and therefore the modified shares still are the shares of the original secret. Roughly speaking, their scheme works as follows. In the initialisation phase a Shamir (t, n) scheme is implemented to share the secret $K \in GF(p)$ among the set of participants $\mathcal{P} = \{P_1, \ldots, P_n\}$. At the beginning of time period, j $(j = 1, 2, \cdots)$, each participant P_i $(1 \leq i \leq n)$ applies the following share renewal protocol.

1. P_i selects $t-1$ random numbers $b_{i_1}, \ldots, b_{i_{t-1}}$
2. P_i secretly chooses (independently at random) $t-1$ elements of $\mathbb{Z}_p$, denoted $b_{i_1}, \ldots, b_{i_{t-1}}$ and forms the polynomial

$$f_{i_j}(x) = 0 + \sum_{\ell=1}^{t-1} b_{i_\ell} x^\ell.$$

3. For $1 \leq m \leq n$, the participant P_i computes s_{i_m}, where

$$s_{i_m} = f_{i_j}(x_i) \pmod{p}.$$

4. P_i gives (in private) share s_{i_m} to participant P_m.

Now, the share of each participant P_m $(1 \leq m \leq n)$ in the time period j is given by:

$$s_m^{j-1} + s_{1_m} + s_{2_m} + \ldots + s_{n_m}$$

where s_m^{j-1} denotes the share of participant P_m at time period $j-1$ and s_{i_m} $(1 \leq i \leq n)$ denotes the shares that participant P_m has received at each share renewal protocol (note that, s_{m_m} is generated by P_m himself).

Note. Here, we have illustrated the idea behind proactive secret sharing schemes. To study the exact and secure protocol, we refer the readers to [79]. It is worth noting that Desmedt and Jajodia [52] have proposed a proactive secret sharing scheme where in every period the access structure can also be changed.

Chapter 4

Generalised Secret Sharing Schemes

This chapter considers secret sharing schemes with arbitrary monotone access structures. In Section 4.2 some methods for constructing general schemes will be studied. Computationally secure secret sharing schemes will be discussed in Section 4.3. Section 4.4 is devoted to the consideration of extended capabilities required in secret sharing schemes. In this section the extension of secret sharing schemes and the cheating problem in the secret reconstruction phase will be discussed. Finally, in Sections 4.5 and 4.6 we present two solutions for constructing secret sharing schemes in multilevel and compartmented access structures.

4.1 Introduction

Although a large fraction of research in the area of secret sharing schemes is devoted to threshold schemes, they can handle only a small fraction of the secret sharing functions which one may wish to form. In fact, threshold schemes were originally formulated for democratic groups in which every share has equal weight. Some applications, however, require different weighting of shares. For example, a bank vault can be opened by either two vice-presidents, P_1 and P_2, or by three senior tellers, P_3, P_4 and P_5, that is,

$$\Gamma^- = P_1P_2 + P_3P_4P_5.$$

This scheme requires that the shares assigned to more privileged participants carry more weight than the shares assigned to less privileged participants.

Ito, Saito and Nishizeki [83] have first generalised the Shamir construction and designed a secret sharing scheme which realizes any arbitrary access structure. In their cumulative scheme, the so called *multiple assignment scheme* (see also [84]), a single share, may be assigned to several participants.

Later, Benaloh and Leichter [8] utilised Ito et al's idea and proposed a more efficient general scheme which assigns to each participant, in general, fewer shares than Ito et al's scheme. They have proved, however, there exist access structures which cannot be realised by giving only one share to each participant.

The first extensive discussion on the generalisation of secret sharing schemes was given by Simmons [154], who introduced the following eight classes of real-word applications in which simple (t, n) threshold schemes are not able to handle their requirements.

1. Compartmented t_i-out-of-ℓ_i shared secret schemes in which the secret reconstruction requires t_i participants of compartment ℓ_i cooperate together.

2. Multilevel t_i-out-of-ℓ_i shared secret schemes in which the reconstruction of the secret requires t_i members of the level ℓ_i or higher levels cooperate.

3. Extrinsic shared secret schemes in which the value of a share to the reconstruction of the secret depends on its functional relationship to other shares, and not on its information content.

4. Prepositioned shared secret schemes in which the shareholders are unable to recover the secret until such time as the scheme is activated by communicating additional information.

5. Prepositioned shared secret schemes in which the same shares can be used to recover different secrets depending on the choice of the activating information.

6. Proof of correctness of the reconstructed secret to a confidence of $\approx 1 - \varepsilon$, where ε is the probability of guessing the correct secret.

7. Tolerance of erroneous inputs of some number, ℓ, of shares. That is, the correct secret will be calculated even though ℓ of the inputs are in error.

8. A cryptographically secure mnemonic technique to enable participants to recover a private piece of information (share) that they cannot remember by using a piece that they can.

4.2 Some General Schemes

4.2.1 The Multiple Assignment Scheme

Ito, Saito and Nishizeki [83] introduced the multiple assignment secret sharing scheme. Their scheme utilises the Shamir construction to design a secret sharing scheme which realizes any arbitrary access structure. The multiple assignment scheme works as follows. Let $\Gamma \subset 2^{\mathcal{P}}$ be an access structure. The dealer, $\mathcal{D}$, constructs Γ^{c+} and utilises a Shamir (t,t) threshold scheme to generate t shares (where $t = |\Gamma^{c+}|$). Then, for any unauthorised set $\mathcal{B}$, $\mathcal{B} \in \Gamma^{c+}$, it assigns a distinct share to all participants in $\bar{\mathcal{B}}$ ($\bar{\mathcal{B}} = \mathcal{P} \setminus \mathcal{B}$).

Example 4.1 *Let $\mathcal{P} = \{P_1, P_2, P_3, P_4\}$ be the set of participants and let*

$$\Gamma^{-} = P_1P_2 + P_2P_3 + P_3P_4 \tag{4.1}$$

be the access structure. In order to share the secret $K \in GF(p)$, the dealer gets

$$\Gamma^{c+} = \{\{P_1, P_3\}, \{P_1, P_4\}, \{P_2, P_4\}\}.$$

Since $|\Gamma^{c+}| = 3$, it designs a Shamir $(3,3)$ threshold scheme and generates three shares, s_1, s_2, s_3. Then it assigns share s_1 to P_2 and P_4 (that do not belong to unauthorised set $\{P_1, P_3\}$). It also assigns share s_2 to P_2 and P_3 (that do not belong to unauthorised set $\{P_1, P_4\}$). Similarly, it assigns share s_3 to P_1 and P_3.

They have proved that, for every access set $\mathcal{A} \in \Gamma$, the number of distinct shares given to its participants is equal to t, while for every unauthorised set $\mathcal{B} \notin \Gamma$, the number of distinct shares given to its members is less than t. That is, the scheme satisfies the requirement of secret sharing schemes, since the knowledge of at least t shares enables an authorised set to recover the secret. The knowledge of less than t shares, however, does not allow an unauthorised set to recover the secret. Figure 4.1 illustrates the multiple assignment scheme corresponding to the access structure $\Gamma^{-} = P_1P_2 + P_2P_3 + P_3P_4$.

4.2.2 The Logical Approach

Definition 4.1 [108] *A positive logical expression is said to be in Conjunctive Normal Form (CNF) if*

$$\Gamma = \mathcal{A}_1\mathcal{A}_2 \cdots \mathcal{A}_t,$$

where each of the $\mathcal{A}_i$ are elementary disjoints.

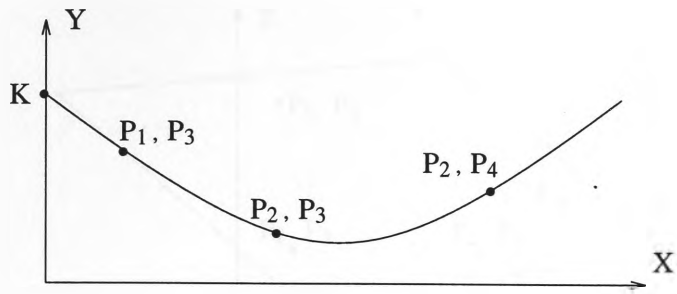

Figure 4.1: Ito et al's scheme for $\Gamma^- = P_1P_2 + P_2P_3 + P_3P_4$

Γ^- is in *minimal* CNF [108] if Γ^- is in CNF and $\mathcal{A}_i \not\subseteq \mathcal{A}_j$ for any $i \neq j$.

Benaloh and Leichter [8] pointed out that the method described by Ito, Saito and Nishizeki [83] corresponds precisely to the case of a minimal CNF-formula in which conjunctions are formed by use of the Shamir (n, n) threshold schemes. They observed that every access structure can be translated into a *monotone formula* which yields, in many cases, a much smaller formula than the CNF-formula. The monotone formula is a logical expression written as a sum of products of the P_i, where each P_i takes the value *true* or *false*. For example, the access structure Γ^-, given in equation 4.1, can be written as

$$((P_1 \bigwedge P_2) \bigvee (P_2 \bigwedge P_3) \bigvee (P_3 \bigwedge P_4)).$$

Since their method combines (if it is possible) sets in the access structure in order to form a (t_i, n_i) threshold scheme, their scheme reduces the number of shares assigned to each participant. There are, however, many cases in which their method is still unable to be applied efficiently.

4.2.3 Cumulative Schemes

Simmons, Jackson and Martin [160] have introduced a constructive algorithm which operates on the logical description of access structures to produce a logical expression that uniquely determines one of the desired geometrical configurations. Their algorithm, in fact, utilises the monotone formulas corresponding to access structures to obtain a *cumulative* scheme. A cumulative scheme for the access structure Γ is a map $\alpha : \mathcal{P} \to 2^S$ (where S is some set), such that for any $\mathcal{A} \subseteq \mathcal{P}$,

$$\bigcup_{P_i \in \mathcal{A}} P_i^\alpha = S \quad \text{if and only if} \quad \mathcal{A} \in \Gamma.$$

That is, in cumulative schemes the secret K can be reconstructed if all shares are known. The scheme can be written as a $|\mathcal{P}| \times |S|$ array $M = [m_{ij}]$, where row i of the matrix M

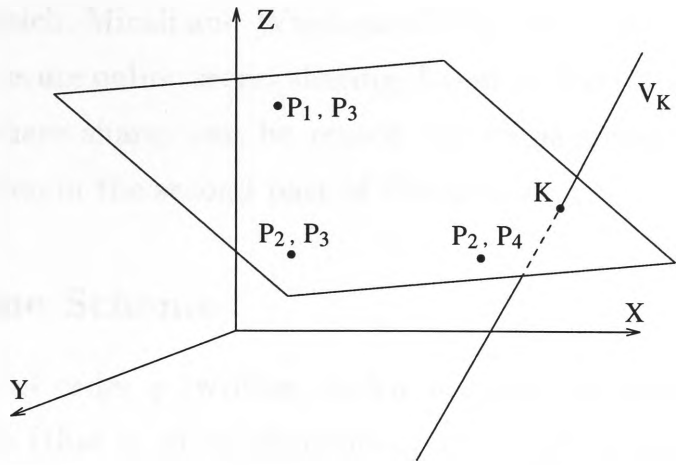

Figure 4.2: Simmons et al's configuration for $\Gamma^- = P_1P_2 + P_2P_3 + P_3P_4$

is indexed by $P_i \in \mathcal{P}$ and column j of the matrix M is indexed by an element $s_j \in S$, such that $m_{ij} = 1$ if and only if P_i is given s_j otherwise $m_{ij} = 0$.

The cumulative scheme proposed in [160] works as follows. Let $\Gamma^- = \mathcal{A}_1 + \cdots + \mathcal{A}_\ell$ be a monotone formula in its minimal form. The *dual access structure* $\Gamma^* = \{\mathcal{B}_1, \ldots, \mathcal{B}_t\}$ is the monotone access structure obtained from Γ^- by interchanging $(+)$ and juxtaposition in the boolean expression for Γ^-. For example, if $\Gamma^- = P_1P_2 + P_2P_3 + P_3P_4$ then $\Gamma^* = (P_1 + P_2)(P_2 + P_3)(P_3 + P_4) = P_1P_3 + P_2P_3 + P_2P_4$. Then the map $\alpha : \mathcal{P} \to 2^S$, which is defined as

$$P_i^\alpha = \{s_j \mid P_i \text{ appears in } \mathcal{B}_j\},$$

gives a cumulative scheme for Γ ($S = \{s_1, \ldots, s_t\}$ is the set of shares). Figure 4.2 illustrates a perfect geometrical configuration for the access structure $\Gamma^- = P_1P_2 + P_2P_3 + P_3P_4$.

Cumulative schemes are also studied by Jackson and Martin [85], Charnes and Pieprzyk [33] and Ghodosi et al [73].

4.3 Computationally Secure Secret Sharing Schemes

In the literature of secret sharing schemes, several computationally secure schemes have also been proposed. For example, Cachin [29] proposed *online* secret sharing schemes. The scheme provides the capability of sharing multiple secrets and allows adding participants dynamically, without having to redistribute new shares. These capabilities are realized by storing additional authentic information at a publicly accessible location.

Pinch [134] pointed out that Cachin's scheme does not allow shares to be reused after the secret has been reconstructed (without a further distributed computation protocol

such as that of Goldreich, Micali and Wigderson [75]). He proposed a modified protocol for computationally secure online secret sharing, based on the intractability of the Diffie-Hellman problem, where shares can be reused (an explanation of terms used in this subsection will be given in the second part of this thesis).

4.3.1 An Online Scheme

G_1 is a cyclic group of order q (written multiplicatively) in which the Diffie-Hellman problem is intractable (that is, given elements g, g^x and g^y in G_1, it is computationally infeasible to obtain g^{xy}) and $f : G_1 \longrightarrow G_2$ is a one-way function. The group operations in G_1 and G_2 respectively are multiplication and addition modulo a large prime p. The set $\mathcal{P}$ of participants is denoted by $P_1, \ldots, P_n$. Certain subsets $\mathcal{A} \in 2^{\mathcal{P}}$ are authorised to recover the secret K. The family of authorised sets of participants is denoted by Γ

Pinch's protocol works as follows : The dealer $\mathcal{D}$, who knows the secret K, randomly chooses shares s_i (integers prime to q) for each participant $P_i \in \mathcal{P}$ and transmits s_i over a secure channel to P_i. For each minimal authorised set $\mathcal{A} \in \Gamma^-$, $|\mathcal{A}| = t$, the dealer randomly chooses $g_{\mathcal{A}}$ to be a generator of G_1 and computes

$$T_{\mathcal{A}} = K - f\left(g_{\mathcal{A}}^{(\prod_{P_i \in \mathcal{A}} s_i)}\right)$$

and posts the pair $(g_{\mathcal{A}}, T_{\mathcal{A}})$ on the notice board. To recover the secret K, a minimal authorised set $\mathcal{A} = \{P_1, \ldots, P_t\}$ of participants comes together and performs the following steps.

1. Member P_1 reads $g_{\mathcal{A}}$ from the notice board, forms $g_{\mathcal{A}}^{s_1}$ and passes the result to P_2.

2. Each subsequent member P_i, for $1 < i < t$, receives $g_{\mathcal{A}}^{s_1 \cdots s_{i-1}}$ and raises this value to the power s_i to form $g_{\mathcal{A}}^{s_1 \cdots s_i}$ which is passed to P_{i+1}.

3. The final participant P_t receives $g_{\mathcal{A}}^{s_1 \cdots s_{t-1}}$ and raises this value to the power s_t to form
$$V_{\mathcal{A}} = g_{\mathcal{A}}^{s_1 \cdots s_t} = g_{\mathcal{A}}^{\prod_{P_i \in \mathcal{A}} s_i}$$

4. On behalf of the access set $\mathcal{A}$, member P_t reads $T_{\mathcal{A}}$ from the notice board and reconstructs K as $K = T_{\mathcal{A}} + f(V_{\mathcal{A}})$.

If there are multiple secrets K_i to share, then it is possible to use the same one-way function f, provided that each entry on the notice board has a fresh value of $g_{\mathcal{A}}$ attached.

Pinch also has a variant proposal which, according to him, avoids the necessity for the first participant P_1 to reveal $g_A^{s_1}$ at step 1. P_1 takes r modulo q at random and forms $g_A^{rs_1}$ and passes the result to P_2, and so on. At the end of the protocol, P_t returns the computed value $g_A^{rs_1 \cdots s_t}$ to P_1 which computes

$$V_A = (g_A^{rs_1 \cdots s_t})^{r^{-1}} \pmod{p}$$

where r^{-1} is the inverse of r, that is $r \times r^{-1} = 1 \bmod q$ (the other parts of the protocol are the same as the original protocol).

4.4 Extended Capabilities for Secret Sharing Schemes

There are several areas in which applications of secret sharing schemes require extended capabilities. In this section, we will consider some of these required capabilities. In particular, we discuss the extension of a secret sharing scheme and cheating detection in the secret reconstruction phase.

4.4.1 Extension of a Secret Sharing Scheme

As mentioned in Section 2.3, a desirable capability of every secret sharing scheme is that the scheme must be able to cope with situations when new members join the group. Indeed it is not very practical to modify the shares each time a new member joins group. This is a common problem with long-lived and sensitive information (e.g., cryptographic master keys) in which the secret must be kept secure even if some shareholders added into the system.

In [83], the authors claimed that their scheme is flexible for the case in which a new member joins the group of shareholders. They considered the following problem [83, Problem 3].

> "Can a scheme realizing an access structure Γ_1 be extended so that a new scheme realizes an access structure Γ_2?"

The question was answered affirmatively provided the new access structure Γ_2 is an extension of Γ_1, that is, $\Gamma_1 \subset \Gamma_2$ and $\Gamma_1^c \subset \Gamma_2^c$.

In Section 2.3 we have presented an example to illustrate the extension of a given access structure. It is not difficult to show, using the abstract model of Section 2.1, that it is always possible to extend any given access structure. However, this is not the case with every secret sharing scheme.

Here we show that the extension of the multiple assignment secret sharing scheme given in [83] does not work.

Theorem 4.1 *The extension of a multiple assignment scheme, proposed in [83], is not secure. That is, in the extended scheme, the secret can be reconstructed by an unauthorised set of participants.*

Proof. Let $\mathcal{P}_1 = \{P_1, \ldots, P_n\}$ and $\Gamma_1 \subset 2^{\mathcal{P}_1}$ be an access structure. Let $\Gamma_1^- = \{\mathcal{A}_1, \ldots, \mathcal{A}_\ell\}$ and $\Gamma_1^{c+} = \{\mathcal{B}_1, \ldots, \mathcal{B}_t\}$. Assume a multiple assignment scheme realizes Γ_1 and we want to extend the set of shareholders to a set $\mathcal{P}_2$, where $\mathcal{P}_2 = \{P_1, \ldots, P_n, P_{n+1}, \ldots, P_m\}$, that is, $\mathcal{P}_1 \subset \mathcal{P}_2$. Let the access structure $\Gamma_2 \subset 2^{\mathcal{P}_2}$ be as follows:

$$\Gamma_2^- = \{\mathcal{A}_1, \ldots, \mathcal{A}_\ell, \{P_i, P_j\}, \text{for } i < j,\ i = 1, \ldots, m-1,\ j = n+1, \ldots, m\}. \qquad (4.2)$$

Clearly, $\Gamma_1 \subset \Gamma_2$. On the other hand, since all subsets consisting of two participants, in which at least one of them belongs to the set of new shareholders, are access sets, we have,

$$\Gamma_2^{c+} = \{\mathcal{B}_1 \ldots, \mathcal{B}_t, \{P_{n+1}\}, \ldots, \{P_m\}\}.$$

That is, $\Gamma_1^{c+} \subset \Gamma_2^{c+}$ and therefore Γ_2 is an extension of Γ_1.

Assume that the multiple assignment scheme, which realizes the access structure Γ_1 on a set $\mathcal{P}_1$, applies the set $S_1 = \{s_1, \ldots, s_t\}$ of shares. That is, s_1 is assigned to participants in set $\mathcal{P}_1 \setminus \mathcal{B}_1$, and in general s_i is assigned to participants in set $\mathcal{P}_1 \setminus \mathcal{B}_i$. In the new scheme, however, the share s_1 will be assigned to participants in set $\mathcal{P}_2 \setminus \mathcal{B}_1$, and in general s_i will be assigned to shareholders in the set $\mathcal{P}_2 \setminus \mathcal{B}_i$. So the set $\mathcal{P}_2 \setminus \mathcal{P}_1 = \{P_{n+1}, \ldots, P_m\}$ will get the set of all shares s_i $(i = 1, \ldots, t)$ from the basic scheme. Since knowing all shares of a secret sharing scheme is sufficient to recreate the secret, every new shareholder can recreate the secret, although none of them individually are supposed to be able to recover the secret. □

Example 4.2 *Let* $\mathcal{P}_1 = \{P_1, P_2, P_3\}$ *and let* $\Gamma_1^- = \{\{P_1, P_2\}, \{P_2, P_3\}\}$. *Since* $\Gamma_1^{c+} = \{\{P_2\}, \{P_1, P_3\}\}$, *the dealer generates a Shamir* $(2, 2)$ *threshold scheme and assigns* s_1 *to set* $\mathcal{P}_1 \setminus \{P_2\}$, *that is, to participants* P_1 *and* P_3. *Similarly, it assigns the share* s_2 *to set* $\mathcal{P}_1 \setminus \{P_1, P_3\}$, *that is, to participant* P_2.

Let $\mathcal{P}_2 = \{P_1, P_2, P_3, P_4, P_5\}$ *and also let* $\Gamma_2^- = \{\{P_1, P_2\}, \{P_2, P_3\}, \{P_1, P_4\}, \{P_1, P_5\}, \{P_2, P_4\}, \{P_2, P_5\}, \{P_3, P_4\}, \{P_3, P_5\}, \{P_4, P_5\}\}$. *Hence,* $\Gamma_2^{c+} = \{\{P_2\}, \{P_1, P_3\}, \{P_4\}, \{P_5\}\}$ *and the dealer generates a Shamir* $(4, 4)$ *threshold scheme to generate four shares* s_1, s_2, s_3, s_4. *However, this set of shares contains the set of shares* s_1, s_2 *which have been*

generated in the basic scheme. In this extended scheme, however, share s_1 will be assigned to set $\mathcal{P}_2 \setminus \{P_2\} = \{P_1, P_3, P_4, P_5\}$, share s_2 will be assigned to set $\mathcal{P}_2 \setminus \{P_1, P_3\} = \{P_2, P_4, P_5\}$, share s_3 will be assigned to set $\mathcal{P}_2 \setminus \{P_4\} = \{P_1, P_2, P_3, P_5\}$ and finally share s_4 will be assigned to set $\mathcal{P}_2 \setminus \{P_5\} = \{P_1, P_2, P_3, P_4\}$. Note that P_1 knows the set of shares $\{s_1, s_3, s_4\}$ and P_2 possesses the set of shares $\{s_2, s_3, s_4\}$. As $\{P_1, P_2\} \in \Gamma_1$, shares s_1 and s_2 are sufficient to recover the secret. Since P_4 holds the set of shares $\{s_1, s_2, s_4\}$ and P_5 possesses the set of shares $\{s_1, s_2, s_3\}$, both P_4 and P_5 can individually recover the secret (they know both shares s_1 and s_2).

Ito, Saito and Nishizeki [83] applied Shamir threshold scheme to construct their extended multiple assignment secret sharing scheme. Their method works as follows. Let $\mathcal{P}_1 = \{P_1, \ldots, P_n\}$ be a set of participants and let a Shamir (t, n) threshold scheme be designed for the set $\mathcal{P}_1$. Assume that we want to design a Shamir (ℓ, m) threshold scheme for a set $\mathcal{P}_2 = \{P_1, \ldots, P_n, P_{n+1}, \ldots, P_m\}$, such that the set of old shares is still acceptable in the new scheme, that is, the new scheme is an extension of the old scheme. In [83] the authors have claimed that; if polynomials $f_1(x)$ and $f_2(x)$ which are associated with the Shamir (t, n) and (ℓ, m) threshold schemes satisfy the condition $\ell \geq t + 2$, then the extension is possible by generating a polynomial $f_2(x)$ (of degree at most $k-1$) such that the t shares generated by polynomial $f_1(x)$ still can be generated by polynomial $f_2(x)$. Here we show how unauthorised collection of $(\ell - 1)$ participants can recover the secret in this extended Shamir (ℓ, m) threshold scheme.

Theorem 4.2 *The extension of a Shamir threshold scheme, proposed in [83], is not secure. That is, any subset of $(\ell - 1)$ participants can also recover the secret.*

Proof. Let a Shamir (t, n) threshold scheme be constructed on a set $\mathcal{P}_1 = \{P_1, \ldots, P_n\}$ of n participants and let the associated polynomial be $f_1(x) = K + a_1 x + \cdots + a_{t-1} x^{t-1}$, that is, $f_1(x)$ is a polynomial of degree at most $t - 1$. Also, let a Shamir (ℓ, m) threshold scheme which is constructed on a set $\mathcal{P}_2 = \{P_1, \ldots, P_n, P_{n+1}, \ldots, P_m\}$ be an extension of the (t, n) scheme. That is, all shares of the old scheme are also acceptable shares in the new scheme. In the extended scheme, however, the associated polynomial is of degree at most $\ell - 1$. We assume $\ell \geq t + 2$, which satisfies the condition given in [83]. Thus, we have, $f_2(x) = K + b_1 x + \cdots + b_{\ell-1} x^{\ell-1}$. Although, without knowing that the (ℓ, m) threshold scheme is an extension of a (t, n) Shamir scheme, less than ℓ participants obtain absolutely nothing about the secret, here we show the knowledge of this fact enables $\ell - 1$ collaborating participants of the extended scheme exactly to determine the secret.

Let $\mathcal{B} \subset \mathcal{P}_2$ ($|\mathcal{B}| = \ell - 1$) be a set of collaborating participants. Since $f_2(x_i) = f_1(x_i)$, $(1 \leq i \leq n)$, the collaborating participants of the set $\mathcal{B}$ know the following set of equations (corresponding to polynomial $f_1(x)$).

$$\begin{aligned} K + a_1 x_1 + \ldots + a_{t-1} x_1^{t-1} &= s_1 \\ &\vdots \\ K + a_1 x_t + \ldots + a_{t-1} x_t^{t-1} &= s_t \end{aligned}$$

They also know the following set of equations regarding the set $\mathcal{P}_1$ on polynomial $f_2(x)$.

$$\begin{aligned} K + b_1 x_1 + \ldots + b_{\ell-1} x_1^{\ell-1} &= s_1 \\ &\vdots \\ K + b_1 x_t + \ldots + b_{\ell-1} x_t^{\ell-1} &= s_t \end{aligned}$$

Without loss of generality, let the collaborating $\ell - 1$ participants be $\{P_{n+i_1}, \ldots, P_{n+i_{\ell-1}}\}$. So, they can provide the following set of $\ell - 1$ equations:

$$\begin{aligned} K + b_1 x_{n+i_1} + \ldots + b_{\ell-1} x_{n+i_1}^{\ell-1} &= s_{n+i_1} \\ &\vdots \\ K + b_1 x_{n+i_{\ell-1}} + \ldots + b_{\ell-1} x_{n+i_{\ell-1}}^{\ell-1} &= s_{n+i_{\ell-1}} \end{aligned}$$

It is not difficult to see that the above three sets of $t + t + \ell - 1$ linearly independent equations have $1 + (t-1) + t + (\ell - 1)$ unknowns (corresponding to K, a_is, shares $s_1 \cdots s_t$ and b_js, respectively). Since the number of equations is equal to the number of unknowns, the system of equations has a unique solution for K, that is $\ell - 1$ participants can exactly recreate the secret. □

It is worth mentioning that the problem arises because the overlapping shares leak some information. In the following section we shall show how to deal with this problem.

4.4.2 How to Extend a Shamir Scheme

So far (also in [71]) we have shown that the extension of Shamir schemes given in [83] is not secure. In this section, we show how to perform this task securely.

Let a Shamir (t, n) threshold scheme be constructed on a set $\mathcal{P}_1 = \{P_1, \ldots, P_n\}$ and let $f_1(x) = K + a_{1,1}x + \cdots + a_{1,t-1}x^{t-1}$ of degree at most $t-1$ be the polynomial associated with this scheme. Suppose we want to extend this scheme to a (ℓ, m) threshold scheme over the set $\mathcal{P}_2 = \{P_1, \ldots, P_n, P_{n+1}, \ldots, P_m\}$. In the following we show how to select a polynomial $f_2(x)$ of degree T such that every subset of ℓ or more participants from the

set $\mathcal{P}_2$ can recover the secret, but for every subset of less that ℓ participants the secret remains absolutely undetermined.

Let $f_2(x) = K + a_{2,1}x \cdots + a_{2,T}x^T$. Since $f_2(x_i) = f_1(x_i)$ for all x_i $(1 \leq i \leq n)$, the following set of $2 \times n$ equations are known.

$$\text{From } f_1(x) \begin{cases} K + a_{1,1}x_1 + \cdots + a_{1,t-1}x_1^{t-1} = s_1 \\ \vdots \\ K + a_{1,1}x_n + \cdots + a_{1,t-1}x_n^{t-1} = s_n \end{cases}$$

$$\text{From } f_2(x) \begin{cases} K + a_{2,1}x_1 + \cdots + a_{2,T}x_1^T = s_1 \\ \vdots \\ K + a_{2,1}x_n + \cdots + a_{2,T}x_n^T = s_n \end{cases}$$

The number of unknowns in this system of equations is $1+(t-1)+T+n$. The system has a unique solution if the number of equations is at least equal to the number of unknowns. In the extended scheme, however, the requirement is that at least ℓ participants from the set $\mathcal{P}_2$ must collaborate in order to recover the secret. Let a set $\mathcal{A} \subset \mathcal{P}_2$ $(|\mathcal{A}| = \ell)$ of participants include the following set of ℓ equations into the system (each participant contributes one equation).

$$\text{From } f_2(x) \begin{cases} K + a_{2,1}x_{j_1} + \cdots + a_{2,T}x_{j_1}^T = s_{j_1} \\ \vdots \\ K + a_{2,1}x_{j_\ell} + \cdots + a_{2,T}x_{j_\ell}^T = s_{j_\ell} \end{cases}$$

where $n+1 \leq j_i \leq m$, $1 \leq i \leq \ell$.

Now, we want the above set of $2 \times n + \ell$ equations to have a unique solution for K. This requires that $2 \times n + \ell = 1 + (t-1) + T + n$ (note that the later set of ℓ equations does not increases the number of unknowns). So, the dealer can select a suitable value for T (knowing t, ℓ and n).

Although we have shown that if ℓ participants from the set $\mathcal{P}_2 \setminus \mathcal{P}_1$ collaborate, then they can determine the secret, we must show that every subset of ℓ participants from the set $\mathcal{P}_2$ can also do so. Let j participants, $0 < j < \ell$ from the set $\mathcal{P}_2 \setminus \mathcal{P}_1$ collaborate in the secret reconstruction process. Thus, $\ell - j$ participants from the set $\mathcal{P}_1$ must collaborate in the secret reconstruction. Although this decreases the number of unknown shares $s_1, \ldots, s_n$ by $\ell - j$, the number of unknown shares in the system is still n (since $\ell - j$ shares regarding the absent participants are now unknown). That is, for every subset of ℓ participants from the set $\mathcal{P}_2$ the above set of equations has n unknown shares.

So, the extended scheme can be constructed if T is chosen such that $\ell + 2n = 1 + (t-1) + T + n$, or simply,

$$T = n + (\ell - t) \tag{4.3}$$

To construct the polynomial $f_2(x)$, the dealer first selects T random coefficients $a_{2,1}, \ldots, a_{2,T}$ such that $f_2(x) = K + a_{2,1}x + \cdots + a_{2,T}x^T$ satisfies $f_2(x_i) = f_1(x_i)$, $1 \leq i \leq n$. Next, it selects m distinct and non zero elements x_i $(n+1 \leq i \leq m)$, such that $x_i \neq x_j$ $(i \neq j,\ 1 \leq i, j \leq m)$ and computes shares $s_i = f_2(x_i)$ $(n+1 \leq i \leq m)$. Then the dealer privately sends the shares to their correspondence (only to the new $m - n$ participants of the extended scheme).

The polynomial $f_2(x)$ can be constructed if $T \geq t + 1$. Because, the condition $f_2(x_i) = f_1(x_i)$, $1 \leq i \leq n$ is equivalent to:

$$\begin{pmatrix} 1 & x_1 & \ldots & x_1^{T-1} \\ 1 & x_2 & \ldots & x_2^{T-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{t-1} & \ldots & x_{t-1}^{T-1} \end{pmatrix} \begin{pmatrix} a_{2,1} - a_{1,1} \\ a_{2,2} - a_{1,2} \\ \vdots \\ a_{2,t-1} - a_{1,t-1} \\ a_{2,t} \\ \vdots \\ a_{2,T} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

If $T \geq t + 1$, then $t - 1 \times T$ matrix above has rank $t - 1 < T - 1$, and hence the dealer can select $a_{2,1}, \ldots, a_{2,T}$ as desired. However, considering equation (4.3) and the fact that $n \geq t$ (this is a basic condition in Shamir scheme) we have, $T \geq \ell$. Since $\ell > t$ (otherwise, the dealer just generates $m - n$ shares in the constructed (t, n) scheme and sends them to their correspondence), we have $T > t$, that is,

$$\deg(f_2(x)) \geq \deg(f_1(x)) + 2$$

and the construction of $f_2(x)$ is possible every time. Thus, we obtain the following theorem.

Theorem 4.3 *For every Shamir (t, n) threshold scheme over a set $\mathcal{P}_1$, there exists a Shamir (ℓ, m) threshold scheme over a set $\mathcal{P}_2$ $(\mathcal{P}_1 \subset \mathcal{P}_2)$ which is an extension of the (t, n) scheme.*

As an example, let a Shamir $(2, 2)$ scheme be constructed over a set $\mathcal{P}_1 = \{P_1, P_2\}$. Let $\mathcal{P}_2 = \{P_1, P_2, P_3, P_4, P_5\}$ and we want to extend the $(2, 2)$ scheme to a $(3, 5)$ scheme (3 members joining the group). Further, let $f_1(x) = K + a_{1,1}x$ of degree at most 1 be the polynomial associated with the $(2, 2)$ scheme. In order to compute T, the degree of

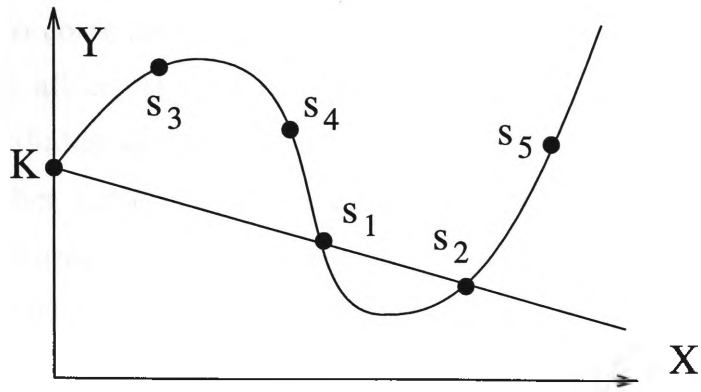

Figure 4.3: Extension of a Shamir $(2,2)$ scheme to a $(3,5)$ scheme

polynomial $f_2(x) = K + a_{2,1}x + \cdots + a_{2,T}x^T$ corresponding to the extended scheme, we obtain (using equation 4.3)

$$T = 2 + 3 - 2 = 3.$$

Figure 4.3 illustrates such an extension. Note that, in the original Shamir schemes a polynomial of degree 3 is associated with a $(4,n)$ scheme. However, as we have shown earlier, knowing the fact that a scheme is an extended scheme enables every set of 3 participants to recover the secret. For example, participants P_1, P_3 and P_4 know that,

$$\begin{aligned}
K + a_{1,1} \times x_1 &= s_1 \\
K + a_{1,1} \times x_2 &= s_2 \\
K + a_{2,1} \times x_1 + a_{2,2}x_1^2 + a_{2,3}x_1^3 &= s_1 \\
K + a_{2,1} \times x_2 + a_{2,2}x_2^2 + a_{2,3}x_2^3 &= s_2 \\
K + a_{2,1} \times x_3 + a_{2,2}x_3^2 + a_{2,3}x_3^3 &= s_3 \\
K + a_{2,1} \times x_4 + a_{2,2}x_4^2 + a_{2,3}x_4^3 &= s_4.
\end{aligned}$$

Since the above set of 6 equations has 6 unknowns (K, $a_{1,1}$, $a_{2,1}$, $a_{2,2}$, $a_{2,3}$ and s_2), the secret K can be easily computed.

Note. The extension of multiple assignment secret sharing schemes [83] was an attempt to solve the problem of sharing a secret in hierarchical groups (see Simmons [154]). However, in their method the above scenario indicates an extension of a Shamir $(2,2)$ scheme to a Shamir $(4,5)$ scheme, which is not the case.

We shall show in a moment how this extension technique can be used in the construction of our multilevel (hierarchical) secret sharing scheme.

4.4.3 Correctness of the Reconstructed Secret

An important issue in a secret sharing scheme is that the reconstruction procedure must provide the valid secret to all participants of an authorised set. There are fundamentally

different approaches to correctness for shared secret schemes. For example, a verifiable secret sharing scheme allows the honest participants to ensure that they can recover a unique secret. Verifiable secret sharing schemes were introduced in [39] and then discussed by many other authors (see, for example, [58]). Although in verifiable secret sharing schemes participants can verify the validity of their own shares, they cannot know whether other participants with whom they might collaborate to reconstruct the secret also have valid shares. This problem is discussed in publicly verifiable secret sharing schemes [164], in which participants of every authorised set are able to convince everybody that their shares are related to the secret. That is, in the secret reconstruction phase they can recover the secret. Even assuming that the dealer is honest and every shareholder received correct shares, there is no guarantee that the secret reconstruction protocol provides the original secret for the authorised set of collaborating participants. That is, a dishonest participant may fool the others so they obtain an invalid secret. This problem has been discussed by several authors (see for example [170], [25], [155] and [107]).

In some applications of a secret sharing scheme no additional confirmation that a correct value for the secret has been recovered is needed. For example, the bank vault door either opens or it does not, after an authorised set of participants have entered their shares. This is not the case if the action controlled by a shared secret scheme is distant, in either time or place, from where an authorised set of participants apply their shares. For example, (following Simmons [155]) if the controlled action is the arming of a missile warhead, it is desirable to have confirmation prior to launch that the correct arming code has been entered, as opposed to learning after the missile arrives at the target that the warhead had not been armed.

We observe two different approaches to solve this problem.

Detection of Cheating - The secret reconstruction protocol has the capability to detect cheating (if it occurs). In this case, in general, the cheater(s) can obtain the correct secret while the honest participants obtain nothing.

Prevention of Cheating - The secret reconstruction protocol provides no information about the secret to the collaborating participants. That is, neither cheater(s) nor honest participants obtain any information about the secret.

In [68] we have proposed a protocol that can detect and prevent cheating in online secret sharing schemes. We have shown the online secret sharing scheme [134] is vulnerable to cheating. In this scheme (see Section 4.3.1), a dishonest participant $P_i \in \mathcal{A}$ may

contribute with fake share $s_i' = \alpha s_i$, where α is a random integer modulo q. Since every participant of an authorised set $\mathcal{A}$ ($|\mathcal{A}| = t$) has access to the final result $g_{\mathcal{A}}^{s_1 \cdots s_i' \cdots s_t}$, the participant P_i can calculate the value,

$$(g_{\mathcal{A}}^{s_1 \cdots s_i' \cdots s_t})^{\alpha^{-1}} = g_{\mathcal{A}}^{s_1 \cdots s_i \cdots s_t} = g_{\mathcal{A}}^{\prod_{P_i \in \mathcal{A}} s_i} = V_{\mathcal{A}}$$

and hence the correct secret as in Pinch's scheme, while the other participants calculate an invalid secret.

How to Detect Cheating

Suppose in the initialisation phase of the Pinch scheme, the dealer publishes $g_{\mathcal{A}}^{V_{\mathcal{A}}}$ (corresponding to every authorised set $\mathcal{A}$). Let the reconstruction protocol be the same as in the original Pinch scheme and let $V_{\mathcal{A}}'$ be the final result. Every participant $P_i \in \mathcal{A}$, can verify whether

$$g_{\mathcal{A}}^{V_{\mathcal{A}}} \stackrel{?}{=} g_{\mathcal{A}}^{V_{\mathcal{A}}'}.$$

If the verification fails, then a cheating has occurred in the protocol and thus the computed secret is not valid. This protocol detects cheating but does not detect the cheater(s) nor prevent cheating. That is, the cheater(s) obtain the secret while the others gain nothing.

How to Prevent Cheating

Let $a = \sum_{P_i \in \mathcal{A}} g_{\mathcal{A}}^{s_i}$ correspond to an authorised set $\mathcal{A}$. We assume that in the initialisation phase of the Pinch scheme the dealer also publishes $a_{\mathcal{A}} = g_{\mathcal{A}}^{a}$. Note that this extra public information gives no useful information about the secret or about participants' shares. Otherwise one could solve the discrete logarithm in G_1 and easily solve the Diffie-Hellman problem.

Let $\mathcal{A}$ be an authorised set of participants. At the reconstruction phase, every participant $P_i \in \mathcal{A}$ computes $g_{\mathcal{A}}^{s_i}$ and broadcasts it to all participants in the set $\mathcal{A}$. Thus, every participant $P_i \in \mathcal{A}$ receives $t-1$ values $g_{\mathcal{A}}^{s_j}$ corresponding to all $P_j \in \mathcal{A}$, $P_j \neq P_i$. Each participant computes a and verifies $a_{\mathcal{A}} \stackrel{?}{=} g_{\mathcal{A}}^{a}$. If the verification fails, then the protocol stops. Let participants agree to perform computation in the cycle $P_1, \ldots, P_t$. If the check $a_{\mathcal{A}} \stackrel{?}{=} g_{\mathcal{A}}^{a}$ is successful, then each participant P_i ($i = 1, \ldots t$) knows the true value $g_{\mathcal{A}}^{s_{i-1}}$ of its predecessor (P_t is the predecessor of P_1). So participant P_i ($i = 1, \ldots, t$) initiates the protocol by computing the value $(g_{\mathcal{A}}^{s_{i-1}})^{s_i}$ and passing it to P_{i+1}. The protocol proceeds as in the Pinch scheme and ends at P_{i-2}. In this way, the participant P_{i-1} cannot directly contribute to the computation which was started by P_i.

Let there exist only one cheater, P_i $(1 \leq i \leq t)$ in the system. So if P_i cheats, the computation initiated by P_{i+1} must be correct (the correctness can be verified as $g_{\mathcal{A}}^{V_{\mathcal{A}}} \stackrel{?}{=} g_{\mathcal{A}}^{V'_{\mathcal{A}}}$, where $V'_{\mathcal{A}}$ is the result obtained by P_{i-1}). That is, although cheating has occurred, the honest set of participants can recover the secret.

If there exists a group of collaborating cheaters, then each participant must play (simultaneously) the role of P_1 for every other participant in set $\mathcal{A}$. Although the number of computations increases rapidly, before completing the protocol any possible cheating will be detected and the protocol will be stopped (for more detail see [68]).

4.5 Secret Sharing in Multilevel Groups

In multilevel t_i-out-of-n_i secret sharing schemes, the set of all participants is divided into levels (classes). The i-th level contains n_i participants. The levels create a hierarchical structure. Any t_i participants on the i-th level can recover the secret. When the number of cooperating participants from the i-th level is smaller than t_i, say r_i, then $t_i - r_i$ participants can be taken from higher levels. For example, a bank may require the concurrence of two vice-presidents or three senior tellers to authenticate an electronic funds transfer (EFT). If there are only two senior tellers available, the missing one can be substituted by a vice president.

The concept of multilevel (or hierarchical) secret sharing was considered by several authors (see, for example Shamir [147], Kothari [96], Ito et al [83] and Charnes et al [32]). Shamir [147] suggests that threshold schemes for hierarchical groups can be realized by giving more shares to higher level participants. Kothari [96] considered hierarchical threshold schemes in which a simple (t_i, n_i) threshold scheme is associated with the i-th level of a multilevel group. The obvious drawback of this solution is that it does not provide concurrency among different levels of hierarchical groups. Ito et al [83] discussed secret sharing for general access structures and proved that every access structure can be realized by a perfect secret sharing scheme. The main drawback of their scheme is that the more privileged participants are assigned longer shares.

Simmons [154] pointed out that the solutions for secret sharing in multilevel groups proposed by earlier authors are not efficient. He suggested efficient geometrical secret sharing schemes with the required properties. However, his solution is applicable only to a particular case of multilevel groups. More precisely, he discussed secret sharing in multilevel groups with particular access structures.

Brickell [22] studied general secret sharing in multilevel groups and proved that it is

possible to construct ideal secret sharing schemes for any multilevel access structure. In Brickell's vector space construction, the lower bound on the size of the modulus p (size of the field in which the calculations are being done) is considerably large.

In this section we present an efficient solution for secret sharing in multilevel groups. Our scheme is based on the Shamir scheme and is perfect and ideal. In our scheme, the lower bound on the modulus is significantly smaller than in Brickell's scheme. Indeed, the condition $p > n$ (as in Shamir's original scheme) is sufficient to implement our proposed scheme.

4.5.1 Notations

Assume that a multilevel (or hierarchical) group consists of ℓ levels. That is, a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n participants is partitioned into ℓ disjoint subsets $\mathcal{P}_1, \ldots, \mathcal{P}_\ell$. The subset $\mathcal{P}_1$ is on the highest level of the hierarchy while $\mathcal{P}_\ell$ is on the least privileged level. Denote the number of participants on the i-th level as $n_i = |\mathcal{P}_i|$. The threshold t_i indicates the smallest number of participants on the i-th or higher levels, who can cooperate to successfully reconstruct the secret. The number of participants in the scheme is $n = |\mathcal{P}| = \sum_{i=1}^{\ell} |\mathcal{P}_i| = \sum_{i=1}^{\ell} n_i$. Let N_i be the total number of participants on the i-th and higher levels. That is, $N_i = \sum_{j=1}^{i} n_j$, $1 \leq i \leq \ell$. Clearly, $N_1 = n_1$ and $N_\ell = n$. Of course, we assume that the thresholds on different levels satisfy the following relation $t_1 < t_2 < \cdots < t_\ell$. Note that the reconstruction of a secret can be initialised by any hierarchical subgroup of $\mathcal{P}_i$. If their number is smaller than the threshold number t_i, the subgroup can ask some participants from the higher levels to collaborate and pool their shares. The total number of participants has to be at least t_i. The access structure is defined as:

$$\Gamma = \{\mathcal{A} \subseteq \mathcal{P} \mid \sum_{j=1}^{i} |\mathcal{A} \cap \mathcal{P}_j| \geq t_i \text{ for } i = 1, \ldots, \ell\} \tag{4.4}$$

Consider again the bank example where monetary transactions can be authenticated by three senior tellers or two vice presidents. So, there exist two levels of hierarchy. The first (higher) level consists of two vice presidents $\mathcal{P}_1 = \{P_1, P_2\}$ with $n_1 = 2$. The second (lower) level consists of three senior tellers $\mathcal{P}_2 = \{P_3, P_4, P_5\}$ with $n_2 = 3$. To recover the secret, it is necessary that two participants on the first level or three participants on the second level or three participants from both levels pool their shares. Thus, $t_1 = 2$, $t_2 = 3$ and $n = 5$.

4.5.2 The Model

Our model utilises a sequence of related Shamir threshold schemes with overlapping shares.

Lemma 4.4 (transitivity of extension) *If* Γ_2 *is an extension of* Γ_1 *and* Γ_3 *is an extension of* Γ_2 *then* Γ_3 *is an extension of* Γ_1.

We say Γ_3 is the second extension of Γ_1. Similarly, we define the i^{th} extension of a Shamir threshold scheme.

Secret sharing for multilevel access structures displays some common features with threshold schemes. However, an implementation of multilevel secret sharing based on a sequence of independent (t_i, n_i) threshold schemes on each level $i = 1, \ldots, \ell$ makes the cooperation among participants existing on different levels difficult to achieve. Denoted by,

$$\mathcal{P}^i = \bigcup_{j=1}^{i} \mathcal{P}_j$$

the set of all participants on the i-th and all higher levels. An alternative implementation of secret sharing for multilevel access structures would involve a sequence of independent threshold schemes (t_i, N_i) for the set $\mathcal{P}^i$ $(i = 1, \ldots, \ell)$ of participants. This solution requires $\ell - i + 1$ shares to be assigned to each participant on the i-th level.

A reasonable implementation of secret sharing for a multilevel access structure can be done as follows. First a (t_1, n_1) threshold scheme (scheme A_1) is designed. It corresponds to the first (highest) level of participants from $\mathcal{P}^1$. Then a (t_2, N_2) threshold scheme (scheme A_2) for $\mathcal{P}^2$ is constructed as an extension of A_1. Next a (t_3, N_3) threshold scheme (scheme A_3) for $\mathcal{P}^3$ is constructed as an extension of A_2. The process continues until a (t_ℓ, N_ℓ) threshold scheme (scheme A_ℓ) for $\mathcal{P}^\ell$ is constructed by extending the threshold scheme $A_{\ell-1}$. In this implementation, each participant will be assigned a single share only.

In Section 2.3 we have shown the extension is possible for every extended set of participants. That is, our model is applicable for any multilevel (hierarchical) access structure. We use $(t_i, N_i)_{T_i}$ scheme to denote an extended Shamir (t_i, N_i) threshold scheme in which the polynomial associated with the scheme has a degree of at most T_i (in general, $T_i > t_i$).

At level i, the dealer can easily calculate the value T_i, which is the degree of the polynomial associated with a $(t_i, N_i)_{T_i}$ threshold scheme for $\mathcal{P}^i$. The dealer observes that the available set of equations with the system is as follows: N_1 equations (for level

1) with T_1 (maximum number of coefficients in $f_1(x)$); N_2 equations (for level 2) with T_2 unknowns and, in general, N_{i-1} equations (for level $i-1$) with T_{i-1} unknowns (plus one unknown, corresponding to the secret itself). Since the requirement is that at least t_i participants must collaborate in order to recover the secret, T_i must must satisfy the following equality:

$$t_i + \sum_{j=1}^{i-1} N_j = 1 + \sum_{j=1}^{i} T_j \tag{4.5}$$

which contains a single unknown value, T_i (all T_j, $1 \leq j \leq i-1$ have been calculated in previous levels).

Hence, a secret sharing for a given multilevel access structure can be implemented according to the following algorithm:

Algorithm 1 – a $(t_i, N_i)_{T_i}$ secret sharing scheme.

1. Select at random a polynomial of degree at most $T_1 = t_1 - 1$ and compute n_1 shares for n_1 participants from $\mathcal{P}_1$. The outcome is a $(t_1, N_1)_{T_1}$ threshold scheme $(N_1 = n_1)$.

2. For $i = 2$ to ℓ do:

   - for the given initial $(t_{i-1}, N_{i-1})_{T_{i-1}}$ threshold scheme, construct its extension $(t_i, N_i)_{T_i}$.
   - compute n_i shares for participants on the i-th level,
   - take the next i,

3. Distribute the shares to corresponding participants via secure channels.

4.5.3 Security of the Scheme

The following theorem demonstrates that secret sharing schemes obtained using Algorithm 1 are perfect.

Theorem 4.5 *Algorithm 1 produces an ideal and perfect secret sharing scheme for an arbitrary multilevel access structure.*

Proof. (Sketch) Algorithm 1 produces ℓ threshold schemes $A_1, \ldots, A_\ell$, where:

A_1 is defined by polynomial $f_1(x)$ for $\mathcal{P}_1$,
A_i is defined by polynomial $f_i(x)$ for $\mathcal{P}^i = \bigcup_{j=1}^{i} \mathcal{P}_j$, and

$f_i(x) = K + a_{i,1}x + \ldots + a_{i,T_i}x^{T_i}$ for $i = 1, \ldots, \ell$.

Without loss of generality, we can assume $\mathcal{B} = \{P_1, \ldots, P_w\} \notin \Gamma$ are the collaborating participants. For each level, we can determine $\mathcal{B}_i = \mathcal{B} \cap \mathcal{P}^i$ and the number $\beta_i = |\mathcal{B}_i|$. Clearly, $\beta_i < t_i$. So, each system of equations for the i-th level does not produce a unique solution. Indeed, according to the method of generating polynomials associated with Shamir threshold scheme for level i, every subset of all equations available to the set $\mathcal{B}$ has more unknowns than the number of equations, and therefore, has no unique solution. That is, the solution is a space equivalent to $GF(p)$, and thus, the secret remains absolutely undetermined.

4.5.4 The Lower Bound on the Modulus

Brickell proved [22, Theorem 1] that there exists an ideal secret sharing scheme for a multilevel access structure over $GF(p)$ if:

$$p > (\ell - 1)\begin{pmatrix} n \\ \ell - 1 \end{pmatrix}.$$

It is easy to show that, in the construction by Algorithm 1, the above condition on size p can be removed.

Corollary 4.6 *Let Γ be a multilevel access structure with ℓ levels. Assume further that the secret sharing scheme for Γ is implemented by the sequence of threshold schemes $A_1, \ldots, A_\ell$; created according to Algorithm 1. That is, A_i is a $(t_i, N_i)_{T_i}$ threshold scheme. Thus, the lower bound of p in our scheme is given by $p > T_\ell$*

Now we give a simple assessment of the required lower bound for p in our scheme. From equation (4.5), since $t_\ell < N_\ell$, we have $\sum_{j=1}^{i} T_j < \sum_{j=1}^{i} N_j$ $(i = 1, \ldots, \ell)$. That is, in general $T_i < N_i$ and therefore $T_\ell < N_\ell$. In other words the basic condition of the original Shamir scheme, that is,

$$p > n$$

is sufficient to implement our scheme (since $N_\ell = n$).

Remark 2 *It is worth mentioning that, although our proposed multilevel secret sharing scheme and Brickell's multilevel scheme[22] both are ideal, and therefore they have the same information rate, they are not equivalent. We observe the following:*

1. *The secret space, $\mathcal{K} = GF(p)$, is fixed beforehand and the dealer must select the secret from this space. Clearly, for every p in which Brickell's construction generates an ideal scheme, our scheme does so. However, for all values*

$$(\ell - 1)\begin{pmatrix} n \\ \ell - 1 \end{pmatrix} > p > n$$

our system produces an ideal scheme, but Brickell's does not.

2. *The secret space, $\mathcal{K} = GF(p)$, is chosen by the dealer. In this case, the dealer of our scheme can select considerably smaller values for p (compared to Brickell's scheme). Hence, the shares in our scheme, in general, are smaller than the shares in Brickell's scheme.*

4.6 Secret Sharing in Compartmented Groups

The notion of compartmented secret sharing was introduced by Simmons [154]. In compartmented t_i-out-of-n_i secret sharing schemes there are several compartments each consisting of n_i participants. The secret is partitioned in such a way that its reconstruction requires the cooperation of at least t_i participants in some, or perhaps all, compartments. Consider the example presented by Simmons in [154]. Let two countries agree to control the recovery of the secret (which may initiate a common action) by a secret sharing scheme. The secret can be recreated only if at least two participants from both countries pool their shares together.

Simmons [154] discussed the construction of secret sharing in compartmented groups with particular access structures. Brickell [22] studied general secret sharing in compartmented groups and proved that it is possible to construct ideal secret sharing schemes for any compartmented access structure. In this section, we present an efficient solution to this class of secret sharing schemes.

4.6.1 The Scheme

Let the set of participants $\mathcal{P}$ be partitioned into ℓ disjoint sets $\mathcal{P}_1, \ldots, \mathcal{P}_\ell$. The compartmented access structure Γ is defined as follows.

Definition 4.2 *A subset $\mathcal{A} \subset \mathcal{P}$ belongs to the access structure Γ if:*

1. $|\mathcal{A} \cap \mathcal{P}_i| \geq t_i$ *for* $i = 1, \ldots, \ell$, *and*

2. $|\mathcal{A}| \geq t$ *where* $t \geq \sum_{i=1}^{\ell} t_i$.

The number of participants in different compartments and integers $t, t_1, \ldots, t_\ell$ *determine an instance of the compartmented access structure.*

We consider two distinct cases.

Case $t = \sum_{i=1}^{\ell} t_i$

In this case the access structure is:

$$\Gamma = \{A \subseteq \mathcal{P} \quad | \quad |A \cap \mathcal{P}_i| \geq t_i \quad \text{for} \quad i = 1, \ldots, \ell\} \tag{4.6}$$

A trivial solution for the above access structure is as follows. The dealer simply chooses $\ell - 1$ random values $c_1, \ldots, c_{\ell-1}$ from elements of $GF(p)$, and defines a polynomial,

$$\kappa(x) = K + c_1 x + \ldots + c_{\ell-1} x^{\ell-1}.$$

The secret $K = \kappa(0)$ and the partial secrets $k_i = \kappa(i)$ for $i = 1, \ldots, \ell$. The dealer constructs a Shamir (t_i, n_i) scheme for each compartment i. The schemes are independently designed and the scheme in the i-th compartment allows recovery of the partial key k_i. The collection of shares for all compartments are later distributed securely to the participants. Obviously, if at least t_i participants of the i-th compartment pool their shares, they can reconstruct the partial secret k_i. A group of fewer than t_i collaborating participants learns absolutely nothing about k_i. Thus, the reconstruction of the secret K needs all partial keys to be reconstructed by at least t_i participants in each compartment i $(i = 1, \ldots, \ell)$.

This solution was also proposed by Brickell [22]. However, prior to the results described here, no efficient solution has been proposed for a general compartmented access structure in which $t > \sum_{i=1}^{\ell} t_i$.

Case $t > \sum_{i=1}^{\ell} t_i$

The corresponding access structure is:

$$\Gamma = \{A \subseteq \mathcal{P} \quad | \quad |A| \geq t, \ |A \cap \mathcal{P}_i| \geq t_i \quad \text{for} \quad i = 1, \ldots, \ell\} \tag{4.7}$$

Let $T = t - \sum_{i=1}^{\ell} t_i$. The secret sharing scheme for a compartmented access structure Γ is designed according to the following algorithm.

Algorithm 2 – a $(t_i, n_i | t; i = 1, \ldots, \ell)$ *secret sharing scheme.*

1. Choose $\ell - 1$ random values $c_1, \ldots, c_{\ell-1} \in GF(p)$ and define the polynomial,

$$\kappa(x) = K + c_1 x + \ldots + c_{\ell-1} x^{\ell-1}.$$

The secret $K = \kappa(0)$ and the partial secrets $k_i = \kappa(i)$ for $i = 1, \ldots, \ell$.

2. Select randomly and uniformly $t_i - 1$ values $a_{i,1}, \ldots, a_{i,t_i-1}$ from $GF(p)$ corresponding to each level i, $i = 1, \ldots, \ell$,

3. Choose randomly and uniformly T values $\beta_1, \ldots, \beta_T$ from $GF(p)$,

4. Determine a sequence of ℓ polynomials,

$$f_i(x) = k_i + a_{i,1}x + \cdots + a_{i,t_i-1}x^{t_i-1} + \beta_1 x^{t_i} + \cdots + \beta_T x^{t_i+T-1}$$

for every level i.

5. Compute shares for all compartments, i.e. $s_{i_j,i} = f_i(x_{i_j,i})$ for $j = 1, \ldots, n_i$ and $i = 1, \ldots, \ell$ ($n_i = |\mathcal{P}_i|$) and send them securely to the participants.

4.6.2 Security of the Scheme

The following theorem demonstrates that secret sharing schemes obtained using Algorithm 2 are perfect.

Theorem 4.7 *The secret sharing scheme obtained from Algorithm 2 for a compartmented access structure Γ of the form given by equation (4.7) is ideal and perfect and allows recovery of the secret only if the set of cooperating participants $\mathcal{A} \in \Gamma$.*

Proof. (Sketch) First we prove that if $\mathcal{A} \in \Gamma$, then the participants from $\mathcal{A}$ can recover the secret K. Note that to determine the secret K, each compartment needs to recover its associated partial secret k_i.

Since $\mathcal{A} \in \Gamma$, there must be at least t_i collaborating participants from each compartment. Let the actual numbers of collaborating participants be $\alpha_1, \ldots, \alpha_\ell$, such that $\alpha_i \geq t_i$ and $\sum_{i=1}^{\ell} \alpha_i \geq t$. The combiner who collects all shares from participants in $\mathcal{A}$

can establish the following system of linear equations:

$$\begin{cases} k_1 + a_{1,1}x_{i_1,1} + \cdots + a_{1,t_1-1}x_{i_1,1}^{t_1-1} + \beta_1 x_{i_1,1}^{t_1} + \cdots + \beta_T x_{i_1,1}^{t_1+T-1} & = s_{i_1,1} \\ & \vdots \\ k_1 + a_{1,1}x_{i_{\alpha_1},1} + \cdots + a_{1,t_1-1}x_{i_{\alpha_1},1}^{t_1-1} + \beta_1 x_{i_{\alpha_1},1}^{t_1} + \cdots + \beta_T x_{i_{\alpha_1},1}^{t_1+T-1} & = s_{i_{\alpha_1},1} \end{cases}$$
$$\vdots$$
$$\begin{cases} k_\ell + a_{\ell,1}x_{i_1,\ell} + \cdots + a_{\ell,t_\ell-1}x_{i_1,\ell}^{t_\ell-1} + \beta_1 x_{i_1,\ell}^{t_\ell} + \cdots + \beta_T x_{i_1,\ell}^{t_\ell+T-1} & = s_{i_1,\ell} \\ & \vdots \\ k_\ell + a_{\ell,1}x_{i_{\alpha_\ell},1} + \cdots + a_{1,t_\ell-1}x_{i_{\alpha_\ell},1}^{t_\ell-1} + \beta_1 x_{i_{\alpha_\ell},1}^{t_\ell} + \cdots + \beta_T x_{i_{\alpha_\ell},1}^{t_\ell+T-1} & = s_{i_{\alpha_\ell},\ell} \end{cases}$$

In the above system of equations, t_i unknown coefficients $k_i, a_{i,j}$ $(j = 1, \ldots, t_i - 1)$ are associated with compartment i, $i = 1, \ldots, \ell$. The T unknown β_i are common in all equations. Since we have at least t equations with t unknowns, the system has a unique solution. Knowing partial secrets k_i, the secret K can be recovered.

Assume that $\mathcal{A} \notin \Gamma$. Then there are two possibilities. The first possibility is that there is a compartment i for which $\alpha_i < t_i$. This immediately implies that the corresponding partial key k_i cannot be found. The second possibility is that all $\alpha_i \geq t_i$, but $\sum_{i=1}^{\ell} \alpha_i < t$. This precludes the existence of the unique solution for $\beta_1, \ldots, \beta_T$.

4.6.3 The Lower Bound on the Modulus

Brickell showed that [22, Theorem 3] there exists an ideal secret sharing scheme for a compartmented access structure over $GF(p)$ if:

$$p > \begin{pmatrix} n \\ t \end{pmatrix}$$

where n and t are the the same as in our scheme. In our proposed scheme, independent Shamir schemes are constructed for every compartment. Since $t_i + T < n$, it is easy to derive the following corollary.

Corollary 4.8 *Let Γ be a compartmented access structure with ℓ levels and $n = |\mathcal{P}|$ participants. Then there is an ideal secret sharing scheme for Γ over $GF(p)$ if:*

$$p > n.$$

That is, to construct our secret sharing scheme in compartmented groups no additional conditions need to be satisfied.

Remark 3 *With similar arguments (see Remark 2 in Section 4.5.4) our proposed scheme for secret sharing in compartmented groups has advantages over Brickell's scheme.*

Part II

Society-Oriented Cryptography

Contents

The main consideration of this part of thesis is society-oriented cryptographic systems, which is the application of secret sharing schemes in cryptography. It consists of Chapters 5, 6 and 7.

The first chapter of this part, Chapter 5, considers the principals of modern cryptographic algorithms. In spite of the fact that all existing cryptographic algorithms are based on computations in finite algebraic structures, it is shown how the concept of floating-point arithmetic can be used in the construction of cryptographic algorithms. This chapter presents two novel cryptographic algorithms which apply transcendental numbers.

In Chapter 6 threshold cryptographic systems are discussed. The concept of society-oriented cryptography and the implementation considerations of such systems are studied. Threshold cryptography and the well known public-key based threshold decryption systems are reviewed. Generalised threshold cryptographic schemes is discussed and a cryptosystem for hierarchical groups is presented. This chapter also discusses the concept of threshold digital signatures. Some well-known approaches to the construction of threshold digital signatures, namely threshold RSA digital signatures and threshold ElGamal-type digital signatures are discussed. A particular RSA-type shared generation of signatures, the so called Boyd's scheme, is reviewed and an improvement of the system is discussed.

Chapter 7 is devoted to the group-oriented cryptographic systems. A related work on this area is studied and a general model for constructing such systems is presented. Group-oriented cryptosystems based on public-key and private-key cryptosystems are presented. Also a self certified group-oriented cryptosystem that works with no help of a combiner has been introduced in this chapter.

Chapter 5

Cryptography

Modern cryptography deals with integers only; this principle is widely accepted in the cryptographic community. Floating-point arithmetic is never used in cryptographic algorithms. In this Chapter, two classes of transcendentals are applied to construct novel encryption algorithms.

5.1 Transcendental Numbers Based Cryptography

Without exceptions, all cryptographic algorithms are based on computations in finite algebraic structures (e.g., groups, rings, fields, and vector spaces). The results of these computations are always exact – no rounding or approximation is involved. On the contrary, a single-bit error in the input of any cryptographic algorithm generates an unintentional result.

In [132] we described a new and efficient algorithm for the multiplication of floating-point numbers. We suggested two secure pseudo-random bit generators based on transcendental numbers. These two classes of transcendentals are applied to construct novel encryption algorithms.

5.1.1 Introduction

The operational disadvantages of perfectly secure stream cipher cryptosystems (e.g. the one-time pad cryptosystem) which apply truly random sequences as the keystream, have led to the development of practically secure stream cipher systems [138]. The strength of such stream cipher cryptosystems depends on the security of their keystreams. Hence, the main challenge in stream cipher design is to produce sequences that appear random. Since such sequences are not truly random they are called *pseudo-random* sequences.

Classical pseudo-random generators are deterministic algorithms with well known mathematical structures that output numbers or binary strings which "appear" random[1].

A number of pseudo-random generators have been proposed in the literature, several of those have been broken (for example see [116], [139], [140], [153], and [76]). The common characteristic of these generators is that they produce periodic sequences. Although the period of the sequence is long, and in practice the whole sequence will not be generated, this makes the sequence vulnerable.

Here we propose a method for the generation of pseudo-random sequences based on the expansion of irrational numbers. In contrast to the conventional pseudo-random generators our method produces non-periodic sequences.

5.1.2 Notations and Definitions

In this section we provide the reader with a collection of basic terms and definitions which are used throughout this chapter.

The decimal representation of a rational number is either finite or periodic. By contrast the decimal representation of every irrational number is infinite and non-periodic. Real numbers satisfying an equation of the form:

$$f(x) = a_0 + a_1 x + \ldots + a_d x^d = 0$$

with integral coefficients, are either integers or irrationals. Such numbers are called *algebraic.* Irrational numbers which satisfy no such equation are called *transcendentals.*

An attractive feature of real numbers is that they can represent any (infinite or finite) sequence of integers. Consider an experiment in which an unbiased coin is flipped an 'infinite' number of times. It is clear that the resulting random sequence is equivalent to some real number. Obviously, this sequence (the real) must not be either a rational or algebraic number, as in both cases a finite subsequence uniquely determines the rest of the (infinite) sequence. All infinite sequences of truly random integers fall into the broad class of transcendental. Algebraic irrationals may look 'random' but their 'randomness' is limited to a finite subsequence.

Definition 5.1 [2] *Assume $a < b$. The closed interval $[a,b]$ is the set $\{x \mid a \leq x \leq b\}$.*

Definition 5.2 [54] *A measure on a set $\mathcal{X}$ is a mapping μ from the set of subsets of $\mathcal{X}$ to interval [0,1], which satisfies:*

[1]The importance of random and pseudo-random number generation in Cryptology is discussed in [148] and [173].

(i) $\mu(\mathcal{X}) = 1$;

(ii) if $y = \cup_{i\in\omega} x_i$ *is a disjoint union and* $y \in \mathcal{X}$, *then*

$$\mu(y) = \sum_{i\in\omega} \mu(x_i),$$

where ω denotes the set of all non-negative integers.

Definition 5.3 [123] *A set* $\mathcal{R}$ *of real numbers is said to have the measure zero if it is possible to cover the points of* $\mathcal{R}$ *with a set of intervals of arbitrary small total length.*

As an example, the set of natural numbers has the measure zero. When the integer 1 is enclosed in the interval $[1 - \frac{\varepsilon}{2}, 1 + \frac{\varepsilon}{2}]$, the integer 2 in the interval $[2 - \frac{\varepsilon}{4}, 2 + \frac{\varepsilon}{4}]$, $\cdots$, and in general the integer i in the interval $[i - \frac{\varepsilon}{2^i}, i + \frac{\varepsilon}{2^i}]$, the natural numbers are covered by intervals of total length $\varepsilon + \frac{\varepsilon}{2} + \frac{\varepsilon}{4} + \cdots = 2\varepsilon$, which can be made arbitrarily small.

Theorem 5.1 [162] *Every closed interval* $[a, b]$ *is a measurable set.*

Definition 5.4 [169] *A function* $f(x)$ *is said to satisfy a Lipschitz condition of order* m *on the closed interval* $[a, b]$ *if there is a constant* c *such that*

$$|f(x_2) - f(x_1)| \leq c\,|x_2 - x_1|^m$$

for all values of x_1, $x_2 \in [a, b]$.

Theorem 5.2 (Approximation) [162] *If* $f(x)$ *is continuous in the closed interval* $[a, b]$ *and* ε *is any given positive number, there exists a polynomial* $g(x)$ *such that the inequality*

$$|f(x) - g(x)| < \varepsilon$$

is satisfied throughout the interval $[a, b]$.

Consider a quadratic equation $ax^2 + bx + c$ $(a \neq 0)$. Let $d = b^2 - 4ac$ be a positive integer which is not a perfect square. Then the root of the above quadratic equation is a *quadratic irrational number.* That is,

$$\alpha = \frac{(b + \sqrt{d})}{2a}$$

is an example of quadratic irrational numbers.

Definition 5.5 [3] *A Diophantine equation is an equation in one or more unknowns which is to be solved for integral values of the unknowns.*

For example, the primitive solutions, in non-negative integers, of the Diophantine equation,

$$x^2 + 2y^2 = z^2$$

are given by $x = \pm(r^2 - 2s^2)$, $y = 2rs$ and $z = r^2 + 2s^2$, where $x > 0$ and r and s are non-negative integers, such that $\gcd(r, 2s) = 1$.

5.1.3 Binary Sequences from Expansion of Irrational Numbers

Assume we have a random generator which allows us to select an irrational number, randomly with a uniform probability distribution, from the interval [0,1]. We now discuss whether the binary expansion of these selected irrational numbers can be used to generate (at least in principle) random sequences.

Lemma 5.3 *There is at least one irrational number in whose binary expansion the finite binary sequence* $W = 0.w_1w_2\cdots w_j$ *occurs.*

Proof. The sequence $W010010001\ldots$, is an infinite non-periodic sequence which contains the sequence W.

Theorem 5.4 *There are infinitely many irrational numbers in whose binary expansions the finite binary sequence* $W = 0.w_1w_2\cdots w_j$ *occurs.*

Proof. Assume that there are a finite number m of irrational numbers whose binary expansions contain the sequence W. Extending W by just one bit gives two different sequences $W \parallel 0$ and $W \parallel 1$. Thus extending W by i bits gives 2^i different finite length sequences (each of length $j + i$). We choose an i such that $2^i > m$. Hence, there is a finite length sequence which can not appear in the binary expansion of any irrational number, a contradiction to Lemma 5.3.

We say that a random bit generator passes *the strong next bit test* if the first j consecutive bits do not provide any information about the $(j + 1)$-th bit (where j is an arbitrary natural number).

We show that the binary expansion of irrational numbers passes the strong next bit test.

Theorem 5.5 *The binary expansion of irrational numbers passes the strong next bit test.*

Proof. Let $\alpha_1, \alpha_2, \cdots, \alpha_j$ be the first j bits of the binary expansion of an irrational number and let the rational number $b = 0.\alpha_1\alpha_2\cdots\alpha_j$ be an approximation of all irrational numbers whose binary expansions begin with sequence $\alpha_1, \alpha_2, \cdots, \alpha_j$ (there are infinitely many such irrational numbers). Clearly, those irrational numbers that produce zero as the next bit, belong to the interval $\mathcal{I}_0 = [b, b + 2^{-(j+1)}]$. Let $\gamma = \alpha_1, \alpha_2, \cdots, \alpha_j, 0, \cdots$ be one such irrational number. No matter what the bits following 0 are, the value of this irrational number is greater than b but less than $b + 2^{-(j+1)}$ and hence $\gamma \in \mathcal{I}_0$.

Irrational numbers of the interval $\mathcal{I}_0$ generate the sequence $\alpha_1, \cdots, \alpha_k, 0$. For any irrational number $\beta \in \mathcal{I}_0$ the irrational number $(\beta - b)$ has the expansion $0.\overbrace{00\cdots0}^{j+1}\cdots$ (but not all zeros, since it is an irrational number). Hence the binary expansion of β is of the form:

$$\alpha_1, \alpha_2, \cdots.\alpha_j, 0, \cdots + 0.\overbrace{00\cdots0}^{j+1}\cdots = \alpha_1, \alpha_2, \cdots, \alpha_j, 0, \cdots.$$

This proves that $\mathcal{I}_0$ contains the set of irrational numbers of the form $\alpha_1, \cdots, \alpha_j, 0, \cdots$ Similarly, it can be shown that all irrational numbers which generate one as the next bit belong to the interval $\mathcal{I}_1 = [b + 2^{-(j+1)}, b + 2^{-j}]$, and vice versa.

We assume that the set of real numbers with initial sequence $\alpha_1, \alpha_2, \cdots, \alpha_j$ is uniformly distributed. Then the probability that α_{j+1} is zero, given the knowledge of $\alpha_1, \alpha_2, \cdots, \alpha_j$, is the measure of the set $\mathcal{I}_0$, which is $\frac{1}{2^{j+1}}$. Likewise, the probability that α_{j+1} is one is $\frac{1}{2^{j+1}}$. Hence, the probability of either bit occurring is the same and this completes the proof.

How to Implement a Random Selection of Irrationals?

All truly random sequences have to be among the irrationals. This phenomenon is related to a well-known property of reals. The set of all reals $\mathbb{R}$, can be split into the subset of all rationals, $\mathbb{Q}$, and the subset of all irrationals, $\mathbb{I}$. The cardinality of the rationals, however, is $|\mathbb{Q}| = \aleph_0$ and they are equinumerous with natural numbers. On the other hand, the cardinality of all irrational numbers is $|\mathbb{I}| = 2^{\aleph_0}$. That is, the set $\mathbb{I}$ is equinumerous with the interval [0,1], and as a matter of fact with any interval $[a, b]$; $a, b \in \mathbb{R}, a < b$. Cantor's theorem [54] gives that $\aleph_0 < 2^{\aleph_0}$.

If there is a measure μ which assigns a positive real to a measurable subset from the interval $\mathcal{I} = [0, 1]$ with the condition that $\mu(\mathcal{I}) = 1$, then the measure of any subset of rationals in $\mathcal{I}$ equals zero.

A possible solution is to flip an unbiased coin an infinite number of times. Clearly this is not a practical solution. Moreover, if the selection of irrationals is to be done within

a limited period of time, we need to use a finite number of indices to mark all possible irrationals. Then we can randomly choose an index and use the corresponding irrational. This gives a practical irrational number generator whose security is determined by the size of the index. From an attacker's point of view, who sees polynomially many output bits, this is equivalent to the difficulty of finding the selected irrational number from its approximation. This brings us to the well-known problem of designing pseudo-random bit generators (PBG).

Pseudo-random sequences have to satisfy the following conditions:

1. they have to be generated 'easily',
2. they have to be indexed by a finite seed which is selected randomly and uniformly from all possible values of the seed (or index),
3. they have to pass the next bit test, i.e., it is computationally infeasible to determine the $(j+1)$-th bit from the sequence of j polynomially many bits.

5.1.4 Weak Irrational Numbers

A particular subset of irrational numbers is the set of algebraic irrationals. This subset of irrationals can be efficiently indexed using coefficients of their minimal polynomials. Hence, the resulting sequences are not secure.

An algebraic irrational can be generated by randomly selecting a minimal polynomial $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_d x^d$ and one of the d roots. The polynomial and the root uniquely define the irrational α. The security parameters of such a generator is the pair (d, H) where d is the degree of $f(x)$ and H is the height of $f(x)$, that is, all coefficients a_i $(i = 1, \ldots, d)$ are selected from the set of integers from the interval $[-H, H]$. Kannan, Lenstra and Lovász [92] have shown that given the knowledge of the first polynomially many bits of an algebraic number, that number can be determined (i.e., its defining polynomial can be identified).

They use the LLL-algorithm [104] to efficiently determine the minimal polynomial $f(x)$ of an algebraic irrational α from its approximation $\tilde{\alpha}$. More precisely, their method (the KLL attack) identifies, in polynomial time, the polynomial $f(x)$ given the first $O(d^2 + d \log H)$ bits of the binary expansion of the algebraic number. Since we assume that both parameters d and H are known to a potential cryptanalyst, the binary expansions of algebraic irrationals are not secure pseudo-random sequences.

Observe that if the minimal polynomial is restricted to binomials $f(x) = a_0 + x^d$, the KLL attack can identify the irrational from a sufficiently long sequence of consecutive

bits (which do not necessarily start at the beginning). Since a sufficiently long sequence of consecutive bits pinpoints the unique irrational with its minimal polynomial $g(x) = b_0 + b_1x + b_2x^2 + \cdots + b_dx^d$, the polynomial $g(x)$ can then be reduced to the corresponding binomial $f(y)$ by applying the transformation $x = y + N$, where N is an integer. Note also that if the analysis of bits in the KLL algorithms does not start from the beginning of a number, the parameter H increases. However, this has a negligible impact on efficiency of the algorithm for 'reasonable' values of N.

The approximation described in [92] extends to certain classes of transcendental numbers. Examples of such numbers are: $cos^{-1}(\alpha)$, $sin^{-1}(\alpha)$ and $log(\alpha)$ where α is an algebraic irrational. This follows from [92, Theorem 2.3]. We examine closely the hypothesis of this theorem.

The assumption that there is a complex valued approximable function f which satisfies the uniform Lipschitz condition, is central to the hypothesis of [92, Theorem 2.3]. Examples of such functions are: $sin(z)$, $cos(z)$. Kannan et al. have extended their algorithm to this setting. The [92, Theorem 2.3] states that there is an "efficient" algorithm which takes as input such a function f, parameters (d,H), a complex number $\tilde{\beta}$, and outputs a complex number β which is approximated by $\tilde{\beta}$, and an algebraic number $f(\beta)$ of degree at most d and height at most H. The assertion about the transcendental numbers $cos^{-1}(\alpha)$, etc., follows from this theorem.

5.1.5 Transcendentals Immune to the KLL Attack

It is obvious that algebraic irrationals should be avoided as they can be "easily" identified. The class of transcendentals looks like the only choice. To avoid the KLL attack we should choose transcendental numbers not of the type $f^{-1}(\beta)$, where f and β satisfy the hypothesis of [92, Theorem 2.3]. We will consider two classes of transcendental numbers which seem suitable for cryptographic applications.

Class 1 - Simple exponentiation. Consider irrationals of the form α^β, where α is a positive integer $\neq 0, 1$ and β is a real quadratic irrational. They are known to be transcendental [5]. In particular $2^{\sqrt{2}}$, $2^{\sqrt{5}}$ are transcendental. Moreover, transcendental numbers of the type α^β cannot be determined with the help of [92, Theorem 2.3] as long as α and β are not revealed. Hence we propose to use the binary expansions of numbers of the form α^β, as "secure" pseudo-random sequences. Both α and β are kept secret and create a seed (index) of the pseudo-random number generator.

Class 2 - Composite exponentiation. For our second class we require a theorem of Baker [5]. It is proved that numbers of the form $\alpha_1^{\beta_1} \ldots \alpha_n^{\beta_n}$, such that: $\alpha_i \neq 0, 1$ and β_i are algebraic, and $1, \beta_1 \ldots, \beta_n$ are linearly independent over the rationals, are transcendentals. The simplest numbers of this type have the form $\alpha_1^{\beta_1} \alpha_2^{\beta_2}$, where $\beta_1 = \sqrt{m}$, $\beta_2 = \sqrt{n}$ and m, n are natural non-square numbers such that mn is a non-square. We take these numbers as our second source. It is easily shown that if mn is a non-square, then 1, β_1 and β_2 are linearly independent over the rationals. Hence for such β_i, $\alpha_1^{\beta_1} \alpha_2^{\beta_2}$ is transcendental. An example of these numbers is $2^{\sqrt{2}} 3^{\sqrt{3}}$. Clearly, the numbers $\alpha_1, \ldots, \alpha_n$, $\beta_1, \ldots, \beta_n$ need to be secret and constitute a seed (index) of the pseudo-random number generator.

5.1.6 Encryption Primitives Based on Transcendentals

We can use transcendentals of the form α^β for encryption in at least two ways. In *stream cipher mode*, the binary sequence generated by the expansion of the transcendental is used as a pseudo-random sequence and is XORed bitwise with the binary representation of the message. That is, an n-bit message $M = (m_1, \ldots, m_n)$ is encrypted to $C = (c_1, \ldots, c_n)$ where $c_i = m_i \oplus r_i$ and r_i is the ith bit of the string $R = (r_1, \ldots, r_n)$, generated from the expansion of transcendentals of either Class 1 or 2. To decrypt a message the receiver needs to know the seed of the PBG, which is used to recreate the pseudo-random string R. For a known-plaintext attack, encryption is as secure as R (if the system is used once only). We note that unpredictability of R (passing the next-bit test) is not proven but we do not know of any algorithm which can predict the next bit of the sequence either.

We consider now another approach to encryption. Let the message space be $\mathcal{M} = \mathbb{Z}_N$. The following notation is used. If there are two irrationals γ and δ, then an equation $\gamma = \delta$ means exact equality. We write $\gamma \stackrel{s}{=} \delta$ if $|\gamma - \delta| < 10^{-s}$.

The basic encryption/decryption algorithm is defined as follows. Let $K = (\alpha, \beta)$ be the cryptographic key, where $\alpha \in \mathbb{Z}$ and β is a quadratic irrational from the set $\Psi = \{\sqrt{u} \mid u \in \mathbb{Z}_N\}$ and u is square free. The cryptogram is generated as

$$C \stackrel{s}{=} (M + \alpha)^\beta. \tag{5.1}$$

Cryptograms are positive rationals which are approximations of transcendentals. Both α and β are chosen according to the requirements for Class 1 transcendentals. The security parameter of this scheme is the number of different keys which can be used (or the size of the key space $\mathcal{K}$). The size of the key space, however, relates to the sizes of the spaces

from which the pair α and β is chosen.

In order to decrypt the cryptogram, the reverse operation is applied, that is,

$$M \stackrel{s}{=} C^{\frac{1}{\beta}} - \alpha.$$

Hence, the message M can be recovered from the cryptogram, since it is an integer which can be determined from the s-place approximation of C.

Proposition 5.6 *The encryption scheme defined by equation* 5.1 *has the multiplicative property for a given pair* (α, β) *whenever the the following Diophantine equation has a solution*

$$(M_1 + M_2) + \alpha = 1.$$

Since the messages and the key α are always positive integers, it follows from Proposition 5.6 that the encryption scheme defined by 5.1 is never multiplicative.

5.1.7 Attacks on Class 1 Sources

In a *ciphertext only attack*, the attacker knows only the ciphertext blocks and is required to find the plaintext (or the key). Using the equations which describe the system, the cryptanalyst must derive α and β from the following set of equations in which the M_i's are unknown.

$$\begin{aligned} C_1 &\stackrel{s}{=} (M_1 + \alpha)^\beta \\ C_2 &\stackrel{s}{=} (M_2 + \alpha)^\beta \\ \ldots\ldots &\ldots\ldots \\ C_t &\stackrel{s}{=} (M_t + \alpha)^\beta \end{aligned}$$

This might seem impossible as the number of equations is always smaller than the number of unknowns. However, properties of α^β might help a cryptanalyst to draw some conclusions about the value of the key. In particular, for a sufficiently long sequence of cipher blocks the enemy can use the monotonicity of x^β to order the cipher blocks and use three consecutive values to solve for α, β and M_1. That is, to solve,

$$\begin{aligned} C_1 &\stackrel{s}{=} (M_1 + \alpha)^\beta, \\ C_2 &\stackrel{s}{=} (M_1 + 1 + \alpha)^\beta, \\ C_3 &\stackrel{s}{=} (M_1 + 2 + \alpha)^\beta. \end{aligned}$$

This can be repeated for a number of triplets C_i, C_j and C_k that are likely to correspond to consecutive message blocks.

Next, we outline a *plaintext attack* on the system defined by 5.1. In this situation, the attacker knows both the M_i's and C_i's in the above set of equations.

Let (C_1, M_1) and (C_2, M_2) be a pair of known plaintexts/ciphertexts. Then we have,

$$\begin{aligned} \log C_1 &= \beta(\log(1 + \alpha/M_1) + \log M_1), \\ \log C_2 &= \beta(\log(1 + \alpha/M_2) + \log M_2). \end{aligned}$$

Now, we will assume that $|\alpha/M| < 1$ for any key/message pair (α, M).

We can approximate $\log(1 + \alpha/M_1)$ and $\log(1 + \alpha/M_2)$ as polynomials $P(\alpha/M_1)$ and $Q(\alpha/M_2)$, by taking a sufficient number of terms in the power series expansion of $\log(1 + x)$. With the remainder, a known function of x, we obtain the approximations,

$$\begin{aligned} \log C_1 &\approx \beta(P(\alpha/M_1) + \log M_1), \\ \log C_2 &\approx \beta(Q(\alpha/M_2) + \log M_2). \end{aligned}$$

By dividing these two expressions we can eliminate β. Now one obtains a polynomial in α with known (approximate) coefficients. This can be solved for α and determined exactly, since α is an integer. Having the α, one can check if the approximation to $\log(1 + \alpha/M_1)$ and $\log(1 + \alpha/M_2)$ by polynomials was sufficiently accurate. If not, more terms of the approximation are taken, until the required degree of accuracy is achieved.

This process yields a number of possibilities $\alpha_1, \ldots, \alpha_r$ (roots of a polynomial) for the key. But the correct α can be determined by comparing (C_1, M_1) with respect to the possibilities $\alpha_1, \ldots, \alpha_r$. Once the correct α is known we can obtain an approximation for $\hat{\beta}$ from the pair (C_1, M_1). Now we use the KLL algorithm to determine β from $\hat{\beta}$.

5.1.8 Encryption Based on Class 2 Numbers

For Class 2 numbers we can, without loss of generality, use composite exponentiation of the form $\alpha_1^{\beta_1}\alpha_2^{\beta_2}$. The secret key is $K = (\alpha_1, \alpha_2, \beta_1, \beta_2)$. Encryption is defined by,

$$C \stackrel{s}{=} \alpha_1^{\beta_1} \times (\alpha_2 + M)^{\beta_2}. \tag{5.2}$$

Decryption, firstly, involves the division of the cryptogram C by $\alpha_1^{\beta_1}$, further processing is the same as in the previous case.

Since $\alpha_1^{\beta_1}$ is fixed, the generator is subject to the attacks applicable on Class 1 sources. In particular, observing a long sequence of cryptograms allows the enemy to choose cryptograms that correspond to consecutive plaintexts. In this case, the following equations

have to be solved,

$$\begin{aligned} C_1 &\stackrel{s}{=} K(M_1+\alpha)^\beta, \\ C_2 &\stackrel{s}{=} K(M_1+1+\alpha)^\beta, \\ C_3 &\stackrel{s}{=} K(M_1+2+\alpha)^\beta, \\ C_4 &\stackrel{s}{=} K(M_1+3+\alpha)^\beta. \end{aligned}$$

The above two schemes have a common characteristic – they are subject to the *cluster attack*. This follows from the property that if the distance $d(M, M')$ is small, then for the corresponding cryptograms, $d(C, C')$ tends to be small. The same observation is applicable for the key. To fix this problem, we can apply several encryption iterations each with the form of either 5.1 or 5.2. As a matter of fact, two iterations are enough.

There is however a better solution, which we describe below.

An Alternative Algorithm

For an irrational $\gamma = b_1 b_2 b_3 \ldots$, we denote,

$$\gamma\,|_t = b_1 b_2 \ldots b_t.$$

$\gamma\,|_t$ is a rational, which is created from the irrational γ by truncating all digits after the t-th position. For an irrational $\gamma \in \mathbb{I}$ and any pair of positive integers t and s $(t < s)$, we define $\gamma\,|_t^s$ as,

$$\gamma\,|_t^s = \gamma\,|_s - \gamma\,|_t\,.$$

For example, if $\gamma = a_1 a_2 \ldots$ then,

$$\gamma\,|_2^4 = a_1 a_2 a_3 a_4 - a_1 a_2 = a_3 a_4,$$

That is, $\gamma\,|_2^4$ holds the digits of γ in positions 3 and 4. We consider $\gamma\,|_t^s$ as subsequence of digits from $t+1$ to s. It can also be treated as an integer or a rational (the decimal point can be placed arbitrarily).

Encryption

A third alternative for the encryption algorithm, is to encode the message in a one-to-one way into the minimal polynomial of a random quadratic irrational. Denote this as β_M; so as long as M is not a square and odd integer, β_M is a root of $x^2 - M = 0$. Next select a sufficiently long subsequence $\beta_M\,|_{t_2}^{t_1}$ from β_M. The cryptogram is,

$$C \stackrel{s}{=} \alpha^{\beta_M}|_{t_2}^{t_1}. \tag{5.3}$$

Note that, the subsequence $\beta_M\ |_{t_2}^{t_1}$ must be selected long enough such that the KLL algorithm can pinpoint the unique irrational number β_M from analysing its subsequence. As mentioned in Section 5.1.4, if the analysis of bits in the KLL algorithms does not start from the beginning of a number, the parameter H increases. Hence, the length of the subsequence $\beta_M\ |_{t_2}^{t_1}$ from β_M depends on the value of t_2 (the starting position to select a subsequence).

Decryption

For decryption, the receiver, who knows α, retrieves the sequence $\beta_M\ |_{t_2}^{t_1}$ and uses the KLL algorithm to find the minimal polynomial and the message M.

Note. This method of encryption creates many different cryptograms for the same message if the subsequence $\beta_M\ |_{t_2}^{t_1}$ is selected from different places, so the *cluster* and *approximation* attacks are not applicable.

5.1.9 Efficiency

First, let us consider the problem of generating the expansions of algebraic irrationals. This problem can be restated as the well-known problem of root-finding of a minimal polynomial (the algebraic irrational is a root of some $f(x)$). Newton's method, and its modifications such as the secant method, gives a very efficient algorithm whose time complexity function is $O(n)$ where n is the degree of $f(x)$ and the underlying computer has fixed precision floating-point arithmetic (see any standard book on numerical analysis such as [172],[28],[94]).

We would like, however, to select the precision parameter as an argument of the algorithm. In this setting the best known algorithm, called the *splitting circle method* [146], runs in time bounded by $O(n^3 \log n + an^2) \log(an) \log\log(an)$, where $n = \deg(f(x))$ and a represents the required precision of computations (the expected error has to be smaller than 2^{-s}).

To generate a pseudo-random sequence (of Class 1 or 2), one has to use floating-point arithmetic with arbitrary precision. This facility is available with mathematical packages such as Maple or Mathematica. The requested precision automatically determines the length of the generated pseudo-random sequence. We experimented with Maple and generated transcendentals of both classes. On average, it took 10 minutes to produce a

pseudo-random string of length 30 Kbits. Obviously dedicated algorithms will generate pseudo-random sequences significantly quicker.

5.1.10 Security Considerations

Assume that $\mathcal{A}$ is a subset of positive integers and $\mathcal{B}$ is a subset of algebraic numbers. Class 1 transcendentals are indexed by two functions. The first, $\mathcal{I}_\alpha : \{0,1\}^{n_\alpha} \to \mathcal{A}$, assigns a positive integer $\alpha \in \mathcal{A}$ to a binary string of length n_α. The second, $\mathcal{I}_\beta : \{0,1\}^{n_\beta} \to \mathcal{B}$, assigns an algebraic number $\beta \in \mathcal{B}$ to a binary string of length n_β (or equivalently its minimal polynomial). We can apply the KLL algorithm to identify the transcendental $T = \alpha^\beta$. Note that T is identified if both α and β are found from a sufficiently long sequence of bits of T. Our identification algorithm is the following. First randomly select $\alpha' \in \mathcal{A}$ and calculate

$$\beta' = \log_{\alpha'} T \ .$$

The KLL algorithm is used to determine β'. This attack succeeds if $\alpha' = \alpha$, for then the KLL algorithm will identify the irrational algebraic number from the approximation of β. The complexity of this algorithm is $O(2^{n_\alpha} p_{KLL}(n_\beta))$, where $p_{KLL}(n_\beta)$ is the time complexity function of the KLL algorithm required to identify the algebraic number β.

Proposition 5.7 *The parameters α and β of Class 1 transcendentals can be identified by an algorithm whose time complexity function is*

$$p_{class1}(n) \leq O(2^{n_\alpha - 1} p_{KLL}(n_\beta)) \ .$$

For certain indices $U(\beta) = \alpha^\beta$ is *multiplicative*, i.e., satisfies the equation $U(\beta_1 \cdot \beta_2) = U(\beta_1) \cdot U(\beta_2)$. If β_1 and β_2 are such that $U(\beta_1 \cdot \beta_2) = U(\beta_1) \cdot U(\beta_2)$, which holds whenever $\beta_1 \cdot \beta_2 = \beta_1 + \beta_2$, then an adversary who could determine a polynomial fraction of the key pairs (α, β) of Class 1 numbers from their approximations in polynomial time could also, using the multiplicative property, determine all such key pairs in random polynomial time (Rivest makes this remark about RSA in [137]). The condition under which $U(\beta)$ is multiplicative is the following.

Proposition 5.8 *Let $\beta_1 = \sqrt{m}$ and $\beta_2 = \sqrt{n}$ be two real irrationals, then $U(\beta_1 \cdot \beta_2) = U(\beta_1) \cdot U(\beta_2)$ is equivalent to the solution of the Diophantine equation*

$$(mn)^2 + m^2 + n^2 = 2mn(1 + m + n).$$

An easy argument using Proposition 5.8 gives that the necessary condition for $U(\beta)$ to be multiplicative is that m and n are even positive integers. Hence, we should avoid these indices when using Class 1 transcendentals.

Chapter 6

Threshold Cryptography

This chapter deals with threshold cryptography. More precisely, it concerns threshold decryption and threshold digital signature schemes.

In Section 6.1 we give an introduction to society-oriented cryptography. Threshold cryptography will be considered in Section 6.2. In Section 6.3 we study the ElGamal and RSA based threshold decryption systems. Generalised threshold cryptosystems will be discussed in Section 6.4. In Section 6.5 we present a solution to deciphering a cryptogram in hierarchical groups. Section 6.6 is devoted to the consideration of threshold signature schemes.

6.1 Society-Oriented Cryptography

Groups play an important role in our modern world. Numerous examples of groups (e.g., banks, companies, the board of directors of a company, the administration of a university, and so on) can be found in all countries around the world. Groups may have different structures. Democratic groups usually exhibit a flat internal structure where every member has the same rights, while in hierarchical groups privileges of members depend upon their position in the hierarchy. No matter what structure the group has, a common aspect of almost all groups is that their roles directly relate to the functionality of their organisation. That is, the functional aspects of groups are independent of their members. This is the reason why letters addressed to a group usually start with some well-known and common expressions like "Dear Sir/Madam" or "To whom it may concern" no matter who is in charge.

So, it is desirable that groups can decrypt a ciphertexts or can generate a digital signature on a document. Traditional private-key cryptographic systems and conventional public-key cryptosystems, however, are adequate for the cases that there are two individuals in the system. Cryptographic transformations by a group of participants is

the subject of investigation in *society-oriented cryptography.*

The notion of society-oriented cryptography was introduced by Croft-Harris [42], Boyd [21] and Desmedt [45]. Unlike classical cryptography, society-oriented cryptography allows groups of cooperating participants to carry out cryptographic transformations. That is, society-oriented cryptography requires distributing the power of performing a cryptographic transformation among a group of participants such that only designated subsets, the so called authorised sets of the group, can perform the required cryptographic operation but unauthorised sets cannot do so.

6.1.1 Implementation Consideration

In general, cryptographic keys are not one-time secrets and they must be kept secret and reused many times. Hence, the trivial implementation of a society-oriented cryptographic system which requires the concatenation of a secret sharing scheme with single user cryptography is usually unacceptable as an authorised set of participants must first recover the cryptographic key, which compromises the system.

Ideally, one would require collaborating participants to apply their shares and perform the required cryptographic transformation at the same time. These results, also called *partial results*, are then sent to the *combiner* who calculates the final result. Note that, in some society-oriented cryptographic systems, cooperating participants first need to apply a function on their shares, and produce their *modified shares*, then use the modified shares to generate partial results. Since groups can exhibit different structures and richer relations among participants, implementations of such services become more complex for groups than for individuals.

It is worth mentioning that almost all implementations of society-oriented cryptographic systems are based on public-key cryptosystems such as the RSA [136], the El-Gamal [55], or the Diffie-Hellman public-key distribution system [53]. There has been no attempt to base threshold cryptographic algorithms on symmetric cryptosystems like DES [122]. We observe two reasons for this.

1. Most existing implementations of society-oriented cryptographic systems are based on the concept of a homomorphic secret sharing scheme. The mathematical structure of the public-key cryptosystems are adequate regarding the homomorphic property, but the nonlinearity property of the private-key cryptosystems does not allow definition of such a homomorphism.

2. Although the necessity of the homomorphism property to implement a society-oriented cryptographic system has not been proved, symmetric cryptosystems are not suitable for this purpose. Assume that a bank wishes to use a symmetric cryptosystem based threshold system. In order to communicate with customers, it must possess separate cryptographic keys corresponding to different customers, otherwise, customers can read the messages communicated between the bank and other customers.

6.2 Threshold Cryptography

A particularly interesting class of society-oriented cryptographic transformations includes threshold cryptographic transformations with a group of n members. Examples of the need for threshold cryptographic transformations include the signature by a majority in a parliament, signing a document in a bank (e.g., when any two-out-of three senior tellers are allowed to authenticate an EFT), deciphering a cryptogram addressed to a company (when the intended receivers might have shared responsibility), and so on.

In a threshold cryptographic system, the groups authorised to perform a required cryptographic transformation consist of all subsets of t (or more) participants. Such schemes are called t-out-of-n threshold cryptographic systems or simply (t, n) threshold systems. So, the access structure of a (t, n) threshold cryptographic system can be expressed as:

$$\Gamma = \{\mathcal{A} \subseteq \mathcal{P} \mid |\mathcal{A}| \geq t\},$$

where t, the so called *threshold parameter*, is an integer, $t \leq n$. More precisely, in a (t, n) threshold system the power to perform a cryptographic transformation is divided among n members of a group, such that the following conditions are satisfied:

- any set of t or more participants can perform the required cryptographic transformation;
- any set of $t - 1$ or fewer participants are not able to perform the required cryptographic transformation successfully;
- neither the group secret key nor the shares of collaborating participants can be derived from the group/partial cryptographic transformation.

Threshold cryptography was first, independently, introduced by Croft-Harris [42], Boyd [21] and Desmedt [45]. In [42] a (t, n) authorisation system was proposed. This

system, however, is insecure for many applications. In [21] a $(2, n)$ threshold RSA signature scheme was introduced. Although the threshold parameter of this system is fixed, it is not ideal, that is, the share assigned to each participant is larger than the secret key of the underlying RSA cryptosystem (indeed, no ideal threshold RSA cryptosystem has been proposed in the literature). In [45] different kinds of groups and their cryptographic needs are discussed. It is shown that in many groups the power to decrypt a cryptogram must be shared. The threshold decryption system proposed in [45], however, was based on mental games [75] and is therefore interactive and impractical.

6.3 Threshold Decryption

Messages are frequently dispatched to a group of people, e.g. a company. Let a group require that any t out of n members be cooperatively able to read the message. In [45] it has been shown that such a (t, n) threshold decryption does not satisfy different needs of a group at different times. For example, some messages are so urgent that every member of the group must be able to read them. On the other hand, there are confidential messages in which the cooperation of a majority of group members is required to decipher the cryptogram. There are many other needs that can not be accommodated by a simple threshold policy with a fixed threshold parameter. However, constructing different threshold systems to satisfy different needs of a group at different times requires assigning different public keys to the group, which is not a practical solution. Since the sender is the only one who knows the nature of the message, Desmedt suggests [45],

> "the ideal solution would be that the sender of the information can add a few bits to the message such that these few bits enable the group to access the information using the method he has decided and no other one. This means the group has only to publish one public key and that it can be used in cases that all members have to access the information simultaneously as well as in case of emergency and all other possible cases (depending on what the sender decides)."

However, no such threshold cryptosystem has been constructed so far. The first attempts on designing threshold decryption systems [45] were based on mental games [75] and therefore completely impractical and interactive. The first practical and non-interactive threshold decryption system was introduced by Desmedt and Frankel [48]. Their scheme applies the ElGamal [55] public-key cryptosystem.

6.3.1 Threshold ElGamal Decryption

Desmedt and Frankel [48] proposed a method in which an organisation can employ a (t, n) threshold decryption system. That is, every message intended for the group can be deciphered if and only if a set of t or more members of the organisation want to read the message. Their proposed threshold system utilises the ElGamal[55] cryptosystem and uses the Shamir (t, n) threshold security policy to determine who can read a message intended for the organisation.

The scheme is implemented in such a way that the cooperating participants do not need to reconstruct the secret. Instead, they perform the decryption algorithm (using their *modified* shares) and transmit their partial results to a designated *combiner*. If the cooperating participants belong to an authorised set, $\mathcal{A}$, that is, $|\mathcal{A}| \geq t$ then the combiner can obtain the correct plaintext.

In order to illustrate the implementation method of this threshold decryption, let K and $k = g^K$ be the secret and public keys of an ElGamal cryptosystem and let $C = (g^r, Mk^r)$ be the cryptogram of a message M. (see Section 1.5.2 for a description of the ElGamal public-key cryptosystem). We assume a secret sharing scheme can distribute the secret K among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n participants, such that for every set $\mathcal{A}$ $(|\mathcal{A}| \geq t)$,

$$\sum_{P_i \in \mathcal{A}} s_i = K \pmod{\varphi(p)},$$

where s_i is the modified share of participant P_i from the secret K. Each participant $P_i \in \mathcal{A}$ computes $(g^r)^{s_i}$ and transmits its partial decryption to the combiner. The combiner computes,

$$\prod_{P_i \in \mathcal{A}} (g^r)^{s_i} = (g^r)^{s_1 + \cdots + s_t} = (g^r)^K = k^r,$$

and hence recovers the message.

However, considering equation (3.2), this can be computed if $\varphi(p)$ is a prime number. Since, for any prime number p, $p > 3$, the integer $\varphi(p) = p - 1$ is not prime, Desmedt and Frankel [48] suggest performing the ElGamal system in $GF(2^\ell)$ for a large enough value of ℓ such that $\varphi(p) = p' = 2^\ell - 1$ is a *Mersenne prime*.

Set-up Phase:

The dealer selects at random a polynomial of degree $t - 1$ and applies the Shamir (t, n) threshold scheme (over $\mathbb{Z}_{p'}$, where $p' = 2^\ell - 1$) in order to distribute the secret K among the shareholders. That is;

1. $\mathcal{D}$ chooses n distinct and non-zero elements of $\mathbb{Z}_{p'}$, denoted $x_1, \ldots, x_n$ and sends x_i to P_i via a public channel,

2. $\mathcal{D}$ chooses (independently at random) $t-2$ elements of $\mathbb{Z}_p$, denoted $a_1, \ldots, a_{t-2}$ and chooses (at random) an element $a_{t-1} \notin \mathbb{Z}_{p'} - \{0\}$. Then he forms the polynomial

$$f(x) = K + \sum_{i=1}^{t-1} a_i x^i,$$

3. for every P_i, $1 \leq i \leq n$, the dealer computes the initial share s_i', where

$$s_i' = f(x_i) \pmod{p'},$$

4. $\mathcal{D}$ gives (in private) s_i' to participant P_i.

Shared Decryption:

Participants of an authorised set $\mathcal{A}$, ($|\mathcal{A}| \geq t$) first compute their modified shares s_i using,

$$s_i = \sum_{i=1}^{t} s_i' \prod_{\substack{j=1 \\ j \neq i}}^{t} \frac{0 - x_j}{x_i - x_j}.$$

According to equation (3.2), for every authorised set $\mathcal{A}$ of at least t participants

$$K = \sum_{P_i \in \mathcal{A}} s_i \pmod{p'}$$

and thus the system works as it was described.

Since in any part of the system no share or the group secret key is disclosed, participants can use their shares many times without risk of the security of the system being compromised.

Remark 4 *With contrast to the first part of this thesis, throughout this part we assume that the polynomial $f(x)$ in the Shamir (t, n) threshold secret sharing scheme is of degree exactly $t-1$. Otherwise, every set of $t-1$ collaborating participants can easily realize that they can perform the required group transformation and therefore they can break the threshold system.*

In [131] it is shown that there is no need for a trusted party (dealer) to distribute the shares of the key among the participants. However, both threshold ElGamal cryptographic systems [48] and [131] do not work without the help of a trusted combiner. That

is, a dishonest combiner can choose a message of his choice claiming that the message has been extracted from the cryptogram. Since there is no way to control the relationship between the retrieved message and the cryptogram the participants have to accept the message.

Here we show that even with the help of trusted combiner their cryptosystems are subject to the following attack. Let a dishonest participant $P_j \in \mathcal{A}$ contribute with a modified share $s_j + \alpha$ and transmit a partial decryption $(g^r)^{s_j+\alpha}$ to the combiner. The combiner computes,

$$\prod_{P_i \in \mathcal{A}} (g^k)^{s_i} = (g^r)^{\sum_{P_i \in \mathcal{A}} s_i} = (g^r)^{K+\alpha} = k^r \times (g^r)^{\alpha}$$

and, using the multiplicative inverse of the result, calculates the message as:

$$M_f = \frac{Mk^r}{(g^r)^{\alpha} k^r} = \frac{M}{(g^r)^{\alpha}}.$$

However, P_j can retrieve the genuine message using,

$$M_f \times (g^r)^{\alpha} = M.$$

In order to avoid these problems, let $h(\cdot)$ be a one-way and collision free hash function, such that $h(\cdot) = GF(p)$. The sender, who wants to communicate a message M, first calculates $h(M)$ and then constitutes the cryptogram $(g^{h(M)}, Mk^{h(M)})$. One can see that the structure of the system and the cryptographic algorithms (for both encryption and decryption) are similar to the original ElGamal system (the random generation part is replaced with the hash value of the message itself). This modified ElGamal system, however, has the following properties:

- The sender does not need to take care about generating a fresh random exponent for each message, since h is collision free.
- The system provides verifiability of the recovered message. The first entity of the cryptogram must be equal to $g^{h(M)}$. Since h is collision free, an opponent cannot find a $M_f \neq M$, such that $g^{h(M)} = g^{h(M_f)}$.

6.3.2 Threshold RSA Decryption

In [59] Frankel proposed a RSA based shared decryption system that allows *n-out-of-n* shareholders to decrypt a cryptogram, however, implementation of a (t, n) threshold RSA system is more difficult.

In [48] Desmedt and Frankel's attempts to implement threshold decryption for the RSA system (similar to that proposed for the ElGamal system) failed. The difficulty of implementation of the RSA threshold cryptosystem is that of how to apply Lagrange interpolation over $\mathbb{Z}_{\varphi(N)}$, while $\varphi(N)$ is kept secret. In [49], Desmedt and Frankel have shown a solution to this problem.

Desmedt-Frankel's Scheme

The RSA [136] applies a composite integer N which is the product of two large and safe primes p and q, that is, $p = 2p' + 1$ and $q = 2q' + 1$. In the original RSA system $\varphi(N) = (p-1)(q-1)$. However, it is suitable to use $\lambda(N) = 2p'q'$ instead, where λ is the Carmichael function, that is, the exponent of $\mathbb{Z}_N^*(\cdot)$. Let k and K be the public and secret keys, respectively (e.g. $k \times K \equiv 1 \bmod \lambda(N)$).

In order to construct a (t, n) threshold RSA decryption, the dealer distributes the secret key K among n members of the group in such a way that for any authorised set $\mathcal{A}$, $|\mathcal{A}| \geq t$, the modified shares of the participants satisfy the following,

$$\sum_{P_i \in \mathcal{A}} s_i = K \bmod \lambda(N).$$

That is, the Shamir scheme, and therefore the Lagrange interpolation formula needs to be performed in $\mathbb{Z}_{\lambda(N)}$. But $\lambda(N)$ is even and thus not all $(x_i - x_j)$ have an inverse, modulo $\lambda(N)$ (see equation 3.2). To solve this problem, the dealer selects a random polynomial, $f(x)$, of degree $t-1$, such that all its coefficients are even and the constant term is equal to $K - 1$ (in RSA K is odd). The dealer gives, in private, the initial share,

$$s_i' = \frac{f(x_i)/2}{\left(\prod_{\substack{P_j \in \mathcal{P} \\ j \neq i}} (x_i - x_j)\right)/2} \pmod{p'q'}$$

to participant P_i, $1 \leq i \leq n$.

Applying this technique, in the secret reconstruction phase no inverse has to be calculated by an authorised set $\mathcal{A}$, since

$$f(x) = \sum_{P_i \in \mathcal{A}} s_i' \prod_{\substack{P_j \notin \mathcal{A} \\ P_j \in \mathcal{P}}} (x_i - x_j) \prod_{\substack{P_j \in \mathcal{A} \\ j \neq i}} (x - x_j) \pmod{2p'q'}.$$

Hence, the modified shares of each participant, $P_i \in \mathcal{A}$, will be given by,

$$s_i = \prod_{\substack{P_j \notin \mathcal{A} \\ P_j \in \mathcal{P}}} (x_i - x_j) \prod_{\substack{P_j \in \mathcal{A} \\ j \neq i}} (0 - x_j) \tag{6.1}$$

However, in this method the value $K-1$ was shared among the group members. That is, for every authorised set $\mathcal{A}$ the collaborating participants have,

$$\sum_{P_i \in \mathcal{A}} s_i = K - 1 \bmod \lambda(N).$$

Shared Decryption:

In order to decipher a cryptogram $C = M^k \pmod N$, participants of an authorised set $\mathcal{A}$ compute their modified shares s_i ($P_i \in \mathcal{A}$) and transmit their partial decryptions, C^{s_i}, to the combiner. The combiner recovers the message M, using

$$C \cdot \prod_{P_i \in \mathcal{A}} C^{s_i} = C \cdot C^{\sum_{P_i \in \mathcal{A}} s_i} = C \cdot C^{K-1} = C^K = M.$$

6.4 Generalised Threshold Cryptosystems

So far we have discussed threshold decryption, that is, all authorised sets have the same cardinality. In [100], Laih and Harn proposed a RSA based generalised shared decryption system that allows any designated sets of participants to perform the required group cryptographic transformation. However, Langford [103] has shown that this scheme is not secure for some access structures. Desmedt [47] has also pointed out that for some access structures the shareholders need exponential computing power.

A widely studied general access structure is the hierarchical structure. Hierarchical access structures have been the subject of investigation by several authors (see for example, [171] and [32]). In [67], we proposed a cryptographic system for hierarchical groups. In that system we examined the simplest case of hierarchical structures in which there was only one participant in each level of the hierarchy. In the following, we present a cryptosystem for general hierarchical groups.

6.5 Cryptosystems for Hierarchical Groups

The concept of threshold cryptography was originally formulated by Desmedt [46] for "democratic" groups where every participant has equal rights in performing cryptographic operations. There were numerous attempts to generalise society-oriented cryptographic systems for an arbitrary access structure. To illustrate the problem consider the following quotation from Desmedt [46].

> "Suppose that the group decided that the messages are first to be accessed by the supervisor and then afterwards by all other members at the same

moment. The group publishes the corresponding public key. The question now is: *does the publication of this key and of the encryption method, reveal what the decision of the group is*? If that would be the case, then everybody knows which hierarchy the group has, or more generally knows which kind of society that corresponds with the group."

Cryptographic operations in hierarchical groups can be done in two different ways: from top to bottom or from bottom to top. "Top-down" cryptographic operations have to be performed after "the go-ahead" is given by the higher level. The permission (or denial) for the lower level may depend upon the result of the operation performed by the higher level. "Bottom-up" cryptographic operations permit the delegation of a specific right from lower levels to the top. The top level successfully executes the cryptographic operation when all lower levels have prepared their partial results.

6.5.1 Top-down Cryptography in Hierarchical Groups

Here we investigate the problem of a secure top-down information flow. The proposed cryptosystems are designed in such a way that the information flow can be stopped by higher level participants. The proposed "top-down" hierarchical cryptosystems are based on the ElGamal and RSA cryptosystems.

Let us consider the case when a cryptogram is broadcast to all participants of a hierarchical organisation. A top-down cryptosystem should allow the participant on the top of the hierarchy to retrieve the plain-text message from the cryptogram. Once the higher level participants have decrypted the cryptogram, they may allow the lower level participants to decrypt the cryptogram by sending a suitable permission – a *go-ahead ticket.*

6.5.2 The model

We assume that the hierarchy of participants is described by a tree structure, where $\mathcal{P}(0)$ is the set of the highest level participants. Every participant $P_j \in \mathcal{P}(0)$ is the head of the group $\mathcal{P}(1; j)$. In general each participant of the group $\mathcal{P}(i; j_1, \ldots, j_i)$ is the head of the group $\mathcal{P}(i+1; j_1, \ldots, j_i, j_{i+1})$. Any participant can be a member and the head of a single group. It is easy to see that each path in the hierarchy is uniquely identified by a suitable sequence of heads (supervisors). We also assume that all participants of the group defined on the i-th level have the same power, i.e. we are dealing with threshold access structures.

Definition 6.1 *A top-down cryptographic system consists of a single trusted dealer and a collection of combiners. Each combiner on the i-th level acts on behalf of a specific group.*

- *The trusted dealer ($\mathcal{D}$) sets up the system. $\mathcal{D}$ chooses both the encryption and decryption keys. The encryption key $k \in \mathcal{K}$ is public. The decryption key $K \in \mathcal{K}$ is secret and is used to calculate shares. Next, $\mathcal{D}$ determines shares for all participants (of all groups of the hierarchy) according to the following function:*

$$f_{\mathcal{D}}^{(i)} : \mathcal{K} \times \mathcal{P}(i; j_1, \ldots, j_i) \to \mathcal{S} \times \mathcal{T}, \quad i = 0, 1, \cdots, \ell - 1$$

for the groups on the i-th levels, and

$$f_{\mathcal{D}}^{(\ell)} : \mathcal{K} \times \mathcal{P}(\ell; j_1, \ldots, j_\ell) \to \mathcal{S}$$

for the lowest level, where $\mathcal{T}$ denotes the set of all go-ahead tickets. All secret sharing schemes used for the groups are designed for threshold access structures. The value of the threshold may be different for each group. The dealer sets up all static secret elements which do not change throughout the life-time of the system. However, participants may need to calculate their modified shares in order to perform the required cryptographic transformations.

- *The combiner of the top level group (on 0-th level) collects partial results from participants of the group $\mathcal{P}(0)$ and applies the function:*

$$f_C^{(0)} : f_0(\mathcal{C}, \mathcal{S}_1) \times \cdots \times f_0(\mathcal{C}, \mathcal{S}_t) \to M \times f_{tic}^0(\mathcal{C}, \mathcal{T}),$$

where $f_{tic}^0(\bullet, \bullet)$ is a function which assigns a go-ahead ticket for a cryptogram. If the number of participants who have provided their shares to the combiner equals or exceeds the threshold parameter, $f_C^{(0)}$ generates the plain message and a valid go-ahead ticket which may be given to the lower level groups.

- *A combiner on the i-th level collects partial results of its group $\mathcal{P}(i; j_1, \ldots, j_i)$, a go-ahead ticket from the higher level and uses the function:*

$$f_C^{(i)} : f_i(\mathcal{C}, \mathcal{S}_1) \times \cdots \times f_i(\mathcal{C}, \mathcal{S}_t) \times f_{tic}^{i-1}(\mathcal{C}, \mathcal{T}) \to M \times f_{tic}^{i+1}(\mathcal{C}, \mathcal{T}), \quad i = 1, 2, \cdots, \ell - 1$$

to retrieve the message and compute the ticket for the lower level groups. The message and the ticket are valid only when the number of contributing participants equals or exceeds the threshold of the level. The combiner on the lowest level, ℓ,

retrieves the message (using partial results of the participants and the go-ahead ticket) as:

$$f_C^{(\ell)} : f_\ell(\mathcal{C}, \mathcal{S}_1) \times \cdots \times f_\ell(\mathcal{C}, \mathcal{S}_t) \times f_{tic}^{\ell-1}(\mathcal{C}, \mathcal{T}) \to M.$$

Go-ahead tickets may be computed by their immediate predecessors – groups one level above the current ones. More specifically, for the group $\mathcal{P}(i; j_1, \ldots, j_i)$, only the group $\mathcal{P}(i-1; j_1, \ldots, j_{i-1})$ can produce the valid ticket. This system is called a *hierarchical cryptosystem with sequential go-ahead.*

The alternative to the above are hierarchical systems in which any higher level group belonging to a single path can authorise the lower level groups by passing go-ahead tickets to them. So the group $\mathcal{P}(i; j_1, \ldots, j_i)$ can obtain a valid ticket from one of the following groups: $\mathcal{P}(i-1; j_1, \ldots, j_{i-1})$, ..., $\mathcal{P}(1; j_1)$, and $\mathcal{P}(0)$. These systems are called *hierarchical cryptosystems with jumping go-ahead.*

Next we will discuss two implementations of hierarchical cryptosystems. The first is based on the ElGamal cryptosystem. The second uses the RSA system.

6.5.3 ElGamal Based Hierarchical Cryptosystem

Assume that we have a group of participants $\mathcal{P} = \{\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_\ell\}$. $\mathcal{P}_0$ is on the top of the hierarchy. $\mathcal{P}_1$ belongs to the second level, $\mathcal{P}_2$ to the third, etc. $\mathcal{P}_\ell$ is at the bottom of the hierarchy. Let $|\mathcal{P}_i| = n_i$ and every $t_i \leq n_i$ participants of level i be an authorised set, that is, they can decipher the cryptogram if they have received a go-ahead ticket from the higher level. Although in hierarchical structures, in general, $t_i < t_j$ and $n_i < n_j$ (for all $i < j$), our proposed cryptosystem does not require these conditions to be satisfied.

System Set-up phase:

Consider an ElGamal threshold scheme, as described in Section 6.3.1, with parameters p, g, the secret key K and the public key $k = g^K$.

1. The dealer, $\mathcal{D}$, applies a $(\ell+1, \ell+1)$ Karnin-Green-Hellman threshold secret sharing scheme and generates $\ell + 1$ partial keys, K_i, such that,

$$K = \sum_{i=0}^{\ell} K_i \pmod{p'}.$$

2. For every level i, $0 \leq i \leq \ell$, the dealer applies the Shamir (t_i, n_i) threshold scheme (as described in Section 6.3.1) and generates the shares of K_i.

3. For every level i, $0 \leq i < \ell$, the dealer applies the Shamir (t_i, n_i) threshold scheme (as described in Section 6.3.1) and generates the shares of $\bar{K}_i = \sum_{j=i+1}^{\ell} K_i \pmod{p'}$.

4. The dealer publishes p, g and k as the public parameters of the system. The secrets (degenerate shares) are distributed via a secured channel to the participants. That is, a participant of level i, $0 \leq i < \ell$ receives two shares (corresponding to K_i and $\bar{K}_i$), while participants of level ℓ receives only one share.

Note. According to the homomorphism property of the Shamir secret sharing scheme, for any participant of level $0 \leq i < \ell$ the sum of his shares gives a share of $K_i + \bar{K}_i \pmod{p'}$. That is, the sum of the shares of participants of the highest level gives a share of the secret K, but the sum of the shares of a participant of level one gives a share of $K - K_0$.

Encryption:

The sender, who wants to communicate a message, $M \in GF(p)$, follows the steps of the ElGamal cryptosystem:

1. he selects a random element $r \in GF(p)$,

2. computes the cryptogram $C = (C_1, C_2) = (g^r, Mk^r)$ for the message $M \in \mathcal{M} = GF(p)$,

3. finally, the sender dispatches the cryptogram C to the group.

Shared Decryption:

No level, except the highest level whose participants' shares can determine the secret K, is able to decipher the cryptogram. So, the highest level deciphers the cryptogram:

- each participant, P_j, of an authorised set, $\mathcal{A} \in \mathcal{P}_0$, computes his modified share, s_{0_j} and $\bar{s_{0_j}}$, corresponding to K_0 and $\bar{K}_0$,

- each participant, P_j, generates his partial decryptions, $(g^r)^{s_{0_j}}$ and $(g^r)^{\bar{s_{0_j}}}$, and sends them to the combiner,

- since the set of collaborating participants is greater than or equal to t_0, the combiner can compute

$$\prod_{P_j \in \mathcal{A}} (g^r)^{\bar{s_{0_j}}} = (g^r)^{\bar{K}_0},$$

and, to generate a go-ahead ticket,

$$T_0 = \prod_{P_j \in \mathcal{A}} (g^r)^{s_{0_j}} = (g^r)^{K_0}.$$

- the combiner computes,

$$(g^r)^{\bar{K}_0} \times T_0 = (g^r)^K$$

and hence retrieves the message M from the cryptogram.

The highest level may want the lower level, level one, to be able to read the message. So, the highest level sends the go-ahead ticket, T_0, to its lower level. Participants of an authorised set, $\mathcal{A} \in \mathcal{P}_1$, do as follows:

1. each participant follows a similar algorithm which was performed at the highest level,

2. thus, the combiner of level one can generate two partial decryptions

$$(g^r)^{\bar{K}_1} \quad \text{and} \quad (g^r)^{K_1},$$

3. since the set of collaborating participants consists of at least t_1 participants, the combiner can generate a go-ahead ticket

$$T_1 = T_0 \times (g^r)^{K_1},$$

and

$$(g^r)^{\bar{K}_1} \times T_1 = (g^r)^K,$$

which enable it to retrieve the message M.

It is not difficult to see that any higher level can issue a go-ahead ticket for its lower level. Note that, a valid go-ahead ticket $T_i = T_{i-1} \times (g^r)^{K_i}$. That is, the shares of K_i are used to create a valid go-ahead ticket for one level below and the shares of $\bar{K}_i$, along with the ticket, allow $\mathcal{P}_i$ to decrypt the cryptogram.

6.5.4 RSA Based Hierarchical Cryptosystem

Assume, as before, that we have a group of participants $\mathcal{P} = \{\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_\ell\}$. $\mathcal{P}_0$ is on the top of the hierarchy. $\mathcal{P}_1$ creates the second level, $\mathcal{P}_2$ the third, etc. $\mathcal{P}_\ell$ is at the bottom of the hierarchy. Level i consists of n_i participant and the requirement is that at least t_i (out-of-n_i) participants need to collaborate in order to perform the proper decryption.

System Set-up phase:

Let $N = pq$ where p, q are two safe primes, that is $p = 2p' + 1$ and $q = 2q' + 1$ (p' and q' are large and distinct primes). The dealer ($\mathcal{D}$) who sets up the system, selects p, q and a random integer k such that k and $\lambda(N)$ are coprime ($\lambda(N)$ is the least common multiple of $(p-1)$ and $(q-1)$, and therefore, is equal to $2p'q'$). Next $\mathcal{D}$ publishes k and N as public parameters of the system, while the values of p, q and K (K has the property that $k \times K \equiv 1 \pmod{\lambda(N)}$) are kept secret. The dealer also selects ℓ random even integers $K_1, \cdots, K_\ell$ and solves the equation $K_0 + K_1 + \cdots + K_\ell = K \bmod \lambda(N)$ for the unknown integer s_0 (note that only K_0 is odd). Finally $\mathcal{D}$ distributes K_i among the participants of level i, $0 \leq i \leq \ell$ and distributes $\bar{K}_i = \sum_{j=i+1}^{\ell} K_j$ among the participants of level i, $0 \leq i < \ell$.

The secret sharing scheme is as described in Section 6.3.2. Since K_0 is odd, $K_0 - 1$ will be shared among corresponding participants (see Section 6.3.2). Other partial keys K_i ($1 \leq i \leq \ell$) and $\bar{K}_i$ ($0 \leq i < \ell$) can be shared without reducing by one, since they are even values.

Encryption:

As in the original RSA system, in order to communicate a message, M, the sender computes the cryptogram $C = M^k \bmod N$ and dispatches it to the group.

Decryption:

Similarly to the ElGamal based hierarchical cryptosystem, the highest level performs as follows:

- each participant of an authorised set, $\mathcal{A} \in \mathcal{P}_0$, sends his partial decryption to the combiner,
- the combiner generates $C^{\bar{K}_0}$ and the ticket T_0,
- since the set of collaborating participants consists of at least t_0 participants, the combiner can compute $C^{K_0} \times C^{\bar{K}_0} = C^{K-1}$ and hence decipher the cryptogram as $C^{K-1} \times C = M \pmod{N}$.

Similarly to the ElGamal based hierarchical cryptosystem, the highest level can issue a permission (go-ahead ticket) to enable the lower level to read the message. Since this system works similarly to the ElGamal based cryptosystem, we do not describe it in detail.

6.5.5 Assessment of the System

In order to assess the proposed system, we briefly discuss the security and efficiency of the system.

Security

The proposed cryptosystem for hierarchical groups applies the ElGamal and the RSA threshold cryptosystems. Since the encryption and decryption algorithms do not give any information more than the corresponding threshold cryptosystems, we can claim that the proposed hierarchical cryptosystems are as secure as their underlying threshold cryptosystems.

Efficiency

Although from a security point of view the proposed hierarchical cryptosystems are equivalent to their underlying threshold cryptosystems, from an efficiency point of view they are not equivalent. We observe two major points.

1. In the underlying threshold cryptosystems each participant is responsible for keeping only one secret share. In our hierarchical cryptosystem, however, each participant, except participants of the lowest level – level ℓ, needs to keep two secret shares.

2. In the proposed hierarchical cryptosystems the amount of computation that is required to decipher a cryptogram is about two times the amount of computation in the corresponding threshold cryptosystems.

It is worth mentioning that the inefficiency of our hierarchical system is due to the generation of go-ahead tickets which enable the higher levels to control the flow of information in the group. In fact, if level i does not agree to give permission to its lower level to read the message, each participant first adds his shares then applies that share to the algorithm. Due to the homomorphism property of Shamir threshold system, the result is correct.

6.6 Threshold Signature

A digital signature is an integer, issued by a signer, which depends on both the signer's secret key and the message being signed. In conventional cryptosystems the signer is a

single user. However, the process of signing may need to be shared by a group of people. For example, a bank may require that any transaction is signed by at least two clerks. The first attempts at designing a shared signature are due to Boyd [21].

In general, we assume that the signature is generated by a group of t people (instead of an individual). Sometimes the t authorised co-signers may not be available at the time when the signature is required. To guard against such a failure of the signature generation, an organisation may choose a larger group of n individuals and allow every t out of n ($t < n$) to sign the message. Such a shared generation of signatures is called a (t, n) *threshold signature.* For the case that $t = n$, the shared generation is called a (n, n) *multisignature.*

6.6.1 Threshold RSA Signature

In the RSA cryptosystem the deciphering algorithm is identical to signing. So to avoid repetition, we refer the reader to Section 6.3.2. However, the proposed (t, n) threshold RSA signature suffers the following shortcoming:

1. In [105] Li, Hwang and Lee have discussed that a (t, n) threshold signature scheme does not only require that less than t users must not be able to generate a correct signature, but also a particular set of t participants should not be forged by another set of t participants. They have pointed out, however, that the Desmedt-Frankel's [49] (t, n) threshold RSA signature is subject to the *conspiracy attack.* That is, if t (or more) participants conspire, then the group secret key and all participants' shares will be revealed. Once the shares are revealed, the set of collaborating participants can impersonate another set of shareholders to sign a message without holding the responsibility of the signatures, and can deny having signed a message though in fact they have signed it.

2. Since $\lambda(N)$ is unknown, the cooperating participants compute their modified shares (equation 6.1) in $\mathbb{Z}$. Although, due to exponentiation, the scheme works properly, the size of each modified share s_i is about t times the size of the secret K. So the system is not very practical for a large group of co-signers (e.g. an electronic election with millions of participants).

3. There is no proof of the security of this threshold system.

In Section 6.6.3 we will study a particular RSA type shared signature scheme, which is due to Boyd [21].

6.6.2 Threshold ElGamal-Type Signatures

Implementation of threshold and multisignature schemes based on the ElGamal and its variant (the DSS) signature schemes was the subject of investigation by several authors (see, for example, [77], [106], [80], [102] and [129]). In the following we give a brief description of the Harn [77] (t, n) threshold digital signature. This system utilises the Shamir threshold scheme and a modified version of the ElGamal signature. A dealer or trusted key authentication centre (KAC) selects the system parameters as,

- p, a prime modulus, where $2^{511} < p < 2^{512}$;
- q, a prime divisor of $p-1$, where $2^{159} < q < 2^{160}$;
- a polynomial $f(x) = K + a_1 x + \cdots + a_{t-1}x^{t-1} \bmod q$, where K is the secret and a_i are random integers in $\mathbb{Z}_q$;
- $g = h^{(p-1)/q} \pmod p$, where $h \in GF(p)$ is any integer such that $g > 1$ (g is an element of order q in $GF(p)$);

The parameters p, q and g are public, but K and $a_1, \ldots, a_{t-1}$ are secret values.

The KAC uses the Shamir (t, n) threshold secret sharing scheme to share the secret K among the set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of participants. That is, it assigns $s_i = f(x_i)$ to participant P_i $(1 \leq i \leq n)$. It also publishes $k = g^K \bmod p$ as the group public key and $k_i = g^{f(x_i)} \bmod p$ as the public key of participant P_i $(1 \leq i \leq n)$.

In order to sign the message M, each participant P_i of an authorised set $\mathcal{A}$ $(|\mathcal{A}| \geq t)$ chooses a random value r_i' $(1 < r_i' < q-1)$ and computes a public value r_i, as

$$r_i = g^{r_i'} \pmod p$$

and makes r_i publicly available through a broadcast channel. Once all r_i are available, collaborating participants of the set $\mathcal{A}$ compute,

$$r = \prod_{P_i \in \mathcal{A}} r_i \pmod p.$$

Participant $P_i \in \mathcal{A}$ uses his secret $f(x_i)$ and his chosen *one-time* random r_i', to sign the message M as,

$$\sigma_i = f(x_i) \times M \times \left(\prod_{\substack{P_i \in \mathcal{A} \\ i \neq j}} \frac{-x_j}{x_i - x_j} \right) - r_i' \times r \pmod q.$$

and transmits his partial signature to the combiner. The combiner first verifies the correctness of the partial signature of participant P_i using,

$$(k_i^M)^{\left(\prod_{\substack{P_i \in \mathcal{A} \\ i \neq j}} \frac{-x_j}{x_i - x_j}\right)} \stackrel{?}{=} r_i^r \times g^{\sigma_i} \pmod{p}.$$

If the equation holds true, the partial signature (k_i, σ_i) of message M received from participant P_i is valid. Once all partial results of an authorised set are received and verified, the combiner computes the group signature of message M as,

$$\sigma = \sum_{P_i \in \mathcal{A}} \sigma_i \pmod{q}.$$

To verify the signature, every one who knows the group public key can check,

$$k^M \stackrel{?}{=} r^r \times g^{\sigma} \pmod{p}.$$

If the equation holds true, the group signature (r, σ) is valid.

6.6.3 Boyd's System

The first attempts at designing a shared generation of RSA type signatures are due to Boyd [21]. He showed how to adapt the RSA system to implement a $(2,2)$ multisignature and a $(2,n)$ threshold RSA signature.

A $(2,2)$ RSA Multisignature

Boyd's $(2,2)$ scheme works as follows.

1. The dealer chooses a modulus N which is the product of two large primes and generates the public and secret keys k and K (respectively) as in the RSA [136] system.

2. The dealer selects at random two secret values $1 \leq s_1 \leq N$ and $1 \leq s_2 \leq N$, subject to the condition that s_1 and s_2 are coprime to $\varphi(N)$ and

$$s_1 \times s_2 \times k \equiv 1 \bmod \varphi(N).$$

   That is, $K = s_1 \times s_2 \bmod \varphi(N)$.

3. The dealer sends (in private) the shares s_1 and s_2 to their correspondents and publishes the public parameters N and k.

In order to sign a message $1 \leq M \leq N$ one of the co-signers (e.g. participant P_1) signs the message using his secret key as

$$\sigma_1 = M^{s_1} \pmod{N}$$

and sends his partial signature to another participant. The second participant computes the signature on the message M using

$$(\sigma_1)^{s_2} = (M^{s_1})^{s_2} = M^{s_1 s_2} = M^K = Sig_K(M) \pmod{N}.$$

The recipient of the signature can verify the signature on message M using the public key k as,

$$(Sig_K(M))^k \stackrel{?}{=} M \pmod{N}.$$

Note that, before signing, the second participant can check

$$(\sigma_1)^{s_2 \times k} \stackrel{?}{=} M \pmod{N}$$

to make sure that he is signing the message M. Moreover, the order of co-signers in this scheme is not fixed, that is, the protocol works if P_2 first signs the message.

It is not difficult to extend the scheme to a (n, n) multisignature, where

$$K = s_1 \times s_2 \times \cdots \times s_n \mod \varphi(N).$$

The signature generation requires that a participant signs the message and pass it to another participant. The message is signed if all participants have signed the previously generated partial signature when it is circulated among the group participants. That is, the signature generation works as follows:

$$(\cdots((M^{s_{i1}})^{s_{i2}})^{\cdots})^{s_{in}} = M^{s_1 \times s_2 \times \cdots \times s_n} = M^K \pmod{N}.$$

An alternative scheme can be generated when,

$$K = s_1 + s_2 + \cdots + s_n \mod \varphi(N).$$

In this case, each participant P_i generates his partial signature, $\sigma_i(M) = M^{s_i} \pmod{N}$, and transmits the result to the combiner. The combiner calculates the final signature using,

$$\prod_{i=1}^{n} \sigma_i = M^{s_1 + s_2 + \cdots + s_n} = M^K \pmod{N}.$$

A $(2,n)$ Threshold RSA Signature

Boyd also proposed an extended version of his $(2,2)$ multisignature scheme to construct a $(2,n)$ threshold RSA signature scheme. His $(2,n)$ threshold RSA signature scheme scheme works as follows.

1. The dealer selects the RSA parameter k, K and N as above.
2. the dealer generates (at random) the set of secret shares $S = \{s_1, \ldots, s_n\}$, subject to the condition that s_is $(1 \leq i \leq n)$ are coprime to $\varphi(N)$, and

$$s_1 \times s_2 \times \cdots \times s_n \times k = 1 \bmod \varphi(N).$$

   That is, $K = s_1 \times \ldots \times s_n \bmod \varphi(N)$.
3. The dealer gives (in private) each participant P_i $(1 \leq i \leq n)$ the set of shares $S \setminus s_i$ and publishes the parameter k and N.

To sign a message $1 \leq M \leq N$ one participant (e.g. the participant P_j, $1 \leq j \leq n$) signs the message as,

$$\sigma_j = M^{s_1 s_2 \cdots s_{j-1} s_{j+1} \cdots s_n} \pmod N.$$

Now, every member of the set $\mathcal{P} \setminus P_j$ can compute the final signature on the message M using

$$(\sigma_j)^{s_j} = M^{s_1 s_2 \cdots s_n} = M^K = Sig_K(M) \pmod N,$$

since he knows s_j.

The verification of the signature is the same as in original RSA system. For more details and other variants, e.g., the secret key $K = s_1 + \cdots + s_n$ in which multiplication is replaced by summation, see [21].

Note. The main drawback of Boyd's $(2,n)$ threshold RSA signature scheme is that the share of each participant is $n-1$ times of the secret in a single user RSA system.

6.6.4 Improvement of the System

In this section we present some improvement[1] on the $(2,n)$ RSA threshold signature systems.

Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ and let each participant, P_i $(1 \leq i \leq n)$ be associated with a binary string $b_{i_1} b_{i_2} \ldots b_{i_\ell}$ of length $\ell = \log_2 n$ that represents the integer value $i-1$.

[1]The results arose by the author from the lectures of Professor Desmedt in the Centre for Computer Security Research, University of Wollongong.

Set-up the System:

In order to construct a $(2, n)$ threshold RSA system, the dealer selects all system parameters k, K and N as in the RSA system and distributes the shares of K as follows:

1. for every j, $1 \leq j \leq \ell$, the dealer applies the Boyd's $(2,2)$ system and generates two shares s_{j_1} and s_{j_2}, such that,

$$K = s_{j_1} + s_{j_2} \mod \varphi(N),$$

2. for every participant P_i, if $b_{i_j} = 0$ then the dealer assigns s_{j_1} (or s_{j_2}) otherwise assigns s_{j_2} (or s_{j_1}).

The dealer sends (via secured channels) the shares to corresponding participants.

Shared Generation of RSA Signature:

Let a set $\mathcal{A} = \{P_{i_1}, P_{i_2}\} \subset \mathcal{P}$, $i_1 \neq i_2$ of two participants want to sign a message M. Since the binary representations of two integers $1 \leq i_1 \leq n$ and $1 \leq i_2 \leq n$ are different, at least for one particular bit b_j, $1 \leq j \leq \ell$ their shares (corresponding to iteration j of the set-up algorithm) will be different, and therefore, applying the basic Boyd's $(2, n)$ RSA signature they can sign the message.

Example 6.1 *Let $\mathcal{P} = \{P_1, \ldots, P_8\}$ be a set of participants. Thus,*

P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1

Table 6.1: Binary strings of participants

In order to distribute the shares corresponding to the first row of the above table, let $K = s_1 + s_2 \mod \varphi(N)$. The dealer assigns s_1 to participants P_1, P_2, P_3 and P_4 (since their first bits are zero) and assigns s_2 to participants P_5, P_6, P_7 and P_8. Similarly, for the second and third rows, the dealer distributes $K = s_3 + s_4$ and $K = s_5 + s_6$ respectively. That is, participants' shares are as in Table 6.2:

Since any set of two participants collectively knows K, it can sign the message. For example, if a set $\{P_1, P_2\}$ wishes to sign a message, M, participants P_1 and P_2 observe that their shares corresponding to the third row are different. Thus they generate M^{K_5}

P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
s_1	s_1	s_1	s_1	s_2	s_2	s_2	s_2
s_3	s_3	s_4	s_4	s_3	s_3	s_4	s_4
s_5	s_6	s_5	s_6	s_5	s_6	s_5	s_6

Table 6.2: Shares assigned to each participant

and M^{K_6} *(respectively) and send them to the combiner. The combiner computes the signature on message* M *as,*

$$M^{K_5} \times M^{K_6} = M^K \pmod{N}.$$

Note. Although the size of the shares assigned to each participant is reduced to $O(\log_2 n) \times K$, the system still is far from being ideal.

Chapter 7

Group-Oriented Cryptography

So far, we assumed that the groups have *anonymous membership*, i.e., it is not necessary to know who is a member of the group. Moreover, the internal structure and the access policy of the group was hidden (from an outsider point of view, such groups act as individuals). In this chapter we consider groups with *known members*. The sender determines an access structure and enciphers the messages such that only authorised subsets can decipher the cryptogram.

In Section 7.1 we give a motivation for group-oriented cryptography. Section 7.2 studies a related work on the construction of group-oriented cryptographic systems. The system is based on the Diffie-Hellman [53] key distribution algorithm. In Section 7.3 we present a model for a group-oriented cryptographic system. Two implementations of group-oriented cryptographic systems are presented in this section. The first one is based on public-key cryptosystems, while the second one uses the private-key systems. Finally, in Section 7.4 we present a self-certified group-oriented cryptosystem.

7.1 Motivation

Although only a small fraction of all groups in our society are groups with known members, there exist numerous examples to justify the need for group-oriented cryptography. For example, the Federal Bank may wish to send a message so that a particular set of banks, according to an access structure can decipher a cryptogram and hence read the message. In this and many other similar examples, the intended group is a group with known members. Each member of the intended group is either an individual or a group (e.g. banks, which from a cryptographic point of view act as individuals).

As has been discussed earlier, in order to decipher a cryptogram the receiver must know the secret key. So, one solution could be to send the message to all members of the group, e.g. using their public keys. A second is that the secret key is known to all

members and that the message is sent only once. These two obvious solutions, however, are not adapted to the security needs specific to the protection of information intended for groups. Moreover, constructing a threshold cryptosystem, as has been discussed in Chapter 6, is not adequate since the group of intended receivers and the access structure that determines how the cryptogram can be deciphered are chosen by the sender (see Section 6.3 for the justification of such needs).

The goal of implementation of group-oriented cryptographic systems is to solve this sort of cryptographic problem.

7.2 Related Works

There has not been much research on group-oriented cryptography. Hwang [81] proposed a shared decryption system in which the sender knows the set of receivers. The Hwang system utilises the Diffie-Hellman [53] key distribution scheme and concatenates the Shamir [147] secret sharing scheme with a *predetermined cryptographic system.*

7.2.1 Hwang's System

Assume that $\mathcal{P} = \{P_1, \ldots, P_n\}$. Let each member P_i, $1 \leq i \leq n$, hold a secret key K_i and publish the public key,

$$k_i = g^{K_i} \pmod{p},$$

where p is a large prime and g is a fixed primitive element in $GF(p)$. Furthermore, let each participant, P_i, also be associated with a public prime number N_i ($N_i > p$), such that $N_i \neq N_j$ if $i \neq j$.

Encryption:

Let a sender wish to send a message M to the group in such a way that M is readable only when any set of t members ($t \leq n$) agree to decipher the cryptogram. The sender performs as the following.

1. Obtain the public values g, p, N_i and k_i ($1 \leq i \leq n$) from the public directory of P_i.

2. Generate a secret random value $K \in \{1, \ldots, p-1\}$ and compute,

$$k = g^K \pmod{p} \qquad \text{and} \qquad x_i = (k_i)^K \pmod{p},$$

for all i, $1 \leq i \leq n$ (repeat this step if $x_i = x_j$ for any $i \neq j$).

3. Construct a polynomial,

$$f(x) = S + a_0 x_1 + \cdots + a_{t-1} x^{t-1} \pmod{p},$$

of degree $t-1$ (S will serve as the encryption/decryption key).

4. Encipher the message M using

$$C_1 = E_S(M),$$

where E denotes the predetermined encryption algorithm and D is the corresponding decryption algorithm.

5. Compute n shares $y_i = f(x_i) \pmod{p}$.

6. Compute a common solution C_2 using the Chinese Remainder Theorem (CRT) from the following system of equations:

$$x = y_i \pmod{N_i}, \qquad i = 1, \ldots, n.$$

7. Broadcast the ciphertext,

$$C = (C_1, C_2, N, k, t),$$

where $N = \prod_{i=1}^{n} N_i$ and t is the threshold parameter.

Group Decryption:

1. The legitimate user $P_i \in \mathcal{A}$ can authenticate himself as a legal receiver by verifying $N_i | N$.

2. P_i computes his share y_i by

$$C_2 = y_i \pmod{N_i},$$

and obtains

$$x_i = k^{K_i} \pmod{p}.$$

3. If $|\mathcal{A}| \geq t$ then the combiner can compute S – using Shamir's (t, n) threshold scheme.

4. Cooperating participants can decipher the cryptogram and recover the message M as

$$M = D_S(M).$$

Note. The sender may wish that anyone in the group can recover the message. Clearly, selecting a polynomial of degree zero does not protect the key. In this case, Hwang suggests that the sender generates $t-1$ dummy users and perform the algorithm (for more detail see [81]).

7.2.2 Other Systems

Franklin and Harber [63] also discussed group-oriented cryptosystems. Their system applies the ElGamal cryptosystem in which each user has his own public key. In their naive scheme, the size of cryptogram grows as the number of participants increases. In their modified system, although the size of an encryption is independent of the number of participants, the size of each encrypted bit is four elements of $\mathbb{Z}_p^*$ (where p is the modulo of their system).

In [69] and [70] we have discussed the model of group-oriented cryptography and proposed two group-oriented decryption systems based on the RSA and ElGamal cryptosystems.

7.3 Model of Group-Oriented Cryptography

So far, by $\mathcal{P} = \{P_1, \ldots, P_n\}$ we denote a group of individuals working together. Throughout this chapter, however, we assume P_i $(1 \leq i \leq n)$ is either an individual or a group with anonymous membership. Moreover, we assume that each participant P_i is assigned a cryptosystem. Construction of our group-oriented cryptosystem satisfies the following conditions:

- the system is set up by individual participants independently (groups with anonymous membership are also considered as individuals);
- each participant sets up their own cryptosystem. There is a trusted public registry ("White Pages") which keeps the list of authentic public keys of all potential participants. It is reasonable to assume that such a registry is already set up for single-user public-key cryptography;
- the group of intended receivers of a cryptogram and the access structure to the data are chosen by the sender;
- every authorised set (by the choice of the sender) can perform the group cryptographic transformation, while no unauthorised set can do so.

We assume that the access policy is threshold. An authorised set $\mathcal{A}$ retrieves the message iff the number of cooperating participants is equal to or greater than the threshold parameter t, that is,

$$\Gamma = \{\mathcal{A} \subseteq \mathcal{P} \mid |\mathcal{A}| \geq t\}.$$

The parameter t is chosen by the sender.

Definition 7.1 *A (t, n) group-oriented cryptosystem is a collection of two algorithms:*

1. *the sender, who composes the group of intended receivers $\mathcal{P}$ (for simplicity we assume that $\mathcal{P} = \{P_1, \ldots, P_n\}$), selects the threshold parameter t, collects authentic public keys $\{k_1, \ldots, k_n\} \in \mathcal{E}^n$ of participants of the group $\mathcal{P}$ from White Pages, and applies the encryption function*

$$E : \mathcal{M} \times \mathcal{E}^n \rightarrow \mathcal{C}.$$

The cryptogram $C \in \mathcal{C}$ is broadcast to all participants $\mathcal{P}$,

2. *the combiner, who collects partial decryptions $D_{K_i}(C) \in \Delta$ from a set $\mathcal{A} \subseteq \mathcal{P}$ (for the sake of simplicity assume that $\mathcal{A} = \{P_1, \ldots, P_\ell\}$) and decrypts the cryptogram as*

$$f_{COM} : \Delta_1 \times \ldots \times \Delta_\ell \rightarrow \mathcal{M}$$

where Δ_i is the set of partial decryptions for P_i $(i = 1, \ldots, \ell)$ and Δ is the set of group decryptions. The decryption is always successful if the number of cooperating participants is equal to or greater than the threshold parameter, that is, $\ell \geq t$.

In order to illustrate the model, we present implementations based on both public-key and private-key cryptosystems. In the public-key based system, the assumption is that all participants have their individual public-key cryptosystems already set up and there is a public registry (*"White Pages"*) which holds the authentic public elements of the participants. The sender creates a group from the public information available from the public registry.

7.3.1 Public-key Based Group-Oriented Cryptosystems

A simple scheme is that the sender first chooses the group and a threshold parameter t. The message is then embedded as a secret in a (t, n) threshold scheme. Shares for the scheme are then encrypted individually with each recipient's public key which can be broadcast along with the names of the users.

Consider a group $\mathcal{P}$ of n participants $P_1, P_2, \cdots, P_n$ each with their own public-key cryptosystem. Their public keys k_i are stored in a trusted public registry. The secret keys K_i are known to their owners only.

Suppose a sender wants to communicate a message, M, to a group $\mathcal{P} = \{P_1, \ldots, P_n\}$ such that any t out of n participants can decrypt the cryptogram.

Encryption:

The sender:

1. Obtains the public values from the public registry.
2. Selects at random a polynomial

$$f(x) = K + a_1 x + \cdots + a_{t-1} x^{t-1} \pmod{p},$$

where K will serve as the encryption key. Note that the $GF(p)$ has to be large enough so the guessing of K is not "easy".

3. Enciphers the message M using, $C_1 = E_K(M)$, where E denotes the predetermined encryption algorithm and D is the corresponding decryption algorithm.
4. Compute shares $s_i = f(x_i) \pmod{p}$ for public values x_i $(i = 1, \ldots, n)$.
5. Encrypts the shares of each recipient as;

$$c_i = \mathcal{E}_{k_i}(s_i), \qquad i = 1, \ldots, n$$

and broadcast to correspondents.

Group Decryption:

Upon receiving the cryptogram,

1. The legitimate user $P_i \in \mathcal{A}$ computes his share by

$$s_i = \mathcal{E}_{K_i}(c_i),$$

2. If $|\mathcal{A}| \geq t$ then the combiner can compute K – using Shamir's (t, n) threshold scheme.
3. Every cooperating participant can decipher the cryptogram and recover the message M as

$$M = D_K(M).$$

7.3.2 Private-Key Based Group-Oriented Cryptosystems

Existing threshold cryptosystems and the proposed group-oriented cryptosystems, so far, are based on public-key cryptosystems. The concept of sharing the power to encrypt a message makes no sense in a public-key context (since the public keys of the receivers are known by every one) however, it makes sense in a private-key context. In this section we discuss implementations of group-oriented cryptographic systems based on private-key cryptosystems.

Although private-key based systems can be applied in the same manner as in public-key based group-oriented cryptosystems, here we show an alternative method. That is, instead of enciphering the message, using a private-key cryptosystem, and sharing corresponding key among the recipient participants, using a secret sharing scheme, the sender transmits the cryptogram of the message itself. In fact, the sender applies Asmuth and Bloom's secret sharing scheme [4] to share a message among the recipient participants (see Section 3.5). However, the shares (partial messages) are enciphered using the recipients' cryptosystems.

Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be a set of participants, each associated with his own private-key cryptosystem. Also let $\mathcal{P}' = \{P'_1, \ldots, P'_{n'}\}$ be another set of participants, such that they have (collectively) all the cryptographic keys corresponding to the participants of set $\mathcal{P}$, that is, if $n' \leq n$ then some participants of set $\mathcal{P}'$ may have more than one private key. Further, let each private key be associated with a publicly known polynomial $f_i(x) \in GF(2^X)$, where X is the maximum block-size of messages in the underlying cryptographic system (e.g. in case of DES, $X = 64$).

The scheme consists of two algorithms: an (n, n) group encryption and a (t, n) group decryption.

Group Encryption:

Let $A(x)$ denote a polynomial attached to a binary vector A. Suppose that the group P' wants to send a message, M, $(M(x) \in GF(2^{tX-1}))$ to the group $\mathcal{P}$, such that any t out of n participants of the group $\mathcal{P}$ can cooperatively decrypt the cryptogram.

1. Each participant P'_i (or the combiner, since in this case there may be a combiner in the sender group as well) reduces M to an element of the field generated by $f_i(x)$. that is,

$$M(x) \equiv M_i(x) \bmod f_i(x), \quad i = 1, 2, \ldots, n.$$

2. P_i' uses his secret key, K_i', to encrypt the partial message $M_i(x)$ as,

$$c_i = E_{K_i'}(M_i).$$

3. Participants send their cryptogram(s) to their correspondents.

Group Decryption:

Upon receiving the cryptogram, users $P_i \in \mathcal{A} \subseteq \mathcal{P}$, such that $|\mathcal{A}| \geq t$ can recover the message as,

1. each P_i deciphers the cryptogram c_i and obtains the partial message M_i,

$$M_i = D_{K_i}(c_i),$$

2. collaborating participants send their partial messages to the combiner;

3. upon receiving at least t partial messages, M_i, the combiner applies CRT and retrieves the message M.

7.4 A Self-Certified Group-Oriented Cryptosystem Without a Combiner

Two important issues in the implementation of a public-key based group-oriented cryptosystem are:

1. the sender needs to collect authenticated public keys of the intended receivers;

2. in order to perform the group cryptographic transformation, the combiner needs a secure channel to collect (privately) the partial results from collaborating participants.

7.4.1 Problem with collecting authenticated public keys

When an individual wishes to encrypt a message for a group of users, he has first to collect their public keys (that are assumed to be stored in a public directory) and make sure that they actually correspond to those users. This assurance may clearly not be obtained, if each user is responsible for creating his pair of keys and publishing his public

key in a directory, because with such a system, nothing can prevent adversaries making fake keys related to a given user[1].

The obvious solution to this problem is to provide authenticated public keys by connecting users' public keys to their identities. There are three known approaches that require the existence of a trusted authority.

1. In the simplest approach, which is often called *certificate-based*, the authority creates a *certificate* for each user, after having checked carefully his identity. In this case, each user visiting the authority is given a certificate of the form $R = S(k, I)$, where I is an identification string based on the user's identity (prepared by the authority), k is the user's public key and S is the authority's signature. The certificate will then be registered in a public directory together with user's public key and his identity. Whenever a user A needs to encrypt a message for another user B, he gets (R_B, k_B, I_B) from the directory and checks the validity of the authority's signatures on the pair (k_B, I_B), using the authority's public key (that everybody is assumed to know). This approach, though having the advantage that even the authority does not know users' secret keys, requires a large amount of storage and computation (which essentially depends on the signature scheme in use).

2. Another approach, known as *identity-based*, is proposed by Shamir [150] and has been adopted in many public key schemes. The advantage of this method is that the user's identity serves as his public key and the related secret key is computed by some trapdoor originated by the authority, so that nobody can determine a valid pair of public and secret keys without knowing that trapdoor. This leads to a scheme that needs no certificate and no verification of signatures, hence reducing the amount of storage and computation. This method has, however, the disadvantage that the secret keys are known to the authority.

3. A more sophisticated technique combining the advantages of certificate-based and identity-based methods is proposed by Girault [74], which is known as *self-certified*. In this approach, contrary to identity-based schemes, each user chooses his secret key and creates a shadow w of that secret key using a one-way function and gives it to the authority. Then, contrary to certificate-based schemes, instead of creating a certificate, the authority computes the public key k from the pair (w, I), in such

[1]Even if there is an authority that controls the public directory and protect the write access to it, an adversary can still substitute a public key on the transmission line between the user who is asking that public key and the server which supports the public directory.

a way that k may not be computed without the knowledge of some trapdoor, while w may be easily determined from k, I.

We adopt the latter approach to guarantee the authenticity of the public keys.

7.4.2 Problem with collecting partial decryption

In society-oriented (threshold or group-oriented) cryptographic systems we assume that the combiner is not necessarily trusted. It is assumed that the combiner applies the required computations reliably. Since there is no secret information known to the combiner, everyone who knows the partial results can compute the final one as well. This requires, in society-oriented cryptographic systems, that the partial results must be sent (privately) to the combiner.

This may not be a serious problem if collaborating participants are working together. For example, members of an organisation may pass their partial results to the combiner via an internal mail or personally and in private. But in general, in order to transmit partial results to the combiner a secure channel is needed. However, providing secure channels may not be available when the group decryption is required.

In order to avoid the above mentioned problems, we present a group-oriented cryptosystem which utilises self-certified public keys and works with no help of any combiner.

7.4.3 Implementation of Self-Certified Public Keys

In this section we briefly consider the implementation of self-certified public keys.

Setup Phase:

As in all self-certified schemes, our system assumes the existence of an authority that delivers certified public keys to the legitimate users. In the setup phase of our scheme, the authority chooses:

- an integer N as the product of two large distinct random primes p and q of almost the same size such that $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also prime integers,
- a prime integer $F > N$,
- a base $\alpha \neq 1$ of order $\lambda = p'q'$ modulo N, and

- a collision free and one-way hash function h, that outputs integers less than the minimum value of p' and q', that is, $h(M) < \min(p', q')$.

The authority makes α, h, F and N public, keeps λ secret and discards p and q.

Key Generation:

Every legitimate user who wishes to receive messages chooses his secret key K, computes the shadow $z = \alpha^K \pmod N$ and gives it to the authority. The authority first interrogates the user about his secret key, using an authentication protocol (e.g., a variation of the Schnorr [143] authentication scheme with composite modulus) who must prove his knowledge of K, which is required to be a positive integer. If the authority is convinced of this fact, it prepares a string I corresponding to the user's identity (his name, his address, ...) and computes $ID = h(I)$. Then, it computes the user's public key as

$$k = (z^{-1} - ID)^{ID^{-1}} \pmod N$$

and registers it together with user's identity in a public directory.

Note that the inverse of ID modulo λ always exists, due to the fact that h outputs integers less than p' and q', which guarantees that ID is co-prime to λ, for any I.

7.4.4 Implementation of a (t, n) Group-Oriented Cryptosystem

Let an individual want to send a message $0 \leq M < N$ to a group $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n users of his choice, such that cooperation of any t members of the group is sufficient to retrieve the message.

Encryption:

The sender,

- randomly chooses an integer r and computes $c = (\alpha^{-1})^r \pmod N$,
- forms at random a polynomial $f(x) = v + a_1 x + \ldots + a_{t-1} x^{t-1}$ in $GF(F)$ such that $f(0) = v = \alpha^{h(M)} \pmod N$,
- computes for $i = 1, \ldots, n$

$$\begin{aligned} w_i &= k_i^{ID_i} + ID_i \pmod N \\ s_i &= w_i^r \pmod N \\ d_i &= f(s_i) \\ e_i &= M \cdot w_i^{h(M)} \pmod N \end{aligned}$$

and sends (t, c, d_i, e_i) to each P_i.

Group Decryption:

Upon receiving the cryptogram, every group, $\mathcal{A} \subseteq \mathcal{P}$, of at least t intended receivers can cooperate to retrieve the plaintext message M. That is, each $P_i \in \mathcal{A}$ first calculates,

$$s_i = c^{x_i} \pmod{N},$$

and broadcasts the pair (d_i, s_i). When t values of such pairs are broadcast, each P_i can recover $v = \alpha^{h(M)} \pmod{N}$, which allows computation of the plaintext message as

$$M = v^{x_i} e_i \pmod{N}.$$

Verification:

Since adversaries may substitute broadcasted messages for some participants or even a malicious participant can broadcast a false message, one needs to verify the correctness of the plaintext message that he has computed. For this purpose, after having computed the message M, each P_i checks,

$$\alpha^{h(M)} \stackrel{?}{\equiv} v \pmod{N}. \tag{7.1}$$

If the equation holds true, the retrieved message is valid, otherwise another collaboration of the intended group can recover the message.

7.4.5 Security Considerations

To discuss the security of the proposed group-oriented cryptographic system we observe the following (for more detail see [141]).

- Computing a pair of secret and public keys for a given ID is equivalent to solving a "hard" problem. That is, for a given pair (z, ID), the corresponding k may be computed from $k^{ID} = z^{-1} - ID \pmod{N}$. However, this is equivalent to breaking an instance of the RSA cryptosystem [136] with modulus N. On the other hand, if we first fix k, then K can be computed form the pair (k, ID) as

  $$z = (k^{ID} + ID)^{-1} \pmod{N}.$$

  However, in order to determine the related secret key K, one has to solve a hard instance of the discrete logarithm problem with composite modulus, i.e., $\alpha^K \pmod{N} = z$.

Note that, as pointed out in [74], the authority can create forged keys based on a given identity. However, this requires that one has more than one valid public key in the public registry.

- The proposed scheme utilises Shamir's (t, n) threshold scheme to share an encrypted message, i.e., $v = \alpha^{h(M)} \pmod{N}$. So, recovering the secret v needs the knowledge of at least t pairs (d_i, s_i). If $t-1$ participants $\{P_1, \ldots, P_{j-1}, P_{j+1}, \ldots, P_t\}$ try to recover v, then they need to know s_j. However, without knowing K_j, one has obviously to solve a Diffie-Hellman problem with a composite modulus, which is believed to be equivalent to factoring N and computing discrete logarithms modulo each prime factor[2].

- Although after broadcasting at least t pairs (d_i, s_i) it is easy to compute $v = \alpha^{h(M)} \pmod{N}$, the knowledge of v does not help an adversary to recover M. In fact, obtaining the message from a pair v and e_i (without knowing K_i) is equivalent to breaking the ElGamal cryptosystem [55] with composite modulus.

[2]This problem has been considered by McCurley [112] and is proven to be hard as long as at least one of the problems of computing discrete logarithms and factoring large integers remains intractable.

Chapter 8

Summary and Future Directions

In many cases, crucial decisions are left to a group of people rather than to an individual. The reasons for this are threefold.

1. A group is more reliable – even in the absence of one or more members, the group can still make proper decisions.

2. A group is more trustworthy than an individual – it is harder to corrupt the majority of the group than a single member.

3. A group is fairer and more equitable in their decision – the selection of a wide range of participants representing different points of views (and interests) can in many cases eliminate or substantially reduce bias or prejudice.

Groups try to work out a decision which is an acceptable compromise to all parties (or to a majority). For example, to activate a nuclear weapon, at least two senior officers must concur. To open a bank vault or a deposit box, the cooperation of at least two senior managers is usually required. In these and many other examples, a group must agree to act if they are to be successful in recovering a secret element. The underlying security mechanisms, however, incorporate some kind of secret sharing schemes.

Society-oriented cryptography can provide for the secrecy of sensitive information in an organisation. All known solutions for society-oriented cryptography, however, implicitly or explicitly use secret sharing schemes.

In this thesis, we looked at two systems for the management of secret information, namely secret sharing schemes and society-oriented cryptography.

In order to study secret sharing schemes we reviewed well known systems proposed in the literature and clarified misconceptions about some schemes. Important issues in secret sharing schemes, such as verifiability of the shares and the reconstructed secret value were discussed and a protocol which prevents cheating in a particular on-line secret

sharing scheme was presented. General schemes have been considered and a method which allows the construction of efficient general secret sharing schemes was presented. We have also proposed efficient solutions to sharing a secret in multilevel, or hierarchical, and compartmented access structures. The author's contributions to the area of secret sharing schemes were presented in [71], [68], [72], [73] and [35].

In the second part, cryptography as a tool for solving security problems was discussed and a novel cryptosystem was presented. It was shown that traditional private-key cryptographic systems and conventional public-key cryptosystems are adequate for cases where there are two individuals in the system. Society-oriented cryptographic systems were investigated and two major classes of society-oriented cryptographic systems, threshold and group-oriented cryptographic systems, were discussed. A cryptographic system which allows control of the flow of information in a hierarchical group was presented. Group-oriented cryptographic systems were discussed. Implementations of group-oriented cryptographic systems based on public-key and private-key cryptosystems were presented. A self-certified group-oriented cryptosystem that works with no help of combiner was discussed. Our contributions to society-oriented cryptography were published in [67], [70], [132], [69] and [141].

8.1 Future Directions

There are several topics that are the subject of ongoing investigation by researchers in the area of secret sharing and society-oriented cryptography. This section outlines some of these directions.

8.1.1 Proactive Secret Sharing Schemes

The goal of the implementation of a secret sharing scheme is to protect sensitive information by distributing it among different locations. The idea behind this technique is that an adversary cannot obtain the secret as long as specific numbers of shares have not been compromised. Compromising the secret information, however, is a matter of time. An adversary or a group of adversaries may have enough time to compromise sufficient numbers of shares and hence to obtain the secret. This indicates that conventional secret sharing schemes might be insufficient to provide the secrecy of *long-lived* sensitive information, such as a contract between two countries, proprietary trade-secret information, and so on.

8.1.2 Ideal Threshold Cryptographic Systems

For ElGamal, DSS, and unconditionally secure threshold authentication (see, for example, [51] and [111]), the size of the shares which each user must apply for his partial cryptographic operation is as small as the secret. But for an RSA-based (t, n) threshold system the size of the (modified) share for each participant is about t times that of the secret. The question is:

in RSA-based threshold systems, is it possible to make shares shorter?

8.1.3 Avoiding the Trusted Dealer

In general, to setup a threshold cryptographic system, a trusted party, also called the *dealer*, distributes the secret among the users. The main drawback of threshold cryptosystems with a trusted party is that the participants may not be able to agree on who can be trusted. Sometimes it is relatively easy to find a trusted party if the members of the group know each other well. This is not always the case in practice. For example, it is difficult to agree on a party trusted by two countries.

ElGamal based threshold decryption without a trusted party has been proposed in [131]. ElGamal based threshold signature without a trusted party also has proposed in [77]. DSS based threshold signature without a trusted party has been considered by Langford [102] and a $(2, n)$ system is proposed. However, for the RSA case the major problem is how to generate (jointly) the prime numbers, such that they are unknown to participants. In addition to several theoretical solutions, Boneh and Franklin [20] have proposed such an RSA based system, but their limitation is that $p \neq 2p' + 1$.

8.1.4 Robust Threshold Cryptography

Another important issue in a threshold cryptographic system is to provide the assurance that the collaborating participants obtain a correct result even if there are some problems with communication channels or some participants intentionally cheating by sending false data. In some cases, like threshold unconditionally secure systems, it is easy to deal with this problem (see for example [51]). However, achieving this goal for arbitrary threshold systems is not an easy task. Gennaro, Jarecki, Krawczyk and Rabin [65] have proposed a robust sharing of RSA functions.

8.1.5 Proactive Threshold Cryptosystems

With similar arguments to those regarding the need for a proactive secret sharing scheme, threshold cryptosystems are needed that can be applied proactively. Since the initial shares of participants are renewed in proactive secret sharing schemes, the collaborating participants of a threshold cryptosystem must still be able to compute their modified shares. A proactive RSA system has been proposed by Frankel, Gammell, MacKenzie and Yung [62].

A major technical difficulty in implementation of a proactive threshold RSA is how to update the shares in each time period, while not learning $\lambda(N)$.

Bibliography

[1] R. Anderson and S. Vaudenay, "Minding your p's and q's," in *Advances in Cryptology - Proceedings of ASIACRYPT '96* (K. Kim and T. Matsumoto, eds.), vol. 1163 of *Lecture Notes in Computer Science*, pp. 26–35, Springer-Verlag, 1996.

[2] T. Apostol, *Mathematical Analysis – A modern Approach to Advanced Calculus.* USA: Addison-Wesley, 1957.

[3] R. Archibald, *An Interoduction to The THEORY OF NUMBERS.* Columbus, Ohio: Charles E. Merrill, 1970.

[4] C. Asmuth and J. Bloom, "A Modular Approach to Key Safeguarding," *IEEE Transactions on Information Theory*, vol. IT-29, pp. 208–210, Mar. 1983.

[5] A. Baker, *Transcendental Number Theory.* Cambridge University Press, 1975.

[6] A. Beimel and B. Chor, "Secret Sharing with Public Reconstruction," in *Advances in Cryptology - Proceedings of CRYPTO '95* (D. Coppersmith, ed.), vol. 963 of *Lecture Notes in Computer Science*, pp. 353–366, Springer-Verlag, 1995.

[7] J. Benaloh, "Secret Sharing Homomorphisms: Keeping Shares of a Secret Secret," in *Advances in Cryptology - Proceedings of CRYPTO '86* (A. Odlyzko, ed.), vol. 263 of *Lecture Notes in Computer Science*, pp. 251–260, Springer-Verlag, 1987.

[8] J. Benaloh and J. Leichter, "Generalized Secret Sharing and Monotone Functions," in *Advances in Cryptology - Proceedings of CRYPTO '88* (S. Goldwasser, ed.), vol. 403 of *Lecture Notes in Computer Science*, pp. 27–35, Springer-Verlag, 1990.

[9] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness Theorem for Non-Cryptographic Fault-Tolerant Distributed Computation," in *20th Annual Symposium on the Theory of Computing (STOC)*, pp. 1–10, 1988.

[10] M. Bertilsson and I. Ingemarsson, "A Construction of Practical Secret Sharing Schemes using Linear Block Codes," in *Advances in Cryptology - Proceedings of AUSCRYPT '92* (J. Seberry and Y. Zheng, eds.), vol. 718 of *Lecture Notes in Computer Science*, pp. 67–79, Springer-Verlag, 1993.

[11] A. Beutelspacher, "How To Say "No"," in *Advances in Cryptology - Proceedings of EUROCRYPT '89* (J.-J. Quisquater and J. Vandewalle, eds.), vol. 434 of *Lecture Notes in Computer Science*, pp. 491–496, Springer-Verlag, 1990.

[12] G. Blakley, "Safeguarding cryptographic keys," in *Proceedings of AFIPS 1979 National Computer Conference*, vol. 48, pp. 313–317, 1979.

[13] B. Blakley, G. Blakley, A. Chan, and J. Massey, "Threshold Schemes With Disenrollment," in *Advances in Cryptology - Proceedings of CRYPTO '92* (E. Brickell, ed.), vol. 740 of *Lecture Notes in Computer Science*, pp. 540–548, Springer-Verlag, 1993.

[14] G. Blakley and C. Meadows, "Security Of Ramp Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '84* (G. Blakley and D. Chaum, eds.), vol. 196 of *Lecture Notes in Computer Science*, pp. 242–268, Springer-Verlag, 1985.

[15] D. Bleichenbacher, "Generating ElGamal Signatures Without Knowing the Secret Key," in *Advances in Cryptology - Proceedings of EUROCRYPT '96* (U. Maurer, ed.), vol. 1070 of *Lecture Notes in Computer Science*, pp. 10–18, Springer-Verlag, 1996.

[16] C. Blundo, A. Cresti, A. Santis, and U. Vaccaro, "Fully Dynamic Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '93* (D. Stinson, ed.), vol. 773 of *Lecture Notes in Computer Science*, pp. 110–125, Springer-Verlag, 1994.

[17] C. Blundo, A. Gaggia, and D. Stinson, "On the Dealer's Randomness Required in Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of EUROCRYPT '94* (A. Santis, ed.), vol. 950 of *Lecture Notes in Computer Science*, pp. 35–46, Springer-Verlag, 1995.

[18] C. Blundo, A. Santis, L. Gargano, and U. Vaccaro, "On the Information Rate of Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '92* (E. Brickell, ed.), vol. 740 of *Lecture Notes in Computer Science*, pp. 148–167, Springer-Verlag, 1993.

[19] C. Blundo, A. Santis, D. Stinson, and U. Vaccaro, "Graph Decompositions and Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of EUROCRYPT '92* (R. Rueppel, ed.), vol. 658 of *Lecture Notes in Computer Science*, pp. 1–24, Springer-Verlag, 1993. also, Journal of Cryptology, vol. 8, no. 1, pp. 39-46, 1995.

[20] D. Boneh and M. Franklin, "Efficient Generation of Shared RSA Keys," in *Advances in Cryptology - Proceedings of CRYPTO '97* (S. Burton and J. Kaliski, eds.), vol. 1294 of *Lecture Notes in Computer Science*, pp. 425–439, Springer-Verlag, 1997.

[21] C. Boyd, "Digital Multisignatures," in *Cryptography and Coding* (H. Beker and F. Piper, eds.), pp. 241–246, Clarendon Press, 1989.

[22] E. Brickell, "Some Ideal Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of EUROCRYPT '89* (J.-J. Quisquater and J. Vandewalle, eds.), vol. 434 of *Lecture Notes in Computer Science*, pp. 468–475, Springer-Verlag, 1990.

[23] E. Brickell and D. Davenport, "On the Classification of Ideal Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '89* (G. Brassard, ed.), vol. 435 of *Lecture Notes in Computer Science*, pp. 278–285, Springer-Verlag, 1990. also, Journal of Cryptology, vol. 4, no. 2, pp. 123-134, 1991.

[24] E. Brickell and D. Stinson, "Some Improved Bounds on the Information Rate of Perfect Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '90* (A. Menezes and S. Vanstone, eds.), vol. 537 of *Lecture Notes in Computer Science*, pp. 242–252, Springer-Verlag, 1991. also, Journal of Cryptology, vol. 5, no. 3, pp. 153-166, 1992.

[25] E. Brickell and D. Stinson, "The Detection of Cheaters in Threshold Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '88* (S. Goldwasser, ed.), vol. 403 of *Lecture Notes in Computer Science*, pp. 564–577, Springer-Verlag, 1990. also, SIAM J. on Discrete Math. vol. 4, pp. 502-510, 1991.

[26] L. Brown, M. Kwan, J. Pieprzyk, and J. Seberry, "Improving resistance to differential cryptanalisis and the redesign of loki," in *Advances in Cryptology - Proceedings of ASIACRYPT '91* (H. Imai, R. Rivest, and T. Matsumoto, eds.), vol. 739 of *Lecture Notes in Computer Science*, pp. 36–50, Springer-Verlag, 1993.

[27] L. Brown, J. Pieprzyk, and J. Seberry, "LOKI: A cryptographic primitive for authentication and secrecy applications," in *Advances in Cryptology - Proceedings of AUSCRYPT '90* (J. Seberry and J. Pieprzyk, eds.), Lecture Notes in Computer Science, pp. 229–236, Springer-Verlag, 1990.

[28] R. Burden and J. Faires, *Numerical Analysis.* Boston: PWS Publishing Company, 1993.

[29] C. Cachin, "On-line Secret Sharing," in *Cryptography and Coding: 5th IMA Conference* (C. Boyd, ed.), vol. 1025 of *Lecture Notes in Computer Science*, (Uk), pp. 190–198, Institute for Theoretical Computer Science, ETH Zürich, Springer-Verlag, Dec. 1995.

[30] R. Capocelli, A. Santis, L. Gargano, and U. Vaccaro, "On the Size of Shares for Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '91* (J. Feigenbaum, ed.), vol. 576 of *Lecture Notes in Computer Science*, pp. 101–113, Springer-Verlag, 1992. also, Journal of Cryptology, vol. 6, no. 3, pp. 157-167, 1993.

[31] M. Carpentieri, A. Santis, and U. Vaccaro, "Size of Shares and Probability of Cheating in Threshold Schemes," in *Advances in Cryptology - Proceedings of EUROCRYPT '93* (T. Helleseth, ed.), vol. 765 of *Lecture Notes in Computer Science*, pp. 118–125, Springer-Verlag, 1994.

[32] C. Charnes, K. Martin, J. Pieprzyk, and R. Safavi-Naini, "Secret Sharing in Hierarchical Groups," in *Proceedings of ICICS '97 -International Conference on Information and Communications Security, Beijing, P. R. China* (Y. Han, T. Okamoto, and S. Qing, eds.), vol. 1334 of *Lecture Notes in Computer Science*, pp. 81–86, Springer-Verlag, 1997.

[33] C. Charnes and J. Pieprzyk, "Cumulative arrays and generalised Shamir secret sharing schemes," in *Seventeenth Annual Computer Science Conference (ACSC-17), New Zealand* (G. Gupta, ed.), vol. 16 of *ISBN 0-473-02313-X*, ch. Part C, pp. 519–528, Australian Computer Science Communications, Jan. 1994.

[34] C. Charnes, J. Pieprzyk, and R. Safavi-Naini, "Conditionally Secure Secret Sharing Scheme with Disenrolment Capability," in *2nd ACM Conference on Computer and Communication Security*, (Fairfax, Virgina, USA), pp. 89–95, Nov. 1994.

[35] G. Chaudhry, H. Ghodosi, and J. Seberry, "Perfect Secret Sharing Schemes from Room Squares," *Combinatorial Mathematics and Combinatorial Computing*, vol. 28, pp. 55–61 (in press), 1998.

[36] D. Chaum, "Computer Systems Established, Maintained, and Trusted by Mutually Suspicious Groups," tech. rep., Memorandum No. UCB/ERL M/79/10, University of California, Berkeley, CA, Feb. 1979.

[37] D. Chaum, "How to Keep a Secret Alive: Extensible Private Key, Key Safeguarding, and Threshold Systems," in *Advances in Cryptology - Proceedings of CRYPTO '84* (G. Blakley and D. Chaum, eds.), vol. 196 of *Lecture Notes in Computer Science*, pp. 481–485, Springer-Verlag, 1985.

[38] D. Chen and D. Stinson, "Recent results on combinatorial constructions for threshold schemes," *Australasian Journal of Combinatorics*, vol. 1, pp. 29–48, 1990.

[39] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults," in 26^{th} *IEEE Symposium on Foundations of Computer Science, Portland*, pp. 383–395, Oct. 1985.

[40] B. Chor and E. Kushilevitz, "Secret Sharing Over Infinite Domains," in *Advances in Cryptology - Proceedings of CRYPTO '89* (G. Brassard, ed.), vol. 435 of *Lecture Notes in Computer Science*, pp. 299–306, Springer-Verlag, 1990.

[41] R. Cramer and I. Damgård, "New Generation of Secure and Practical RSA-Based Signatures," in *Advances in Cryptology - Proceedings of CRYPTO '96* (N. Koblitz, ed.), vol. 1109 of *Lecture Notes in Computer Science*, pp. 173–185, Springer-Verlag, 1996.

[42] R. Croft and S. Harris, "Public-key Cryptography and Re-usable Shared Secrets," in *Cryptography and Coding* (H. Beker and F. Piper, eds.), pp. 189–201, Clarendon Press, 1989.

[43] I. Damgård, "A Design Principle for Hash Functions," in *Advances in Cryptology - Proceedings of CRYPTO '89* (G. Brassard, ed.), vol. 435 of *Lecture Notes in Computer Science*, pp. 416–427, Springer-Verlag, 1990.

[44] D. Denning, "Digital Signatures with RSA and Other Public-Key Cryptosystems," *Communications of the ACM*, vol. 27, no. 4, pp. 388–392, 1984.

[45] Y. Desmedt, "Society and group oriented cryptography: A new concept," in *Advances in Cryptology - Proceedings of CRYPTO '87* (C. Pomerance, ed.), vol. 293 of *Lecture Notes in Computer Science*, pp. 120–127, Springer-Verlag, 1988.

[46] Y. Desmedt, "Abuses in Cryptography and How to Fight Them," in *Advances in Cryptology - Proceedings of CRYPTO '88* (S. Goldwasser, ed.), vol. 403 of *Lecture Notes in Computer Science*, pp. 375–389, Springer-Verlag, 1990.

[47] Y. Desmedt, "Threshold Cryptography," *European Transactions on Telecommunications and Related Technologies*, vol. 5, pp. 449–457, Jul-Aug 1994.

[48] Y. Desmedt and Y. Frankel, "Threshold cryptosystems," in *Advances in Cryptology - Proceedings of CRYPTO '89* (G. Brassard, ed.), vol. 435 of *Lecture Notes in Computer Science*, pp. 307–315, Springer-Verlag, 1990.

[49] Y. Desmedt and Y. Frankel, "Shared generation of authenticators and signatures," in *Advances in Cryptology - Proceedings of CRYPTO '91* (J. Feigenbaum, ed.), vol. 576 of *Lecture Notes in Computer Science*, pp. 457–469, Springer-Verlag, 1992.

[50] Y. Desmedt and Y. Frankel, "Homomorphic Zero-Knowledge Threshold Schemes over any Finite Abelian Group," *SIAM Journal on Disc. Math.*, vol. 14, pp. 667–679, 1994.

[51] Y. Desmedt, Y. Frankel, and M. Yung, "Multi-receiver/Multi-sender network security: Efficient authenticated multicast/feedback," *IEEE Infocom '92*, pp. 2045–2054, 1992.

[52] Y. Desmedt and S. Jajodia, "Redistributing secret shares to new access structures and its applications," tech. rep., Available from Desmedt's Home-Page (http://www.cs.uwm.edu/faculty/desmedt/index.html).

[53] W. Diffie and M. Hellman, "New Directions in Cryptography," *IEEE Trans. on Inform. Theory*, vol. IT-22, pp. 644–654, Nov. 1976.

[54] F. Drake, *SET THEORY An Introduction To Large Cardinals.* 1976.

[55] T. ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Trans. on Inform. Theory*, vol. IT-31, pp. 469–472, July 1985.

[56] Federal Register, USA, *A proposed federal information processing standard for digital signature standard (DSS)*, Aug. 1991.

[57] H. Feistel, "Cryptography and Computer Privacy," *Scientific American*, vol. 228, pp. 15–23, May 1973.

[58] P. Feldman, "A Practical Scheme for Non-interactive Verifiable Secret Sharing," in 28^{th} *IEEE Symposium on Foundations of Computer Science*, pp. 427–437, Oct. 1987.

[59] Y. Frankel, "A Practical Protocol for Large Group Oriented Networks," in *Advances in Cryptology - Proceedings of EUROCRYPT '89* (J.-J. Quisquater and J. Vandewalle, eds.), vol. 434 of *Lecture Notes in Computer Science*, pp. 56–61, Springer-Verlag, 1990.

[60] Y. Frankel and Y. Desmedt, "Classification of Ideal Homomorphic Threshold Schemes Over Finite Abelian Groups," in *Advances in Cryptology - Proceedings of EUROCRYPT '92* (R. Rueppel, ed.), vol. 658 of *Lecture Notes in Computer Science*, pp. 25–34, Springer-Verlag, 1993.

[61] Y. Frankel, Y. Desmedt, and M. Burmester, "Non-existence of homomorphic general sharing schemes for some key spaces," in *Advances in Cryptology - Proceedings of CRYPTO '92* (E. Brickell, ed.), vol. 740 of *Lecture Notes in Computer Science*, pp. 549–557, Springer-Verlag, 1993.

[62] Y. Frankel, P. Gemmell, P. MacKenzie, and M. Yung, "Proactive RSA," in *Advances in Cryptology - Proceedings of CRYPTO '97* (S. Burton and J. Kaliski, eds.), vol. 1294 of *Lecture Notes in Computer Science*, pp. 440–454, Springer-Verlag, 1997.

[63] M. Franklin and S. Haber, "Joint Encryption and Message-Efficient Secure Computation," in *Advances in Cryptology - Proceedings of CRYPTO '93* (D. Stinson, ed.), vol. 773 of *Lecture Notes in Computer Science*, pp. 266–277, Springer-Verlag, 1994.

[64] M. Garey and D. Johnson, *Computers and Intractability.* San Francisco, USA: Freeman, 1979.

[65] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust and Efficient Sharing of RSA Functions," in *Advances in Cryptology - Proceedings of CRYPTO '97* (S. Burton and J. Kaliski, eds.), vol. 1294 of *Lecture Notes in Computer Science*, pp. 157–172, Springer-Verlag, 1997.

[66] R. Gennaro and S. Micali, "Verifiable Secret Sharing as Secure Computation," in *Advances in Cryptology - Proceedings of EUROCRYPT '95* (L. Guillou and J.-J. Quisquater, eds.), vol. 921 of *Lecture Notes in Computer Science*, pp. 168–182, Springer-Verlag, 1995.

[67] H. Ghodosi, J. Pieprzyk, C. Charnes, and R. Safavi-Naini, "Cryptosystems for Hierarchical Groups," in *Proceedings of ACISP '96 -Australasian Conference on Information Security and Privacy* (J. Pieprzyk and J. Seberry, eds.), vol. 1172 of *Lecture Notes in Computer Science*, pp. 275–286, Springer-Verlag, 1996.

[68] H. Ghodosi, J. Pieprzyk, G. Chaudhry, and J. Seberry, "How to prevent cheating in Pinch's scheme," *Electronics Letters*, vol. 33, pp. 1453–1454, Aug. 1997.

[69] H. Ghodosi, J. Pieprzyk, and R. Safavi-Naini, "Dynamic Threshold Cryptosystems: A New Scheme in Group Oriented Cryptography," in *Proceedings of PRAGOCRYPT '96 -International Conference on the Theory and Applications*

of Cryptology (J. Přibyl, ed.), (Prague, Czech Republic), pp. 370–379, CTU Publishing house, ISBN: 80-01-01502-5, 1996.

[70] H. Ghodosi, J. Pieprzyk, and R. Safavi-Naini, "A Flexible Threshold Cryptosystem," in *Proceedings of ISITA '96 -The 1996 IEEE International Symposium on Information Theory and Its Applications*, (Victoria, B.C., Canada), pp. 75–77, 1996.

[71] H. Ghodosi, J. Pieprzyk, and R. Safavi-Naini, "Remarks on the Multiple Assignment Secret Sharing Scheme," in *Proceedings of ICICS '97 -International Conference on Information and Communications Security, Beijing, P. R. China* (Y. Han, T. Okamoto, and S. Qing, eds.), vol. 1334 of *Lecture Notes in Computer Science*, pp. 72–80, Springer-Verlag (Berlin), 1997.

[72] H. Ghodosi, J. Pieprzyk, and R. Safavi-Naini, "Secret Sharing in Multilevel and Compartmented Groups," in *Proceedings of ACISP '98 -Australasian Conference on Information Security and Privacy* (C. Boyd and E. Dawson, eds.), vol. 1438 of *Lecture Notes in Computer Science*, pp. 367–378, Springer-Verlag (Berlin), 1998.

[73] H. Ghodosi, J. Pieprzyk, R. Safavi-Naini, and H. Wang, "On Construction of Cumulative Secret Sharing Scheme," in *Proceedings of ACISP '98 -Australasian Conference on Information Security and Privacy* (C. Boyd and E. Dawson, eds.), vol. 1438 of *Lecture Notes in Computer Science*, pp. 379–390, Springer-Verlag (Berlin), 1998.

[74] M. Girault, "Self-certified public keys," in *Advances in Cryptology - Proceedings of EUROCRYPT '91* (D. Davies, ed.), vol. 547 of *Lecture Notes in Computer Science*, pp. 490–497, Springer-Verlag, 1991.

[75] O. Goldreich, S. Micali, and A. Wigderson, "How to Play any Mental Game," in *Proceedings of the Nineteenth ACM Symp. Theory of Computing, STOC*, pp. 218–229, May 25–27, 1987.

[76] J. D. Golić, "Intrinsic Statistical Weakness of Keystream Generators," in *Advances in Cryptology - Proceedings of ASIACRYPT '94* (J. Pieprzyk and R. Safavi-Naini, eds.), vol. 917 of *Lecture Notes in Computer Science*, pp. 91–103, Springer-Verlag, 1995.

[77] L. Harn, "Group-oriented (t, n) threshold digital signature scheme and digital multisignature," *IEE Proc.-Comput. Digit. Tech.*, vol. 141, pp. 307–313, Sept. 1994.

[78] J. He and E. Dawson, "Multistage secret sharing based on one-way function," *Electronics Letters*, vol. 30, pp. 1591–1592, Nov. 1994.

[79] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive Secret Sharing Or: How to Cope With Perpetual Leakage," in *Advances in Cryptology - Proceedings of CRYPTO '95* (D. Coppersmith, ed.), vol. 963 of *Lecture Notes in Computer Science*, pp. 339–352, Springer-Verlag, 1995.

[80] P. Horster, M. Michels, and H. Petersen, "Meta-Multisignature schemes based on the discrete logarithm problem," in *Information Security - the Next Decade,* (J. H. Eloff and S. H. Solms, eds.), IFIP/Sec '95, pp. 128–142, Proceedings of IFIP TC11 eleventh international conference on information security, Chapman and Hall, 1995.

[81] T. Hwang, "Cryptosystem for Group Oriented Cryptography," in *Advances in Cryptology - Proceedings of EUROCRYPT '90* (I. Damgård, ed.), vol. 473 of *Lecture Notes in Computer Science*, pp. 352–360, Springer-Verlag, 1991.

[82] I. Ingemarsson and G. Simmons, "A Protocol to Set Up Shared Secret Schemes Without the Assistance of a Mutually Trusted Party," in *Advances in Cryptology - Proceedings of EUROCRYPT '90* (I. Damgård, ed.), vol. 473 of *Lecture Notes in Computer Science*, pp. 266–282, Springer-Verlag, 1991.

[83] M. Ito, A. Saito, and T. Nishizeki, "Secret Sharing Scheme Realizing General Access Structure," in *Proceedings IEEE Global Telecommun. Conf., Globecom '87, Washington*, pp. 99–102, IEEE Communications Soc. Press, 1987.

[84] M. Ito, A. Saito, and T. Nishizeki, "Multiple Assignment Scheme for Sharing Secret," *Journal of Cryptology*, vol. 6, no. 1, pp. 15–20, 1993.

[85] W.-A. Jackson and K. Martin, "Cumulative Arrays and Geometric Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of AUSCRYPT '92* (J. Seberry and Y. Zheng, eds.), vol. 718 of *Lecture Notes in Computer Science*, pp. 48–55, Springer-Verlag, 1993.

[86] W. Jackson and K. Martin, "Efficient Constructions for One Sharing of Many Secrets," *Australian Journal of Combinatorics*, vol. 14, pp. 283–296, Sept. 1996.

[87] W.-A. Jackson, K. Martin, and C. O'Keefe, "Efficient Secret Sharing Without a Mutually Trusted Authority," in *Advances in Cryptology - Proceedings of EUROCRYPT '95* (L. Guillou and J.-J. Quisquater, eds.), vol. 921 of *Lecture Notes in Computer Science*, pp. 183–193, Springer-Verlag, 1995.

[88] W.-A. Jackson, K. Martin, and C. O'Keefe, "Mutually Trusted Authority-Free Secret Sharing Schemes," *Journal of Cryptology*, vol. 10, no. 4, pp. 261–289, 1997.

[89] W. De Jonge and D. Chaum, "Attacks on some RSA signatures," in *Advances in Cryptology - Proceedings of CRYPTO '85* (H. Williams, ed.), vol. 218 of *Lecture Notes in Computer Science*, pp. 18–27, Springer-Verlag, 1986.

[90] W. De Jonge and D. Chaum, "Some Variations on RSA Signatures & their Security," in *Advances in Cryptology - Proceedings of CRYPTO '86* (A. Odlyzko, ed.), vol. 263 of *Lecture Notes in Computer Science*, pp. 49–59, Springer-Verlag, 1987.

[91] D. Kahn, *The Codebreakers: the story of secret writing.* New York: Macmillian, 1967.

[92] R. Kannan, A. Lenstra, and L. Lovász, "Polynomial Factorization and Nonrandomness of Bits of Algebraic and Some Transcendental Numbers," *Mathematics of Computation*, vol. 50, pp. 235–250, Jan. 1988.

[93] E. Karnin, J. Greene, and M. Hellman, "On Secret Sharing Systems," *IEEE Transactions on Information Theory*, vol. IT-29, pp. 35–41, Jan. 1983.

[94] D. Knuth, *The Art of Computer Programming / Seminumerical Algorithms*, vol. 2. Addison-Wesley, Reading MA, 2 ed., 1981.

[95] N. Koblitz, "Eliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[96] S. Kothari, "Generalized Linear Threshold Scheme," in *Advances in Cryptology - Proceedings of CRYPTO '84* (G. Blakley and D. Chaum, eds.), vol. 196 of *Lecture Notes in Computer Science*, pp. 231–241, Springer-Verlag, 1985.

[97] K. Koyama, U. Maurer, T. Okamoto, and S. Vanstone, "New public-key schemes based on elliptic curves over the ring z_n," in *Advances in Cryptology - Proceedings of CRYPTO '91* (J. Feigenbaum, ed.), vol. 576 of *Lecture Notes in Computer Science*, pp. 252–266, Springer-Verlag, 1992.

[98] K. Kurosawa, S. Obana, and W. Ogata, "t-Cheater Identifiable (k, n) Threshold Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '95* (D. Coppersmith, ed.), vol. 963 of *Lecture Notes in Computer Science*, pp. 410–423, Springer-Verlag, 1995.

[99] K. Kurosawa, K. Okada, K. Sakano, W. Ogata, and S. Tsujii, "Nonperfect Secret Sharing Schemes and Matroids," in *Advances in Cryptology - Proceedings of EUROCRYPT '93* (T. Helleseth, ed.), vol. 765 of *Lecture Notes in Computer Science*, pp. 126–141, Springer-Verlag, 1994.

[100] C. Laih and L. Harn, "Generalized Threshold Cryptosystems," in *Advances in Cryptology - Proceedings of ASIACRYPT '91* (H. Imai, R. Rivest, and T. Matsumoto, eds.), vol. 739 of *Lecture Notes in Computer Science*, pp. 159–166, Springer-Verlag, 1993.

[101] C. Laih, L. Harn, J. Lee, and T. Hwang, "Dynamic threshold scheme based on the definition of cross-product in an N-dimentional linear space," in *Advances in Cryptology - Proceedings of CRYPTO '89* (G. Brassard, ed.), vol. 435 of *Lecture Notes in Computer Science*, pp. 286–297, Springer-Verlag, 1990.

[102] S. Langford, "Threshold DSS Signatures without a Trusted Party," in *Advances in Cryptology - Proceedings of CRYPTO '95* (D. Coppersmith, ed.), vol. 963 of *Lecture Notes in Computer Science*, pp. 397–409, Springer-Verlag, 1995.

[103] S. Langford, "Weaknesses in Some Threshold Cryptosystems," in *Advances in Cryptology - Proceedings of CRYPTO '96* (N. Koblitz, ed.), vol. 1109 of *Lecture Notes in Computer Science*, pp. 74–82, Springer-Verlag, 1996.

[104] A. Lenstra, H. Lenstra, and L. Lovász, "Factoring Polynomials with Rational Coefficients," *Math. Ann.*, vol. 261, pp. 513–534, 1982.

[105] C. Li, T. Hwang, and N. Lee, "Remark on the Threshold RSA Signature Scheme," in *Advances in Cryptology - Proceedings of CRYPTO '93* (D. Stinson, ed.), vol. 773 of *Lecture Notes in Computer Science*, pp. 413–419, Springer-Verlag, 1994.

[106] C.-M. Li, T. Hwang, and N.-Y. Lee, "Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders," in *Advances in Cryptology - Proceedings of EUROCRYPT '94* (A. Santis, ed.), vol. 950 of *Lecture Notes in Computer Science*, pp. 194–204, Springer-Verlag, 1995.

[107] H. Lin and L. Harn, "A Generalized Secret Sharing Scheme With Cheater Detection," in *Advances in Cryptology - Proceedings of ASIACRYPT '91* (H. Imai, R. Rivest, and T. Matsumoto, eds.), vol. 739 of *Lecture Notes in Computer Science*, pp. 149–158, Springer-Verlag, 1993.

[108] K. Martin, *Discrete Structures in the Theory of Secret Sharing.* PhD Thesis, University of London, Royal Hollway and Bedford New College, 1991.

[109] K. Martin, "New Secret Sharing Schemes from Old," *Journal of Combin. Math Combin. Comput.*, vol. 14, pp. 65–77, 1993.

[110] K. Martin, "Untrustworthy Participants in Secret Sharing Schemes," in *Cryptography and Coding III* (M. Ganley, ed.), The Institute of Mathematics and Its Applications Conference, (Uk), pp. 255–264, Oxford University Press, 1993.

[111] K. Martin and R. Safavi-Naini, "Multisender Authentication Systems with Unconditional Security," in *Proceedings of ICICS '97 –International Conference on Information and Communications Security, Beijing, P. R. China* (Y. Han, T. Okamoto, and S. Qing, eds.), vol. 1334 of *Lecture Notes in Computer Science*, pp. 130–143, Springer-Verlag (Berlin), 1997.

[112] M. McCurley, "A key distribution system equivalent to factoring," *Journal of Cryptology*, vol. 1, no. 2, pp. 95–105, 1988.

[113] R. McEliece, "A Public-Key Cryptosystem Based on Algebraic Coding Theory," tech. rep., DSN Progress Report, JPL Pasadena, 1978.

[114] R. McEliece and D. Sarwate, "On Sharing Secrets and Reed-Solomon Codes," *Communications of the ACM*, vol. 24, no. 9, pp. 583–584, 1981.

[115] C. Meadows, "Some Threshold Schemes Without Central Key Distributors," in *Congressus Numerantium*, vol. 46, pp. 187–199, 1985.

[116] W. Meier and O. Staffelbach, "Correlation Properties of Combiners with Memory in Stream Ciphers," in *Advances in Cryptology - Proceedings of EUROCRYPT '90* (I. Damgård, ed.), vol. 473 of *Lecture Notes in Computer Science*, pp. 204–213, Springer-Verlag, 1991.

[117] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography.* USA: CRC Press, 1997.

[118] A. Menezes, T. Okamoto, and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE Trans. on Inform. Theory*, vol. IT-39, pp. 1639–1646, 1993.

[119] R. Merkle and M. Hellman, "Hiding Information and Signatures in Trapdoor Knapsacks," *IEEE Trans. on Inform. Theory*, vol. IT-24, pp. 525–530, Sept. 1978.

[120] M. Mignotte, "How to Share a Secret," in *Cryptography* (T. Beth, ed.), vol. 149 of *Workshop on Cryptography*, pp. 371–375, Burg Feuerstein, Germany: Springer-Verlag, 1983.

[121] V. Miller, "Use of Elliptic Curves in Cryptography," in *Advances in Cryptology - Proceedings of CRYPTO '85* (H. Williams, ed.), vol. 218 of *Lecture Notes in Computer Science*, pp. 417–426, Springer-Verlag, 1986.

[122] National Bureau of Standards, Federal Information Processing Standard (FIPS), US, Department of Commerce, *Data Encryption Standard*, 46 ed., Jan. 1977.

[123] I. Niven, *Irrational Numbers.* USA: Mathematical Association of America, 1956.

[124] K. Nyberg and R. Rueppel, "Message Recovery for Signature Schemes Based on the Discrete Logarithm Problem," in *Advances in Cryptology - Proceedings of EUROCRYPT '94* (A. Santis, ed.), vol. 950 of *Lecture Notes in Computer Science*, pp. 182–193, Springer-Verlag, 1995. also, Designs, Code and Cryptography, vol. 7, pp. 61-81 , 1996.

[125] S. Obana and K. Kurosawa, "Veto is impossible in secret sharing schemes," *Information Processing Letters*, vol. 58, pp. 293–295, June 1996.

[126] L. O'Conner and J. Seberry, *Cryptographic Significance of the Knapsack Problem.* Canada: Aegean Park Press, 1988.

[127] A. Odlyzko, "Discrete logarithms in finite fields and their cryptographic significance," in *Advances in Cryptology - Proceedings of EUROCRYPT '84* (T. Beth, N. Cot, and I. Ingemarsson, eds.), vol. 209 of *Lecture Notes in Computer Science*, pp. 225–314, Springer-Verlag, 1985.

[128] W. Ogata, K. Kurosawa, and S. Tsujii, "Nonperfect Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of AUSCRYPT '92* (J. Seberry and Y. Zheng, eds.), vol. 718 of *Lecture Notes in Computer Science*, pp. 56–66, Springer-Verlag, 1993.

[129] C. Park and K. Kurosawa, "New ElGamal Type Threshold Digital Signature Scheme," *IEICE Trans. Fundamentals*, vol. E79-A, pp. 86–93, Jan. 1996.

[130] T. Pedersen, "Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing," in *Advances in Cryptology - Proceedings of CRYPTO '91* (J. Feigenbaum, ed.), vol. 576 of *Lecture Notes in Computer Science*, pp. 129–140, Springer-Verlag, 1992.

[131] T. Pedersen, "A Threshold Cryptosystem without a Trusted Party," in *Advances in Cryptology - Proceedings of EUROCRYPT '91* (D. Davies, ed.), vol. 547 of *Lecture Notes in Computer Science*, pp. 522–526, Springer-Verlag, 1991.

[132] J. Pieprzyk, H. Ghodosi, C. Charnes, and R. Safavi-Naini, "Cryptography Based on Transcendental Numbers," in *Proceedings of ACISP '96 -Australasian Conference on Information Security and Privacy* (J. Pieprzyk and J. Seberry, eds.), vol. 1172 of *Lecture Notes in Computer Science*, pp. 96–107, Springer-Verlag, 1996.

[133] J. Pieprzyk and B. Sadeghiyan, *Design of Hashing Algorithms.* Berlin: Springer-Verlag, 1993.

[134] R. Pinch, "Online multiple secret sharing," *Electronics Letters*, vol. 32, pp. 1087–1088, June 1996.

[135] R. Rivest, "Remarks on a proposed cryptanalytic attack on the MIT public-key cryptosystem," *Cryptologia*, vol. 2, pp. 62–65, 1978.

[136] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, Feb. 1978.

[137] R. Rivest, "Cryptography," in *Handbook of Theoretical Computer Science, Chap. 13* (J. Leeuwen, ed.), Elsevier Science, 1990.

[138] R. Rueppel, "New approaches to stream ciphers," in *PhD Thesis to Swiss Federal Institute of Technology, Zurich*, 1984.

[139] R. Rueppel, "Correlation immunity and the summation combiner," in *Advances in Cryptology - Proceedings of CRYPTO '85* (H. Williams, ed.), vol. 218 of *Lecture Notes in Computer Science*, pp. 260–272, Springer-Verlag, 1986.

[140] R. Rueppel, "Stream ciphers," in *Contemporary Cryptology - The Science of Information Integrity*, (New York), IEEE Press, 1992.

[141] S. Saeednia and H. Ghodosi, "A Self-Certified Group-Oriented Cryptosystem Without a Combiner," in *Proceedings of ACISP '99 -Australasian Conference on Information Security and Privacy*, Lecture Notes in Computer Science, Springer-Verlag (To Appear).

[142] A. Santis and M. Yung, "On the Design of Provably-Secure Cryptographic Hash Functions," in *Advances in Cryptology - Proceedings of EUROCRYPT '90* (I. Damgård, ed.), vol. 473 of *Lecture Notes in Computer Science*, pp. 412–431, Springer-Verlag, 1991.

[143] C. Schnorr, "Efficient Signature Generation by Smart Cards," *Journal of Cryptology*, vol. 4, no. 3, pp. 161–174, 1991.

[144] J. Seberry and J. Pieprzyk, *Cryptography - An Introduction to Computer Security.* Australia: Prentice Hall, 1989.

[145] P. Schellenberg and D. Stinson, "Threshold Schemes fom Combinatorial Designs," *Combinatorial Mathematics and Combinatorial Computing*, vol. 5, pp. 143–160, 1989.

[146] A. Schönhage, "The fundamental theorem of algebra in terms of computational complexity," tech. rep., Preliminary report, Mathematisches Institut der Universität Tübigen, 1982.

[147] A. Shamir, "How to Share a Secret," *Communications of the ACM*, vol. 22, pp. 612–613, Nov. 1979.

[148] A. Shamir, "On the Generation of Cryptographically Strong Pseudo-Random Sequences," in *Proc. 8th International Colloquium on Automata, Languages and Programming*, 1981. also, ACM Transactions on Computer Systems, 1(1):38-44, 1983.

[149] A. Shamir, "A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem," *IEEE Trans. on Inform. Theory*, vol. IT-30, pp. 699–704, Sept. 1984.

[150] A. Shamir, "Identity-based Cryptosystems and Signature Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '84* (G. Blakley and D. Chaum, eds.), vol. 196 of *Lecture Notes in Computer Science*, pp. 47–53, Springer-Verlag, 1985.

[151] C. Shannon, "Communication Theory of Secrecy Systems," *Bell System Technical Journal*, vol. 28, pp. 565–715, 1948.

[152] A. Shimizu and S. Miyaguchi, "Fast Data Encipherment Algorithm FEAL," in *Advances in Cryptology - Proceedings of EUROCRYPT '87* (D. Chaum and W. Price, eds.), vol. 304 of *Lecture Notes in Computer Science*, pp. 267–278, Springer-Verlag, 1987.

[153] T. Siegenthaler, "Correlation Immunity of Nonlinear Combining Functions for Cryptographic Applications," *IEEE Trans. on Inform. Theory*, vol. IT-30, pp. 776–780, 1984.

[154] G. Simmons, "How to (Really) Share a Secret," in *Advances in Cryptology - Proceedings of CRYPTO '88* (S. Goldwasser, ed.), vol. 403 of *Lecture Notes in Computer Science*, pp. 390–448, Springer-Verlag, 1990.

[155] G. Simmons, "Robust Shared Secret Schemes or 'How to be Sure You Have the Right Answer Even Though You Don't Know the Question'," in *18th Annual Conference on Numerical mathematics and Computing*, vol. 68 of *Congressus Numerantium*, (Manitoba, Canada), pp. 215–248, Winnipeg, May 1989.

[156] G. Simmons, "Prepositioned Shared Secret and/or Shared Control Schemes," in *Advances in Cryptology - Proceedings of EUROCRYPT '89* (J.-J. Quisquater and J. Vandewalle, eds.), vol. 434 of *Lecture Notes in Computer Science*, pp. 436–467, Springer-Verlag, 1990.

[157] G. Simmons, "Geometric Shared Secret and/or Shared Control Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '90* (A. Menezes and S. Vanstone, eds.), vol. 537 of *Lecture Notes in Computer Science*, pp. 216–241, Springer-Verlag, 1991.

[158] G. Simmons, "An Introduction to Shared Secret and/or Shared Control Schemes and Their Application," in *Contemporary Cryptology - The Science of Information Integrity* (G. Simmons, ed.), (New York), pp. 441–497, IEEE Press, 1992.

[159] G. Simmons, "The Consequences of Trust in Shared Secret Schemes," in *Advances in Cryptology - Proceedings of EUROCRYPT '93* (T. Helleseth, ed.), vol. 765 of *Lecture Notes in Computer Science*, pp. 448–452, Springer-Verlag, 1994.

[160] G. Simmons, W.-A. Jackson, and K. Martin, "The Geometry of Shared Secret Schemes," *Bulletin of the Institute of Combinatorics and its Applications (ICA)*, vol. 1, pp. 71–88, Jan. 1991.

[161] G. Simmons and M. Norris, "Preliminary comments on the MIT public-key cryptosystem," *Cryptologia*, vol. 1, pp. 406–414, 1977.

[162] V. Smirnov, *A Course of Higher Mathematics – vol. II.* USA: Addison-Wesley, 1964.

[163] M. De Soete and K. Vedder, "Some New Classes of Geometric Threshold Schemes," in *Advances in Cryptology - Proceedings of EUROCRYPT '88* (C. Günther, ed.), vol. 330 of *Lecture Notes in Computer Science*, pp. 389–401, Springer-Verlag, 1988.

[164] M. Stadler, "Publicly Verifiable Secret Sharing," in *Advances in Cryptology - Proceedings of EUROCRYPT '96* (U. Maurer, ed.), vol. 1070 of *Lecture Notes in Computer Science*, pp. 190–199, Springer-Verlag, 1996.

[165] D. Stinson, "An Explication of Secret Sharing Schemes," *Designs, Codes and Cryptography*, vol. 2, pp. 357–390, 1992.

[166] D. Stinson, "New General Lower Bounds on the Information Rate of Secret Sharing Schemes," in *Advances in Cryptology - Proceedings of CRYPTO '92* (E. Brickell, ed.), vol. 740 of *Lecture Notes in Computer Science*, pp. 168–182, Springer-Verlag, 1993.

[167] D. R. Stinson, *CRYPTOGRAPHY Theory and Practice.* USA: CRC Press, 1995.

[168] D. Stinson and S. Vanstone, "A Combinatorial Approach to Threshold Schemes," *SIAM J. Disc. Math.*, vol. 1, pp. 230–236, May 1988.

[169] G. Thomas, *Calculus – second edition.* USA: Addison-Wesley, 1961.

[170] M. Tompa and H. Woll, "How To Share a Secret with Cheaters," *Journal of Cryptology*, vol. 1, no. 2, pp. 133–138, 1988.

[171] H. Tsai and C. Chang, "A cryptographic implementation for dynamic access control in a user hierarchy," *Computers & Security*, vol. 14, no. 2, pp. 159–166, 1995.

[172] J. Vandergraft, *Introduction to Numerical Computations.* New York: Academic Press, 1983.

[173] A. Yao, "Theory and Applications of Trapdoor Functions," in *the 23rd IEEE Symposium on the Foundations of Computer Science*, pp. 80–91, 1982.

[174] Y. Zheng, *Priciples for Designing Secure Block Ciphers and One-Way Hash Functions.* PhD Thesis, Electrical and Computer Engineering, Japan, 1990.