CODING SCHEMES FOR DIGITAL FINGERPRINTING AND TRAITOR TRACING

Thesis submitted for the award of
Doctor of Philosophy

V. RAVI SANKAR

Department of Computer & Information Sciences
School of Mathematics and Computer/ Information Sciences
University of Hyderabad
Hyderabad - 500 046

JUNE 2004

dedicated to

Sri Ganapathi Sachchidananda

Department of Computer & Information Sciences
School of Mathematics and Computer/ Information Sciences
University of Hyderabad
Hyderabad - 500 046

This is to certify that I, V. Ravi Sankar, have carried out the research embodied in the present thesis for the full period prescribed under the Ph.D ordinances of the University.

I declare to the best of my knowledge that, no part of this thesis was earlier submitted for the award of any research degree of any university.

V. Ravi Sankar

Prof. A.K. Pujari	Dr. Ashutosh Saxena	Dr. V. P. Gulati
Supervisor,	Supervisor,	Supervisor,
University of Hyderabad	IDRBT	IDRBT

Prof. Arun Agarwal	Prof. A.K. Pujari
Head of the Department	Dean of the School

ACKNOWLEDGEMENTS

I would like to thank Director, IDRBT for providing me all the facilities and support to carryout my work.

I also thank Head, Dept of Computer and Information Sciences and Dean, School of MCIS, University of Hyderabad for providing me the required facilities at University to carryout my work.

I sincerely express my deep sense of gratitude to my guide Prof. A.K.Pujari for his constant encouragement and valuable suggestions throughout this work. I also express my gratitude to my co-guides Dr.Ashutosh Saxena and Dr.V.P.Gulati for their unflinching cooperation, guidance, and encouragement throughout the work.

Also it is a pleasure to express my gratitude to the faculty members of Dept of Computer & Information Sciences, UoH and all the Guest Speakers of the Seminar series. I also thank the faculty members of IDRBT for their constant encouragement and suggestions.

I gratefully acknowledge the Librarian, publication officer, the administration and other staff members of IDRBT whose help need a special mention here.

I also thank all my friends at University and colleagues at IDRBT especially Manik lal Das and Sanjay Rawat for their help and cooperation during this work.

At this moment I fondly recall the support of my parents and family members especially my mother who motivated and encouraged me for higher studies. And above all I owe a lot to the grace of Almighty.

V.Ravi Sankar

Notations and Abbreviations

Notations	Abbreviation
q	alphabet size
Q	alphabet
${}^{n}C_{r}$	n choose r
x	codeword
y	”
z	”
$d(x, y)$	hamming distance between x and y
$w(x)$	weight of x
W	collusion set
W_i	”
Γ	fingerprinting code
S	A set
F_p	finite field
N	code length
c	maximum collusion size
b	number of private keys
X	A set
B	set of all blocks in a set system
s, r, v, k	integers
B_i	i^{th} block in (X, B)
F	pirate decoder
w	collusion size
d	hamming distance
λ	integer
$D(C)$	descendent set of C
$desc(C_i)$	descendent set of C_i
Ψ	public fingerprinting code
e_N	number of erasures
e or ?	erasure
ε	error probability
E	elliptic curve
G	base point

Notations	Abbreviation
P_M	message point
θ_i	i^{th} private (key) value
$\overline{d}$	pirate decoder
$\overline{d}_i$	legitimate decoder
β	number of pirate fingerprints
I_n	n × n identity matrix
T	set of keys
R	A relation
k -set	set of size k
λ_s	blocks that contain the s -set
$\overline{\lambda}_s$	blocks that do not contain the s -set
(X, B)	Set system
$V(M, c)$	volume of a sphere in hamming space
$H_2(x)$	binary entropy function
$H_q(x)$	q'ary entropy function
$GF(q)$	Galois field of size q
c -group	group of size c
c-secure	secure against collusions of size c
c-TA	c-traceability code
c-FPC	c-trameproof code
$S(t,k,v)$	Steiner system
(N, M, q) -code	An error correcting code
c-$TS(k, b, v)$	c-Traceability Scheme
t-(v, k, λ)	t-design
c-$Gossip(l, M, q)$	c-Gossip code
CFF	Cover-Free Family
PHF	Perfect Hash Family
SHF	Secure Hash Family
IPP	Identifiable Parent Property
SFP	Secure Frameproof Code
DF	Digital fingerprinting

List of Figures

List of Tables

Abstract

This work deals with the problem of constructing collusion secure fingerprinting codes with short length, which can identify at least one member of the pirate collusion responsible for creating an illegal digital copy. The goal of this work is to construct and analyse non-binary fingerprinting codes such that it is possible for the distributor to identify at least one pirate of the guilty coalition even in the presence of erasures in the recovered fingerprint. This thesis examines various coding schemes and their applicability in collusion secure digital fingerprinting and pirate tracing. The study of the combinatorial properties of these codes in turn assist in constructing codes with good performance.

The work exposes the relations between fingerprinting codes and combinatorial designs [CD96]. Fingerprinting codes that offer collusion resistance and the combinatorial properties of these codes are investigated. A simple fingerprinting model that uses a code where every collusion group creates a unique pirate fingerprint is considered. In this model, the theoretical bounds on non-binary fingerprinting codes, namely, fingerprint length and collusion size are presented. In the traitor tracing scenario, the work presents a tracing method to find the contribution made by authorised users to pirate decoders in Boneh and Franklin's broadcast encryption method [BF99].

This work presents two new construction techniques for Gossip codes, which are IPP codes, from t-designs and Traceability Schemes. These construction techniques for Gossip codes achieve the minimum code length specified for Gossip codes in terms of code parameters (*i.e.,* the number of codewords M, alphabet size q and maximum collusion size c) .

This work also describes the construction of embedded Gossip codes for extending an existing Gossip code into a bigger code. Embedded Gossip codes can also be used to construct embedded Traceability Schemes and embedded frameproof codes, which can expand a broadcast application such that it is compatible to the existing scheme.

A new aspect of this work is handling the erasures in the shortest Gossip codes. In the erasures model, tracing pirates is possible when pirates create erasures (non-alphabet symbols) or alphabet symbols that occur in one of their copies in detected positions. Undetected positions by the pirates remain unaltered in the pirate copy created by them. The tracing in the presence of erasures is categorised (and carried out) into three explicit scenarios - no erasures, only erasures and selective erasures.

The work also presents the realisation of erasure model through concatenated codes. In this model, the pirates choose an alphabet symbol found in their copies or an erasure (denoted by '*e*' or '?') at detected positions. In order to enforce this choice for pirates (and to prevent pirates from choosing a alphabet symbol not found in their copies at a detected mark), the use of concatenated code consisting of a frameproof code as inner code and an IPP code as outer code is proposed. The erasure analysis is also extended to concatenated codes. The equivalence between Gossip codes and t-designs may help in further analysis and in the use of these codes and designs. Performance issues of digital fingerprinting systems are also discussed. The attacks on fingerprinting including collusion attacks and some attacks similar to attacks on watermarks too are discussed. This thesis is divided into seven chapters.

TABLE OF CONTENTS

Chapter 1

Introduction

1.1 Motivation

Protecting digital content against illegal copying and distributing is one of the key issues being faced by owners and distributors across the digital world. Copyrights are an accepted means of legally encouraging the creation of works and their rightful exploitation. Recent developments in digital information processing and distribution have had a tremendous impact on the creators and their copyrights. On one hand, these advancements open new ways of creating and exploiting digital works while on the other, the same advancements open up novel ways of circumventing copyrights. These techniques can be used against the interests of copyright holders by creation of less copies and their efficient distribution. Therefore, new digital rights protection techniques are needed, which allow the creators and other right-holders to enforce their legal rights and interests. The overall goal of these techniques is to assure sufficient incentives for creating works and making them available in the digital form.

Access control techniques such as encryption ensure that a digital copy is accessible to the person who owns the decryption key. However, when the content is decrypted, multiple copies can be created and redistributed. This raises an additional requirement to produce evidences for copyright infringes to law enforcement agencies apart from detection. The identification of culprits involved in creating pirated copies is very much required to protect the distributor's rights and identify any violations of Intellectual Property Rights (IPR).

Digital fingerprinting is an emerging and promising solution to identify copyright infringes and for management of digital rights. Digital fingerprinting makes the copies of a digital object user specific by embedding a unique identifier or serial number inside each copy. This allows the distributor to distinguish illegal users among the authorised users. Marking in digital copies deters users from violating copyrights from the fear of being caught. There are numerous applications where digital fingerprinting plays an important role, which include pay TV, content monitoring and digital evidence etc.

1.2 Problem Statement

Digital fingerprinting refers to the act of embedding a unique identifier (*i.e.*, a codeword of a q'ary code) in a digital object, in such a way that it is difficult for others to find or destroy the identifier. This marking makes every user's copy unique, while still being close to the original. If the users do not cheat, this creates no problem, but a malicious user may try to erase the unique identifier (*fingerprint*) from his copy before distributing the illegal copies. In order to prevent such frauds, it is natural to embed the digits of the fingerprint into locations of the digital document that are unknown to the users. This way, the user cannot erase just the fingerprint without damaging the digital document.

The unique fingerprinted copies of an object prevent simple redistribution problems. When a group of malicious users (pirates) collaborates, each of them gains access to one unique fingerprinted copy of the document. Comparing these copies, they can isolate the positions where the copies differ and these positions hold digits of the fingerprint. They can erase these digits of the fingerprint or they can even introduce arbitrary digits in these positions. Such a strategy, results in a document (pirated copy) that is not identical to any of the legitimate (fingerprinted) copies but is identical with all of them on relevant positions.

Thus in a q'ary digital fingerprinting system the colluders construct pirate copies by comparing legitimate copies and finding the differences to detect the embedded

marks. In this process of creating pirate copies, the pirates may try to frame innocents who are not involved in piracy. Further, they can also erase some of these marks to make them unreadable or can even introduce arbitrary digits in these positions.

A fingerprinting code Γ is a set of fingerprinting codewords with the following identification property: If z is a word that is generated by a coalition of codewords $C \subset \Gamma$, then there exists a tracing algorithm that recovers at least one codeword in C. In a fingerprinting scheme, a watermarking algorithm is used to embed a different fingerprinting codeword in each copy of the object, thus making each copy unique. Therefore, if dishonest users redistribute their fingerprinted copy without modification, they unambiguously incriminate themselves.

The problem that is dealt within this work is construction of collusion secure fingerprinting codes with short length, which can identify at least one member of the pirate collusion responsible for creating an illegal digital copy.

The overall goal of this work is to construct and analyse non-binary fingerprinting codes such that it is possible for the distributor to identify at least one pirate of the guilty coalition even in the presence of erasures in the recovered fingerprint, and without accusing any innocent user. In this thesis, we examine various coding schemes for their applicability in collusion secure digital fingerprinting and pirate tracing. The study of the combinatorial properties of these codes in turn assist in constructing codes with good performance. This work also aims at exposing these relations between fingerprinting codes and combinatorial designs.

We also investigate the factors that influence the performance of these fingerprinting codes and the choices available for the content owners or authorised distributors to handle digital piracy. Specifically, we concentrate on non-binary codes that provide deterministic tracing and resist erasures created by a coalition. We deal with fingerprinting codes for digital data in the context of several users colluding together to create an unauthorised and rather untraceable copy. For protection of copyrights,

we investigate non-binary fingerprinting systems (design) and their effectiveness in tracing traitors. We examine IPP codes and Gossip codes, in particular. We show the combinatorial equivalence between Gossip codes and a variety of block designs by explaining their constructions from Traceability schemes, t-designs and vice versa.

1.3 Contribution

A comprehensive study of digital fingerprinting techniques is carried out to understand and investigate their scope and applications in various domains. We have observed that various (binary) codes with probabilistic tracing and fewer codes with deterministic tracing are proposed in literature. We have explored the choices available during the use of binary and non-binary codes for fingerprinting applications. In this work, fingerprinting codes that offer collusion resistance and the combinatorial properties of these codes are investigated. We have considered a simple fingerprinting model that uses a code where every collusion group creates a unique pirate fingerprint. Under this model, the theoretical bounds on non-binary fingerprinting codes namely fingerprint length and collusion size are presented. In the traitor tracing scenario, we have also provided a tracing method to find the contribution made by authorised users to pirate decoders in Boneh and Franklin's [BF99] broadcast encryption method.

In our work, we have presented two new construction techniques for Gossip codes to achieve the minimum code length specified for Gossip codes in terms of code parameters (*i.e.*, number of codewords M, alphabet size q and collusion size c). These codes are non-binary IPP codes and can identify one parent codeword from the pirate word. The first method is construction of Gossip codes from t-designs, which are combinatorial block designs. The partition method (based on t-designs) described in this work (chapter 5) construct shortest Gossip codes by partitioning the set of all collusion groups (of size c) into disjoint sets. Each disjoint set (equivalence class) defines the accusation sets for one gossip column. The collusion groups whose members can be traced (accused) by a gossip column are known as accusation groups

of that column. These accusation sets in turn define the non-zero positions (indices) for each gossip column that constitute a Gossip code.

The second construction method is from c-Traceability Schemes, which are used in broadcast encryption applications. In a Traceability Scheme each authorised user has a decoder with a set of k keys, from a base key set X that uniquely determines the owner and allows him to decrypt the encrypted broadcast. In this construction method each decoder (private key) defines one gossip column, which in turn constitute the code.

These results are also reliable to make out whether a *shortest* Gossip code exists or not, for the chosen code parameters namely M, q and c. We also present the construction of Traceability Schemes and t-designs from Gossip codes. Gossip codes allow deterministic tracing of pirates and also come up with an efficient tracing algorithm. Most of the earlier works on fingerprinting are usually probabilistic in some way or the other. There are no explicit construction methods for the shortest Gossip codes prior to our work.

In many cases such as broadcast encryption applications and fingerprinting, the number of users in a scheme will increase after the system is set up. Initially the distributor will construct a scheme that will accommodate a fixed number of users say M. If the number of users exceeds M, the scheme need to be extended such that it is compatible with the existing scheme. Such scenarios are possible in Traceability Schemes and digital fingerprinting applications. In Traceability Schemes, it is not advisable to change the keys already issued, when the users in the system grow. In the case of fingerprinting it is not possible to recall the digital copies that are distributed with embedded fingerprints. In our work, the construction of embedded Gossip codes for extending an existing Gossip code to a bigger code is also described. Embedded Gossip codes can also be used to construct embedded Traceability Schemes and embedded frameproof codes which can expand a broadcast application such that it is compatible to existing scheme.

There are many advantages of shortest Gossip codes. Some of them are listed here. In fingerprinting, shortest Gossip codes cause less distortion during embedding due to the shorter length of codewords as compared to normal Gossip codes. This is due to the fact that the modifications to be made in the original object are less. These codes provide deterministic tracing for pirates and tolerate erasures. It is also possible to construct t-designs and Traceability Schemes from these codes. Many applications of t-designs in design theory and broadcast applications for Tractability Schemes are known. The construction of frameproof codes is also possible from shortest Gossip codes.

In our work, another new aspect is handling of erasures in these (shortest) Gossip codes. In the erasures model, tracing pirates is possible when pirates create erasures (non-alphabet symbols) or alphabet symbols that occur in one of their copies in detected positions. Undetected positions by the pirates remain unaltered in the pirate copy. In our work, tracing in presence of erasures is categorised (and carried out) into three explicit scenarios - namely no erasures, only erasures and selective erasures.

We also present the realisation of pirate model (unreadable digit model) through concatenated codes. In the unreadable digit model, the pirates choose an alphabet symbol found in their copies or an erasure (denoted by '*e*' or '?') at detected positions. In order to enforce this choice for pirates (and to prevent pirates from choosing an alphabet symbol not found in their copies at a detected mark), the use of concatenated code consisting of a frameproof code as inner code and an IPP code as outer code is proposed in this work. The erasure analysis is also extended to concatenated codes. The efficient and deterministic tracing of Gossip codes is the advantage. The equivalence between Gossip codes and t-designs may help further analysis and the use of these codes and the designs. Performance issues of digital fingerprinting systems are also discussed. The attacks discussed on fingerprinting include collusion attacks and some are similar to that of attacks on watermarks.

1.4 Organisation of the thesis

The chapter 2 begins with a discussion on fingerprinting and presents the areas closely related to digital fingerprinting namely coding theory, traceable schemes, watermarking and traitor tracing methods. It also provides a comparison between binary vs. non-binary digital fingerprinting codes and gives an elaborative description on fingerprinting codes namely frameproof, IPP and traceability codes. It also presents general pirate strategies such as majority choice and minority choice. An introduction of various fingerprinting models is also presented. It also presents elliptic curve analog of Boneh-Franklin's [BF99] broadcast encryption scheme and related work. This chapter concludes with various applications of digital fingerprinting.

The chapter 3 details various tracing methods for stored content. In particular it explains soft tracing, deterministic and probabilistic tracing methods apart from possible errors in tracing. It presents a generic tracing algorithm for IPP codes. In key fingerprinting scenario it also presents a tracing algorithm for Boneh-Franklin's [BF99] broadcast encryption scheme.

The chapter 4 describes some theoretical bounds on digital fingerprinting namely fingerprint length and collusion size. It considers a simple fingerprinting model and presents these bounds for non-binary fingerprinting codes. This model requires exactly one unique fingerprint to be created by each collusion group to trace the whole collusion group responsible for creating an illegal copy. This chapter also presents a bound on error probability while trying to identify a member of the collusion group responsible for creating an unauthorised copy.

The chapter 5 presents the construction and analysis of a non-binary IPP code namely Gossip code. Two methods of construction for Gossip codes, that achieve shortest code length, are presented namely from t-designs and Traceability Schemes. The converse part *i.e.*, construction of Traceability Schemes and t-designs from Gossip codes is also demonstrated. In addition to this the construction of embedded Gossip codes is also described for extending an existing Gossip code to a bigger code. It also

provides the expressions for finding shortest length, weight and distance of Gossip codes based on code parameters M, q and c. Concatenated codes are presented for the realization of the pirate model and tracing. The performance issues are also discussed in this chapter.

The chapter 6 discusses erasures in non-binary codes mainly in Gossip codes. We discuss three types of erasures namely no-erasures, only-erasures and selective-erasures in embedded fingerprints. We also present the tracing methods for all these cases. The analysis also extended to concatenated codes. We also analyze various other erasure models and their applicability to Gossip codes.

The chapter 7 consists of conclusions and future scope followed by annexure.

Chapter 2

Digital fingerprinting

2.1 Introduction

In this chapter, we present the background of digital fingerprinting. It describes the areas related to fingerprinting such as error correcting codes, Traceability Schemes, combinatorial designs and watermarking techniques. The chapter details various design choices available for fingerprinting namely different types of codes with emphasis on fingerprinting models and pirate strategies. It also includes a brief summary of related works and applications of digital fingerprinting.

Digital fingerprinting is making copies of the same data uniquely identifiable by hiding additional information (a fingerprint) in the data [GP99]. Watermarking techniques [CMB01], a branch of signal processing, can embed the additional information in the digital content. If the distributor finds an illegal copy of a sold (and fingerprinted) work, the embedded (customer's) information can be extracted to identify the culprits. There are two varieties of codes available to construct fingerprinted objects, namely binary and non-binary codes. These choices have been arrived at using the embedding alphabet size q. Binary fingerprinting codes use two symbols denoted by 0 and 1, where as non-binary codes use more than two alphabet symbols. Thus the alphabet size '$q = 2$' for a binary code where as '$q > 2$'for a non-binary code.

In a typical binary fingerprinting system '*zero*' may be encoded by modifying a particular location (mark) of the original, and '*one*' may be encoded as no change in the original. An *original* is the digital object created at first, for distribution of its fingerprinted copies, by someone who has the legal right to do so. The distributor keeps the marking positions secret. Binary fingerprinting systems allow only one alternative at each mark. It is possible to use more than one alternative for each mark,

which leads to non-binary fingerprinting. This marking makes each copy unique and all the copies are similar otherwise. A detected mark differs among compared objects and an undetectable mark is same in all compared objects.

One of the major applications of digital fingerprinting is copyright protection. There are many approaches for copyright protection, which can be classified as *preventive measures* and *deterring measures*.

Deterring measures do not aim at preventing copyright violation but make copyright violations detectable, verifiable and thus prosecutable. Prominent examples of deterring measures are dispute-resolving techniques and forensic analysis [SK04]. *Dispute resolving techniques* are used to resolve conflicts about digital works, where two or more disputants claim to be the creator or copyright holder of a certain work. These techniques ideally aim at identifying the *real* creator of the disputed work and resolve the dispute in favor of the *real* creator.

2.2 Related areas

Digital fingerprinting incorporates many disciplines including error correcting codes, cryptographic traceable schemes, design theory, traitor tracing methods and elements of watermarking. Error correcting codes with large minimum distance are suitable for fingerprinting applications. They help in constructing fingerprinting codes and correcting errors in embedded fingerprints. The decoding algorithm of error correcting codes can also be a tracing algorithm for identifying the culprits who created pirate copies. The traitor tracing methods provide efficient tracing algorithms for fingerprinting applications. Combinatorial designs provide the construction techniques for fingerprinting codes and assist in studying the combinatorial properties of these codes. Watermarking techniques provide methods for embedding these fingerprints imperceptibly so that the malicious users do not detect the embedded fingerprint.

2.2.1 Error correcting codes

A code over a finite alphabet Q is a non-empty subset Γ of size M of Q^N, where Q^N is a vector space of N-tuples. In a (N,M,q) code the parameter N is called the code length, M is the code size and q is the alphabet size. The elements of Q are called as the digits of the alphabet or alphabet symbols. Each code consists of a set of words of constant length, also known as codewords. The dimension (or information length) of Γ is defined by $k = \log_q M$, and the rate of Γ is $\hat{R} = k/N$.

The Hamming distance $d(x,y)$ between two words $x, y \in Q^N$ is the number of digits in which x and y differ. Let Γ be a (N,M,q) code over Q with $M > 1$. The distance of the code Γ is the minimum Hamming distance between any two distinct codewords of Γ, which is given by

$$d_{\min} = \min_{c_1, c_2 \in C : c_1 \neq c_2} d(c_1, c_2)$$

A (N,M,q) code with minimum distance d is called a $(N,M,q:d)$ code.

Any subset of the code Γ is known as a subcode of Γ. The number of non-zero digits in a codeword x is called the *weight* of x and is commonly denoted as $w(x)$.

An error correcting code is a code that is capable of correcting errors occurred due to noise in transmission channels. If a codeword is transmitted through a communication channel, the received word is usually a corrupted version of the sent word due to inherent presence of noise in the channel. If x the transmitted codeword and y is received word then the error pattern $\hat{e}$ of y satisfies $y = x + \hat{e}$. The task of the decoder is to estimate the sent word from the received word. If the number of errors $w(\hat{e})$ is greater than $\lfloor (d-1)/2 \rfloor$, then there can be more than one codeword within distance $w(\hat{e})$ from the received word. Then the decoder may either decode incorrectly or fail to decode at all. To overcome this problem the concept of *list decoding* [VG01, KD98, GSW00] is introduced, where the decoder outputs a list of all the codewords within distance $w(\hat{e})$ of the received word, thus offering a potential way to recover from errors beyond the error correction bound of the code. We present the definitions

of some error correcting codes namely Linear code, Cyclic code and Reed Solomon code.

Linear Code: A Linear Code is a code in which the sum of two codewords is also a codeword.

Cyclic code: A code is called cyclic if $(x_N, x_0, x_1, ..., x_{N-1})$ is also a codeword whenever $(x_0, x_1, ..., x_{N-1}, x_N)$ is a codeword.

The cyclical nature of these codes helps in efficient implementation of encoding and decoding processes inside hardware. The cyclic codes are also linear codes.

Reed Solomon code: Reed-Solomon (RS) codes [RS60] are a special type of cyclic codes. To define it we consider F_q as the finite field with q elements (information symbols) and $1 \le k < N \le q$. We fix any N elements of F_q, say $a_1, a_2, ..., a_N \in F_q$. To encode a packet of k information symbols $(m_1, m_2, ..., m_k)$ where each symbol is an element of F_q, first the message polynomial $f(z) = m_1 + m_2 z + ... + m_k z^{k-1}$ is formed and then $b_i = f(a_i) \in F_q$, $i = 1, 2, ..., N$ are computed. The corresponding codeword is $(b_1, b_2, ..., b_k)$. When information symbols take all possible values in F_q, we get q^k codewords of length N that define the code.

RS codes are commonly used in compact disc players and in construction of fault tolerant systems such as RAID [JP97]. Recent studies [SSW01, KY02] revealed that these codes have a potential application in cryptography and digital fingerprinting. The hardness in decoding RS codes can be exposed to construct new cryptographic applications. These codes are also used in constructing fingerprinted codes [SW02b] such as IPP codes and concatenated codes [FS02].

2.2.2 Traceability Schemes

Traceable Schemes [SW99, SW98, FN93, KD98, GSY99] are cryptographic systems that provide protection against illegal copying and redistribution of digital data. A closely related concept is traceability systems introduced by Chor, Fiat and Naor [CFN94] used in the context of broadcast encryption schemes. Broadcast encryption systems [SW99] allow targeting of an encrypted message to a privileged group of receivers. Each receiver has a decoder with a set of keys that allows him to decrypt

the encrypted messages if he is a member of the target group. When a group of colluders (up to c) construct a pirate decoder to decrypt the content, broadcast encryption systems [NP00, DFKY03] can identify at least one of the colluders who has contributed to the pirate decoder. In tracing mechanism of [SW99], the merchant finds the numbers of common keys between the pirate decoder and all authorised decoders. The merchant then chooses the decoder with the maximum number of common keys with the pirate decoder, which exposes a culprit. These schemes are known as Traceability Schemes and are also called *key fingerprinting schemes* [SW99], which allow tracing of one of the colluders.

A Traceability Scheme $c\text{-}TS(k,b,v)$ is a broadcast encryption scheme, where k is the number of keys provided to each user, b is the total number of users, v is the total number of base keys and c is the collusion size of pirates that the scheme can withstand. Suppose X is a set and B is a family of subsets of X (called blocks) each of size k. A Traceability Scheme can be considered as a set system (X,B) where each block corresponds to a decoder with k keys from the base key set X. The traceability scheme $c\text{-}TS(k,b,v)$ is described in [SW98] as a set system (X,B) with an additional property as follows.

Theorem 2.1 *[SW98]* There exists a $c\text{-}TS(k, b, v)$ if and only if there exists a set system (X, B) such that $|X|=v$, $|B|=b$ and $|P|=k$ for every $P\in B$, with the property that for every choice $w\le c$ blocks $B_1,...,B_w\in B$ and for any k-subset $F\subseteq \bigcup_{j=1}^{w} B_j$, there does not exist a block $P\in B-\{B_1,...,B_w\}$ such that $|F\cap B_i| > |F\cap P|$ for $1\le j\le w$.

2.2.3 Traitor tracing

Digital fingerprinting for content distribution is an important scenario for traitor tracing, where the legitimate subscribers may try to find the differences of their fingerprinted copies to create illegal copies by modifying or erasing the marks embedded in their content. The method of identifying illegal users who created a pirate copy is called pirate tracing, also known as traitor tracing.

Fiat and Tassa [FT99] introduced *dynamic* traitor tracing schemes. Their schemes adapt themselves throughout the algorithm in order to force the pirate to reveal more and more information and eventually, the algorithm locates all traitors (or stops piracy). An additional feature of the dynamic scheme of [FT99] is that there is no need for a priori bound on the possible number of traitors, as the algorithm adapts itself when a new traitor is discovered. The traitor's aim is to bypass the security mechanism of the system, not by constructing a pirate decoder but by re-broadcasting the content after it is decoded. The colluders use their decoders to decrypt the content and after decryption they rebroadcast the content in plain text to another group of users. In this case, the only way to trace the traitors is to use different versions of the content obtained by embedding a distinct mark, for different users. This allows a re-broadcasted message to be linked to the subgroup of users who were given that version. The two main characteristics of this dynamic setting are (*i*) the plain text is marked and, (*ii*) there is a feedback from the channel which allows the traitor to become localised.

2.2.4 Combinatorial designs

Combinatorial designs [CD96] mainly consist of Balanced Incomplete Block Designs (BIBD), orthogonal arrays and latin squares. *A block design* is a pair (X, B), where X is a set of v points and B is a family of k-subsets (called blocks) of X. A BIBD is a pair (X, B) where X is a v-set and B is a collection of k-subsets of X (called blocks) such that each element of X is contained in exactly r blocks and any 2-subset of X is contained in exactly λ blocks. A BIBD (X, B) with parameters v, b, r, k, λ is complete if no two blocks are identical and contains v_{C_k} blocks.

The necessary conditions for the existence of a BIBD(v, b, r, k, λ) are

1. $vr = bk$
2. $r(k-1) = \lambda(v-1)$

t-designs [CD96]: A block design is called t-design if any t-subset of X occurs in exactly λ blocks in B. Tables 2.1, 2.2 and 2.3 present few t-designs. Each column in these examples represent a block corresponding to that particular t-design.

Isomorphism in t-designs[CD96]: A block design can be modeled as a coloured graph. Given a design $D=(X,B)$ where $X=\{x_1,x_2,...,x_v\}$ and $B=\{B_1,B_2,...,B_b\}$, let $G(D)$ be a graph with vertex set $\{x_1,x_2,...,x_v,B_1,B_2,...,B_b\}$, with the element vertices having one colour, and block vertices having a second colour, and the edge set $\{\{x_i,B_j\}\,|\,x_i\in B_j\}$. The designs D_1 and D_2 are isomorphic if the graphs $G(D_1)$ and $G(D_2)$ are isomorphic.

Table 2.3 presents two non-isomorphic t-designs. In chapter 5 we present the construction of Gossip codes from t-designs. Non-isomorphic t-designs can be used to construct different Gossip codes [TL01] for the same set of code parameters M, q and c. Gossip codes constructed from isomorphic t-designs behave in a similar manner in handling erasures.

1	1	1	1	2	2	2	3	3	3	4	7
2	4	5	6	4	5	6	4	5	6	5	8
3	7	9	8	9	8	7	8	7	9	6	9

Table 2.1. 2-(9, 3, 1) design

1	1	1	1	2	2	2	3	3	4	4	5	6
2	3	5	7	3	6	8	4	7	5	8	9	a
4	9	6	b	5	7	c	6	8	7	9	a	b
a	d	8	c	b	9	d	c	a	d	b	c	d

Table 2.2. 2-(13, 4, 1) design

1	1	1	1	1	1	2	2	2	2	2	3	3	3	3	3	4	4	4	5	5	5	6	6	6	7
2	4	6	8	a	c	4	5	7	a	b	4	5	7	8	9	7	8	9	7	9	b	8	9	a	8
3	5	7	9	b	d	6	8	9	c	d	a	6	b	d	c	c	b	d	d	a	c	c	b	d	a

1	1	1	1	1	1	2	2	2	2	2	3	3	3	3	3	4	4	4	5	5	5	6	6	6	7
2	4	6	8	a	c	4	5	7	a	b	4	5	7	8	9	7	8	9	7	9	b	8	9	a	8
3	5	7	9	b	d	6	8	9	c	d	a	6	b	d	c	c	b	d	d	a	c	b	c	d	a

Table 2.3 Two non-isomorphic *t*-designs

An *orthogonal array [CD96]* $OA(t, k, v)$ is an $k \times v^t$ array with entries from a set of $v \geq 2$ symbols, such that in any t rows, every $t \times 1$ column vector appears exactly once.

2.2.5 Watermarking techniques

Watermarking is about hiding the copyright data in images or multimedia objects, which are subject to illegal use. Digital watermarking is the process of embedding a signal (watermark) imperceptibly into a host signal (cover). When copyright issue arises, the watermark is unambiguously recovered from the watermarked signal (stego signal) to resolve rightful ownership issues. The main requirements of a watermarking technique are robustness, imperceptibility and security. Robustness is the ability of a watermark to survive intentional and inadvertent distortion. The watermarking process must not affect the fidelity of the content and for this reason the embedded watermark must be imperceptible. Watermarking must be secure to prevent unauthorised detection, embedding or removal. There are several watermarking techniques [KP00, CMB01] such as Least Significant Bit (LSB) insertion, Discrete Cosine Transform (DCT) [CKLS97], and Discrete Wavelet Transform (DWT) [XBA97] methods. The DWT separates an image into a lower resolution approximation (*LL*) as well as horizontal (*HL*), vertical (*LH*) and diagonal (*HH*) detail components.

2.3 Design choices in digital fingerprinting

Fingerprinting techniques embed a unique codeword into a digital object to make it distinct. The embedded codeword is extracted from the pirate copy for tracing its supporting traitors. There are two varieties of codes available to construct fingerprinted objects, namely binary and non-binary codes. These choices have been arrived at using the embedding alphabet size q. Some objects to be fingerprinted may be more suitable for embedding with a specific q. In the following sections, we detail various choices available for fingerprinting namely the types of codes, different fingerprinting models, the strategies available with pirates and the distributor's counter measures that come along with different tracing methods.

2.3.1 Digital fingerprinting codes

A fingerprinting code is a set of codewords that are embedded in each copy of a digital object, with the purpose of making each copy unique. In order to protect a product, the owner or distributor marks each copy with some codeword and then ships it. This marking allows the distributor to detect the unauthorised copy and trace it back to its owner. The choices for the malicious user for creating a pirate copy is governed by the feasible set of pirate words, explained below.

Feasible or Descendent Set [SSW01]: Consider a code Γ of length N on an alphabet Q with $|Q|=q$ whose symbols are denoted by $\{0, 1,...,q-1\}$. Then $\Gamma \subseteq Q^N$ and we will call it a (N, n, q)-code if $|\Gamma|=n$. The elements of Γ are called codewords. Each codeword is given by $x=(x_1,...,x_N)$, where $x_i \in Q, 1 \le i \le N$. For any subset of codewords $C \subseteq \Gamma$, we define the set of descendents of C, denoted $D(C)$ by $D(C)=\{x \in Q^N : x_i \in \{z_i : z \in C\}, 1 \le i \le N\}$.

The set $D(C)$ consists of N-tuples that could be produced by a coalition holding the codewords of the set C. Descendent set $D(w^i, w^j)$ is the set of all possible illegal codewords that the collusion set $\{w^i, w^j\}$ can create, where w^i is the i^{th} codeword.

A. Frameproof, Secure, IPP and TA codes

A coalition of users may detect the marks by comparing multiple fingerprinted objects and modify or erase them. In this process, the pirates may further try to frame innocent users. To prevent this, Boneh et al [BS98] introduced frameproof (FP) codes in which the pirates cannot frame innocent users outside their coalition, *i.e.* they cannot create a pirate codeword that is not a member of their coalition. Secure codes, IPP and TA codes additionally allow the distributor to identify the pirates.

Secure code: A code Γ is called totally c-secure if there exists a tracing algorithm A satisfying the following condition: if a collusion $C \subseteq \Gamma$, $|C| \le c$, generates a word x then $A(x) \in C$. That is, if a fingerprinting code is c-secure, then the decoding of a

pirate word created by a coalition of utmost c dishonest users will expose at least one of the guilty parties. An example for c-secure code is presented in [BS98].

Frameproof code [BS98]: Let Γ a (N,n) code and $D(W)$ be the feasible set of W where $W \subseteq \Gamma$. Γ is called c-frameproof code *i.e.* c-$FP(v,b)$ code if, for every $W \subseteq \Gamma$ such that $|W| \leq c$, we have $D(W) \cap \Gamma = W$.

IPP code [HLLT98]: A code Γ has c-Identifiable Parent Property (c-IPP) if no coalition of size at most c can produce a N-tuple that cannot be traced back to at least one member of the coalition. IPP codes meet an additional condition over frameproof codes. The condition is that the tracing algorithm must identify at least one traitor who participated in the coalition. Thus a fingerprinting scheme incorporating codewords of IPP code can find at least one user who participated in constructing a pirate copy. Non-binary IPP codes can handle a large collusion size c (> 2) [BCEKZ01] unlike binary IPP codes. Gossip codes are introduced in [TL01] with a tracing algorithm in for collusion secure fingerprinting. Every Gossip code is an IPP code.

Gossip Code [TL01]: A gossip column corresponds to a (q-1) subset of $\{1, 2,..., M\}$ representing the indices of the nonzero alphabet symbols in that column, and in a Gossip code each c-subset of $\{1, 2,..., M\}$ must be contained in at least one such subset. A detailed explanation of Gossip codes is presented in Chapter 5.

Traceability code [SSW01]: A code Γ is called a c-traceability code, if for all $C_i \subseteq \Gamma$, for all $x \in desc(C_i)$, and for all $z \in \Gamma \setminus C_i$, there exists $y \in C_i$ such that $d(x, y) < d(x, z)$.

The relationship between error correcting codes and c-TA codes is given by the following theorem.

Theorem 2.2 *[SSW01]*

Let Γ be an $(L,n,d)_q$ error correcting code, and c be an integer. If $d > \left(1-\frac{1}{c^2}\right)L$ then Γ is a c-$TA(L,n)$ code.

Figure 2.1 was originally presented in [SSW01] which describes the relations between various fingerprinting codes *and* Hash Families [SWZ00]. It can be observed from the figure that an IPP code is also a frameproof code and every traceability code is an IPP code. Further a (*w, w*)-SHF is identical to a *w*-SFP code. Similarly the relations between perfect hash families and IPP codes can also interpreted.

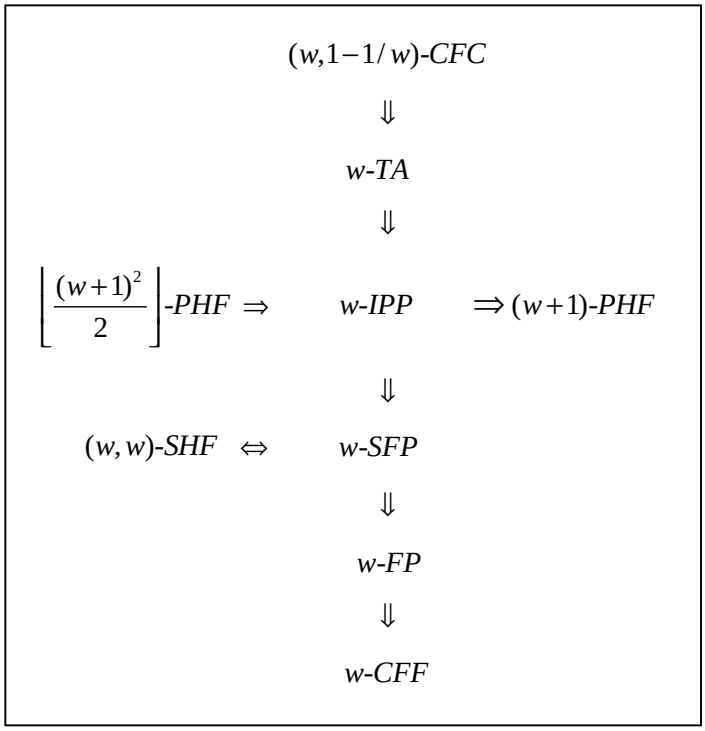

Figure 2.1. Relationships among different types of codes and hash families

CFC	Cover-Free Code
CFF	Cover-Free Family
PHF	Perfect Hash Family
SHF	Secure Hash Family

TA	Traceability Code
IPP	Identifiable Parent Property Code
FP	Frameproof Code
SFP	Secure Frameproof Code

B. Binary and non-binary DF codes

We compare binary ($q=2$) and non-binary ($q>2$) codes and focus on their advantages and disadvantages. If an embedded mark is non-binary, the illegal word extracted from a pirate copy will most likely exclude a large segment of innocent users. In this case, the number of different symbols q is large but they are distributed among the same set of users. This makes the number of users with the same symbol at each position smaller, as compared to a binary fingerprinting system, and thereby we can exclude more users. Thus the probability for accusing innocents will diminish. This is an advantage for non-binary codes over binary codes.

An embedded word in the non-binary system will probably have more detectable positions than in a binary system because the number of different symbols is large. This makes the number of users with the same symbol at each position smaller in non-binary codes and thereby increases the probability of detecting a marked position. This makes collusion attacks stronger for non-binary case. An advantage for binary codes over non-binary codes is that it will probably be easier to fingerprint digital objects with binary codes as q is small. In case of non-binary codes it is the embedding technique, which encodes the alphabet symbols during fingerprinting, that limits this choice of (large) q.

C. Random codes and known codes

If a random code [LW98] is used for fingerprinting purpose, the code itself is the embedded code. So it must be kept secret. If known code (public code) is chosen for fingerprinting, a code equivalent to the public code is used for embedding. However, the marking positions and embedded code are kept secret in known code model. Keeping embedded code secret guarantees that the pirates do not know which mark corresponds to which public code symbol. They also do not know which codeword

belong to which user and how the symbols in the codeword are represented in the object.

2.3.2 Digital fingerprinting models

There are many possible scenarios for fingerprinting. In each scenario the settings are different and the properties of digital fingerprinting depends on these settings. A good number of fingerprinting models have been proposed in existing literature. Some of the possible situations and their requirements along with the notations are described here under.

A. Static fingerprinting models

Fingerprinting schemes for static scenario were introduced and discussed by Boneh and Shaw in [BS98]. In their study they assumed that the content is watermarked once, prior to its broadcast. The schemes of [BS98] were designed to trace the source of piracy once a pirate copy of the content is captured. Fingerprinting of stored content is required for preventing the piracy of multimedia files or images or subscriptions of a digital library. The content remain offline with the users. So unless a pirate copy is identified or suspected it cannot be possible to trace the culprits. This is static scenario and there is no feedback from the pirate network or users of the distributed copies for detection of piracy.

In the static setting there is a one time marking of the content per user. Only when a black market copy is found, the tracing and incrimination algorithm is activated. This model is suitable for, e.g., DVD movie protection. The performance in such a rigid setting with no on-line feedback is less efficient than that in the dynamic setting. This setting is also somewhat less useful than the dynamic setting because there are fewer effective countermeasures like legal action, as opposed to the dynamic setting that allows immediate disconnection of the traitors. In the following text we present the marking assumption [BS98] for stored content.

Assumptions in fingerprint marking

Every fingerprint is a collection (or a sequence) of embedded marks. The number of possible states for these marks is given by the alphabet size of the fingerprinting code. By colluding, users can detect a specific mark if it differs between their copies; otherwise a mark cannot be detected. The main property that the marks should satisfy is that, the users cannot change the state of an undetected mark without rendering the object useless. This is known as marking assumption originally proposed by Boneh and Shaw [BS98]. The assumption is motivated by the fact that it is much easier for the colluding pirates to make changes in places where the copies differ. In case of binary fingerprinting, the state of the detected mark can be changed to any of the following states {0, 1, ?}. The symbol '?' means that the mark is unreadable, *i.e.*, it is impossible to determine if the mark is 0 or 1. Thus any collusion can only create pirate objects containing fingerprints in the collusion's *feasible set* [SW98], possibly with some marks erased. The conjectures of marking assumption are:

1. No mark is affected by another mark.
2. Marked (fingerprinted) object is close to the original regardless of which combinations of alphabet symbols (say 0's and 1's) the marks encode.
3. Colluding groups can find a specific mark if and only if it is detectable. They cannot change an undetectable mark without destroying the fingerprinted object.

There are many methods in static fingerprinting. Some of them are symmetric, asymmetric and anonymous fingerprinting.

A.1 Symmetric fingerprinting

Symmetric fingerprinting assumes the merchant or distributor to be honest. A symmetric fingerprinting model [LL00] is described here under:

1. A fingerprinting code Ψ is chosen by the distributor for usage in the fingerprinting system. The code Ψ is of length N and publicly known. The original digital object that is to be fingerprinted is prepared by locating marks in it, and making alterations in such a way that the object as a whole is slightly

degraded or not at all. In general this is done in such a way that the marking assumption holds.

2. A code Γ is chosen randomly, with equal probability, among all codes equivalent to Ψ.
3. The numbering of the codewords is chosen such that $\Psi = \langle \psi_{\alpha_1}, \psi_{\alpha_2}, .. \psi_{\alpha_M} \rangle$, $\Gamma = \langle \gamma_1, \gamma_2, .. \gamma_M \rangle$ and γ_i is the coordinate-permuted and translated ψ_i for all i.

The objects that are to be distributed are fingerprinted by embedding the codewords of Γ into them. The fingerprinted objects are distributed to the users so that i^{th} user gets the object fingerprinted with codeword γ_i.

A.2 Asymmetric fingerprinting

Asymmetric fingerprinting schemes were first proposed by Pfitzmann and Schunter [PS96] to protect buyer's rights. In this scenario, the buyer need not trust the merchant. The merchant cannot falsely implicate an innocent user for violating copyrights. These schemes internally use a symmetric fingerprinting technique but with additional cryptographic primitives. Thus the computational overload is also more in the case of asymmetric and anonymous fingerprinting as compared to symmetric fingerprinting schemes.

Pfitzmann and Schunter's asymmetric fingerprinting scheme [PS96], removes honesty assumption of the merchant and in case of piracy allows the merchant to identify the original buyer, and is able to construct a *proof* for a judge. Only the buyer can see the fingerprinted data, but if a merchant can capture such an object he can *prove* the identity of the buyer to the judge. Their system requires four protocols; *key-gen*, *fing*, *identity*, and *dispute*, and they all are computationally in polynomial time. The security requirements are:

1. a buyer should obtain a fingerprinted copy if all the protocol participants honestly execute the protocols
2. the merchant is protected from colluding buyers who would like to generate an illegal copy of the data and

3. buyers must be protected from the merchant and other buyers. Although security of the merchant is against collusions of maximum size c, a buyer must remain secure against any size collusion.

Pfizmann and Waidner [PS97] extended this work by giving constructions for large collusions. In these schemes, a symmetric fingerprinting scheme with unconditional security is combined with a number of computationally secure primitives such as signature and one-way functions, and so the final security is computational.

A.3 Anonymous fingerprinting

Anonymous fingerprinting scheme was first proposed by Pfitzmann and Waidner [PW97]. They allow an honest buyer to keep his identity unknown to the merchant who is not assumed to be honest. The merchant only knows the buyer through a pseudo-identity and cannot even link various purchases made by the same buyer. In their model there are four types of participants: buyer, a merchant, a registration center and an arbitrator. The salient features of the scheme are:

1. *key distribution*, which is an algorithm for registration center to generate a key pair
2. *registration*, which is a two party protocol between a buyer and registration center to produce buyer's registration record and registration center's record
3. *data initialization*, which is an algorithm for the merchant to produce merchant's initial data record, which describes the data item to be sold
4. *fingerprinting*, which is a two party protocol between the merchant and a buyer to produce (*i*) fingerprinted data item for the buyer and (*ii*) a purchage record for the merchant
5. *identification*, which is either an algorithm for the merchant or a two party protocol between the merchant and the registration center to trace a traitor
6. *enforced identification*, which is a three party protocol among the merchant, the registration center and the arbiter to trace a traitor and
7. *trail*, which is a two to four party protocol between the merchant and the arbitrator and possibly buyer and the registration center.

Pfitzmann and Waidner's scheme protects against cheating buyers. If a group of buyers produces another illegal copy, and the group size does not exceed a pre-determined size then the merchant is able to obtain a string *proof*, which coincides with a buyer's record.

B. Dynamic fingerprinting Models

Dynamic fingerprinting models [FT99] are used in broadcast scenario. This scenario is applicable for Pay TV or pay per view type of scenarios. The content is broadcasted to a set of legitimate subscribers. Fingerprinting of a broadcast content is required to prevent the rebroadcast threat.

In this case it is possible to obtain feedback from the legitimate users, as they are online or some how connected with the distributor or the broadcaster. The center can therefore see the current pirate broadcast and adapt its watermark distribution in the next segments in order to trace the traitors efficiently. Dynamic schemes decide about the number of active traitors on the fly, based on the feedback from the pirate network, and adapt their behavior accordingly. That is impossible in the static model, where an a priori bound on the number of traitors is required. In dynamic models even if an a priori bound is known, but false incriminations of innocent users are strictly prohibited, there is an exponential performance improvement of dynamic methods over static ones. Some of the methods used in this scenario are dynamic traitor tracing [FT99] and sequential traitor tracing [SW03].

B.1 Dynamic traitor tracing

In dynamic tracing [BPS01, FT99], the *content* is divided into consecutive *segments*, for example, one-minute interval in an audio track. A watermarking algorithm is used to embed one of the marks in the segment, hence creating q *versions* of the segment. In each interval, the user group is divided into q subsets and each subset receives one version of the segment. The subsets are varied in each interval using the rebroadcasted content. The basic idea is to break time into consecutive intervals and modify the watermarking strategy of the system in each interval using the rebroadcasted content. After observing the rebroadcast for a sufficient time, one or

more colluders can be traced. The identified colluders are disconnected from the system and the system proceeds until all colluders are found one by one and get disconnected.

The Model

The assumptions of this model are as follows.

1. Given two variants of a movie segment, v_1 and v_2, the only possible choice for the pirate is to transmit either v_1 or v_2 (*CFN*-model).
2. All variants carry the same information to the extent that humans cannot distinguish between them easily.
3. Given any set of variants, $v_1,...,v_k$, it is impossible to generate another variant that cannot be traced back to one of the original variants, v_i, $1 \le i \le k$.

The setup and terminology used in this model is as follows

1. The center is the source of the content and its watermarked copies.
2. The users, or subscribers, denoted by $U = \{u_1,...,u_n\}$, are recipients of the content.
3. Some of the users may collude in order to distribute illegal copies of the content to non-legitimate users. Such users are referred as traitors to form collusion group. The number of traitors is denoted henceforth by *p*, while the collusion group (pirate) and traitors are denoted by $W = \{t_1,...,t_k\}$, $W \subset U$
4. The marking alphabet that is used to generate codewords is denoted by $Q = \{\sigma_1,...,\sigma_r\}$.
5. For a given segment $1 \le j \le m$ and a mark σ_k; $1 \le k \le r$, $S_k^j \subset U$ denotes the subset of subscribers that got variant σ_k of segment *j*.

Tracing Algorithm

In the dynamic scenario, the pirate W broadcasts at every time segment j (≥ 1), one of the variants owned by the traitors controlled by him, t_i, $1 \leq i \leq c$. Let us denote that variant by s_j and the pirate transmission up to time j by F_j, where $F_j = (s_1, ..., s_j) \in Q^j$.

The goal of the watermarking scheme is to disconnect all subscribers in W, thus rendering the pirate inoperative. Additionally, it would be bad to disconnect innocent subscribers $u \in U \setminus W$. Hence, only deterministic schemes are considered in this case.

Formally, a dynamic watermarking scheme is a function

$$f : U \times Q^* \sim Q \cup \{\sigma_0\}$$

For all $j \geq 1$, f induces a partition of U into the disjoint sets

$$S_k^j = \{u \in U : f(u, F_{j-1}) = \sigma_k\}, 0 \leq k \leq r$$

This is interpreted as follows:

1. At time $j \geq 1$, users $u \in S_k^j$, $k \geq 1$, get variant σ_k of content segment j.
2. At time $j \geq 1$, users $u \in S_0^j$ are disconnected, *i.e.*, get no variant of content segment j.

The following tracing scheme of [FT99] makes use of $r = c + 1$ variants in each segment to trace and disconnect all traitors.

1. *Set* $t = 0$.
2. *Repeat forever:*
   - *(a) For all selections of* r *users out of* U, $\{w_1, ..., w_r\} \subset U$, *produce* $r+1$ *distinct variants of the current segment and transmit the* i^{th} *variant to* w_i, $1 \leq i \leq r$, *and the* $(r+1)^{th}$ *variant to all other users, until the pirate transmits one of the variants.*
   - *(b) If the pirate ever transmits variant* i *for some* $i \leq r$, *disconnect the single user* w_i *and decrement* r *by one. Otherwise, increment* r *by one.*

The performance of the deterministic schemes can be described in terms of the number of variants (r) that they require in each segment, and the number of steps (m) required tracing and disconnecting all traitors. The above algorithm uses $c + 1$ distinct variants and converges in $O(3^c c \log n)$ steps.

Shortcomings of dynamic tracing

The first limitation is that regrouping of the users and mark allocation to users in each interval depends on the rebroadcasted content, also called feedback from the channel. This means that if there is no feedback from the channel no regrouping will occur and so the system is vulnerable to a delayed rebroadcast attack. In this attack, the attackers do not immediately rebroadcast, but record the content and rebroadcast it with some delay so that the broadcaster has no alternative but keeping the mark allocation unchanged. With this attack the system fails completely and cannot trace any colluder.

The second limitation of the system is high real-time computation for regrouping the users and allocating marks to subsets. This means that the length of a segment cannot be very short. In dynamic tracing, the number of segments required by the algorithm to converge grows with the number of users.

Hence, to trace colluders given a fixed length content, the length of the segment must reduce as the size of the user population grows. On the other hand, the computation for repartitioning the group and allocating the versions grows with the size of the group and because of the real-time nature of the computation, the length of the segment cannot be decreased. The conflicting requirements on the segment size, that is, requiring shorter length to provide for longer convergence length, and at the same time the need to have it longer to give time for real-time computation, would result in unworkable systems for large groups.

B.2. Sequential traitor tracing

In [SW03] the authors consider the same scenario as dynamic tracing and propose a different solution, called sequential tracing which removes the shortcomings of dynamic tracing. In sequential tracing, the channel feedback is only used for tracing and *not* for allocation of marks to users. Similar to dynamic tracing, the system can trace all colluders using the channel feed back.

The mark allocation table is predefined and there is no need for real-time computation to determine the mark allocation of the next interval. All other computations related to key management of the group can be performed as precomputation and so the need

for real-time computation will be minimized. Mark allocation in each interval will be according to the table irrespective of the channel feedback. Using a predefined table also protects against the delayed rebroadcast attack, however, the tracing ability of the system will reduce to one traitor. That is, even if the rebroadcast is delayed until the whole content is received, at least one of them will be traced once colluders start rebroadcasting. The system is called *sequential traitor tracing scheme* to emphasise the fact that the traitors are identified sequentially, that is, when a colluder is found he is disconnected and the system proceeds to trace the remaining colluders, and at the same time differentiate it from dynamic schemes. A sample construction of Mark allocation table and tracing algorithm are presented in [SW03]. This construction identifies one of the colluders in steps c^2+1, and *all* colluders in at most c^2+c steps. That is, the scheme converges in c^2+c steps and the convergence length is independent of the size of the user group.

2.3.3 Pirate's choices

Pirate strategy means the technique chosen by pirates to create an unauthorised copy, under some restrictions like marking assumption. In practice, this means that they can choose in each detected mark the alternative from among the q alphabet symbols or something else not in the alphabet *i.e.*, an erasure e. Given the codewords of the pirate group, a strategy f is a mapping from the set of c-tuples of embedded codewords to the set of all possible fingerprints, possibly with erasures. Without knowledge of which codewords the pirates have, the exact result of the pirate's strategy will in general be unknown to pirates.

For the pirates, a strategy f is a random mapping from the set Λ of distinguishable sets of tuple equivalent c-tuples of public codewords to a subset of all possible fingerprints, possibly with erasures, *i.e.*, $f:\Lambda \to \hat{F} \subseteq (GF(q) \cup e)^N$ where $\hat{F}$ is the set of words that can be created by some pirate group. A pirate may try different types of attacks to remove the fingerprint, in order to distribute illegal copies anonymously. Assuming that the pirate has access to a single document copy that has been marked

for him, he may either try to restore the original document by identifying and removing the fingerprint, or try to corrupt the fingerprint by distorting the document. At a detected position of the embedded fingerprint they can also create erasures.

A. Majority choice strategy

Majority choice [BS98] attack requires that the number of pirates is odd and greater than two. In this case, at every detectable mark pirates choose the alternative that occurs the greatest number of times in the available copies.

Minority choice attack can be carried out by inverting the result of the majority choice attack. The difference compared to majority choice is that the illegal fingerprint, on an average, will be as distant as possible to the fingerprints of the pirates.

B. Random choice

In Random choice [GP99] the pirates choose randomly every detectable mark, with equal probability among the possible alternatives, and decides what to output to the illegal copy. A stronger attack results if several pirates collude and compare their independently marked copies. Then they can detect and locate the differences, and combine their copies into a new copy, the fingerprint of which differs from all the pirates.

C. Binary addition

Binary addition takes every detectable mark and the pirates choose to output to the illegal copy the alternative that occurs an odd number of times in the pirate's objects. This does not differ significantly from the normal binary addition.

2.4 Key fingerprinting

Chor et al [CFN94] introduced a special form of cryptography that uses one encryption key and multiple distinct decryption keys, with the property that one cannot compute a new decryption key from a given set of keys. The traitor tracing

schemes of Chor et al [CFN94] adopt the static model, which is suitable for electronic data distribution systems. Two cost measures are to be considered when implementing such schemes: storage requirements at the user end and the necessary increase of bandwidth when the number of users increases. These schemes would be ineffective if the pirate were simply to rebroadcast the original (decrypted) content using a pirate broadcast system.

Pirate decoders that allow access to the content may be manufactured but such decoders, if captured, must inherently contain identifying information that will allow the broadcaster to cut them off from future broadcasts. Additionally, the source of piracy can be detected and legal means can be taken. A broadcast encryption technique and tracing method proposed for this dynamic scenario is public key traitor tracing [BF99]. We present the elliptic curve analog of the encryption technique proposed in [BF99].

2.4.1 Boneh –Franklin Public key Traitor tracing Scheme

Let E be the elliptic curve and G the base point of order p on it.

Key Generation:

The following steps are performed for key generation.

1. For $i = 1,2,...,k$ choose an integer $r_i \in Z_p$ and compute points

   $h_i = r_i G$.

2. Compute $Y = \sum_{i=1}^{k} (\alpha_i h_i)$ for random $\alpha_1, \alpha_2, ..., \alpha_k \in Z_p$.

3. The public key is $\langle Y, h_1, ..., h_k \rangle$.

   A private key is an element $\theta_i \in Z_p$ such that $\theta_i \gamma^{(i)}$ is a representation of Y with respect to base $h_1, ..., h_k$. The i^{th} key, θ_i, is derived from the i^{th} codeword $\gamma^{(i)} = (\gamma_1, ..., \gamma_k) \in \Gamma$ by

$$\theta_i = (\sum_{j=1}^{k} r_j \alpha_j) / (\sum_{j=1}^{k} r_j \gamma_j) (\mathrm{mod}\ p)$$

We refer to the private key as being the representation $\bar{d}_i = \theta_i \gamma^{(i)}$. However, it may be noted that only θ_i needs to be kept secret since the code Γ is public. One can verify that $\bar{d}_i$ is indeed a representation of Y with respect to base $h_1,...,h_k$.

Encryption: To encrypt a message $\bar{m}$ in F_{2^n} first the message $\bar{m}$ is to be embedded onto the elliptic curve E as a point. Let P_M be such a message point. Next a random element $a \in Z_p$ is generated to compute:

$$\hat{S} = P_M + (aY) \text{ and}$$

$$H_i = ah_i \text{ for } i = 1,...,k .$$

The ciphertext $\bar{C}$ is $\bar{C} = \langle \hat{S}, H_1,..., H_k \rangle$

Decryption: To decrypt a ciphertext $\bar{C} = \langle \hat{S}, H_1,..., H_k \rangle$ using the i^{th} user's private key, θ_i, $P_M = \hat{S} - (\theta_i U)$ is computed where $U = \sum_{j=1}^{k} (\gamma_j H_j)$. The original message is recovered as $M = (x$ coordinate of $P_M)$. Here $\gamma^{(i)} = (\gamma_1,...,\gamma_k) \in \Gamma$ is the codeword from which i^{th} user's private key θ_i is derived. Observe that any private key θ_i correctly decrypts the given cipher text.

It is possible to construct pirate decoders for this scheme and decrypt the encrypted content successfully. A tracing algorithm for such pirate decoders is presented in chapter 3.

2.5 Related work

In this section we present the summary of earlier work mainly on digital fingerprinting, which is closely related to our work.

In [SW98], Stinson and Wei presented the construction of frameproof codes from Traceability Schemes. In our work (Chapter 5) we present the construction of much stronger codes *i.e.*, IPP codes from these Traceability Schemes. Safavi-Naini and Wang [SW01] presented a lower bound on maximum possible number of private keys in Traceability Schemes. To enhance the advantage of these Traceability Schemes,

Gafni et al [GSY99] presented a method for adding any desired level of broadcasting capability to the schemes.

With regard to secure codes Boneh and Shaw [BS98] first presented an explicit construction for collusion secure codes. Their code has a length of $O\left(c^3 \log(1/\varepsilon)\right)$ and attained security against coalitions of size c with ε error. Peikert et al [PSS03] demonstrated (on lower bound of code length) that no secure code can have a length of $o\left(c^2 \log(1/\varepsilon)\right)$. To handle erasures created by collusion groups, Guth and Pfitzmann [GP99] constructed binary c-secure codes with ε error with a weaker assumption. Weak marking assumption assumes that an adversary can create only a certain percentage of erasures in place of embedded marks in pirate fingerprints and these erasures are randomly distributed.

For the case of non-binary codes with erasures Safavi-Naini et al [SW02a] gave a construction of q'ary c-secure code that can recover the deleted marks of a shortened fingerprint. In [SW02a] the codeword is repeated adequate number of times so that at least one copy of the embedded codeword can be recovered and thereby increasing erasure tolerance. Tardos [GT03] constructed optimal probabilistic (binary) fingerprinting codes. These codes are for M users that are ε secure against *c* pirates and have a length $O(c^2 \log(M/\varepsilon))$. This construction improved the codes proposed by Boneh and Shaw [BS98] whose length is approximately the square of this length. These codes can withstand erasures in two distinct models for fingerprinting namely weak marking assumption model by Guth and Pfitzmann [GP99] and unreadable digit model [GT03].

Gossip codes are introduced as IPP codes with a deterministic tracing algorithm in [TL01] for collusion secure fingerprinting. The erasure tolerance capability in Gossip codes was not studied earlier to our work. The Gossip code with $c = q - 1$ is studied in [LLS02]. In specific, the conditions for a Gossip code to become traceability (TA) code are presented in [LLS02]. For $q = 3$ the use of this code as an error correcting

code is analysed in [MS99]. Fernández and Soriano [FS02] presented concatenated fingerprinting codes with efficient identification. In this case the inner code use efficient decoding algorithms (Chase algorithms) that correct beyond error correction bound of the code.

2.6 Applications of digital fingerprinting

Digital fingerprinting plays an important role in many real applications. Some of them are owner identification, copyright protection, content distribution, pay television, content monitoring, digital evidence or forensics and Digital Rights Management.

2.6.1 Broadcast monitoring

Consider the scenario of pay TV systems where subscribers may access specific channels or programs by purchasing their viewing rights. In such systems, the content is distributed via terrestrial, cable or satellite broadcast and, hence, a *conditional access* system must be utilised in order to guarantee that only paying subscribers can access the content for which they have paid. To prevent any unauthorised access, cryptography is often used: the conditional access system makes use of secret keys in order to allow only legitimate users access to the content. In this scenario it is also necessary to identify the culprits who simply rebroadcast the content after decryption.
In dynamic traitor tracing [FT99] even if the pirate rebroadcasts the original content to illegitimate users, countermeasures can be activated in order to trace and disconnect such *traitors*. Safavi-Naini and Wang [SW03] introduced sequential traitor tracing schemes for dynamic (active monitoring) systems, which can trace all the traitors one after the other, who are involved in rebroadcast of the received content.

2.6.2 Copyright protection

Digital piracy is one of the major threats being faced by owners and distributors of multimedia content. Contemporary developments in digital information processing and distribution have had a remarkable impact on the creators and their copyrights.

These techniques open up novel ways of circumventing copyrights by creation of multiple copies and their efficient circulation.

In order to protect the digital content the merchant or distributor of a digital work embeds customer-specific information in the form of a unique serial number (*fingerprint*) into the sold work. If the merchant finds an illegal copy of a sold (and fingerprinted) work, he is able to extract the embedded customer information and can identify pirates on the basis of this information. A particular challenge is the security of digital fingerprints against *collusions* of pirate customers, which take the advantage of differently marked versions of the same work and try to derive a new version, which contains no information that identifies one of the pirates. To offer protection against such collusions, the embedded customer information must be *collusion-tolerant*, *i.e.*, it must be guaranteed that no collusion (up to a certain maximum size) can produce a copy of the work for which no member of the collusion can be identified. The challenge from a cryptographic perspective is to find collusion-tolerant encoding of customer identities. Fingerprinting techniques [BS98, LLS02, GP99, SW02a] support copyright protection by deterring the copyright violators.

2.6.3 Content distribution

Consider the example of a digital library, where authorised subscribers of the library should get access to the digital documents. Encryption techniques protect against eavesdroppers, but the main attacks are likely to be from connected legitimate subscribers who proceed to redistribute the received data more than they are entitled to. A mechanism to ensure the security of digital subscriptions is through digital fingerprinting. A *q*'ary fingerprinting system for stored digital objects such as images, videos and audio clips is proposed in [SW02a]. In a scenario where subscribers of digital library, create a pirate copy using two or more valid digital copies, this technique identifies a culprit.

2.6.4 Digital evidence

Digital evidence and forensics [SK04] is required by the law enforcement agencies to resolve disputes arising due to content ownership and distribution. Digital

fingerprinting equipped by traitor tracing methods offers promising methods to check copyright infringes and in constructing evidence to prosecute the culprits.

2.6.5 Digital rights management

Digital Rights Management (DRM) systems comprise many different integrated mechanisms and functionalities such as content/right distribution, payment-systems, access control, copy protection and deterring mechanisms. DRM will enable new content to be made available in safe, open and trusted environments. It enables industry and users interoperability so as not to force content owners and managers to encode their works in proprietary formats or systems. DRM systems can be used to specify the access control mechanisms and user rights like copy once, copy no more, print, read, modify etc. along with copyright notification and transaction tracking. Digital fingerprinting [SW02a, EK03] is an effective means of copyright protection and thereby digital rights management in the digital world.

2.7 Goals of digital fingerprinting

A good digital fingerprinting technique consists of the following:

- a robust embedding algorithm to encode letters from a chosen alphabet at the marking positions,
- a coding algorithm that selects the codewords to be embedded in respective marking positions and
- a pirate tracing algorithm that identifies at least one copy that was used in constructing the pirate or modified object, when a modified object or document is provided as an input.

Ideally the scheme should handle large collusion sizes with shorter fingerprint length to be practically useful. They should also withstand attacks on fingerprints under different pirate strategies including resistance to erasures. There can be many possible goals for digital fingerprinting which include:

- Handle large collusion sizes with shorter fingerprint length.
- Withstand attacks on fingerprints and resist erasures under different pirate strategies.

- Provide an investigation aid to identify the pirates.
- Identify the pirate or, in the case of several pirates working together, identify as many pirates as possible.
- Test whether a certain user, or group of users, is guilty of having created the illegal copy.
- Achieve a trade off between false alarm error and false dismissal error.

2.8 Summary

This chapter details the background on digital fingerprinting. A brief description on some of the areas closely related to digital fingerprinting such as error correcting codes, combinatorial designs and traitor tracing methods is presented. In this chapter we also described the design choices available in digital fingerprinting, the first choice being between binary and non-binary codes. The other options available were choosing from amongst frameproof, IPP and TA codes. Subsequently, we emphasize various fingerprinting models, their requirements and describe relevant notation in each case. The distributor can choose from these models as per the requirement. The choices available for pirates and the countermeasures (tracing methods) are also discussed. Various applications of digital fingerprinting such as copyrights protection, DRM, digital evidence and forensics are covered along with the goals of digital fingerprinting.

Chapter 3

Tracing methods

3.1 Introduction

In this chapter we describe some of the tracing methods for identifying traitors. The chapter distinguishes tracing methods used in fingerprinting of digital content and fingerprinting cryptographic keys. In static scenario it presents a simple tracing method for identifying culprits using a generic IPP code for fingerprinting. It also presents a tracing algorithm for an encryption scheme with multiple decoders [BF99] in key fingerprinting scenario.

3.2 Pirate tracing

Traitor tracing is carried out in two related scenarios: data fingerprinting in the contest of copy protection and key fingerprinting in the contest of broadcast encryption. Traitor tracing in data fingerprinting is a process that takes an illegal fingerprint as input and identify a subset of users who participated in creating the illegal copy. Fingerprinting of digital content protects it from illegal copying and redistribution.

There are two possible scenarios for tracing namely static and dynamic as explained earlier in chapter 2. Tracing can be carried out in static setting (symmetric fingerprinting [LL00], asymmetric fingerprinting [PS96], anonymous fingerprinting [PW97]) or in dynamic scenario (dynamic traitor tracing [FT99], sequential traitor tracing [SW03]).

In Key fingerprinting (public key traitor tracing [BF99]) the aim is to identify the illegal cryptographic keys created from existing legitimate keys for decryption of the encrypted content.

3.2.1 Soft and hard tracing for stored content

There are two complementary types of tracing algorithms for stored content: soft-tracing algorithms and hard-tracing algorithms of traceability codes. In soft-tracing (soft-decision decoding [KV00]), the decoding process takes advantage of side information generated by the receiver. Instead of using the received word symbols, the decoder uses probabilistic reliability information about these received symbols. In [FS04] the authors use soft tracing methods for protecting multimedia content.

A soft-detected fingerprint, also called a *pirate matrix*, is a matrix whose columns correspond to the probability distributions on the alphabet symbols. That is the column j in the matrix gives a probability distribution on the symbols of the fingerprinting code to occur in the j^{th} position of the fingerprint. In soft tracing the algorithm takes a pirate matrix and outputs a colluder. It uses the *generalised* distance (Euclidean distance, log likelihood etc) instead of Hamming distance. Tracing is carried out by finding the codeword that has the highest similarity or shortest generalised distance with the pirate matrix. This tracing is computationally expensive in general, but if the fingerprinting code is a RS code, then the soft-decision decoding of Koetter et al [KV00] can be used to efficiently find a colluder. Soft tracing is used in [SW02b] for identifying culprits for a shortened and corrupted fingerprint.

In hard tracing (hard decision decoding) the pirate word (fingerprint) as extracted from the pirate copy is used for tracing the culprits [SSW01]. For a binary fingerprinting scheme the hard-decision decoding chooses between the two symbols 0 and 1 at each location in the fingerprint. The metric used for hard tracing is hamming distance. Algebraic decoding techniques generally use algebraic codes along with hard tracing techniques.

3.2.2 Deterministic tracing

A tracing algorithm is called deterministic if it traces and increments all traitors and no one else but the traitors [FT99]. On the other hand, schemes in which there is a small chance of false incrimination are referred to as probabilistic. In general, deterministic tracing methods come with a computational overload. It may also be required to have codewords of larger code length, derived from a non-binary code.

3.2.3 Probabilistic tracing

In probabilistic tracing [BS98, GP99], a pirate user is accused with a finite error probability ε, irrespective of its (very small) size. Allowing a small error probability in tracing, permits the distributors to construct and use efficient codes with smaller length.

3.2.4 Pirate testing vs. tracing

Testing is a simple method that takes a group of users and an illegal fingerprint to output either 'yes' or 'no' based on whether a group should be considered guilty [LW98]. The testing methods are useful incase a large number of users are present in a fingerprinting scheme. Since the accused groups contain partially innocent people apart from pirates the performance of the testing method is measured in terms of probability of accusing innocents and the probability of failing to accuse only pirates.

One simple approach for testing whether a group is guilty or not is to consider only the positions that are undetectable for the proposed group and declare the group guilty if its fingerprints agree with illegal fingerprint in all detectable positions. The probability of this happening by chance for innocent group is very low.

3.2.5 Tracing errors

The performance of a tracing method is measured in terms of probability for not tracing the pirates correctly. There are two types of errors in tracing as noted in [LW98]. One is called false alarm error (false positive) and the other is false dismissal error (false negative). The false alarm error corresponds to accusing a user who is innocent and the false dismissal error corresponds to failure in accusing a user who is guilty. It is desirable to keep both error probabilities to a minimum. Unfortunately they are conflicting goals, so a trade-off is required. As a special case when exactly one user is accused after every tracing, the two kinds of errors coincide. Thus the trade-off is correctly accusing a culprit.

3.3 Distributor's choices

Distributor should choose a fingerprinting model such that it is possible to trace the culprits by investigating the illegal copy. For this, the distributor needs to model the pirate strategies and design the counter measures. The choice of codes should be such that the size of feasible set (the set of all possible pirate codewords) is small. Limiting the number of pirate fingerprints allows the distributor to device efficient tracing algorithms.

3.4 A Tracing method for IPP codes

This section describes a generic tracing method for IPP codes in static fingerprinting scenario of stored content. It is to test whether a suspected user has really contributed to a particular pirate copy or not. The following theorem is used in tracing algorithm to find the culprit.

Theorem 3.1 *[SS01]*

Let Γ be a c-IPP code and $C_i \subseteq \Gamma, (i = 1 \text{ to } r)$, $|C_i| = w$, where $w \le c$

If $\bigcap_{i=1}^{r} D(C_i) \neq \phi$, then $\bigcap_{i=1}^{r} C_i \neq \phi$

3.4.1 A general tracing algorithm for *c*-IPP codes

Let Γ be a c-IPP code and W_i be the coalition set of size ω, *i.e.*, $W_i \subseteq \Gamma$ and $|W_i| = \omega$, where $\omega < c$ and the pirate codeword be x. The steps for tracing algorithm are as follows:

1. Consider a testing (suspect) set W among W_i.
2. Find $D(W)$, the descendent set of W. For simplicity we do not allow erasures in $D(W)$.
3. Compute $d_N = |mismatch(x, D(W))|$. This means d_N is equal to the total number of locations of x, whose values are not permitted values as per marking assumption [BS98] at their respective locations.

4. The probability of accusing innocent is d_N/l where l is length of the codeword x.
5. If $d_N = 0$ then clearly $x \in D(W)$ i.e. the coalition is capable of creating the pirate codeword x
6. Further let there be m subsets $W_1, W_2, ..., W_m$ for which $d_N = 0$. This means that $x \in D(W_i) \; \forall \, i = 1,...,m$. Then accuse $\mathbf{f}_{i=1}^{m} W_i$. Note that $\mathbf{f}_{i=1}^{m} W_i$ is a non-empty set since the code is IPP as per Theorem 3.1.
7. Without loss of generality, we assume that W_i is minimal before accusing $\mathbf{f}_{i=1}^{m} W_i$. The set W_i is said to be minimal if any codeword is removed from W_i then $x \notin D(W_i)$.

The aim of steps 1 to 5 is to find a coalition that is capable of creating a pirate codeword. Step 6 is for accusing a culprit without whose involvement the particular pirate codeword cannot be created. IPP codes identify at least one member in step 6. Now d_N is equal to the total number of mismatched positions of x with descendent set $D(W)$. The value of d_N should be equal to zero if the suspect-set has created the pirate codeword under marking assumption [BS98]. Thus the value of d_N/l should be equal to zero for the pirate group. But if d_N is non-zero, it indicates the coalition under consideration may be innocent. This fraction d_N/l is high when an innocent is accused, and hence we consider it as an error probability in tracing. However, more efficient algorithms may exist based on the construction of IPP codes.

3.5 Tracing pirate decoders in a broadcast encryption scheme

In broadcast scenario, a distributor broadcasts encrypted data that should be available only to a certain set of users. Each user has a unique decryption key that can be used to decrypt the broadcast data. It is possible that some users collude and create a new decoder (decryption key), different from any of theirs, and able to decrypt the broadcast data. In a traitor-tracing scheme for broadcast scenario, if the number of

users colluding is less than a threshold, it is possible, with low probability of error, to trace at least one of the colluder.

We present a tracing algorithm for [BF99] scheme presented in chapter 2, where there is one public key and k corresponding private keys. The tracing scheme defends against collusion of any number of parties. For tracing, we make use of the linear space tracing code Γ used in key generation of section 2.4.1.

3.5.1 Construction of Pirate decoder

The construction of a pirate decoder for [BF99] scheme (explained in chapter 2) is as follows.

If Y is the public key such that $Y = \sum_{i=1}^{k} \delta_i h_i$, where $h_i's$ are points on elliptic curve and $\delta_i \in Z_q$, we say that $(\delta_1,...,\delta_k)$ is a representation of Y with respect to base $(h_1,...,h_k)$. Each representation can be a decoder (private key) for the encryption scheme.

If $\bar{d}_1,...,\bar{d}_m$ are representations of Y with respect to the same base, then so is any "convex combination" of these representations: $\bar{d} = \sum_{i=1}^{m} \alpha_i \bar{d}_i$ where $\alpha_1,...,\alpha_m$ are the scalars such that $\sum_{i=1}^{m} \alpha_i = 1$. We call the resultant key $\bar{d}$ as pirate decoder.

3.5.2 Tracing scheme for BF scheme

In this section, we illustrate how to derive complete information about traitors involved in constructing a pirate decoder [VSG02]. Boneh and Franklin [BF99] showed that the efficient way of constructing pirate decryption box is only through convex combinations of existing private keys. The following tracing algorithm is simple, deterministic and traces all the traitors involved in creating a pirate decoder.

Suppose that the pirate gets hold of $m(\leq k)$ decryption keys used to create a pirate box D . The decryption scheme is said to be m resistant if there is a tracing algorithm that can determine at least one of the $\bar{d}_i$'s in the pirate's possession. The pirate

decoder contains at least one representation of Y to correctly decrypt the encrypted content. Further, by examining the decoder, it is possible to obtain one of these representations, which is a pirate key $\bar{d}$ constructed by m keys $\bar{d}_1,...,\bar{d}_m$. Since $\bar{d}$ found in the pirate decoder is a convex combination of $\bar{d}_i$'s, it must lie in the linear span of $\bar{d}_1,...,\bar{d}_m$. The construction of tracing algorithm is such that, given input $\bar{d}$, it outputs all $\bar{d}_i$'s with corresponding weights.

The Set Γ: We describe the set Γ containing k codewords over Z_p^k used in key generation. The code Γ is constructed with k code words by choosing a matrix B of order $k \times k$ as follows.

$$\hat{B} = \begin{pmatrix} 1 & 1 & 1 & \mathbf{h} & 1 \\ 1 & 2 & 3 & \mathbf{h} & k \\ 1^2 & 2^2 & 3^2 & \mathbf{h} & k^2 \\ 1^3 & 2^3 & 3^3 & \mathbf{h} & k^3 \\ \mathbf{j} & \mathbf{j} & \mathbf{j} & \mathbf{j} & \mathbf{j} \\ 1^{k-1} & 2^{k-1} & 3^{k-1} & \mathbf{h} & k^{k-1} \end{pmatrix} = \Gamma$$

Consider Γ as the set of rows of matrix $\hat{B}$ such that Γ contains k codewords each of length k. The matrix $\hat{B}$ is non-singular since it is in Vander Monde form. If $\hat{B}$ is non-singular, so is Γ. The private keys are constructed using the above set $\Gamma \subseteq Z_p^k$ containing k codewords. Each of the k users is given a private key $\bar{d}_i \in Z_p^k$, which is multiple of a codeword in Γ. It is clear that $\bar{d}$ is a point in the linear span of some m codewords $\gamma^{(1)},...,\gamma^{(m)} \in \Gamma$. Then at least one γ in $\gamma^{(1)},...,\gamma^{(m)}$ must be a member of the coalition that created $\bar{d}$. This γ identifies one of the private keys that *must* have participated in the construction of the $\bar{d}$. In fact, the tracing algorithm will output every $\gamma^{(i)}$ (hence all the traitors) which contributed to pirate decoder $\bar{d}$ with the corresponding weights in the linear combination.

Tracing: Let $\bar{d} \in Z_p^k$ be the vector formed by taking a linear combination of $m (m \le k)$ vectors from Γ. We show that for a given $\bar{d}$, one can efficiently determine the unique set vectors in Γ used to construct $\bar{d}$. There exists a vector $\varpi \in Z_p^k$ such that $\varpi \hat{B}^T = \bar{d}$ since the vector $\bar{d}$ is linear combination of these rows of

matrix $\hat{B}$. Also $\overline{d} = \theta.\gamma^{(\theta)}$ for some $\gamma^{(\theta)}$ since $\gamma^{(\theta)}$ is uniquely expressed as linear combination of $\gamma^{(i)}$'s. We show how to recover these$\gamma^{(i)}$'s with their corresponding weights for which we solve the system $\varpi\ \hat{B}^T = \overline{d}$.

Considering $\varpi\ \hat{B}^T = \overline{d}$ we get k equations in k unknowns. Since $\hat{B}^T$ is non-singular the system $\varpi\ \hat{B}^T = \overline{d}$ has a unique solution vector, which is simply ϖ. The i^{th} element of ϖ gives the contribution of the i^{th} user in terms of $\gamma^{(i)}$. If the element is *zero* the corresponding user has no contribution towards the pirate representation $\overline{d}$.

3.6 Summary

In this chapter, we described a variety of traitor-tracing methods in different settings. The distributor can choose one of these methods as per the requirements of the fingerprinting scenario. In this chapter we presented a generic tracing method for IPP codes. Subsequently, we presented a tracing method for a public key broadcast encryption scheme.

Chapter 4

Bounds for Digital fingerprinting

4.1 Introduction

In this chapter, we present a simple fingerprinting model and arrive at a bound on fingerprint length for non-binary codes. The bound is based on the idea that different collusions must generate different fingerprints in order to be distinguishable. This choice additionally makes tracing possible and reveals the collusion group responsible for creating a pirate copy. It also constitutes a lower bound on the fingerprint length, necessary for tracing the collusion group.

4.2 A simple fingerprinting model

Let Γ be a code containing M codewords equal to the number of users in the fingerprinting system. Let $W \subseteq \Gamma$ be a set of at the most c pirates and x be the illegal fingerprint created by collusion W. The following postulates define the fingerprinting model.

1. Number of pirate fingerprints = Number of possible collusion sets = β say
2. Number of possible pirate fingerprints created by pirates $\beta \leq q^l$ where l is length of the fingerprinting code and q is the alphabet size.
3. Every collusion set creates a unique fingerprint *i.e.* no two pirates cannot create the same pirate fingerprint.

To trace the collusion group responsible for creating an illegal copy, this model requires exactly one unique fingerprint to be created by each collusion group. If there are as many distinct fingerprints as the possible collusions (of size at most c) then tracing is simple, since this choice allows a one-to-one mapping between the pirate

groups and pirate fingerprints. This mapping further contributes to a lower bound on fingerprint length.

We present such fingerprinting codes both for binary case (identity matrix I_4) and non-binary case (Table 4.2). In the only erasures case, during creation of pirate fingerprints a collusion group (of size $c = 2$) come together and erases all the detected marks. In this scenario, each pirate group creates a unique fingerprint in these codes as given below.

Example 4.1

Consider the fingerprinting code as identity matrix (I_4). Let the collusion set be $W = \{w^1, w^2\}$ where w^i denote the i^{th} codeword of I_4. The set $\{w^1, w^2\}$ is also denoted as $\{1, 2\}$ in short form. Then $W = \{w^1, w^2\} = \{(1,0,0,0),(0,1,0,0)\}$. The detected positions for the collusion set W are 1st and 2nd. Erasing these positions will result in the pirate copy with the fingerprint (*e, e, 0, 0*). Repeating the same for all collusion sets we get Table 4.1 consisting of all pirate groups and the corresponding fingerprints created by them. The following table provides all the pirate fingerprints (feasible sets of pirates).

Collusion Sets	**Pirate fingerprints**
{1, 2}	(*e, e, 0, 0*)
{1, 3}	(*e, 0, e, 0*)
{1, 4}	(*e, 0, 0, e*)
{2, 3}	(*0, e, e, 0*)
{2, 4}	(*0, e, 0, e*)
{3, 4}	(*0, 0, e, e*)

Table 4.1. Collusion sets vs. pirate fingerprints

In Table 4.1, a non-binary code in which every collusion of size *c*=2 can create a unique fingerprint (only erasures case) is presented.

Example 4.2

0	0	0	0	0	0
0	1	1	1	1	0
1	0	2	1	1	1
1	2	0	2	1	1
2	1	2	0	2	1
2	2	1	2	0	2

Table 4.2. A non-binary code with *M* = 6 and *c* = 2

In Table 4.2 $M = 6,$ and $c = 2$. So there are ${}^{6}C_{2}$ =15 pirate groups. Assume that during the creation of pirate fingerprints, a collusion of size $c = 2$ comes together and erases all the detected marks. For the collusion set $W = \{w^1, w^2\}$ all the positions except the 1st and 6th positions are detected. So the fingerprint created by the pirates will be (*0, e, e, e, e, e, 0*). We can also observe that the illegal words created by each pirate group are different from the other. Other examples for this model are Gossip codes with shortest code length (in only erasures case) as explained in Chapter 6.

Sphere packing (Hamming) Bound [PW72]: For any (n, M, d) code over an alphabet Q of size q, $M.V_q(n, \lfloor (d-1)/2 \rfloor) \le q^n$, Where $V_q(n,t) = \sum_{i=0}^{t} {}^{n}C_i.(q-1)^i$.

Gilbert-Varshamov bound [MS77]: Let $F_q = GF(q)$ and let n, k and d be positive integers such that $V_q(n-1, d-2) = q^{n-k}$. Then there exists a linear code (n, k) code over F_q with minimum distance at least d.

4.3 Theoretical bounds on fingerprinting

We derive a lower bound on required fingerprint length to accuse a collusion set of size at most c, responsible for creating a pirate fingerprint as per the model. If every possible pirate group can create a unique illegal fingerprint there is a deterministic method to find the whole group through a mapping from the illegal word to the pirate group that is able to create it.

From postulate (2) of the model it follows that $l \geq \log_q \beta$ where β is the number of collusion groups. To find the minimum value of l, we estimate the value of β.

In binary fingerprinting schemes if the total number of users is M, the collusion size is at most c then the number of pirate groups (equal to number of pirate words) is $\sum_{i=1}^{c} {}^{M}C_i = \sum_{i=1}^{c} {}^{M}C_i - 1 = V(M,c) - 1$, where $V(M,c)$ is the volume of a sphere of radius c in an M-dimensional Hamming space. This means that there must be $\lceil \log(V(M,c)-1) \rceil$ bits in the fingerprint for it is to be possible to uniquely accuse each of the possible user sets. Now consider a non-binary code of alphabet of size q over a field F_q. The sphere of radius t in F_q^n, is a set of words in F_q^n at Hamming distance t or less, from a given word in F_q^n. The number of words, or the volume, of such a sphere is given by $V_q(n,t) = \sum_{i=0}^{t} {}^{n}C_i.(q-1)^i$.

The binary entropy function $H_2(x)$ is given by $H_2(x) = -x\log_2 x - (1-x)\log_2(1-x)$. It can be observed that $H_2(0) = H_2(1) = 0$. To estimates the value $V_q(n,t)$ we make use of the q'ary entropy function $H_q : [0,1] \to [0,1]$, which is defined as $H_q(x) = -x\log_q x - (1-x)\log_q(1-x) + x\log_q(q-1)$. The function $x \sim H_q(x)$ is strictly *concave*, non-negative, and attains a maximum value 1 at $x = 1 - (1/q)$. The estimation of $V_q(n,t)$ is required for calculating the bound on fingerprint length which is similar to estimation of the sphere packing bound and the Gilbert-Varshamov bound asymptotically.

We express the limits on $V_q(n,t)$ in terms of q'ary entropy function $H_q(x)$ using the following Lemmas (4.1 and 4.2) presented in [Roth04].

Lemma 4.1

Let $[n,t]$ be a code over $GF(q)$ and $0 \leq \nu/n \leq 1 - (1/q)$. Then in a sphere of radius t in n-dimensional Hamming space, the number of codewords

$$V_q(n,t) \leq q^{nH_q(\nu/n)}$$

Proof. Let $\tau = t/n$. Then,

From the definition of entropy function $H_q(x)$ we get $q^{-nH_q(\tau)} = \tau^t (1-\tau)^{n-t}.(q-1)^{-t}$

$$\text{Thus} \quad q^{-nH_q(\tau)}.V_q(n,t) = \tau^t (1-\tau)^{n-t} (q-1)^{-t}.\sum_{i=0}^{t} {}^nC_i.(q-1)^i$$

$$\leq \tau^t (1-\tau)^{n-t} (q-1)^{-t}.\sum_{i=0}^{n} {}^nC_i.(q-1)^i \left(\frac{(1-\tau)(q-1)}{\tau} \right)^{t-i}$$

$$\text{for } \tau \leq 1-(1/q)$$

$$= \sum_{i=0}^{n} {}^nC_i \tau^i (1-\tau)^{n-i}$$

$$= (\tau + (1-\tau))^n = 1,$$

Hence, we have the result $V_q(n,t) \leq q^{nH_q(\tau)}$.

Lemma 4.2 presents a lower bound on number of pirate fingerprints $V_q(n,t)$ in terms of q'ary entropy function $H_q(x)$. Lemma 4.3 presents a bound on collusion size c, considering a generic error correcting code.

Lemma 4.2

For $[n,t]$ code over $GF(q)$, for $0 \leq t \leq n$,

$$V_q(n,t) \geq {}^nC_t (q-1)^t \geq \frac{1}{n+1}.q^{nH_q(t/n)}.$$

Proof. Let $\tau = t/n$ and let $N_i = {}^nC_i.\tau^i (1-\tau)^{n-i}$.

We first show that N_i is maximal when $i = t$. Since

$$\frac{N_{i+1}}{N_i} = \frac{{}^nC_{i+1}.\tau^{i+1}(1-\tau)^{n-i-1}}{{}^nC_i.\tau^i (1-\tau)^{n-i}} = \frac{n-i}{i+1}.\frac{\tau}{1-\tau} = \frac{n-i}{i+1}.\frac{t}{n-t}$$

Thus, $N_{i+1}/N_i < 1$ if and only if $i \geq t$. It follows that

$$(n+1).N_t \geq \sum_{i=0}^{n} N_i = (\tau + (1-\tau))^n = 1 \text{ and so } N_t \geq \frac{1}{n+1}.$$

On the other hand,

$$N_t = {}^nC_t \tau^t (1-\tau)^{n-t} = {}^nC_t (q-1)^t . q^{-nH_q(t/n)}.$$

Hence, $V_q(n,t) \geq {}^nC_t (q-1)^t = N_t . q^{nH_q(t/n)} \geq \frac{1}{n+1} . q^{nH_q(t/n)}$

From Lemma 4.1 and 4.2 we can estimate the value of $V_q(n,t)$ with the following in equality, $\frac{1}{n+1} . q^{nH_q(t/n)} \leq V_q(n,t) \leq q^{nH_q(\tau)}$

4.3.1 Length of fingerprint

The number of different pirate fingerprints in our model is one less than the number of words present in a M-dimensional sphere (volume) with radius c. Thus we can say that in order to identify any user set (of at most size c) from a pirate fingerprint, the length l of the fingerprints has to fulfill the following inequality:

$$l \geq \log_q (V_q(M,c) - 1) > \log_q \left(\frac{V_q(M,c)}{q} \right)$$

$$= \log_q \left(V_q(M,c) \right) - \log_q q$$

$$= \log_q \left(\frac{1}{M+1} . q^{M.H_q(c/M)} \right) - 1$$

$$= \log_q \left(\frac{1}{M+1} \right) + M.H_q \left(\frac{c}{M} \right) - 1$$

$$= -\log(M+1) - c \log \left(\frac{c}{M} \right) - (M-c) \log \left(1 - \frac{c}{M} \right) + c \log(q-1) - 1$$

This result shows that the choice of fingerprint length must be higher than a certain value, in order to identify any user set of at most size c. A graphical representation of the variations of minimum fingerprint length with variations in number of users (M) and collusion size (c) are presented in Figure 4.1 and Figure 4.2 respectively. From these figures it can be observed that the bound on fingerprint length varies rapidly to changes in collusion size as compared to the variations in number of users.

In most of the scenarios, the collusion groups have a considerable freedom in their choice of fingerprints and their feasible sets are larger. In such scenarios the number

of pirate fingerprints are much larger than the number of possible collusion groups. If the feasible sets of the collusions are larger, it becomes more difficult to identify the culprits. Consequently, for a general non-binary fingerprinting scheme outside this model, the required fingerprint length is expected to be much greater than this lower bound. We now make a comparison between the current bound on fingerprint length with other known bounds.

In the current model the minimum length of the fingerprint $= \lceil \log(V(M,c)-1) \rceil$

$$\approx \log\left(\sum_{i=1}^{c} {}^{M}C_i\right) \text{W} \log(M) \text{ for small } c$$

The bound holds good for the model we have considered which is slightly restricted model. For probabilistic binary codes Boneh et al [BS98], Tordos [GT03] and Guth et al [GP99] have presented the bounds on fingerprint length under strong marking assumption, unreadable digit model and weak marking assumption respectively. Secure codes of [BS98] have a length of $O\left(c^3 \log\left(\frac{1}{\varepsilon}\right)\right)$ and Tordos codes achieve $O(c^2 \log(M/\varepsilon))$. The model we currently considered varies from these models considerably.

4.3.2 Collusion size

Each fingerprinting code has a maximum collusion size that it can withstand. If the collusion size of pirates that created a pirate fingerprint exceeds the maximum collusion size, then it is not possible to trace the culprits. This is due to the fact that as the collusion size increases, the number of detected positions increases and the choices available for pirates at each detected position increases. Thus tracing becomes more difficult and so, fingerprinting codes that can withstand large collusion sizes require large fingerprint length. Here we consider a generic error correcting code as the fingerprinting code.

Lemma 4.3

If colluding pirates create an illegal copy by making erasures in every detectable mark, it may be impossible to trace any of them by decoding the illegal word to the closest codeword.

Proof. Consider exactly two pirates. In order to correctly find one of the pirates, the number of erasures, e_N must not be greater than $d_{\min} - 1$, for correct decoding. The minimum Hamming distance between two codewords is $d_{\min}$, which means that the number of detectable marks for the pirates is at least $d_{\min}$. But $e_N = d_{\min} \geq d_{\min} - 1$, so it may be impossible to decode this correctly. This is true when the coalition size $c = 2$, for any alphabet size q, and adding more pirates will not alter any of the detectable marks undetectable. So this holds for any number of pirates $c \geq 2$.

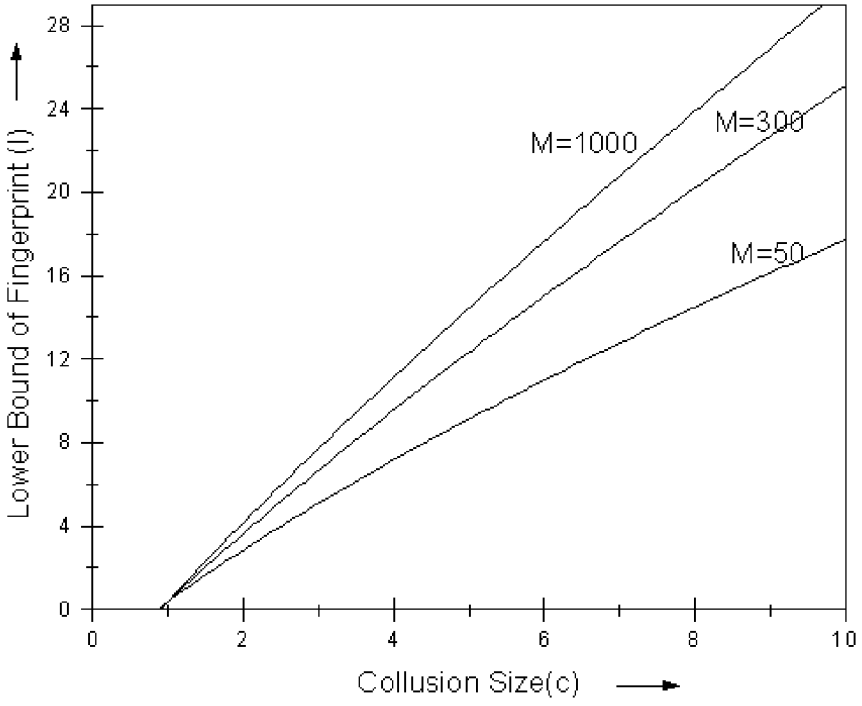

Figure 4.1. Lower bound vs. collusion size

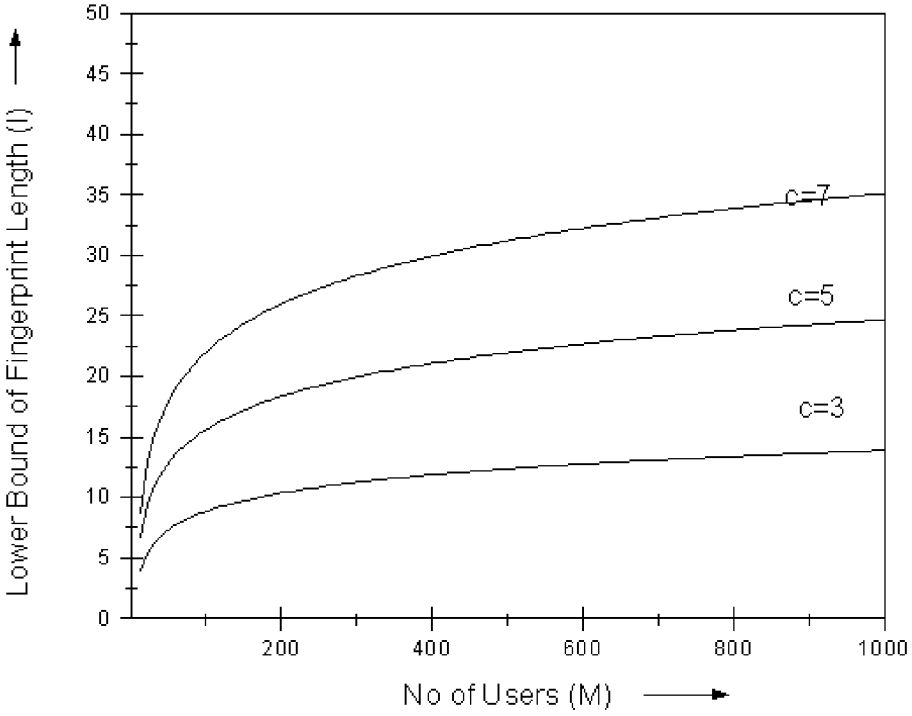

Figure 4.2. Lower Bound vs. number of users

4.4 To identify a single pirate in collusion

In this section, we trace explicitly one member of the coalition rather than trying to trace the whole collusion group responsible for creating an unauthorised copy.

Let U be the whole fingerprinting space. If the fingerprinting code Γ contains q alphabet symbols then the size of U is $|U| = q^n$. Let $C \subseteq \Gamma$ be the collusion set. We denote the set of all fingerprints the pirates can create as $\hat{P} = \bigcap_{C \subseteq \Gamma, |C|=c} FS(C)$ and $\hat{P} \subseteq U$.

Let A_i denote the accusation set for the i^{th} user. A_i is the set of all pirate fingerprints (created by any collusion in which user i is a member) for which i^{th} will be accused.

Let A denote the union of all accusation sets i.e. $A = \bigcap_{i=1}^{M} A_i$. This means that A contains the set of all pirate fingerprints for which at least one user can be traced back. In other words, A will contain the pirate fingerprints for which the tracing algorithm can identify a parent.

For a deterministic code $A = \bar{P}$ i.e. for every possible pirate fingerprint (under any pirate strategy) there exists a tracing algorithm, which identifies one member of the pirate group responsible for creating that pirate fingerprint. Define ε_P as the fraction of pirate fingerprints for which no user is accused.

So $\varepsilon_P = \frac{|\bar{P} - A|}{|\bar{P}|} \geq \frac{|\bar{P}| - |A|}{|\bar{P}|} = 1 - \frac{|A|}{|\bar{P}|}$ is the maximum possible failure probability to accuse a culprit for any pirate strategy.

For example consider the identity matrix I_3.

Let $C = \{W^1, W^2\}$ be the collusion set where w^i is the i^{th} codeword in I_3

Now $D(C) = \{(1, 0, 0), (0, 1, 0), (1, 1, 0), (0, 0, 0)\}$. Thus

$\bar{P} = \{(1, 0, 0), (1, 1, 0), (1, 0, 1), (0, 1, 0), (0, 0, 1), (0, 1, 1), (0, 0, 0)\}$

Tracing is possible by looking at the non-zero position in the pirate fingerprint. The tracing algorithm accuses the i^{th} user if the i^{th} position of the pirate fingerprint is '1'. Using the tracing algorithm one can define the members of accusation groups as follows.

$A_1 = \{(1, 0, 0), (1, 1, 0), (1, 0, 1)\}$

$A_2 = \{(0, 1, 0), (1, 1, 0), (0, 0, 1)\}$

$A_3 = \{(0, 0, 1), (0, 1, 1), (1, 0, 1)\}$

It follows that $A = \bigcup_{i=1}^{3} A_i = \{(1, 0, 0), (1, 1, 0), (1, 0, 1), (0,1,0), (0, 0, 1), (0, 1, 1)\}$

$\varepsilon_P = \frac{|\bar{P} - A|}{\bar{P}} = \frac{1}{7}$.

4.5 Summary

There are many parameters on which the performance of fingerprinting scheme depends. Some of these parameters are collusion size, length of fingerprint, choice of codes, error correcting capability of underlying codes, pirate strategies and tracing algorithm. In this chapter, we illustrated a simple fingerprinting model in which each pirate group creates a unique fingerprint when they collude. This makes the tracing to be possible for the distributor. Subsequently, we arrived at the theoretical bounds on minimum fingerprint length and collusion size for a non-binary code based on the model. We also present a method to find the error probability in identifying a single pirate in the collusion where we can trace explicitly one member of the coalition rather than trying to trace the whole collusion group.

Chapter 5

On Gossip Codes, *t*-designs and Traceability Schemes

5.1 Introduction

This chapter presents the construction methods for a non-binary IPP code, namely Gossip code. IPP codes allow the identification of at least one pirate involved in creating the illegal copy. In this chapter, two techniques for construction of Gossip codes are presented that exploit combinatorial t-designs and Traceability Schemes. The codes thus constructed have the shortest code length achievable by Gossip codes as per the code parameters. The converse part, *i.e.,* construction of Traceability Schemes and t-designs from Gossip codes is also explained. These results help in analysing Gossip codes and understanding their combinatorial properties. We also present the construction of embedded Gossip codes for extending an existing Gossip code to a bigger code. Concatenated codes are presented for the realisation of tracing in presence of collusion attacks. The performance issues of Gossip codes are also discussed.

5.2 Gossip codes as IPP codes

Gossip codes [TL01] are fingerprinting codes that can provide protection against illegal copying of digital objects when the codewords are embedded as digital fingerprints. They are also c-IPP codes, which can identify at least one user involved in creating an illegal copy.

We first explain the construction of Gossip codes originally presented in [LLS02].
Let $B(M, q)$ be the $0/1$- matrix consisting of l columns and M rows such that each column is created by placing $q-1$ ones and $M-(q-1)$ zeros. The parameters q and M are so chosen satisfying $q \geq 3$ and $M \geq q+1$. The codeword matrix $G(M, q)$ is constructed from $B(M, q)$ by replacing the $q-1$ ones in each column, with the $q-1$

different non-zero symbols of Q and retaining the zeros unaltered. In a gossip column, the occurrence of non-zero alphabet symbols (among themselves) is immaterial as long as they are distinct.

This construction does not guarantee the shortest code length for Gossip codes whereas our constructions [VSGP04a, VSGP04b] achieve it. We also denote the matrix $G(M, q)$ by $c\text{-}Gossip(l, M, q)$ code where c is the collusion size, M is the number of code words, q is the alphabet size and l is the length of the code.

A 2-Gossip (7, 7, 4) code can be constructed as follows.

Example 5.1

Consider a 0/1 matrix (Table 5.1) such that M = 7 and l = 7 and each column contains four zeros. We replace the ones in each column of Table 5.1, with three different non-zero symbols namely 1, 2 and 3 to obtain the Gossip code presented in Table 5.2. A Gossip code with alphabet size $q = 4$ contains three nonzero symbols 1, 2, 3 and zero.

1	1	1	0	0	0	0
1	0	0	1	1	0	0
1	0	0	0	0	1	1
0	1	0	1	0	0	1
0	1	0	0	1	1	0
0	0	1	1	0	1	0
0	0	1	0	1	0	1

Table 5.1. *B*(*M, q*) Matrix

1	1	1	0	0	0	0
2	0	0	1	1	0	0
3	0	0	0	0	1	1
0	2	0	2	0	0	2
0	3	0	0	2	2	0
0	0	2	3	0	3	0
0	0	3	0	3	0	3

Table 5.2. 2-Gossip (7, 7, 4) code

Definition 5.1 *[TL01]* A gossip column is a code column of a *q*'ary code, with all non-zero symbols of the alphabet Q appearing exactly once, and the symbol 0 in the remaining positions.

Tracing using Gossip columns: Consider the 6^{th} gossip column [0 0 1 0 2 3 0] in Table 5.2. In this gossip column, the non-zero symbols are distinct and appear at positions 3, 5 and 6. Thus this gossip column is responsible for tracing the 3^{rd} or 5^{th} or 6^{th} users (based on 6^{th} position in the pirate word), if they are involved as traitors. The collusion groups whose members can be traced (accused) by a gossip column are known as accusation groups of that column. Thus this gossip column traces or accuses the collusion groups {3, 5}, {5, 6} and {3, 6} of collusion size $c = 2$. The remaining collusion groups are accusation groups for some other columns. The collusion groups {3, *i*}(for *i*=1, 2, 4 or 7) certainly can create a pirate word with a 1 at that column's position. But the tracing of these groups is left over to other Gossip columns. Thus a gossip column contributes to tracing traitors involved in creating a pirate word.

The following lemma presents another definition of Gossip code based on the IPP property of Gossip codes.

Lemma 5.1 *[TL01]*
A Gossip code is the code with a gossip column for each collusion group to trace.

The tracing method uses the fact that every illegal fingerprint contains at least one non-zero symbol. If a particular collusion group is traceable by a gossip column, then all its subgroups are also traceable by the same gossip column since they will have a subset of possible alphabet symbols. If there exists a gossip column for each c -group, it is possible to trace at least one member of all pirate groups whose size is less than or equal to *c*. So the Gossip code will always find a member of the pirate group and thus it is a *c*-IPP code.

Definition 5.2 The set of indices of non-zero positions of a gossip column is termed as *column key* corresponding to that column.

In the Gossip code presented in Example 5.1, the column key for the 6th column is {3, 5, 6}. *Column keys* of Gossip code identify the set of pirate groups that each gossip column is capable of tracing. From a gossip column of a Gossip code, where $c < q < M$, exactly ${}^{q-1}C_c$ collusion groups are traceable. If M is the total number of codewords in a Gossip code, then the total number of possible pirate groups of size *c* is given by ${}^{M}C_c$. For $c \le q-1$, each gossip column can at the most accuse ${}^{q-1}C_c$ groups, since there are $(q-1)$ non-zero symbols, which are distinct. Hence, in a Gossip code the total number of gossip columns $l \ge {}^{M}C_c / {}^{q-1}C_c$ for $c < q < M$. This gives a lower bound on length of the Gossip code. If the Gossip code length attains the equality (bound) condition, then it has the shortest possible length. We denote this shortest length by l.

The structure of the Gossip code is such that every non-zero symbol in the pirate codeword will lead to only one culprit. Since Gossip codes are q'ary IPP codes and come up with a deterministic tracing algorithm unlike q'ary c-secure code with error probability ε [BS98], they have a significant role to play in fingerprinting applications.

Lemma 5.2 presents the properties of Gossip codes with $c = q-1$. A generalisation of this lemma is presented in Theorem 5.6.

Lemma 5.2

For a shortest *c-Gossip*(*l*, *M*, *q*) code with $c = q-1$

1. Length of the code $l = n(M,q) = {}^{M}C_{q-1}$
2. Weight of the code $w(M,q) = {}^{M-1}C_{q-2}$
3. Distance of the code $d(M,q) = {}^{M-1}C_{q-2} + {}^{M-2}C_{q-2}$

Proof.

(*i*) In a shortest *c-Gossip*(l,M,q) code the length of the code is given by $l = {}^{M}C_{c} \big/ {}^{q-1}C_{c}$.

If $c = q-1$, then the length of the code become $l = {}^{M}C_{q-1}$

(*ii*) In a *c-Gossip*(*l*, *M*, *q*) code, the number of undetectable positions for a collusion of size c is given by ${}^{M-c}C_{q-1}$. Thus the number of zeros in a codeword is ${}^{M-1}C_{q-1}$.

We know that weight of the Gossip code = Length of the code - Number of zeros

$$= {}^{M}C_{q-1} - {}^{M-1}C_{q-1}$$

$$= {}^{M-1}C_{q-2} \quad \left\{ \because {}^{M}C_{q-1} = {}^{M-1}C_{q-1} + {}^{M-1}C_{q-2} \right\}$$

(*iii*) The distance of the code $d(M,q) = n(M,q) - {}^{M-2}C_{q-1}$

But $n(M,q) = {}^{M}C_{q-1}$

$$d(M,q) = n(M,q) - {}^{M-2}C_{q-1}$$

$$\therefore d(M,q) = {}^{M}C_{q-1} - {}^{M-2}C_{q-1}$$

$$= {}^{M-1}C_{q-1} + {}^{M-1}C_{q-2} - {}^{M-2}C_{q-1}$$

$$= {}^{M-2}C_{q-2} + {}^{M-1}C_{q-2}$$

5.3 Construction of shortest Gossip codes

In Gossip codes that achieve the bound, the gossip columns accuse distinct groups and thus every collusion group is an accusation group for some or the other gossip column. Such Gossip codes are known as the shortest Gossip codes for the chosen code parameters. The following lemma defines a shortest Gossip code in terms of traceability of collusion groups. A shortest Gossip code satisfies Lemma 5.3 and also the bound condition.

Lemma 5.3

A *c-Gossip*(l, M, q) code is the shortest code if and only if each gossip column accuses ${}^{q-1}C_c$ distinct collusion groups (c-groups).

Proof. The condition for gossip code to achieve its equality bound of code length is that each gossip column should accuse distinct groups. Further, each column can at the most accuse ${}^{q-1}C_c$ groups. Thus a shortest Gossip code must accuse ${}^{q-1}C_c$ distinct groups.

5.3.1 Partition Method

In this method, we construct shortest Gossip codes by partitioning the set of all collusion groups (of size c) into disjoint sets. Each disjoint set (equivalence class) defines the accusation sets for one gossip column. The accusation sets in turn define the non-zero positions for each gossip column that constitute a Gossip code.

Gossip codes that achieve the bound constitute a partition on the set S of all collusion groups (of size c). This means there exists an equivalence relation R, which partitions S into equivalence classes. Let Γ be a *c-Gossip*(l, M, q) code. Let C_i and C_j be two collusion sets of size c. We say $C_i \, R \, C_j$ (C_i is related to C_j) if they are accusations groups for the same gossip column. The construction of Γ from partition method is as follows.

Let X be a set of M users. Let S be the set of all subsets of X of size c. Let C_i and C_j be two members of S. Consider a t-(v,k,λ) design with $t = c$, $v = M$, $k = q-1$ and $\lambda = 1$. We define a relation R as follows.

$C_i \, R \, C_j$ if and only if C_i and C_j are subsets of the same block in the t-(v, k, λ) design.

It is transparent that the relation R is reflexive, symmetric and transitive and hence an equivalence relation that partitions S into equivalence classes. Let $C_1, C_2, ..., C_r$ be the exhaustive members of the i^{th} equivalence class. We now compute the set $\bigcap_{i=1}^{r} C_i$, which gives the non-zero positions (*column key*) for the i^{th} gossip column. Repeating the same process for all equivalence classes, one can arrive at all the *column keys*. It is from these column keys that the indices of non-zero positions for each column are known, and hence a *c-Gossip*(l, M, q) code can be constructed.

The construction of 2-Gossip (7, 7, 4) code from partition method is as follows.

Example 5.2

Let $M = 7$, $q = 4$, and $c = 2$ thus the total number of pirate groups $= {}^{M}C_c = {}^{7}C_2 = 21$

The maximum number of accusation groups for each gossip column $= {}^{q-1}C_c = {}^{3}C_2 = 3$

The number of Gossip columns $= l = 21/3 = 7$

Coalition	Pirate members in coalition	Gossip column to accuse the coalition
C_1	{1, 2}	1
C_2	{1, 3}	1
C_3	{1, 4}	2
C_4	{1, 5}	2
C_5	{1, 6}	3
C_6	{1, 7}	3
C_7	{2, 3}	1
C_8	{2, 4}	4
C_9	{2, 5}	5
C_{10}	{2, 6}	4
C_{11}	{2, 7}	5
C_{12}	{3, 4}	7
C_{13}	{3, 5}	6
C_{14}	{3, 6}	6
C_{15}	{3, 7}	7
C_{16}	{4, 5}	2
C_{17}	{4, 6}	4
C_{18}	{4, 7}	7
C_{19}	{5, 6}	6
C_{20}	{5, 7}	5
C_{21}	{6, 7}	3

Table 5.3 Accusation groups for gossip columns

Now, we can partition the total pirate groups of size 2, (namely C_1 to C_{21}) as per the above choice. The accusation groups for any gossip column form one equivalence class. The accusation groups for the first column, which are C_1, C_2, and C_7 form the first equivalence class. The union of these accusation groups of the first column will effectively become the first column key.

Column	Accusation Groups	Column Key	Equivalence class
1	C_1, C_2, C_7	{1, 2, 3}	I
2	C_3, C_4, C_{16}	{1, 4, 5}	II
3	C_5, C_6, C_{21}	{1, 6, 7}	III
4	C_8, C_{10}, C_{17}	{2, 4, 6}	IV
5	C_9, C_{11}, C_{20}	{2, 5, 7}	V
6	C_{13}, C_{14}, C_{19}	{3, 5, 6}	VI
7	C_{12}, C_{15}, C_{18}	{3, 4, 7}	VII

Table 5.4. Partitioning of the collusion groups into equivalence classes

Consider the 2nd column key *i.e.* {1, 4, 5}. This column key provides the indices of non-zero symbols for the 2nd gossip column. By filling the 1st, 4th and 5th positions in the 2nd column with the non-zero alphabet symbols, namely 1, 2 & 3 and the rest with zeros, we get the second gossip column. Repeating the same process for all the columns, we get the following Gossip code presented in Table 5.5.

The constructed 2 -Gossip (7, 7, 4) code is as follows

1	1	1	0	0	0	0
2	0	0	1	1	0	0
3	0	0	0	0	1	1
0	2	0	2	0	0	2
0	3	0	0	2	2	0
0	0	2	3	0	3	0
0	0	3	0	3	0	3

Table 5.5. 2-Gossip (7, 7, 4) code

Example 5.3 presents an alternate construction of 2-Gossip(7, 7, 4) code based on an alternate partition (Table 5.6) of the collusion groups.

Example 5.3

Considering an alternate choice, we get the following partition on the set of all collusion groups.

Column	Accusation Groups	Column Key	Equivalence class
1	C_1, C_3, C_8	{1, 2, 4}	I
2	C_2, C_4, C_{13}	{1, 3, 5}	II
3	C_5, C_6, C_{21}	{1, 6, 7}	III
4	C_7, C_{10}, C_{14}	{2, 3, 6}	IV
5	C_9, C_{11}, C_{20}	{2, 5, 7}	V
6	C_{12}, C_{15}, C_{18}	{3, 4, 7}	VI
7	C_{16}, C_{17}, C_{20}	{4, 5, 6}	VII

Table 5.6 Alternate partition of the collusion groups into equivalence classes

The alternate Gossip code is presented in Table 5.7.

1	1	1	0	0	0	0
2	0	0	1	1	0	0
0	2	0	2	0	1	0
3	0	0	0	0	2	1
0	3	0	0	2	0	2
0	0	2	3	0	0	3
0	0	3	0	3	3	0

Table 5.7. Alternate 2-Gossip (7, 7, 4) code

The above construction shows that there exist multiple 2-Gossip (7, 7, 4) codes, which are distinct. Thus in general, there may exist several Gossip codes for the same code parameters.

5.3.2 Gossip codes from *t*-designs

In this section, we present the construction of Gossip codes from t-designs. Theorem 5.1 presents the conditions for construction of c-*Gossip*(l, M, q) codes from t-designs along with the construction technique. Here the combinatorial equivalence between t-(v, k, λ) design with $t = c,\ v = M,\ k = q-1$ and $\lambda = 1$ and c-*Gossip*(l, M, q) code is exploited to construct and analyse Gossip codes. Theorem 5.1 is also reliable to make out whether a *shortest* Gossip code exists or not, for the chosen code parameters namely M, q and c. A program DISCRETA [AB97] to compute t-designs is available in public domain. In Gossip codes that achieve the bound, the gossip columns accuse distinct groups and thus every collusion group is an accusation group for some or the other gossip column.

Definition 5.3 *[CD96] A* t-(v, k, λ) design is a set system (X, B), where $|X| = v,\ |B_i| = k$ for every $B_i \in B$, and every t-subset of X occurs in exactly λ blocks in B.

Theorem 5.1

A c-*Gossip*(l, M, q) code exists if and only if c-$(M, q-1, 1)$ design exists where $l = {}^{M}C_c \,/\, {}^{q-1}C_c$.

Proof. Consider a t-(v,k,λ) design with $t = c, v = M, k = q-1$ and $\lambda = 1$. Then there exists a set system (X, B) such that $|X| = v, |B_i| = k$ for every $B_i \in B$. Further, every c-subset of X occurs in exactly one block in B. Let B_i be the i^{th} *column key* of the Gossip code. Then the accusation groups of the i^{th} gossip column are identical to the c-subsets of B_i. Since every c-subset of X occurs in exactly one block, the accusation groups of all gossip columns are distinct. It is known that the number of blocks in c-$(M, q-1, 1)$ design is ${}^{M}C_c \,/\, {}^{q-1}C_c$. Thus the t-design c-$(M, q-1, 1)$ completely defines c-*Gossip*(l, M, q) code.

Conversely let (X, B) be the set representation of *c-Gossip*(l, M, q) code. Then we have $|X| = M$, and $|B_i| = q - 1$ for every $B_i \in B$, where each block B_i correspond to one *column key*. The number of blocks in B is ${}^{M}C_c / {}^{q-1}C_c$ equal to the number of *column keys*. This implies that this (X, B) system represent a t-(v, k, λ) design with $t = c,\ v = M,\ k = q - 1,$ and $\lambda = 1$.

Lemma 5.4 presents the condition for existence of 2-Gossip codes for alphabet sizes 4, 5 and 6. Lemma 5.5 presents the existence of a Gossip code with collusion size 3. These lemmas are based on Theorem 5.1.

Lemma 5.4

For $3 \le k \le 5$, a 2-*Gossip*$(l, M, k+1)$ code exists if and only if $M \equiv 1$ or $k \mod(k^2 - k)$.

Proof. For $3 \le k \le 5$, a 2-$(M, k, 1)$ design exists if and only if $M \equiv 1$ or $k \mod(k^2 - k)$ (see Chapter I.2 in [CD96]). We have shown in Theorem 5.1 that *c-Gossip*(l, M, q) code exists if and only if *c*-$(M, q - 1, 1)$ design exists.

Lemma 5.5

There exists a 3-*Gossip*$(9 \times 82, 82, 11)$ code.

Proof. It is known from [CD96] that 3-$(p^2 + 1, p + 1, 1)$ design exist when p is prime power. Let p = 9 then it follows that 3-(82,10,1) design exists. Using Theorem 5.1 we can construct c-Gossip code from this design.

The code parameters of the Gossip code that can be constructed from 3-(82,10,1) design are M = 82, c = 3 and q = k+1 = 11.

So ${}^{M}C_c = \dfrac{82 \times 81 \times 80}{6}$; ${}^{q-1}C_c = \dfrac{10 \times 9 \times 8}{6}$

$\therefore l = 9 \times 82$

It follows that 3-*Gossip*$(9 \times 82,\ 82,\ 11)$ code exists.

Consider 2-Gossip(7, 7, 4) code presented in Example 5.1. The t-design corresponding to this Gossip code is 2-(7, 3, 1) design, which is given by

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

$B_1 = \{1, 2, 3\}, B_2 = \{1, 4, 5\}, B_3 = \{1, 6, 7\}, B_4 = \{2, 4, 6\},$
$B_5 = \{2, 5, 7\}, B_6 = \{3, 5, 6\}, B_7 = \{3, 4, 7\}$

Steiner Systems [CD96]: Given three integers t, k, v such that $2 \le t < k < v$ a Steiner system $S(t, k, v)$ is a v-set X together with a family B of k-subsets of X (called blocks) with the property that every t-subset of X is contained in exactly one block. This implies a $S(t, k, v)$ system is equivalent to a t-$(v, k, 1)$ design. Thus each Steiner system is combinatorially equivalent to one Gossip code.

The known infinite families of $S(t, k, v)$ are

1. $S(2, p, p^r)$, where p is a prime power , $r \ge 2$
2. $S(3, p+1, p^r+1)$, p is a prime power , $r \ge 2$
3. $S(2, p+1, p^r + ... + p + 1)$ p is a prime power, $r \ge 2$
4. $S(2, p+1, p^3+1)$ p is a prime power , $r \ge 2$
5. $S(2, 2^r, 2^{r+s} + 2^r - 2^s)$ for $2 \le r < s$.

From the first result presented above, we can say an infinite family of 2-Gossip code exists with $M = p^r, q = p+1$ where p is prime power and $r \ge 2$.

Examples for Steiner 5-designs are
$S(5, 6, 12)$, $S(5, 8, 12)$, $S(5, 8, 84)$, $S(5, 8, 108)$, $S(5, 8, 132)$, and $S(5, 8, 168)$. All these designs can be used to construct Gossip codes with collusion size '5'.

5.3.3 Gossip codes from Traceability Schemes

A c-Traceability scheme c-$TS(k, b, v)$ [3, 4] is a broadcast encryption scheme, where k is the number of keys provided to each user, b is the total number of users, v is

the total number of base keys and c is the collusion size of pirates that the scheme can withstand. A trusted authority generates the set T of v base keys and assigns k keys chosen from T to each user. The i^{th} user's personal key or private key is denoted by $P(i)$ that uniquely determines the owner and allows him to decrypt the encrypted broadcast. The broadcast message consists of an enabling block E and a cipher block Y. The cipher block is the encryption of the actual plaintext data $\bar{m}$ using a secret key 'a'. That is, $Y = \xi_a(\bar{m})$, where $\xi()$ is the symmetric encryption function. The enabling block consists of the shares of a, which are encrypted using some or all of the v keys in the base set. The decryption of enabling block will allow the recovery of the secret key a. Every authorised user should be able to recover a using his personal key and then decrypt the cipher block using a to obtain the plaintext data *i.e.*, $\bar{m} = \zeta_a(Y)$, where $\zeta()$ is the decryption function for the cryptosystem. A collusion of users may conspire and create an unauthorised pirate decoder F. This pirate decoder will consist of a subset of base keys such that $F \subseteq \bigcup_{i \in W} P(i)$, where W is the coalition of traitors. Once a pirate decoder is found, the broadcaster can trace those who have participated in producing the pirate decoder. Traitor detection is carried out by computing $|F \cap P(U)|$ for all users U. If $|F \cap P(U)| \geq |F \cap P(V)|$ for all users $V \neq U$, then U is defined to be exposed user.

In this section, we present an alternate construction for shortest length Gossip codes from Traceability Schemes c-$TS(k, b, v)$ and vice versa [VSGP04a]. The required condition for the construction of Gossip codes from c-Traceability scheme is that the number of private keys in c-Traceability Scheme should be $b = {}^{v}C_c / {}^{k}C_c$. Stinson et al [SW98] proved the existence of an infinite class of Traceability Schemes used in this case.

In Lemma 5.6 we present a condition on the private keys of the Traceability Scheme used in Theorem 5.2.

Lemma 5.6

Let (X, B) denote the set system corresponding to c-$TS(k, b, v)$ with B_i s representing the private keys of c-$TS(k, b, v)$ and c-$Gossip(b, v, k+1)$ denote the Gossip code constructed from c-$TS(k, b, v)$. If $|B_i \cap B_j| \geq c$ for $i \neq j$, then the accusation groups of the gossip columns are not distinct.

Proof. Assume $|B_i \cap B_j| \geq c$. Then there exist at least c elements say $\{x_1, x_2, ..., x_c\}$ which are common in both B_i and B_j. Since the set $\{x_1, x_2, ..., x_c\}$ is formed by both B_i and B_j, the c-sets formed by B_i s are not distinct. The set $\{x_1, x_2, ..., x_c\}$ will be accusation set for both i^{th} and j^{th} gossip columns. Hence the accusation sets that correspond these gossip columns in the Gossip code are not distinct.

Theorem 5.2 presents the construction of shortest Gossip codes from Traceability Schemes.

Theorem 5.2

If c-$TS\left(k, {}^{v}C_c / {}^{k}C_c, v\right)$ exists then c-$Gossip\left({}^{v}C_c / {}^{k}C_c, v, k+1\right)$ code exists.

Proof. For a c-$TS(k, b, v)$, the total number of c-groups that can be formed from v users is ${}^{v}C_c$. Then the maximum number of c-groups that can be formed by each private key containing k elements is ${}^{k}C_c$. If the number of c-traceable keys is equal to ${}^{v}C_c / {}^{k}C_c$, then c-groups formed by each private key, should be distinct. It follows that $|B_i \cap B_j| < c$ for $i \neq j$ where B_i s correspond to (X, B) representation of a $c-TS(k,b,v)$. Let each B_i be equal to one *column key* in the Gossip code. Then the c-groups created by k elements of each B_i *i.e.*, ${}^{k}C_c$ groups will be the accusation groups for the i^{th} gossip column. Since $|B_i \cap B_j| < c$, from Lemma 5.6 each column in Gossip code accuses ${}^{k}C_c$ distinct groups. If we consider v codewords in the Gossip code, equal to the number of base keys in $c-TS(k,b,v)$, then the code is a c-$Gossip\left({}^{v}C_c / {}^{k}C_c, v, k+1\right)$ code, and each column contains $k+1$ distinct elements

$(0 \text{ to } k)$. Since we have considered $l = {}^{v}C_c / {}^{k}C_c$ gossip columns, the Gossip code achieves the minimum code length.

In Lemma 5.7, we present an infinite family of 2-Gossip codes, which can be constructed from 2-Traceability schemes. The construction of Traceabilty schemes from Gossip codes (*i.e.* converse of Theorem 5.2) is presented as Theorem 5.3.

Lemma 5.7

There exists a $2-Gossip\left(v(v-1)/20, v, 6\right)$ code for all $v \equiv 1 \text{ or } 5 \bmod 20$.

Proof. Stinson and Wei (see Theorem 3.4 in [SW98]) showed that there exists $2-TS(5, v(v-1)/20, v)$ whenever $v \equiv 1 \text{ or } 5 \bmod 20$. Observe that ${}^{v}C_2 / {}^{5}C_2 = v(v-1)/20$. From Theorem 5.2, it follows that $2\text{-}Gossip\left(v(v-1)/20, v, 6\right)$ code exists.

Theorem 5.3

If $c\text{-}Gossip\left({}^{M}C_c / {}^{q-1}C_c,\ M,\ q\right)$ code exists then $w-TS\left(q-1, {}^{M}C_c / {}^{q-1}C_c, M\right)$ exists where $w = \left\lfloor \sqrt{(q-2)/(c-1)} \right\rfloor$.

Proof. Let (X, B) be the set representation of the Gossip code. Then we have $|X| = M$, and $|B_i| = q-1$ for every $B_i \in B$, where each block B_i corresponds to one *column key*. The number of blocks in B is equal to ${}^{M}C_c / {}^{q-1}C_c$. Since the Gossip code achieves the minimum code length, the accusation sets of each gossip column are distinct. This implies that this (X, B) system represent a t-(v, k, λ) design, where $t = c,\ v = M,\ k = q-1,$ and $\lambda = 1$. This implies that there exists corresponding Traceability Scheme $w-TS\left(q-1, {}^{M}C_c / {}^{q-1}C_c, M\right)$ (see Theorem 3.2 in [SW98]) where $w = \left\lfloor \sqrt{(q-2)/(c-1)} \right\rfloor$.

Example 5.4

Consider the Traceability Scheme 2-TS (5, 21, 21)

Base Keys $= T =$ {1 to 21}

Private Key $P(i) = \{2+i, 5+i, 6+i, 11+i, 13+i\}$

The private keys in Table 5.8 are constructed from $P(i)$. The first private key is constructed by substituting i=1 in $P(i)$ and the process is repeated for all the users.

User No (*i*)	Private Key P(*i*) = i^{th} Column Key
1.	{3, 6, 7, 12, 14}
2.	{4, 7, 8, 13, 15}
3.	{5, 8, 9, 14, 16}
4.	{6, 9, 10, 15, 17}
5.	{7, 10, 11, 16, 18}
6.	{8, 11, 12, 17, 19}
7.	{9, 12, 13, 18, 20}
8.	{10, 13, 14, 19, 21}
9.	{11, 14, 15, 20, 1}
10.	{12, 15, 16, 21, 2}
11.	{13, 16, 17, 1, 3}
12.	{14, 17, 18, 2, 4}
13.	{15, 18, 19, 3, 5}
14.	{16, 19, 20, 4, 6}
15.	{17, 20, 21, 5, 7}
16.	{18, 21, 1, 6, 8}
17.	{19, 1, 2, 7, 9}
18.	{20, 2, 3, 8, 10}
19.	{21, 3, 4, 9, 11}
20.	{1, 4, 5, 10, 12}
21.	{2, 5, 6, 11, 13}

Table 5.8. 2-Traceablity Scheme (2-TS (5, 21, 21))

The total number of user groups of size 2 is equal to

$$^{v}C_c = {}^{21}C_2 = \frac{21(20)}{2} = 210.$$

Number of accusation groups per each gossip column $^{q-1}C_c = {}^{k}C_c = {}^{5}C_2 = 10$,

and so $l = \frac{210}{10} = 21$

We denote the ith private key by $P(i)$ which is also equal to the column key B_i. That is $B_1 = P(1)$, ..., $B_i = P(i)$.

Considering the first column key (first private key) we get the following accusation groups. The 2-subsets that can be formed from B_1 are the accusation groups for the first column.

Column	Accusation Groups
1	{3, 6}, {3, 7}, {3, 12}, {3, 14}, {6, 7}, {6, 12}, {6, 14}, {7, 12}, {7,14}, {12, 14}

Table 5.9. Accusation group of 1st column in 2-Gossip (21, 21, 6) code

The private key $P(i)$ gives the indices of non-zero symbols for the i^{th} column. So for the 1st column the nonzero positions are 3, 6, 7, 12 and 14. We replace these positions with the non-zero alphabet symbols *i.e.*, 1, 2, 3 and 4 respectively. The Gossip code is constructed by repeating the process for all the columns.

0	0	0	0	0	0	0	0	5	0	4	0	0	0	0	3	2	0	0	1	0
0	0	0	0	0	0	0	0	0	5	0	4	0	0	0	0	3	2	0	0	1
1	0	0	0	0	0	0	0	0	0	5	0	4	0	0	0	0	3	2	0	0
0	1	0	0	0	0	0	0	0	0	0	5	0	4	0	0	0	0	3	2	0
0	0	1	0	0	0	0	0	0	0	0	0	5	0	4	0	0	0	0	3	2
2	0	0	1	0	0	0	0	0	0	0	0	0	5	0	4	0	0	0	0	3
3	2	0	0	1	0	0	0	0	0	0	0	0	0	5	0	4	0	0	0	0
0	3	2	0	0	1	0	0	0	0	0	0	0	0	0	5	0	4	0	0	0
0	0	3	2	0	0	1	0	0	0	0	0	0	0	0	0	5	0	4	0	0
0	0	0	3	2	0	0	1	0	0	0	0	0	0	0	0	0	5	0	4	0
0	0	0	0	3	2	0	0	1	0	0	0	0	0	0	0	0	0	5	0	4
4	0	0	0	0	3	2	0	0	1	0	0	0	0	0	0	0	0	0	5	0
0	4	0	0	0	0	3	2	0	0	1	0	0	0	0	0	0	0	0	0	5
5	0	4	0	0	0	0	3	2	0	0	1	0	0	0	0	0	0	0	0	0
0	5	0	4	0	0	0	0	3	2	0	0	1	0	0	0	0	0	0	0	0
0	0	5	0	4	0	0	0	0	3	2	0	0	1	0	0	0	0	0	0	0
0	0	0	5	0	4	0	0	0	0	3	2	0	0	1	0	0	0	0	0	0
0	0	0	0	5	0	4	0	0	0	0	3	2	0	0	1	0	0	0	0	0
0	0	0	0	0	5	0	4	0	0	0	0	3	2	0	0	1	0	0	0	0
0	0	0	0	0	0	5	0	4	0	0	0	0	3	2	0	0	1	0	0	0
0	0	0	0	0	0	0	5	0	4	0	0	0	0	3	2	0	0	1	0	0

Table 5.10. 2-Gossip (21, 21, 6) code constructed from 2-TS (5, 21, 21)

Generalised Gossip codes: Consider the Gossip code construction presented in the above sections. It is easy to see that in each gossip column, every non-zero symbol appears exactly once. However, it is possible to construct an IPP code, with every

non-zero symbol appearing at the most once in each column such that all possible groups of size c are accusation groups for some code-column or the other. We call these codes as Generalised Gossip codes (GGC). Here is an example presented in [HLLT98].

Take $n = 3$. Let $m := \left\lfloor \frac{q-1}{2} \right\rfloor$, $Q = \{1, 2, ..., q\}$. The code Γ consists of the following words.

(i) (0, 0, 0)

(ii) $(0, i, i)$ with $1 \le i \le m$,

(iii) $(i, 0, i+m)$ with $1 \le i \le m$,

(iv) $(i, i, 0)$ with $m+1 \le i \le q-1$

Clearly Γ has $q+m$ words. In each position, every non-zero symbol occurs at the most in one codeword. For $q = 5$, this is 2-Gossip(3, 7, 5) code with an extra codeword namely zero word. But for $q = 6$, the third column does not contain the symbol '5'.

Frameproof codes from Gossip codes: Let g_{ij} denote the $(i, j)^{\text{th}}$ element of the Gossip code under consideration. Let f_{ij} denote the $(i, j)^{\text{th}}$ element of the frameproof code to be constructed from Gossip code. This construction is also based on the result that there exists a *t*-design that corresponds to a given Gossip code (see Theorem 5.1) and the construction of frameproof codes from *t*-designs presented in [SW98]. The construction of frameproof code from Gossip code is as follows.

Choose $f_{ij} = 1$ if $g_{ji} \neq 0$

$f_{ij} = 0$ otherwise.

This choice provides the construction technique of frameproof codes from Gossip codes.

Example 5.5

Consider the 2-Gossip (7, 7, 4) code in Table 5.11. Replacing all the non-zero symbols with 'ones' and transposing the resultant matrix we get the frameproof code presented in Table 5.12.

1	0	0	0	1	0	1
2	1	0	0	0	1	0
0	2	1	0	0	0	2
3	0	2	1	0	0	0
0	3	0	2	2	0	0
0	0	3	0	3	2	0
0	0	0	3	0	3	3

Table 5.11. 2-Gossip (7, 7, 4) code

The frameproof code constructed is a 2-FP(7, 7) code. It can be verified that Table 5.12 is indeed a frameproof code. This construction is also helpful to construct embedded frameproof codes from embedded Gossip codes (See section 5.4 below).

1	1	0	1	0	0	0
0	1	1	0	1	0	0
0	0	1	1	0	1	0
0	0	0	1	1	0	1
1	0	0	0	1	1	0
0	1	0	0	0	1	1
1	0	1	0	0	0	1

Table 5.12. 2-FP(7, 7) code

5.4 Embedded Gossip codes

In many cases such as broadcast encryption applications and fingerprinting, the number of users in a scheme will increase after the system is set up. Initially the data supplier will construct a scheme that will accommodate a fixed number of users say M. If the number of users exceeds M, we need to extend the scheme that is compatible with the existing scheme. Such scenarios are possible in Traceability

Schemes apart from digital fingerprinting applications. In Traceability Schemes, it is not advisable to change the keys already issued, when the users in the system grow. Embedded Traceability Schemes extend the scheme compatible to existing Traceability Scheme. Embedded Gossip codes (see *Definition 5.5*) can be used to construct the Embedded Traceability Schemes and frameproof codes. Here we explain how a Gossip code can be embedded into another.

Definition 5.4 Let Γ be a c-*Gossip*(l, M, q) code and Γ' be a c-*Gossip*(l', M', q) code, where $M < M'$ and $l < l'$. Suppose that for every codeword $x \in \Gamma$ there exists a codeword $x' \in \Gamma'$ such that the first l symbols are the same as x. Then we say Γ is embedded into Γ'.

It is simpler to understand the embeddings in terms of set systems.
Let (X, B) and (X', B') be two set systems. (X, B) is said to be embedded into (X', B') if $X \subseteq X'$ and $B \subseteq B'$. Suppose (X, B) correspond to t-(v, k, λ) and (X', B') to t-(v', k, λ) design. We say t-(v, k, λ) is embedded into t-(v', k, λ) if (X, B) is embedded into (X', B').

Lemma 5.8

A c-*Gossip*(l, M, q) code can be embedded into c-*Gossip*(l', M', q) code if and only if the respective c-$(M, q-1, 1)$ design is embedded in c-$(M', q-1, 1)$.

Proof. The proof follows from Theorem 5.1.

For example a 2-*Gossip*$(7, 7, 4)$ code can be embedded into 2-*Gossip*$(35, 15, 4)$ code.

5.5 Significance of shortest Gossip codes

The constructions of Gossip codes solve two of the open problems proposed by Staddon et al [SSW01] on the existence of

1. c-IPP code such that $c < q < \left\lfloor \frac{(c+2)^2}{4} \right\rfloor$
2. c-TA code such that $q < c^2$ and $M > q$.

We give examples for these codes from our constructions. Consider a Gossip code with $c=3,\ q=4$ and $M=5$. This implies the length of the code is $l=10$. Further $3<4<\left\lfloor \frac{(3+2)^2}{4} \right\rfloor$. The construction of the Gossip code in Example 5.6 is possible since $c=q-1$.

Example 5.6

1	1	1	1	0	0	0	1	1	0
2	2	2	0	1	1	1	0	0	0
3	0	0	2	2	2	0	2	0	1
0	3	0	0	3	0	2	3	2	2
0	0	3	3	0	3	3	0	3	3

Table 5.13. 3-Gossip(10, 5, 4) code

The following conditions for which a Gossip code can become a TA code are presented in [LLS02].

1. A $(q-1)$-*Gossip*$(l,\ M,\ q)$ code is a 2-*TA* code whenever $q\geq 3$.
2. A $(q-1)$-*Gossip*$(l,\ q+2,\ q)$ is a 3-*TA* code when ever $q\geq 6$.

Now, we construct *c*-TA code (Example 5.7) such that $q<c^2$ and $M>q$.

Consider 2-Gossip(10, 5, 3) code. Here $c=2,\ M=5,\ q=3$ and $l=10.$

This choice satisfies the condition $q<c^2$ and $M>q$.

Example 5.7

1	1	1	1	0	0	0	0	0	0
2	0	0	0	1	1	1	0	0	0
0	2	0	0	2	0	0	1	1	0
0	0	2	0	0	2	0	2	0	1
0	0	0	2	0	0	2	0	2	2

Table 5.14. 2-Gossip(10, 5, 3) code

Theorem 5.4 and 5.5 are two known and important results about *t*-designs.

Theorem 5.4 *[CD96]* If (X, B) is a t-(v, k, λ) design and S is any s-element subset of X, with $0 \le s < t$, then the number of blocks containing S is $\lambda_s = |\{P \in B : S \subseteq P\}| = \lambda.\left({}^{v-s}C_{t-s} \big/ {}^{k-s}C_{t-s}\right)$.

Theorem 5.5 *[CD96]* If (X, B) is a $t-(v,k,\lambda)$ design and S is any $s-$element subset of X, with $0 \le s \le t$, then the number of blocks does not contain any point of S is $\bar{\lambda}_s = |\{P \in B : P \cap S = \phi\}| = \lambda.\left({}^{v-s}C_{k} \big/ {}^{v-t}C_{k-t}\right)$.

Theorem 5.6 presents the length, weight and distance of a shortest Gossip code.

Theorem 5.6 For a shortest *c-Gossip*(l, M, q) code

1. Length of the code $l = n(M,q) = {}^{M}C_{c} \big/ {}^{q-1}C_{c}$
2. Weight of the code $w(M,q) = \lambda_1 = {}^{M-1}C_{c-1} \big/ {}^{q-2}C_{c-1}$
3. Distance of the code $d(M,q) = l - {}^{M-2}C_{q-1} \big/ {}^{M-c}C_{q-1-c}$

Proof.

1. From a gossip column of a Gossip code, where $c < q < M$, exactly ${}^{q-1}C_{c}$ collusion groups are traceable. If M is the total number of codewords in a Gossip code, then the total number of possible pirate groups of size *c* is given by ${}^{M}C_{c}$. For $c \le q-1$, each gossip column at the most can accuse ${}^{q-1}C_{c}$ groups since there are $(q-1)$ non-zero symbols, which are distinct. Hence, in a Gossip code the total number of gossip columns $l \ge {}^{M}C_{c} \big/ {}^{q-1}C_{c}$ for $c < q < M$. This gives a lower bound on length of the Gossip code. Hence the shortest Gossip code length attains the equality condition.

2. We have shown in Theorem 5.1 that a shortest *c-Gossip*(l, M, q) code exists if and only if t-$(v,k,1)$ design exists with $t = c, v = M, k = q-1$. Looking at the t-design's blocks, we can tell which group's pirate words will have 0 at a given position, and which groups will not. Further the j^{th} position in a codeword is

non-zero if $W \cap B_j \neq \phi$. Consider $|W|=1$, thus from Theorem 5.4, the number of blocks that contain W are λ_1 where λ_1 is computed with $t=c, v=M$, and $k=q-1$. Thus λ_1 is the number positions not equal to 0.

3. Distance between two codewords is the number of positions in which they differ. If the distance between any two codewords is constant, then distance of the code will be equal to that distance. Consider $|W|=2$. If $W \subseteq X - B_i$ (implies $W \cap B_i = \phi$) and $|W| \leq t$ then the collusion group W *cannot* detect the i^{th} position since the i^{th} position is *zero* in all the codewords of W. From Theorem 5.5, the number of blocks that do not contain W is $\overline{\lambda_2}$. This means the number of undetectable positions (zeros) for W is $\overline{\lambda_2}$. Thus the number of undetectable positions for any collusion size '2' is $\overline{\lambda_2}$. The number of mismatched positions between any two codewords is $l - \overline{\lambda_2}$. Thus the distance between any two code words is $l - \overline{\lambda_2}$ where $\overline{\lambda_2}$ is as defined above with $t=c, v=M, k=q-1$. Thus the distance of the code becomes $l - {}^{M-2}C_{q-1} \big/ {}^{M-c}C_{q-1-c}$.

Substituting $c=q-1$ in this theorem Lemma 5.2 can be derived. From this theorem, the computed weight of the 2-Gossip (7, 7, 4) code is 3 and distance is equal to 5, which can be verified with the code presented in Table 5.2.

5.5.1 Advantages of shortest Gossip codes

There are various advantages of shortest Gossip codes. Some of them are listed here.

1. Shortest Gossip codes cause less distortion due to shorter length when compared to normal Gossip codes.
2. They provide deterministic tracing for pirates
3. These codes tolerate erasures (see chapter 6 for details).
4. It is possible to construct *t*-designs and Tractability Schemes from these codes
5. Construction of frameproof codes is also possible from shortest Gossip codes.

6. They are constant weight codes.
7. The distance between two codewords of a shortest Gossip code is also equal.
8. Embedded Gossip codes can be to construct Embedded Traceability Schemes and embedded fingerprinting codes which can expand an existing system to a bigger one.
9. Concatenated code that exploit shortest Gossip codes are also presented in chapter 6.

Shortest Gossip codes are constant weight codes. The distance between two codewords of a shortest Gossip code is also equal.

These Gossip codes cause less distortion during embedding due to their shorter length as compared to normal Gossip codes, since the modifications to be made in the original are less. These codes provide deterministic tracing for pirates and tolerate erasures. It is possible to construct t-designs and Tractability Schemes from these codes. The construction of frameproof codes is also possible from shortest Gossip codes. Embedded Gossip codes can be used to construct Embedded Traceability Schemes and embedded fingerprinting codes which can expand an existing broadcast system to a bigger one. Concatenated codes that exploit shortest Gossip codes are presented in chapter 6.

5.6 Construction of Concatenated Codes

A Concatenated code consists of an inner code and an outer code, where the symbols in the outer code are the codewords of the inner code. Thus every codeword in the Concatenated code is the concatenated collection of inner codewords. Further, the number of codewords in inner code is equal to the number of alphabet symbols in the outer code. We denote the codewords of the inner code as $\{\overline{0}, \overline{1}, \overline{2}, ..., \overline{q-1}\}$ for convenience. We replace the symbols of outer code with inner codewords for constructing the Concatenated code. The length of the concatenated code is same as that of the outer code.

When a frameproof code is used as inner code, the pirate members cannot create alphabet symbols not found in their copies in the detected positions of the concatenated code. Thus during comparison whenever a mark is detected in the concatenated code, the pirates need to create an alphabet symbol they find in their copies or a non-alphabet symbol in the detected positions. Thus this concatenated code presents a sensible way for the implementing fingerprinting schemes under pirate model assuming marking assumption [BS98] for non-binary codes. Consider a frameproof code with $M = 4,\ q = 2$ and $c = 2$. Let $\overline{0}$ denote the first inner codeword, $\overline{1}$ denote the second, and so on.

Example 5.8

1	0	0	≡	$\overline{0}$
0	1	0		$\overline{1}$
0	0	1		$\overline{2}$
1	1	1		$\overline{3}$

Table 5.15. 2-FP(3, 4) code

If we use the Gossip code in Table 5.2 as outer code for the Concatenated code, the Concatenated code will be:

$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$		w^1
$\overline{2}$	$\overline{0}$	$\overline{0}$	$\overline{1}$	$\overline{1}$	$\overline{0}$	$\overline{0}$		w^2
$\overline{3}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{1}$	$\overline{1}$		:
$\overline{0}$	$\overline{2}$	$\overline{0}$	$\overline{2}$	$\overline{0}$	$\overline{0}$	$\overline{2}$	≡	:
$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{0}$	$\overline{2}$	$\overline{2}$	$\overline{0}$		:
$\overline{0}$	$\overline{0}$	$\overline{2}$	$\overline{3}$	$\overline{0}$	$\overline{3}$	$\overline{0}$		:
$\overline{0}$	$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{3}$		w^7

Table 5.16. A concatenated Gossip code

The collusion size c is two for the concatenated code. Now, we discuss the tracing for the Concatenated code defined above.

Tracing: Consider the collusion set W, consisting of w^1 and w^2. The descendent set is $D(W) = \{(a_1,...,a_5,\overline{0},\overline{0}) / a_1 \in \{\overline{1},\ \overline{2}\}, a_i \in \{\overline{1},\ \overline{0}\} \text{ for } 2 \le i \le 4\}$ since the last two positions are undetected. The choices in the other positions are based on the choices available in the inner code (i.e. code given in Table 5.14). Let a pirate word created by the collusion set $W = \{w^1, w^2\}$ be $x = \{0\ 0\ 1\ \ 0\ 0\ 0\ \ 0\ 0\ 0\ \ 0\ 1\ 0\ \ 0\ 0\ 0\ \ 1\ 0\ 0\ \ 1\ 0\ 0\}$. This can be re-written as $\{\overline{2},\ \overline{e},\ \overline{e},\ \overline{1}\ ,\overline{e},\ \overline{e},\ \overline{0},\ \overline{0}\}$. This is the selective erasures case of simple Gossip code. Further, the pirate word contains a nonzero symbol. In this case, the tracing algorithm runs in time $O(Ml)$ and will trace w^2.

5.7 Performance Analysis

We need to know what the pirates can possibly do to analyze a fingerprinting code and the tracing method. Further, we need to model the possible pirate strategies to examine the performance of a fingerprinting system. Since only a fraction of the total available locations are marked, it is highly difficult for the pirates to guess the marked positions. Pirates can detect only those positions that differ in their copies under *marking assumption* [BS98]. The pirates may choose one of the marks they have found at the respective positions when creating a pirate copy. A mark is detected when it is different in two copies during comparison. Thus the detected positions in a binary system reveal all types of embedded marks (0 and 1) but in a non-binary system, they reveal only a few of them. For creating an illegal word, at each detected position the pirates may choose different types of erasures namely:

- a symbol, which they cannot find in their copies
- a symbol not in the alphabet of the fingerprinting code
- a symbol that pirates cannot see or a non-alphabet
- any symbol without restrictions

The fingerprinting model should identify whether it is possible for pirates to create marks they cannot *find*. Encoding the alphabets prior to embedding is one way to limit the pirate's choices at the detected positions.

With a non-binary code, two different user groups can have the same detectable

positions (and undetectable positions) but still have different illegal words. The number of erasures will probably increase with a non-binary code (with large q) and erasures will not reveal so much information about the pirate group as in binary (small q) case.

5.7.1 Performance of Gossip codes

One advantage of Gossip codes is that they can accuse a traitor deterministically. Nevertheless all the previous constructions were probabilistic in some way or other. In general, the focus in all probabilistic code constructions is to keep the length of each codeword short. Short length is important to design efficient and effective embedding techniques of a codeword into an object. Some of the previous probabilistic codes managed to achieve polylog size in the number of the users. But the length of the code tends to be very large (to be interpreted as proportional to the number of users) if deterministic full tracing solutions are sought. Thus Gossip codes in general, have much greater length than probabilistic codes. Our constructions for Gossip codes achieve the minimum possible code length specified for Gossip codes. The additional advantage of Gossip codes comes from the fact that they can withstand erasures and tracing is possible in presence of erasures. For improved results concatenated codes with outer code as Gossip code and inner codes with probabilistic tracing can be used.

5.8 Additional Examples

In this section we present some additional examples on construction of Gossip codes based on the methods presented earlier in this chapter.Example 5.9 presents the construction of 2-Gossip(13, 13, 5) code from partition method. The number of codewords is equal to length of the code in this example.

Example 5.9

Consider M =13, q =5, c =2

Then ${}^{M}C_{c} = {}^{13}C_{2} = \frac{13\times12}{2} = 78$, ${}^{q-1}C_{c} = {}^{4}C_{2} = \frac{4\times3}{2} = 6$

Number of gossip columns= $l = \frac{78}{6} = 13$

Coalition	Pirate members in coalition	Gossip column	Coalition	Pirate members in coalition	Gossip column
C_1	{1,2}	1	C_{40}	{4,11}	13
C_2	{1,3}	1	C_{41}	{4,12}	10
C_3	{1,4}	1	C_{42}	{4,13}	11
C_4	{1,5}	2	C_{43}	{5,6}	2
C_5	{1,6}	2	C_{44}	{5,7}	2
C_6	{1,7}	2	C_{45}	{5,8}	5
C_7	{1,8}	3	C_{46}	{5,9}	8
C_8	{1,9}	3	C_{47}	{5,10}	10
C_9	{1,10}	3	C_{48}	{5,11}	5
C_{10}	{1,11}	4	C_{49}	{5,12}	10
C_{11}	{1,12}	4	C_{50}	{5,13}	8
C_{12}	{1,13}	4	C_{51}	{6,7}	2
C_{13}	{2,3}	1	C_{52}	{6,8}	11
C_{14}	{2,4}	1	C_{53}	{6,9}	6
C_{15}	{2,5}	5	C_{54}	{6,10}	12
C_{16}	{2,6}	6	C_{55}	{6,11}	12
C_{17}	{2,7}	7	C_{56}	{6,12}	6
C_{18}	{2,8}	5	C_{57}	{6,13}	11
C_{19}	{2,9}	6	C_{58}	{7,8}	9
C_{20}	{2,10}	7	C_{59}	{7,9}	13
C_{21}	{2,11}	5	C_{60}	{7,10}	7
C_{22}	{2,12}	6	C_{61}	{7,11}	13
C_{23}	{2,13}	7	C_{62}	{7,12}	9
C_{24}	{3,4}	1	C_{63}	{7,13}	7
C_{25}	{3,5}	8	C_{64}	{8,9}	3
C_{26}	{3,6}	12	C_{65}	{8,10}	3
C_{27}	{3,7}	9	C_{66}	{8,11}	5
C_{28}	{3,8}	9	C_{67}	{8,12}	9
C_{29}	{3,9}	8	C_{68}	{8,13}	11
C_{30}	{3,10}	12	C_{69}	{9,10}	3
C_{31}	{3,11}	12	C_{70}	{9,11}	13
C_{32}	{3,12}	9	C_{71}	{9,12}	6
C_{33}	{3,13}	8	C_{72}	{9,13}	8
C_{34}	{4,5}	10	C_{73}	{10,11}	12
C_{35}	{4,6}	11	C_{74}	{10,12}	10
C_{36}	{4,7}	13	C_{75}	{10,13}	7
C_{37}	{4,8}	11	C_{76}	{11,12}	4
C_{38}	{4,9}	13	C_{77}	{11,13}	4
C_{39}	{4,10}	10	C_{78}	{12,13}	4

Table 5.17. Accusation groups for gossip columns (2-Gossip (13,13,5) code)

Now we can partition the total pirate groups of size 2, namely C_1 to C_{78}

Column	Accusation Groups	Equivalence class
1.	$C_1,C_2,C_3,C_{13},C_{14},C_{24}$	I
2.	$C_4,C_5,C_6,C_{43},C_{44},C_{51}$	II
3.	$C_7,C_8,C_9,C_{64},C_{65},C_{69}$	III
4.	$C_{10},C_{11},C_{12},C_{76},C_{77},C_{78}$	IV
5.	$C_{15},C_{18},C_{21},C_{45},C_{48},C_{66}$	V
6.	$C_{16},C_{19},C_{22},C_{53},C_{56},C_{71}$	VI
7.	$C_{17},C_{20},C_{23},C_{60},C_{63},C_{75}$	VII
8.	$C_{25},C_{29},C_{33},C_{46},C_{50},C_{72}$	VIII
9.	$C_{27},C_{28},C_{32},C_{58},C_{62},C_{67}$	IX
10.	$C_{34},C_{39},C_{41},C_{47},C_{49},C_{74}$	X
11.	$C_{35},C_{37},C_{42},C_{52}, C_{57},C_{68}$	XI
12.	$C_{26},C_{30},C_{31},C_{54},C_{55},C_{73}$	XII
13.	$C_{36},C_{38},C_{40},C_{59},C_{61},C_{70}$	XIII

Table 5.18. Partitioning of the collusion groups (2-Gossip (13,13,5) code)

The 2- Gossip(13, 13, 5) code is presented in Table 5.19.

1	1	1	1	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	1	0	0	0	0	0	0
3	0	0	0	0	0	0	1	1	0	0	1	0
4	0	0	0	0	0	0	0	0	1	1	0	1
0	2	0	0	2	0	0	2	0	2	0	0	0
0	3	0	0	0	2	0	0	0	0	2	2	0
0	4	0	0	0	0	2	0	2	0	0	0	2
0	0	2	0	3	0	0	0	0	0	3	0	0
0	0	3	0	0	3	0	3	3	0	0	0	3
0	0	4	0	0	0	3	0	0	3	0	3	0
0	0	0	2	4	0	0	0	0	0	0	4	4
0	0	0	3	0	4	0	0	4	4	0	0	0
0	0	0	4	0	0	4	4	0	0	4	0	0

Table 5.19. 2- Gossip(13,13,5) code from Partition method

Example 5.10 presents the construction of 2-Gossip(82, 41, 6) code from the 2-Traceability Scheme namely 2-TS(5, 82, 41). In this example number of gossip columns are 82 and total number of codewords is 41.

Example 5.10

Base Keys =T = {1 to 41}

Private Keys of 2-TS (5, 82, 41) are given by ($i = 1, 2, .., 41$):

$P(i) = \{i,\ 9+i,\ 17+i,\ 15+i,\ 36+i\}$

$P(41+i) = \{35+i,\ 31+i,\ 32+i,\ 1+i,\ 19+i\}$

The private keys in 2-TS(5, 82, 41), which are same as column keys are

User No	Private Key	User No	Private Key
1.	{1,10,18,16,37}	42.	{1,10,18,16,37}
2.	{2 11,19,17,38}	43.	{2,11,19,17,38}
3.	{3,12,20,18,39}	44.	{3,12,20,18,39}
4.	{4 13,21,19,40}	45.	{4,13,21,19,40}
5.	{5,14,22,20,41}	46.	{5,14,22,20,41}
6.	{6,15,23,21,1}	47.	{6,15,23,21,1}
7.	{7,16,24,22,2}	48.	{7,16,24,22,2}
8.	{8,17,25,23,3}	49.	{8,17,25,23,3}
9.	{9,18,26,24,4}	50.	{9,18,26,24,4}
10.	{10,19,27,25,5}	51.	{10,19,27,25,5}
11.	{11,20,28,26,6}	52.	{11,20,28,26,6}
12.	{12,21,29,27,7}	53.	{12,21,29,27,7}
13.	{13,22,30,28,8}	54.	{13,22,30,28,8}
14.	{14,23,31,29,9}	55.	{14,23,31,29,9}
15.	{15,24,32,30,10}	56.	{15,24,32,30,10}
16.	{16,25,33,31,11}	57.	{16,25,33,31,11}
17.	{17,26,34,32,12}	58.	{17,26,34,32,12}
18.	{18,27,35,33,13}	59.	{18,27,35,33,13}
19.	{19,28,36,34,14}	60.	{19,28,36,34,14}
20.	{20,29,37,35,15}	61.	{20,29,37,35,15}
21.	{21,30,38,36,16}	62.	{21,30,38,36,16}
22.	{22,31,39,37,17}	63.	{22,31,39,37,17}
23.	{23,32,40,38,18}	64.	{23,32,40,38,18}
24.	{24,33,41,39,19}	65.	{24,33,41,39,19}
25.	{25,34,1,40,20}	66.	{25,34,1,40,20}
26.	{26,35,2,41,21}	67.	{26,35,2,41,21}
27.	{27,36,3,1,22}	68.	{27,36,3,1,22}
28.	{28,37,4,2,23}	69.	{28,37,4,2,23}
29.	{29,38,5,3,24}	70.	{29,38,5,3,24}
30.	{30,39,6,4,25}	71.	{30,39,6,4,25}
31.	{31,40,7,5,26}	72.	{31,40,7,5,26}
32.	{32,41,8,6,27}	73.	{32,41,8,6,27}
33.	{33,1,9,7,28}	74.	{33,1,9,7,28}
34.	{34,2,10,8,29}	75.	{34,2,10,8,29}
35.	{35,3,11,9,30}	76.	{35,3,11,9,30}
36.	{36,4,12,10,31}	77.	{36,4,12,10,31}
37.	{37,5,13,11,32}	78.	{37,5,13,11,32}
38.	{38,6,14,12,33}	79.	{38,6,14,12,33}

39.	{39,7,15,13,34}	80.	{39,7,15,13,34}
40.	{40,8,16,14 35}	81.	{40,8,16,14,35}
41.	{41,9,17,15,36}	82.	{41,9,17,15,36}

Table 5.20. 2-Traceablity Scheme (2-TS (5,82,41))

Total number of user groups of size '2' $= {}^{v}C_{c} = {}^{41}C_{2} = \frac{41(40)}{2} = 820$

Accusation groups per column $= {}^{q-1}C_{c} = {}^{k}C_{c} = {}^{5}C_{2} = 10$

Number of gossip columns= $l = \frac{820}{10} = 82$

Accusation groups for i^{th} column are the set of all groups of size 2 constructed from i^{th} column key. Considering the first column key i.e. {1,10,18,16,37} we get the following accusation groups for the first gossip column.

Column	Accusation Groups
1	{1,10}, {1,18}, {1,16}, {1,37}, {10,18}, {10,16}, {10,37}, {18,16}, {18,37}, {16,37}

Table 5.21. Accusation groups for the 1st gossip column (2-Gossip(82,41,6) code)

Similar accusation groups exist corresponding to for other gossip columns. Now Gossip code similar to Example 5.4 can be constructed by taking each private key as column key. Some of the entries are filled here under.

Column à **Row No** â	1	2	3	……	81	82
1	1	1	1	……	0	0
2	0	0	0	……	0	0
:	:	:	:		:	:
:	:	:	:		:	:
:	:	:	:		:	:
:	:	:	:		:	:
40	0	0	0	……	1	0
41	0	0	0	……	0	1

Table 5.22. 2-Gossip(82,41,6) code constructed from 2-TS(5,82,41)

The following example (Example 5.11) presents an alternate construction for 2-Gossip(21, 21, 6) code. It also confirms the observation that there exist multiple Gossip codes for the same code parameters.

Example 5.11

Consider M =21, q =6, c =2

The 2-Traceability Scheme under consideration is as follows.

User No	Private Key
1.	{1, 2, 3, 4, 5}
2.	{1, 6, 7, 8, 9}
3.	{1, 10, 11, 12, 13}
4.	{1, 14, 15, 16, 17}
5.	{1, 18, 19, 20, 21}
6.	{2, 6, 10, 14, 18}
7.	{2, 7, 11, 15, 19}
8.	{2, 8, 12, 16, 20}
9.	{2, 9, 13, 17, 21}
10.	{3, 6, 11, 16, 21}
11.	{3, 7, 10, 17, 20}
12.	{3, 8, 13, 14, 19}
13.	{3, 9, 12, 15, 18}
14.	{4, 6, 12, 17, 19}
15.	{4, 7, 13, 16, 18}
16.	{4, 8, 10, 15, 20}
17.	{4, 9, 11, 14, 21}
18.	{5, 6, 13, 15, 21}
19.	{5, 7, 12, 14, 20}
20.	{5, 8, 11, 17, 18}
21.	{5, 9, 10, 16, 19}

Table 5.23. 2-Traceablity Scheme (Alternate 2-TS (5,21,21))

We have ${}^{M}C_{c} = {}^{21}C_{2} = \frac{21\times 20}{2} = 21\times 10$

and ${}^{q-1}C_{c} = {}^{5}C_{2} = \frac{5\times 4}{2} = 10$

Thus $l = \frac{21\times 10}{10} = 21$

Accusation groups for i^{th} column are the set of all groups of size 2 constructed from i^{th} column key. Considering the first column key, we get the following accusation groups. Similar accusation groups exist corresponding to for other gossip columns.

Column	Accusation Group
1	{1,2}, {1,3}, {1,4}, {1,5}, {2,3}, {2,4}, {2,5}, {3,4}, {3,5}, {4,5}

Table 5.24. Accusation group for 1st gossip column from Table 5.23

The 2-Gossip(21, 21, 6) code constructed is as follows.

1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
0	2	0	0	0	2	0	0	0	2	0	0	0	2	0	0	0	2	0	0	0
0	3	0	0	0	0	2	0	0	0	2	0	0	0	2	0	0	0	2	0	0
0	4	0	0	0	0	0	2	0	0	0	2	0	0	0	2	0	0	0	2	0
0	5	0	0	0	0	0	0	2	0	0	0	2	0	0	0	2	0	0	0	2
0	0	2	0	0	3	0	0	0	0	3	0	0	0	0	3	0	0	0	0	3
0	0	3	0	0	0	3	0	0	3	0	0	0	0	0	0	3	0	0	3	0
0	0	4	0	0	0	0	3	0	0	0	0	3	3	0	0	0	0	3	0	0
0	0	5	0	0	0	0	0	3	0	0	3	0	0	3	0	0	3	0	0	0
0	0	0	2	0	4	0	0	0	0	0	4	0	0	0	0	4	0	4	0	0
0	0	0	3	0	0	4	0	0	0	0	0	4	0	0	4	0	4	0	0	0
0	0	0	4	0	0	0	4	0	4	0	0	0	0	4	0	0	0	0	0	4
0	0	0	5	0	0	0	0	4	0	4	0	0	4	0	0	0	0	0	4	0
0	0	0	0	2	5	0	0	0	0	0	0	5	0	5	0	0	0	0	5	0
0	0	0	0	3	0	5	0	0	0	0	5	0	5	0	0	0	0	0	0	5
0	0	0	0	4	0	0	5	0	0	5	0	0	0	0	5	0	0	5	0	0
0	0	0	0	5	0	0	0	5	5	0	0	0	0	0	0	5	5	0	0	0

Table 5.25. An alternate construction for 2-Gossip(21,21,6) code

In Example 5.12 we present the construction of 3-Gossip(30, 10, 5) code. It can also be observed that Gossip codes that sustain larger collusion sizes will have larger length and/or larger alphabet size.

Example 5.12

The 30 blocks of a 3-(10, 4, 1) design are as follows.

B_1={1 2 3 4} B_2={1, 5 6 7} B_3={2, 5, 8, 9}

B_4={3, 6, 8, 10} B_5={4, 7, 9, 10} B_6={6, 7, 8, 9}

B_7={3, 4, 8, 9} B_8={3, 4, 6, 7} B_9={2, 4, 5, 7}

B_{10}={2, 3, 5, 6} B_{11}={5, 7, 8, 10} B_{12}={2, 4, 8, 10}

B_{13}={1, 4, 6, 10} B_{14}={1, 4, 5, 9} B_{15}={1, 3, 5, 8}

B_{16}={5, 6, 9, 10} B_{17}={2, 3, 9, 10} B_{18}={1, 3, 7, 10}

B_{19}={1, 2, 7, 9} B_{20}={1, 2, 6, 8} B_{21}={1, 8, 9, 10}

B_{22}={2, 6, 7, 10} B_{23}={3, 5, 7, 9} B_{24}={4, 5, 6, 8}

B_{25}={3, 4, 5, 10} B_{26}={2, 4, 6, 9} B_{27}={2, 3, 7, 8}

B_{28}={1, 4, 7, 8} B_{29}={1, 3, 6, 9} B_{30}={1, 2, 5, 10}

From this t-design the Gossip code constructed (as per Theorem 5.1) is as follows.

1	1	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1	1	1	0	0	0	0	0	0	1	1	1	0
2	0	1	0	0	0	0	1	1	0	1	0	0	0	0	1	0	2	2	0	1	0	0	0	1	1	0	0	2	0
3	0	0	1	0	0	1	0	2	0	0	0	0	2	0	2	2	0	0	0	0	1	0	1	0	2	0	2	0	1
4	0	0	0	1	0	2	2	0	0	2	2	2	0	0	0	0	0	0	0	0	0	1	2	2	0	2	0	0	2
0	2	2	0	0	0	0	3	3	1	0	0	3	3	1	0	0	0	0	0	0	2	2	3	0	0	0	0	3	0
0	3	0	2	0	1	0	0	4	0	0	3	0	0	2	0	0	0	3	0	2	0	3	0	3	0	0	3	0	3
0	4	0	0	2	2	0	4	0	2	0	0	0	0	0	0	3	3	0	0	3	3	0	0	0	3	3	0	0	4
0	0	3	3	0	3	3	0	0	3	3	0	0	4	0	0	0	0	4	2	0	0	4	0	0	4	4	0	0	0
0	0	4	0	3	4	4	0	0	0	0	0	4	0	3	3	0	4	0	3	0	4	0	0	4	0	0	4	0	0
0	0	0	4	4	0	0	0	0	4	4	4	0	0	4	4	4	0	0	4	4	0	0	4	0	0	0	0	4	0

Table 5.26. 3-Gossip(30, 10, 5) code constructed from 3-(10, 4, 1) design

5.9 Embedding

We present simple watermarking techniques, which are based on wavelet transforms to embed (hide) the codewords of a Gossip code into images. The distributor may choose any of the embedding method from the available watermarking techniques based on the requirement and type of the media.

5.9.1 Wavelet Watermarking Techniques

The Discrete Wavelet Transform (DWT) separates an image into a lower resolution approximation (*LL*) as well as horizontal (*HL*), vertical (*LH*) and diagonal (*HH*) detail components. The process can be repeated to compute multiple 'scale' wavelet decomposition similar to the two-scale wavelet transform as shown below.

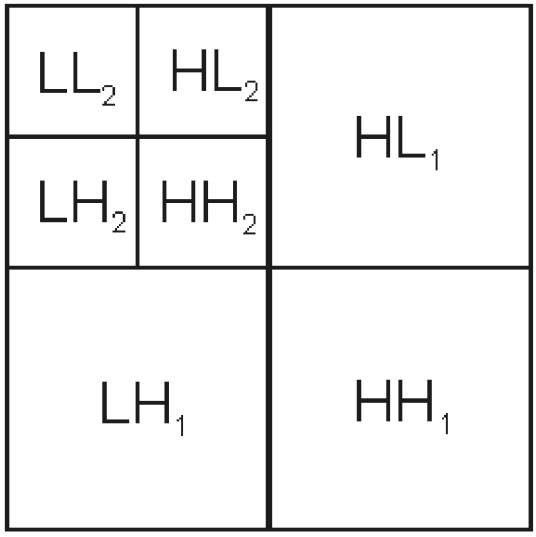

Figure 5.1. Wavelet Decomposition

One of the advantages of the Wavelet transform is that it is believed to more accurately model the Human Visual System (HVS) as compared to Fast Fourier Transform (FFT) and Discrete Cosine Transform (DCT). This allows us to use higher energy watermarks in regions that the HVS is known to be less sensitive to; such as the high-resolution detail bands *LH*, *HL*, and *HH*.

Method I

We use a straightforward technique for embedding the Gossip codeword as a CDMA watermark sequence in the detail bands according to the equation [CKLS97] shown below:

$$I_{V_{a,b}} = \begin{cases} V_i + \alpha . |V_i| . wm_i, & a,b \in HL, LH \\ V_i & a,b \in LL, HH \end{cases} \qquad (1)$$

where V_i denotes the coefficient of the transformed image, wm_i one bit of the watermark to be embedded, and a scaling factor α. In order to detect the watermark, we generate the same random number sequence and determine its correlation with the two transformed detail bands. If the correlation exceeds some threshold $\hat{T}$, the watermark is detected. Thus, in this method watermark detection does not require the original image. This method can be easily extended to embed multiple watermarks into the image. The robustness evaluation was limited to testing against JPEG

compression and the addition of random noise.

The pseudo-code for embedding and recovery of CDMA watermark into H1, V1, D1 components of a 1-scale DWT, using Matlab^{TM} [MAT 03] toolkits is given below:

Embedding

1. *read in the cover object*
2. *determine size of cover image*
3. *decompose the image (use dwt2 and 'harr' wavelet)*
4. *read in message (watermark i.e., Gossip codeword)*
5. *reshape the message (Gossip codeword) into a 2D-vector.*
6. *read the seed (key) for PN generator*
7. *reset MATLAB's PN generator to state 'key'*
8. *fix the seed using a key to get same set of random numbers (pn_sequence even for recovery)*
9. *set the gain factor k (=2)*
10. *add pn sequences to H1 and V1 componants when message bit = 0*
    a. *cH1=cH1+k*pn_sequence_h;*
    b. *cV1=cV1+k*pn_sequence_v;*
11. *perform IDWT*
12. *convert back to unsigned int (uint8)*

Recovery

1. *reset MATLAB's pseudo random number generator to state 'key'*
2. *initalize message to all ones*
3. *Compute dwt2 of watermarked image*
4. *generate same pseudo random number sequences*
5. *Find the correlations*
    a. *correlation_h(kk)=corr2(cH1,pn_sequence_h);*
    b. *correlation_v(kk)=corr2(cV1,pn_sequence_v);*
    c. *correlation(kk)=(correlation_h(kk)+correlation_v(kk))/2;*
6. *if (correlation(kk) > mean(correlation))*
   then flip the i^{th} message bit to 0.
7. *Message recovered itself is the recovered watermark*

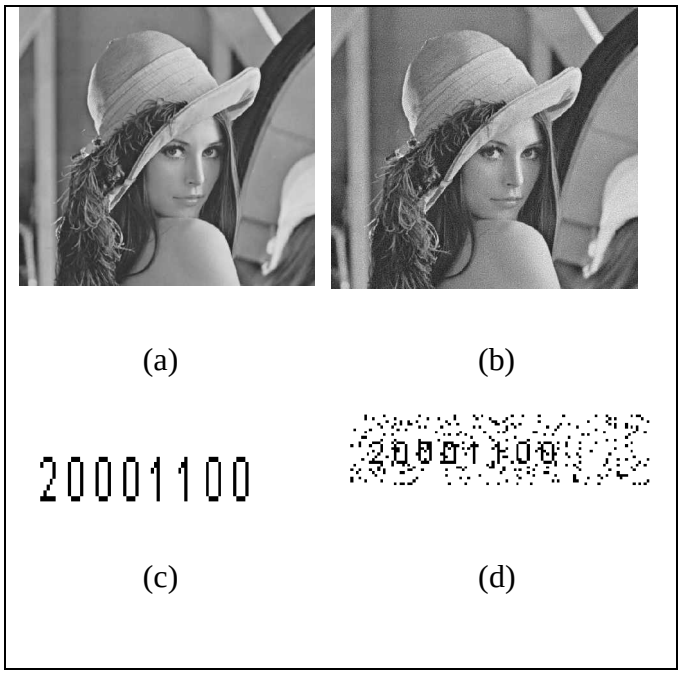

Figure 5.2. (a) Original Image (b) Watermarked Image (c) Embedded Watermark image (d) Recovered Watermark

Method II

This method is based on a technique proposed by Xia *et al.* in [XBA97] and uses non-oblivious watermark. In copyright protection, it can be assumed that the owner who verifies the presence of a particular watermark has access to the original image. Such watermarking schemes are referred to as non-oblivious (or private) watermarking as against oblivious (or public) techniques. Our aim is to implement a perceptual multi-resolution watermarking scheme for digital images and test its robustness for different common image alterations. Here the watermark (A Gossip codeword) is embedded as binary bit sequence. We have tested the overall robustness of the watermarked image to several basic image processing operations among which are included additive white gaussian noise, JPEG compression and image halftoning. In order to weigh the watermark according to the magnitude of the wavelet coefficients, we use *first* one of the two following relations *i.e.*, equation (2) between the original coefficients y, modified coefficients $\tilde{y}$, which contain the watermark wm:

$$\hat{y}[m,n] = y[m,n] + \alpha(y[m,n]^2).wm[m,n] \quad (2)$$

Or

$$\hat{y}[m,n] = y[m,n] + \alpha.abs(y[m,n]).wm[m,n] \quad (3)$$

It may be noted that the relations (2) and (3), even though mathematically different, achieve exactly the same goal. This choice provides more weight to the watermark added to high value wavelet coefficients. The parameter α controls the strength of the watermark; it is in fact a good way to choose between transparency and robustness or a tradeoff between the two. Finally, the 2D- IDWT of $\hat{y}$ is computed to form the watermarked image.

At the other end of the communication channel, a decoder is used to extract the watermarked information from the received image. Upon reception of the supposedly watermarked image, the algorithm first isolates the *signature* included in this image by comparing the DWT coefficients of the image with those of the original image. Using a method close to Cox *et al.* in [CKLS97] and Inoue *et al.* [IMK99], we compared the found *signature* with the watermark key bank by correlating them. Then, a decision is made based on the threshold. Thus the detection phase consists of taking the watermark key and contrasting it with the found signature (Figure 5.3(c)) by computing the cross correlation at the first resolution level (*i.e.*, highest frequency coefficients).

The pseudo-code for embedding of watermark using a multi resolution wavelet watermarking technique is given below:

Embedding

1. *read the cover image*
2. *decompose the image using wavedec2 (3 levels, 'harr' wavelet)*
3. *choose an 'α ' and the watermark (a Gossip codeword)*
4. *Embed the watermark using equation 2 in the wavelet coefficients*

Detection

1. *non-blind watermark detection requires the original watermark (wm)*
2. *consider a bank, the collection of possible watermarks*

3. *use 2-Phase detection: Suspect and Confirm*

   Suspect: Is there any watermark that we are looking for?

   Confirm: Is this identified watermark is close enough to the watermark extracted (difference image) from the received image.
4. *Find the signature (difference image of watermarked and original image)*
5. *Correlate the signature with the all watermarks from the bank*
6. *When the correlation exceeds the threshold for a particular watermark from the bank, it is suspect*
7. *To confirm the existence of the watermark use signature image and the suspect watermark to find the cross correlation*
8. *When a peak is detected it is positive identification of the watermark.*

5.9.2 Testing

In order to test the robustness of our implementation, we have used the same testing procedure as the one used by Xia *et al.* in [XBA97]. The characteristics of chosen attacks are summarized as follows.

- Noise: Modifies the intensity of pixels of the watermarked image. Gaussian noice is added to the watermark image before detection. See Figure 5.3 (e).
- JPEG compression: Introduces blocking artifacts in the compressed images. Figure 5.3(d).
- Halftoning: It is used for printing; which gives the impression of gray levels, whereas, there are actually only two levels (black & white). Figure 5.3(f).

(a)

(b)

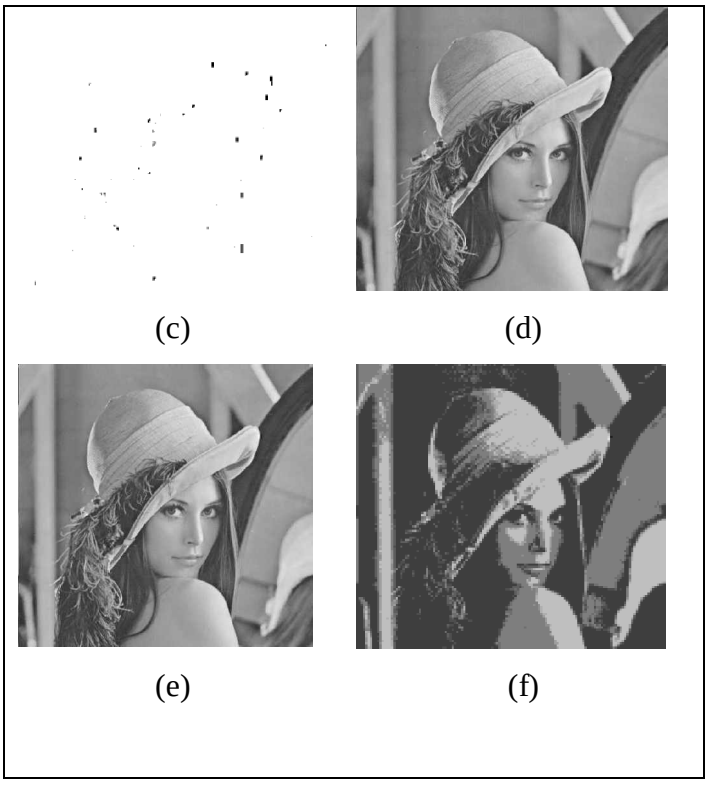

Figure 5.3. (a) Original Image (b) Watermarked Image (c) Difference/ signature Image (d) Compressed Image (e) Noise Image (f) Half Toned Image

5.10 Summary

We present the construction and analysis of a non-binary IPP code namely Gossip code. Two methods of construction for Gossip codes that achieve shortest code length are presented namely from t-designs and Traceability Schemes with vice versa. Apart from this, the construction of embedded Gossip codes is also described for extending an existing Gossip code to a bigger code. Concatenated codes are presented for the realization of the pirate model and tracing as per the scenario is also presented. The performance issues are also discussed.

Chapter 6

Erasures in non-binary fingerprinting systems

6.1 Introduction

In this chapter, we consider erasures in non-binary codes and Gossip codes with shortest code length, in particular. We primarily analyse erasures in the shortest Gossip codes in unreadable digit model [GT03]. We also consider the applicability of other erasure models to Gossip codes by extending the analysis to other erasure models. We also extend the analysis to concatenated codes that use shortest Gossip codes.

6.2 Erasures in Gossip codes

6.2.1 Erasure model

In our discussions, *erasure* means *a non-alphabet symbol* chosen by the pirates in a detected position of the embedded fingerprint. We also assume that marking assumption [BS98] holds in our model. The pirates can *find* only those alphabet symbols that match with any one of their copies at each detected position. In practice the embedding mechanism, which encodes codewords into digital objects, will determine the type of alphabet symbols or erasures that are possible at each detected position for creating an illegal copy.

Definition 6.1 Fingerprinting System with Optional Erasures (FSOPE) is a system where the pirate controlling a group of traitors may optionally choose to create an alphabet symbol that he can find in one of their copies or an erasure (non-alphabet symbol) at each detected mark.

The model assumes that pirates can create one of the alphabet symbols matching any one of their copies, or an erasure (denoted by '*e*' or '?') in the place of a detected mark. When the pirates choose any symbol that matches with a symbol in the alphabet, it will not be possible to differentiate between the symbols chosen by the pirates and the valid symbols embedded under the selected fingerprinting scheme.

This may lead to errors in tracing. For this reason the embedding algorithm should be so chosen that it hides the actual alphabet from being detected by the pirates (through an encoding mechanism).

FSOPE can be sub-divided into the following cases:

1. No erasures
2. Selective erasures
3. Only erasures

We now discuss erasures in non-binary fingerprinting systems considering Gossip codes.

6.2.2 Gossip Codes with no erasures

In this case, the pirates are not allowed to create erasures, however, at each position they are free to choose the alphabet symbols they find in their copies. The tracing algorithm identifies a culprit from the extracted codeword. The tracing mechanism is obvious when the coalition size is equal to one. This is because the pirate's codeword has to match with one of the valid codewords of the Gossip code.

To illustrate tracing in Gossip codes with no erasures, we consider the code in Example 5.1, *i.e.*, 2-Gossip (7, 7, 4) code.

1	1	1	0	0	0	0
2	0	0	1	1	0	0
3	0	0	0	0	1	1
0	2	0	2	0	0	2
0	3	0	0	2	2	0
0	0	2	3	0	3	0
0	0	3	0	3	0	3

Table 6.1. 2-Gossip (7, 7, 4) code

Let the pirate set be $W = \{w^1, w^2\}$ where w^i is the i^{th} codeword in Table 6.1. Let the pirate word found in the illegal copy be $x = \{2,0,0,0,0,0,0\}$. Clearly $x \in D(w^1, w^2)$, but x also belongs to $D(w^2, w^3)$ where $D(w^i, w^j)$ is the descendent set of $\{w^j, w^j\}$.

Tracing: Consider the non-zero value in the pirate codeword x of the above example. Its position is first. We observe that the second codeword contains 2 in the first position. We accuse the 2nd user as culprit because the position of non-zero alphabet symbol and the value in the pirate codeword match with that of the second user's codeword. It may also be noted that the 2nd user is the member of both the coalitions that are capable of creating the pirate codeword.

The following lemma presents the tracing algorithm for Gossip codes.

Lemma 6.1

Every pirate word created from a Gossip code, containing at least one non-zero alphabet symbol can reveal one member of the pirate group.

Proof. The construction of the Gossip code makes it clear that every non-zero symbol in the pirate codeword, say x, can reveal one member of the collusion. Let the i^{th} position in the pirate codeword be a valid alphabet symbol say '1', then there exists exactly one codeword say j^{th} codeword, which contains '1' in the i^{th} position. It is clear that without the involvement of j^{th} user, it is impossible to create the pirate codeword x and so the algorithm accuses j^{th} user.

The following lemma shows that there exists a (trivial) Gossip code that can accuse all the active traitors from the pirate fingerprint when no erasures are present.

Lemma 6.2

A (q-1)-Gossip(q, q, q) code has a deterministic tracing algorithm and the tracing algorithm accuses the active traitors.

Proof. The general construction of Gossip codes that can trace all active colluders (in *no erasures* case) who contributed to piracy is as follows:

Choose the number of codewords equal to the number of alphabet symbols i.e. $M = q$.

Let the collusion size c be equal to $q-1$ and the number of gossip columns be $l = q$.

Construct q gossip columns, where each column contains the alphabet symbols $\{0,1,...,q-1\}$ only once. It is known from [TL01] that when $c = q-1$, there exists a

Gossip code such that each column accuses one of the collusion sets. Since every symbol in the alphabet appears only once in each column, the alphabet symbol 0 also contributes to tracing and each gossip column can trace ${}^{q}C_{c}$ pirate groups. This leads to tracing all the active users who contributed in creating the pirate codeword by using their legitimate copies.

It may be noted that (q-1) -Gossip(q, q, q) code is not shortest Gossip code. If $M = q$ then $c - Gossip(1, M, M)$ code is shortest code since all symbols are distinct. But for some concatenated codes we consider Gossip codes with $l = q = M$.

The following example (Example 6.1) presents a 4-Gossip(5,5,5) code constructed as per Lemma 6.2.

Example 6.1

0	1	1	1	1
1	2	2	2	0
2	3	3	0	2
3	0	4	3	3
4	4	0	4	4

Table 6.2. 4-Gossip (5, 5, 5) code

6.2.3 Gossip codes with selective erasures

In this case, pirate users are free to create erasures or the alphabet symbols they *found* in their copies, at each detected position. Further, the undetected positions of the collusion remain the same in the illegal copy, when such a copy is created by the colluded set. Lemma 6.1 holds good for all Gossip codes even in case of erasures. Thus any pirate word containing one non-zero symbol can reveal one traitor even in presence of erasures. In a c-$Gossip(l, M, q)$ code if the pirates create *only erasures* at all detected positions, then the tracing is same as *only erasures* case. If the pirates choose to create zeros at all detected positions, whenever there is a zero in any of the codewords, the tracing is possible against all collusion sets of sizes less than c, (provided there are undetectable positions for the collusion groups).

The following lemma explains the possibility of tracing in (q-1)-Gossip(q, q, q) code in the presence of erasures. Lemma 6.2 explains the tracing in no erasures case.

Lemma 6.3

The (q-1)-Gossip(q,q,q) code can trace all the pirate codewords except $(e,e,e,e,e...l$ times$)$ to reveal at least one culprit.

Proof. This follows from *Lemma 6.1*.

In Example 6.1, it can be seen that the pirate codeword (e,e,e,e,e) contains all erasures and so the tracing will not yield any pirate. It will be the same as choosing only erasures in detected positions, since all the positions are detected. It may be recalled that $(q-1)-Gossip(q,q,q)$ code is not shortest Gossip code. For shortest Gossip code there exists some undetectable positions for each collusion set. Thus a collusion group (with size $< c$) cannot create a pirate word with all erasures in the case of shortest Gossip codes.

6.2.4 Gossip codes with only erasures

An embedded position is detected when its value is found different in two copies during comparison. In *only erasures* case, the pirates are allowed to create only erasures in the detected position and hence can create ony one pirate codeword. If the number of zeros in each gossip column is equal to the number of pirates c, i.e. $c = M-(q-1)$ this will guarantee a unique coordinate (in the pirate word) for each pirate group, which has only zeros. Thus the pirate words created by different pirate groups are all distinct. Consider a c-*Gossip*(l, M, q) code with $c = q-1$ and $M = 2.(q-1)$. Here is an example.

Example 6.2

1	1	0	0	1	0
2	0	1	1	0	0
0	2	2	0	0	1
0	0	0	2	2	2

Table 6.3. 2-Gossip(6, 4, 3) code

In Table 6.3, the number of pirate sets of size 2 are equal to ${}^4C_2$ =6. The exhaustive list of the pirate codewords (equal to number of pirate groups here) created by all the collusion sets are as follows, where *e* denotes an erasure at detected position and *0* means the position is undetected.

S.No	Collusion sets	Pirate fingerprints
1	{1, 2}	(*e*, *e*, *e*, *e*, *e*, *0*)
2	{1, 3}	(*e*, *e*, *e*, *0*, *e*, *e*)
3	{1, 4}	(*e*, *e*, *0*, *e*, *e*, *e*)
4	{2, 3}	(*e*, *e*, *e*, *e*, *0*, *e*)
5	{2, 4}	(*e*, *0*, *e*, *e*, *e*, *e*)
6	{3, 4}	(*0*, *e*, *e*, *e*, *e*, *e*)

Table 6.4. Collusion sets vs pirate fingerprints

Tracing: We observe that each collusion set creates a unique fingerprint in o*nly erasures* case. Therefore the number of collusion sets is equal to number of pirate fingerprints. Moreover, all the pirate codewords are distinct. Since there is one-to-one matching between a particular collusion set and the pirate fingerprint, it is possible to find the collusion group as a whole. This tracing method has $O\left({}^M C_c\right)$ complexity where c is collusion size. An alternative method is to find all the zeros in the pirate word, and accuse everybody who has zeros in *all* these positions. This works even if the actual coalition size is less than c, and runs in time $O(M \cdot l)$ where M is the number of users and l is the length of the code. This is significantly more efficient than $O\left({}^M C_c\right)$, which is the running time of the earlier algorithm.

General case for only Erasures: Let (X, B) denote the equivalent set system of a t-design that corresponds to a *c-Gossip*(l, M, q) code. Looking at the t-design's blocks we can tell which group's pirate words will have 0 at a given position, and which groups will have erasures. For example the block B_i gives the list of groups, which will not be able to detect the i^{th} position. If the collusion set $W \subseteq X - B_i$ (implies $W \cap B_i = \phi$) and $|W| \leq t$ then the collusion group W cannot detect the i^{th} position. W will create an erasure (since the mark is detected) in j^{th} position if $W \cap B_j \neq \phi$. Let x be the pirate word created by a collusion group W and x_i be the i^{th} position in the pirate word. The pirate word created by W can be completely specified by the blocks of the t-design.

x_j is zero (undetected position) if $W \cap B_j = \phi$ and

x_j is e (detected) if $W \cap B_j \neq \phi$

Theorem 6.1 *[HP85, CD96]*

Let Ω be a $t-(v,k,\lambda)$ design. For any integer s satisfying $t > s \geq 0$, there are exactly λ_s blocks of Ω which are incident with any given s-set of points of Ω, where

$$\lambda_s = \frac{\lambda(v-s)(v-s-1)..(v-(t-1))}{(k-s)(k-s-1)...(k-(t-1))}$$

From this formula, it can be seen that the number of nonzero symbols in each codeword for a *c-Gossip*(l, M, q) code is equal to λ_1. (λ_1 is computed from the above formula with $\lambda = 1, v = M, t = c$ and $s = 1$.) Thus each codeword contain $l - \lambda_1$ zeros.

Lemma 6.4

If *c-Gossip*(l, M, q) code is a Gossip code with minimum code length, then the number of undetectable positions (number of zeros in only erasures case) for a collusion W of size $w \leq c$ in the pirate word is $\bar{\lambda}_w$ which is also equal to $l - \left(\sum_{i=1}^{w} (-1)^{i-1} \binom{w}{i} \lambda_i \right)$ where λ_i is defined as above, with $\lambda = 1, v = M$ and $t = c$.

Proof. The proof follows from the fact that there exists a $t-(v,k,\lambda)$ design with $\lambda=1, v=M$ and $t=c$ that corresponds to c-*Gossip*(l, M, q) code (Theorem 5.1). Let W be the collusion set of size $w \le c$. The number of blocks in the $t-(v,k,\lambda)$ design disjoint from W will give the number of undetectable positions for the collusion.

For example consider 2-Gossip(7, 7, 4) code presented in Example 5.1. The t –design corresponding to this Gossip code is 2-(7, 3, 1) design, which is given by

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

$B_1 = \{1,\ 2,\ 3\}, B_2 = \{1,\ 4,\ 5\}, B_3 = \{1,\ 6,\ 7\}, B_4 = \{2,\ 4,\ 6\},$
$B_5 = \{2,\ 5,\ 7\}, B_6 = \{3,\ 5,\ 6\}, B_7 = \{3,\ 4,\ 7\}$

Let W be a collusion set, and let $W = \{1,2\}$.

Number of blocks that do not contain any member of W =

Total number of blocks – Number of blocks that contain 1 –Number of blocks that contain 2 + Number of blocks that contain {1, 2}

$= l - 2.\lambda_1 + 1 = 7\text{-}2 \times 3 + 1 = 2$

This implies in only erasures case for 2-Gossip (7, 7, 4) code each pirate codeword contain two zeros. In a Gossip code that achieves the bound, if $c < M-(q-1)$ then the number of zeros in a column is more than c. This ensures that there is at least one undetected position for the pirate codewords. It is known that $2-(p^2+p+1, p+1, 1)$ design exists when p is prime power. When a Gossip code is constructed from $2-(p^2+p+1, p+1, 1)$ design, the number of zeros in each Gossip column $= M-(q-1) = p^2+p+1 - (p+1) = p^2 > c$. Also the number of c-groups for which, a given position in the pirate code word is undetectable is ${}^{p^2}C_c$. Thus for 2-*Gossip*(7, 7, 4) code number of zeros in each column is 4, and ${}^4C_2 = 6$ groups contain 0 in a given position of the pirate word. Let us consider the Gossip code as given in example 5.1, where $M=7$, $q=4$ and $c=2$. Here, the number of pirate sets of size 2 are equal to ${}^7C_2 = 21$. The exhaustive list of the pirate codewords (equal to number of pirate groups here) created by all the collusion sets are as follows.

S.No	Collusion sets	Pirate fingerprints
1	{1, 2}	(*e*, *e*, *e*, *e*, *e*, *0*, *0*)
2	{1, 3}	(*e*, *e*, *e*, *0*, *0*, *e*, *e*)
3	{1, 4}	(*e*, *e*, *e*, *e*, *0*, *0*, *e*)
4	{1, 5}	(*e*, *e*, *e*, *0*, *e*, *e*, *0*)
5	{1, 6}	(*e*, *e*, *e*, *e*, *0*, *e*, *0*)
6	{1, 7}	(*e*, *e*, *e*, *0*, *e*, *0*, *e*)
7	{2, 3}	(*e*, *0*, *0*, *e*, *e*, *e*, *e*)
8	{2, 4}	(*e*, *e*, *0*, *e*, *e*, *0*, *e*)
9	{2, 5}	(*e*, *e*, *0*, *e*, *e*, *e*, *0*)
10	{2, 6}	(*e*, *0*, *e*, *e*, *e*, *e*, *0*)
11	{2, 7}	(*e*, *0*, *e*, *e*, *e*, *0*, *e*)
12	{3, 4}	(*e*, *e*, *0*, *e*, *0*, *e*, *e*)
13	{3, 5}	(*e*, *e*, *0*, *0*, *e*, *e*, *e*)
14	{3, 6}	(*e*, *0*, *e*, *e*, *0*, *e*, *e*)
15	{3, 7}	(*e*, *0*, *e*, *0*, *e*, *e*, *e*)
16	{4, 5}	(*0*, *e*, *0*, *e*, *e*, *e*, *e*)
17	{4, 6}	(*0*, *e*, *e*, *e*, *0*, *e*, *e*)
18	{4, 7}	(*0*, *e*, *e*, *e*, *e*, *0*, *e*)
19	{5, 6}	(*0*, *e*, *e*, *e*, *e*, *e*, *0*)
20	{5, 7}	(*0*, *e*, *e*, *0*, *e*, *e*, *e*)
21	{6, 7}	(*0*, *0*, *e*, *e*, *e*, *e*, *e*)

Table 6.5. Collusion sets vs. pirate fingerprints

In Theorem 6.2 we present the Gossip codes that are suitable for handing '*only erasures*' strategy of pirates.

Theorem 6.2

If *c-Gossip*(l, M, q) code is a Gossip code with minimum length such that $c < M - (q-1)$, then no two collusion groups can create the same pirate word in fingerprinting with only erasures.

Proof. In *only erasures* case (of any Gossip code) it is always true that each pirate group can create only one pirate word. If $c < M - (q-1)$ in a Gossip code with minimum possible length, the pattern of undetectable zeros is different for each group. Thus the pirate words are unique. Further, if two pirate words can create the

same pirate word then each Gossip column cannot accuse ${}^{q-1}C_c$ distinct groups, which is a contradiction.

Examples for these codes are Gossip codes constructed from $2\text{-}(p^2+p+1,\ p+1,\ 1)$ designs and $3\text{-}(p^2+1,\ p+1,\ 1)$ designs where p is prime power and greater than 2. Specific examples are $3\text{-}Gossip(30,\ 10,\ 5)$, and $2\text{-Gossip}(21,\ 21,\ 6)$ codes apart from $2\text{-Gossip}(7,\ 7,\ 4)$ code.

6.3 Erasures in concatenated codes

In this section we explain how to handle erasures in concatenated codes. Erasures inside concatenated codes are also divided in to three cases similar to that of erasures in Gossip codes.

6.3.1 Concatenated codes with no erasures

In the following example (Example 6.3) we present a concatenated code for no erasures case along with the analysis.

Example 6.3

Let us consider a Gossip code with $M = q = 4$, and $c = 3$. It would result in the following code presented in Table 6.6, where $\overline{0}$ denotes the first inner codeword, $\overline{1}$ denotes the second, and so on.

1	1	1	0		$\overline{0}$
2	2	0	1	≡	$\overline{1}$
3	0	2	2		$\overline{2}$
0	3	3	3		$\overline{3}$

Table 6.6. 3-Gossip(4, 4, 4) code

This code can trace all its pirates if the collusion size is less than or equal to three [refer *Lemma 6.1*]. If we use the Gossip code in *Example 5.1* as outer code for the Concatenated code, the Concatenated code will be:

$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$		w^1
$\overline{2}$	$\overline{0}$	$\overline{0}$	$\overline{1}$	$\overline{1}$	$\overline{0}$	$\overline{0}$		w^2
$\overline{3}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{1}$	$\overline{1}$		⋮
$\overline{0}$	$\overline{2}$	$\overline{0}$	$\overline{2}$	$\overline{0}$	$\overline{0}$	$\overline{2}$	≡	⋮
$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{0}$	$\overline{2}$	$\overline{2}$	$\overline{0}$		⋮
$\overline{0}$	$\overline{0}$	$\overline{2}$	$\overline{3}$	$\overline{0}$	$\overline{3}$	$\overline{0}$		⋮
$\overline{0}$	$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{3}$		w^7

Table 6.7. 2-(7,7,4) Concatenated Gossip code

The collusion size c is two for the concatenated code. Now, we discuss the optional erasures case for the Concatenated code defined above. In this case, the pirates are allowed to create an alphabet they *see* in the detected positions.

Tracing: Consider the collusion set W, consisting of w^1 and w^2. The descendent set is $D(W) = \{(a_1,...,a_5,\overline{0},\overline{0}) / a_1 \in \{\overline{1},\ \overline{2}\}, a_i \in \{\overline{1},\ \overline{0}\} \text{ for } 2 \le i \le 4\}$ since the last two positions are undetected. The choices in the other positions are based on the choices available in the inner code.

Let a *pirate* word created by the collusion set $W = \{w^1, w^2\}$ be $x = \{2\ 2\ 2\ 2\ \ 1\ 2\ 0\ 0\ \ 1\ 1\ 1\ 1\ \ 1\ 1\ 0\ 1\ \ 1\ 2\ 1\ 1\ \ 1\ 1\ 1\ 0\ \ 1\ 1\ 1\ 0\}$. The inner codewords that are exposed by applying tracing algorithm on inner pirate word $\{2\ 2\ 2\ 2\}$ are $\{\overline{1},\ \overline{2}\}$. Similarly, all other inner pirate codewords, reveal $\{\overline{0}, \overline{1}\}$ except the last two pirate codewords, which reveal $\overline{0}$. With this information, we can completely reconstruct the descendent set of pirates as $\{(a_1,...,a_5,\overline{0},\overline{0}) / a_1 \in \{\overline{1},\ \overline{2}\}, a_i \in \{\overline{1},\ \overline{0}\} \text{ for } 2 \le i \le 4\}$. This traces the pirate set to be $\{w^1, w^2\}$. Further, as the outer code is also an IPP code, the tracing algorithm does not accuse any innocents. More such contractions are possible using any of the Gossip codes given in the Appendix as outer code, and choosing a suitable and deterministic Gossip code as inner code, as per *Lemma 6.1*. The condition that must be taken care is that the number of inner code words is to be equal to the number of alphabet symbols of the outer code.

6.3.2 Concatenated codes with only erasures

In the following example (Example 6.4) we present a concatenated code for only erasures along with the analysis.

Example 6.4

The construction of the Concatenated code for only erasures scenario is as follows. Let us consider the 2-Gossip(7, 4, 7) code presented in Table 6.1 as inner code.

1	1	1	0	0	0	0		$\overline{0}$
2	0	0	1	1	0	0		$\overline{1}$
3	0	0	0	0	1	1		:
0	2	0	2	0	0	2	≡	:
0	3	0	0	2	2	0		:
0	0	2	3	0	3	0		:
0	0	3	0	3	0	3		$\overline{6}$

Table 6.8. 2-Gossip(7, 7, 4) inner code

The symbols $\{\overline{0},\overline{1},...,\overline{6}\}$ are codewords in the inner code but they are alphabets in the outer code. The constructed Concatenated code, where w^i denotes the i^{th} codeword, will be as follows. The collusion size is 2 for the concatenated code.

$\overline{0}$	$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{1}$		w^1
$\overline{1}$	$\overline{0}$	$\overline{2}$	$\overline{2}$	$\overline{2}$	$\overline{2}$	$\overline{2}$		w^2
$\overline{2}$	$\overline{2}$	$\overline{0}$	$\overline{3}$	$\overline{3}$	$\overline{3}$	$\overline{3}$		:
$\overline{3}$	$\overline{3}$	$\overline{3}$	$\overline{0}$	$\overline{4}$	$\overline{4}$	$\overline{4}$	≡	:
$\overline{4}$	$\overline{4}$	$\overline{4}$	$\overline{4}$	$\overline{0}$	$\overline{5}$	$\overline{5}$		:
$\overline{5}$	$\overline{5}$	$\overline{5}$	$\overline{5}$	$\overline{5}$	$\overline{0}$	$\overline{6}$		:
$\overline{6}$	$\overline{6}$	$\overline{6}$	$\overline{6}$	$\overline{6}$	$\overline{6}$	$\overline{0}$		w^7

Table 6.9. 2-(7, 7, 7)Concatenated Gossip code

Tracing: Let the pirate set be $W=\{w^1,w^2\}$. The descendent set will be $D(W)=\{(\overline{e}\ \overline{e}\ \overline{e}\ \overline{e}\ \overline{e}\ \overline{e}\ \overline{e})\}$ where $\overline{e}$ contains all erasures in the detected positions of

inner Gossip codewords. Each pirate set can create one unique pirate fingerprint, assuming only erasures are possible in detected positions. Let a *pirate* word created by the collusion set $W = \{w^1, w^2\}$ be $x = \{e\ e\ e\ e\ e\ 0\ 0;\ e\ e\ e\ e\ e\ 0\ 0;\ e\ 0\ 0\ e\ e\ e\ e;\ldots\}$. The first erasure word in the pirate word x created by W is $\{e, e, e, e, e, 0, 0\}$ and the third erasure word is $\{e, 0, 0, e, e, e, e\}$. In tracing, the first erasure in the pirate word of Concatenated code will trace pirates as $\{\overline{0}, \overline{1}\}$ and the third erasure will reveal that the inner pirate words are of $\{\overline{1}, \overline{2}\}$. This will lead to tracing the complete pirate set $W = \{w^1, w^2\}$.

6.3.3 Concatenated Gossip codes with selective erasures

In this construction, the outer code is also a Gossip code (that has a tracing algorithm) and the inner code has a deterministic decoding algorithm, so the Concatenated code also traces back the members of the pirate group. The pirates have to create erasures in inner codes in order to create erasures in Concatenated codes. Thus, dealing with erasures in inner Gossip codes would automatically mean dealing with erasures in Concatenated codes.

Example 6.5

Consider a Gossip code with $M = q = 4$, and $c = 3$. It would result in the following code, where $\overline{0}$ denote the first inner codeword, $\overline{1}$ denote the second, and so on.

Consider 3-Gossip(4, 4, 4) code

1	1	1	0
2	2	0	1
3	0	2	2
0	3	3	3

$\equiv$

$\overline{0}$
$\overline{1}$
$\overline{2}$
$\overline{3}$

Table 6.10. 3-Gossip(4, 4, 4) code

This code can trace all its pirates if the collusion size is less than or equal to three (refer Lemma 6.2*)*. If we use the Gossip code in E*xample 5.1* as outer code for the Concatenated code, the Concatenated code will be:

$\overline{1}$	$\overline{1}$	$\overline{1}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$		w^1
$\overline{2}$	$\overline{0}$	$\overline{0}$	$\overline{1}$	$\overline{1}$	$\overline{0}$	$\overline{0}$		w^2
$\overline{3}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{1}$	$\overline{1}$		:
$\overline{0}$	$\overline{2}$	$\overline{0}$	$\overline{2}$	$\overline{0}$	$\overline{0}$	$\overline{2}$	$\equiv$	:
$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{0}$	$\overline{2}$	$\overline{2}$	$\overline{0}$		:
$\overline{0}$	$\overline{0}$	$\overline{2}$	$\overline{3}$	$\overline{0}$	$\overline{3}$	$\overline{0}$		:
$\overline{0}$	$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{3}$	$\overline{0}$	$\overline{3}$		w^7

Table 6.11. 2-(7, 7, 4) Concatenated Gossip code

The collusion size c is two for the concatenated code. Now, we discuss the optional erasures case for the concatenated code defined above. In this case, the pirates are allowed to create an alphabet symbol they *find* in the detected positions.

Tracing: Consider the collusion set W, consisting of w^1 and w^2. The descendent set is given by $D(W)=\{(a_1,...,a_5,\overline{0},\overline{0})/\, a_1\in\{\overline{1},\ \overline{2}\}, a_i\in\{\overline{1},\ \overline{0}\} \text{ for } 2\le i\le 4\}$ since the last two positions are undetected. The choices in the other positions are based on the choices available in the inner code.

Let the pirate word created by the collusion set W be $x=\{2\ 2\ 2\ 2\ \ 2\ 2\ 0\ 1\ \ 1\ 1\ 1\ 1\ \ 1\ 1\ 0\ 1\ \ 1\ 2\ 1\ 1\ \ 1\ 1\ 1\ 0\ \ 1\ 1\ 1\ 0\}$. This can be re-written as $\{\overline{e},\overline{1},\overline{e},\overline{0},\overline{e},\overline{0},\overline{0}\}$. This is same as simple Gossip code with selective erasures. This will reveal a pirate namely w^1. Suppose the pirateword is $x=\{2\ 2\ 2\ 2\ \ 1\ 2\ 0\ 0\ \ 1\ 1\ 1\ 0\ \ 1\ 1\ 0\ 1\ \ 1\ 2\ 1\ 1\ \ 1\ 1\ 1\ 0\ \ 1\ 1\ 1\ 0\}$. This can be rewritten as $\{\overline{e},\overline{e},\overline{e},\overline{0},\overline{e},\overline{0},\overline{0}\}$. This does not contain any non-zero symbol of the outer code. But here the inner codewords that are exposed by applying tracing algorithm on inner pirate word $\{2\ 2\ 2\ 2\}$ are $\{\overline{1},\ \overline{2}\}$. Similarly, all other inner pirate words, reveal $\{\overline{0},\overline{1}\}$ except the last two pirate words, which reveal $\overline{0}$. With this information, we can

completely reconstruct the descendent set of pirates as $\{\overline{1}$ *or* $\overline{2},\ \overline{1}$ *or* $\overline{0},\ \overline{0},\ \overline{1}$ *or* $\overline{0},\ \overline{1}$ *or* $\overline{0},\ \overline{0},\ \overline{0}\}$. This traces the pirate set to be $\{w^1, w^2\}$.

Lemma 6.5 presents the condition for the construction of concatenated IPP code.

Lemma 6.5

From an inner *c-Gossip*(l, M, q) code and an outer *c-Gossip*(l', M', q') code a concatenated IPP code can be constructed provided $M = q'$.

To construct a generic concatenated code the condition that must be taken care is that the number of inner codewords is equal to the number of alphabet symbols of the outer code Further, this concatenated code is an IPP code, since the inner and outer codes are IPP codes. Such a concatenated code can be constructed by considering Table 6.3 as inner code and Table 6.1 as outer code. The *no erasures* and o*nly erasures* cases in these concatenated codes are straightforward and omitted here as the tracing in the inner codewords is similar to Gossip codes for *no erasures* and o*nly erasures*.

6.4 Average, majority and minority attacks on non-binary codes

In averaging attack, the pirate object is obtained by finding the average values of object elements, for example pixels in images. This attack reduces the strength of the embedded marks and in watermark recovery phase introduces errors in the recovered fingerprints. If this attack is used on all blocks and if there are enough colluders [GP99], then the majority of the marks will be erased and tracing will fail. In [EKK97] a bound on the size of collusion to make the mark undecidable is derived. Using c-traceability codes for fingerprinting sequences will still have the same upper bound on collusion security. However if large (sufficient) portion of the fingerprinting sequence is recovered, the pirate object can be traced to one of the colluders.

Non-binary alphabet symbols may be represented in binary form for embedding. Alternately, the non-binary alphabet symbols may be encoded during embedding. So it may be possible for pirates to directly compare non-binary alphabet symbols embedded inside different fingerprinted objects.

When a non-binary alphabet symbol is embedded in binary form, the majority and minority choice attacks work on binary bits (that represent a non-binary alphabet symbol), which in turn may affect non-binary alphabet symbols that are part of an IPP/TA codeword. Pirates may compare the binary bits of two non-binary alphabet symbols to create attack on non-binary fingerprint code. The detection of at least one bit in a non-binary alphabet symbol may be considered as detection of the alphabet symbol and we may proceed as per marking assumption in those detected positions. Sometimes the non-binary error correcting codes are not used directly in fingerprinting but rather used as outer codes for constructing IPP/TA codes. The inner codes may be binary (frameproof) code. Then the actual fingerprinting code becomes a concatenated code.

In this case, the concatenated code may be seen as a non-binary code whose alphabet symbols are represented (encoded as) as binary vectors of a frameproof code. It is interesting to see how the collusion attacks vary when the embedding algorithm *is LSB, FFT, and wavelet transform.* After embedding a watermark in fourier domain, the object undergoes inverse fourier transform. Thus the watermark embedded in Fourier domain after undergoing Inverse transform will not behave similar to the one that is embedded in spatial domain. If the embedding technique is in transform domain (like FFT) then the embedded information may go in fourier (transform) coefficients. It is interesting to know whether the collusion attacks exist on fingerprinted objects whose embedding technique is based on Wavelet/ FFT/DCT transforms. For that we analyse collusion attacks on watermarks in transform domain.

6.4.1 Detection of fingerprint marks in fourier space

It is possible to detect the differences between fingerprinted copies in Transform domain. If anyone takes difference of FFT of first user (U_1)'s copy and FFT of second user (U_2)'s copy, the difference will have some relation to detected WMs in Fourier space. As an example, we use the simple equation $v_i' = v_i + \left(\frac{1-\alpha}{1+\alpha}\right)x_i$ where v_i' is the modified DCT coefficient of the original value v_i, x_i is an element of the fingerprint sequence and α is a scaling factor. We can observe that (FFT(U_1's copy)-FFT(U_2's copy)) is non-zero in case of detected marks and 0 in case of undetected marks. It may be observed that if the underlying watermarking technique is not secure the fingerprinting scheme cannot be secure for any fingerprinting scheme.

To simulate the attacks (majority/ minority choice) we need to consider an embedding technique to start with. These attacks can in principle be extended to different embedding algorithms like fourier transform and other wavelet-based algorithms.

6.5 Other erasure models

In the absence of the marking assumption, both the distributor and the users are allowed to add relatively small distortion to the original document. The main application of this scenario is fingerprinting of digital media such as image, video, audio, and speech content. In such applications, slight differences from the original version will not effect the quality of users' copies. It is assumed that both the signal and the fingerprint are real random variables that obey some probability distribution. Establishing resilience of the system against collusion attacks relies on probabilistic or information-theoretic analysis.

6.5.1 Weak marking assumption

Fingerprinting schemes in general, assume that the pirates do not alter the digital document on positions where all of the copies agree. This is called the *marking condition*. Guth and Pfitzmann [GP99] introduced a relaxation of marking condition. They assume the following relaxed version of the marking condition: At any position where the codewords of all the pirates agree, the pirates have a definite probability δ of being able to output a different digit. This happens independently for all the positions of agreement and if they can output a different digit they are not restricted about the digit they output. This models the situation where the users cannot detect the positions where the fingerprint is embedded in a digital document, but they are allowed to modify a δ fraction of the entire document, thus also modifying some digits of the fingerprinting code where such modification is against the marking condition. If the fingerprint is embedded in digital images, audio or video files this relaxation seems to be natural. Thus to handle erasures created by collusion groups, Guth and Pfitzmann [GP99] constructed binary c-secure codes with ε error with a weaker assumption, which assumes that adversary can create only a certain percentage of erasures in place of embedded marks in pirate fingerprints and these erasures are (approximately) randomly distributed.

Guth and Pfitzmann's marking assumption [GP99]: A collusion $C = \{u_1,...,u_b\} \subseteq \Gamma$, $b \le c$, is only capable of creating an object whose fingerprint lies in the following set

$$P(C) = \{x = (x_1,...,x_L) : x_j \in \{w_j^{(i)} : 1 \le i \le b\} \cup \{?\}, 1 \le j \le L\}.$$

This is q-ary version of Guth and Pfitzmann's marking assumption and allows erasure to occur both in the detected and undetected positions.

Definition 6.2 *[SW02a]* Let $\Gamma = q^L$ be a code and $C = \{u_1,..u_b\} \subseteq \Gamma, b \le c$ be a collusion set. Γ is called c-Traceability code tolerating $\hat{e}$ erasures , and denoted by $c - TA(L,n;\hat{e})_q$, if the following condition holds: for any $(x_1,...,x_L) \in P(C;\hat{e})$, where $P(C;\hat{e})$ is the subset of $P(C)$ containing at most $\hat{e}$ erasures, there is a $u_i \in C$ such that $|\{j : x_j = w_j^i\}| > |\{j : x_j = w_j\}|$ for any $(w_1,...,w_L) \in \Gamma \setminus C$.

The following theorem (Theorem 6.3) presents the conditions for an error correcting code to become a Traceability code in presence of erasures.

Theorem 6.3 *[SW02a]* Let be an $(N,n,d)_q$ error correcting code, and c, $\hat{e}$ be integers. If d satisfies $d > \left(1-\frac{1}{c^2}\right)L + \frac{\hat{e}}{c^2}$ then Γ is $c-TA(L,n;\hat{e})_q$ code.

6.5.2 Cut and paste model

In a q'ary fingerprinting system for digital objects (such as images, videos and audio clips) a fingerprint is embedded as a q'ary sequence. The object is divided into blocks and each symbol of the fingerprint is embedded into one block. Colluders are able to construct a pirate object by assembling parts from their copies. They can also erase some of the marks or cut out part of the object resulting in a shortened fingerprint with some unreadable marks. In [SW02a] the codeword is repeated adequate number of times so that at least one copy of the embedded codeword can be recovered and thereby increasing erasure tolerance.

6.5.3 Shortened fingerprint model

The pirate object is constructed by (*i*) cut and paste of parts of different copies, (*ii*) using averaging attack to weaken the marks and defeat the mark detection algorithm, and (*iii*) cropping the object and removing some parts of the fingerprint. The pirate fingerprint recovered from a pirate object will be a sequence, possibly of shorter length compared to the original fingerprints, having some erased marks and in each of the non-erased positions, a mark from one of the colluders. The result of cut and paste attack on the object is that each component of the pirate fingerprint is from the fingerprint of one of the colluders or is an *erased mark*. Cropping attacks will remove parts of the fingerprint and will result in a shorter fingerprint. Collusion secure fingerprinting codes and traceability codes protect against attacks of type (*i*) and (*ii*), as long as the number of erased (unrecognisable) marks are not too many. However

they fail to trace the culprits if the pirate fingerprint is shortened even by one component.

Tracing shortened fingerprints was considered in [SW02a] and a tracing algorithm based on Levenshtein distance was proposed. The tracing algorithm, instead of hamming distance, used the length of the longest sub-string common between the pirate word and each codeword to measure similarity between the two, and chose the most similar codeword as a colluder. Deletion correcting codes were used to construct fingerprinting codes instead of error-correcting codes. The drawback of this method is that (*i*) tracing algorithm is computationally expensive and (*ii*) construction of deletion correcting codes that satisfy the required conditions is an open problem.

Safavi-Naini and Wang [SW02b] considered the problem of tracing shortened fingerprints and showed that using certain generalized Reed-Solomon (GRS) codes it is possible to trace shortened fingerprints correctly. They showed that for GRS codes deletion decoding can be formulated as a polynomial interpolation problem and the list decoding algorithm of Guruswami and Sudan [GS99] can be used to find the closest, *i.e.*, most similar code vector to the given shortened pirate word. This removes shortcomings of the earlier method [SW02a] by giving a construction for codes that can protect against deletion of fingerprint components, and also giving an efficient algorithm for tracing. In [SW02a] the authors use the list-decoding algorithm to find a bound on the number of deletions that can be tolerated, if the fingerprinting code is from the special class of GRS codes. Their work considers fingerprints in which strength of the marks is weakened and so the mark detection algorithm cannot easily produce a single output for each mark. Attacks such as averaging results in uncertainty in detection of the marks and so the recovered fingerprints will be likely to have errors which would result in incorrect tracing. They proposed a *combined mark detection and tracing algorithm* and showed that it can be used to construct a more powerful tracing algorithm.

6.5.4 Unreadable digit model *[GT03]* This model restricts the pirates to use at each position of the document a digit that one of them has at that position or a special unreadable digit. The marking condition still applies: if all of their documents agree at a position they cannot put an unreadable digit there. This model is more restrictive to the pirates than the weak marking assumption proposed by Guth and Pfitzmann [GP99].

Gossip codes in different erasure models: Gossip codes have deterministic tracing capabilities under strong marking assumption. However the tracing becomes probabilistic when erasures are created and various models discussed above. It can be noted that the unreadable digit model is identical to the model we discussed in this chapter namely *FPOPE* model which was elaborated into three different sub cases namely no erasures, only erasures and selective erasures. For tracing gossip words with erasures created under Guth and Pfitzmann we still need to assume that pirates cannot change the digits undetectable for them. Further the pirate word needs to contain a non-binary alphabet symbol. The cut and paste model is acceptable to the extent that the fingerprint is not shortened during the attack. To handle attacks that shorten the fingerprint it is required to have redundancy in fingerprints embedded.

6.6 Summary

In this chapter, we investigated erasures in Gossip codes. We discussed three types of erasures namely no erasures, only erasures and selective erasures in embedded fingerprints. We also presented the tracing methods in all these cases. The analysis also extended to concatenated codes. Efficient concatenated codes can be constructed with inner codes that withstand these erasure models and the outer code being an IPP or a Gossip code. We also discussed various other erasure models and their applicability to Gossip codes.

Chapter 7

Conclusions and future scope

With the increasing availability of copying devices for digital data, the need to restrain illegal redistribution of digital objects is an important issue. Digital fingerprinting makes the copies of a digital document unique by embedding a unique identification number such that it is difficult to find or destroy the identification number. This deters users from redistributing digital copies with the fear of being caught.

There are several design choices available in digital fingerprinting, the first choice being between binary and non-binary codes. The other options available are choosing from amongst frameproof, IPP and TA codes. The applications of digital fingerprinting include copyrights protection, DRM, and digital evidence. This work presented various fingerprinting models and the choices available for pirates and the countermeasures. In this work, we also described various traitor-tracing methods and presented a tracing method for Boneh and Franklin's broadcast encryption scheme. The combinatorial properties of IPP codes and a generic tracing method are also presented. The distributor may choose one of the tracing methods to identify the culprits based on the pirate model.

There are several parameters that influence the performance of a fingerprinting scheme. Some of these parameters are collusion size, length of fingerprint, choice of codes, error correcting capability of underlying codes, pirate strategies and tracing algorithm. A minimum fingerprint length is required to identify the collusion groups responsible for creating an illegal fingerprint based on the pirate model. Every fingerprinting scheme will have a limit on the maximum collusion size it can withstand. It is also possible to estimate the error probability associated with identifying a single pirate in the collusion to trace explicitly one member of the coalition through a systematic method.

Gossip codes that achieve shortest code length can be constructed from a class t-designs and of c-Traceability Schemes. The fifth chapter presented the construction and analysis of Gossip codes. Two methods of construction for Gossip codes that achieve shortest code length are presented that include construction of Gossip codes from t-designs, and from c-Traceability Schemes, which are used in broadcast encryption applications. The converse part *i.e.*, construction of Traceability Schemes and t-designs from Gossip codes is also presented. The converse part *i.e.*, construction of Traceability Schemes and t-designs from Gossip codes is also feasible.

These results are also reliable to make out whether a *shortest* Gossip code exists or not, for the chosen code parameters namely M, q and c. Gossip codes as IPP codes can identify one parent of the pirate copy during tracing. They allow deterministic tracing of pirates and also come up with an efficient tracing algorithm where as earlier works on fingerprinting are usually probabilistic in some way or the other.

The construction of embedded Gossip code is also possible and it can be used for extending an existing Gossip code to a bigger code. Embedded Gossip codes in turn can be used to construct embedded Traceability Schemes and embedded frameproof codes. These (embedded) codes and Traceability schemes can expand a broadcast application such that it is compatible to the existing scheme.

When a frameproof code is used as inner code, the pirate members cannot create alphabet symbols not found in their copies in the detected positions of the concatenated code. Thus, during comparison, whenever a mark is detected in the concatenated code, the pirates need to create an alphabet symbol they find in their copies or a non-alphabet symbol in the detected positions. Thus this concatenated code presents a sensible way for implementing fingerprinting schemes under pirate model, assuming marking assumption [BS98] for non-binary codes. Thus concatenated codes are presented here for the realization of the pirate model and tracing.

Gossip codes suitable for selective erasures case are also suitable for no erasures and only erasures cases. In chapter 6, we discussed erasures in Gossip codes. We discussed three types of erasures namely no erasures, only erasures and selective erasures in embedded fingerprints. In *only erasures* case, the pirates are allowed to create only erasures in the detected positions and cannot choose any valid alphabet symbol. In selective erasures case, the pirates may choose one of the alternatives found in their copies or a non-alphabet symbol at each detected mark. We also presented the tracing methods in all these cases. The analysis is also extended to concatenated codes. We also discussed various other erasure models, which include random erasures in fingerprints and shortening the fingerprint. Efficient concatenated codes can be constructed with inner codes that withstand these erasure models and the outer code being an IPP or a Gossip code.

There are various advantages of shortest Gossip codes. Shortest Gossip codes are likely to cause less distortion during embedding due to shorter length as compared to normal Gossip codes since the modifications to be made to the original are less. Shortest Gossip codes are constant weight codes. These codes provide deterministic tracing for pirates and tolerate erasures. Construction of frameproof codes is also possible from shortest Gossip codes.

We envisage that the following may constitute the future scope of this work.

1. If a pirate copy is found, original users are caught in tracing but the distributor is not. One can investigate remarking techniques to improve tracing from a node-to-node basis. Such a technique would help in traffic analysis since it is possible to track the movement of pirate copy.

2. An ideal fingerprinting scheme should meet the goals of fingerprinting (collusion tolerance, object quality tolerance, robustness, efficient tracing etc) in design. It would be interesting to know how close a given fingerprinting scheme is to the ideal. It may also be interesting to identify

the issues related to practical implementations of non-binary fingerprinting for a large user community.

3. There are many approaches to analyse a fingerprinting scheme, which are based on collusion tolerance, error-correcting capacity, code length, trade offs (Game theory approach), resistance to various pirate strategies and Tracing capabilities. One can study whether it feasible to have a uniform evaluation procedure that analyses a fingerprinting scheme considering many of these parameters. Another area of further study can be in identifying the performance limits for a generic non-binary fingerprinting scheme under good erasure tolerance (weak marking condition) and deterministic tracing conditions.

4. An interesting approach to fingerprinting may be information theoretic approach, *i.e.*, finding out how much information should be contained in a pirate codeword about its parent codewords in order to accuse the collusion groups completely or partially or exactly a user inside a coalition. One may fix a pirate strategy (that includes known attacks) before going actually for this analysis. Sometimes, it is beneficial to test whether a particular coalition is involved in creating pirate copies.

5. There are some already known limitations for binary and non-binary codes. It would be interesting to identify the other possible limitations of fingerprinting in this direction. This will be useful in identifying the scope of fingerprinting with a particular choice of codes.

Bibliography

[BCEKZ01] A. Barg, G. Cohen, S. Encheva, G. Kabatiansky, and G. Zémor, "A Hypergraph Approach to the Identifying Parent Property: The Case of Multiple Parents", SIAM Journal on Discrete Mathematics, Vol. 14, Number 3, pp. 423-431, 2001.

[BPS01] Omer Berkman, Michal Parnas, and Jirí Sgall, "Efficient Dynamic Traitor Tracing", SIAM Journal on Computing, Vol 30, Number 6, pp. 1802-1828, 2001.

[AB97] Anton Betten et al, "DISCRETA - a tool for constructing t-designs", Lehrstuhl II fur Mathematik, Universitat Bayreuth, 1997 available at http://btm2xl.mat.uni-bayreuth.de/discreta/

[BS98] D. Boneh and J. Shaw, "Collusion-Secure Fingerprinting for Digital Data", IEEE Trans. on Inform. Theory, Vol. IT-44, pp. 1897-1905, 1998.

[BF99] D. Boneh and M. Franklin, "An Efficient Public Key Traitor Tracing Scheme", Proceedings of Crypto '99, Santa Barbara, California, 1999, pp. 338-353.

[BPS01] O. Berkman, M. Parnas and J. Sgall, "Efficient Dynamic Traitor Tracing", Proc. of the 11th ACM-SIAM Symp. on Discrete Algorithms (SODA), 2000, pp. 586-595,.

[CFN94] B. Chor, A. Fiat, and M. Naor, "Tracing Traitors", in Proc. Crypto '94, Santa Barbara, California, 1994, pp. 257-270.

[CD96] C. J. Colbourn and J.H.Dinitz, eds, CRC Handbook of Combinatorial Designs, CRC Press, Inc., 1996.

[CKLS97] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure Spread Spectrum Watermarking for Multimedia", IEEE Transactions on Image Processing", Vol.6, pp. 1673-1687, 1997.

[CMB01] I. J. Cox, Matthew L. Miller, and Jeffrey A. Bloom, "Digital Watermarking", Morgan Kaufmann Publishers, Inc., San Francisco, 2001.

[DFKY03] Yevgeniy Dodis, Nelly Fazio, Aggelos Kiayias and Moti Yung, “Scalable Public-Key Tracing and Revoking”, in Principles of Distributed Computing (PODC’03), Boston, 2003, pp.190-199.

[EK03] Sabu Emmanuel and Mohan S. Kankanhalli, “A digital rights management scheme for broadcast video”, Multimedia Systems, Vol 8, Number 6, pp. 444-458, 2003.

[EKK97] Funda Ergün, Joe Kilian, and Ravi Kumar, “A Note on the Limits of Collusion-Resistant Watermarks”, in Advances in Cryptology - EUROCRYPT '99, Prague, Czech Republic, 1999, pp.140-149.

[FS02] M. Fernández and M.Soriano “Fingerprinting Concatenated Codes with Efficient Identification”, Information Security Conference 2002 (ISC'02), Sao Paulo, (Brazil), 2002, pp. 459-470.

[FN93] A. Fiat and M. Naor, “Broadcast Encryption,” in Proc. Crypto '93, Santa Barbara, California, 1993, pp. 480-491.

[FS04] M. Fernandez, M. Soriano, "Soft-decision tracing in fingerprinted multimedia content", Multimedia, IEEE, Volume 11, Issue 2, pp. 38- 46, 2004

[FT99] A. Fiat and T. Tassa, “Dynamic traitor tracing,” in Proc. Crypto'99, Espoo, Finland, 1999, pp. 354-371.

[GSY99] E. Gafni, J. Staddon and Y. L. Yin, “Efficient Methods for Integrating Traceability and Broadcast Encryption,” in Proc. Crypto’99, Espoo, Finland, 1999, pp. 372-387.

[GSW00] J. Garay, J. Staddon and A. Wool, “Long-Lived Broadcast Encryption.,” in Proc. Crypto '00, Santa Barbara, California, 2000, pp.333-352

[VG01] V. Guruswami, “List Decoding of Error-Correcting Codes”, Ph.D thesis, MIT, August 2001.

[GS99] V. Guruswami and M. Sudan, “Improved decoding of Reed-Solomon and algebraic geometric codes”, IEEE Transactions on Information Theory, vol.45, pp.1757-1768, 1999.

[GP99] H. Guth and B. Pfitzmann, “Error- and Collusion-Secure Fingerprinting for Digital Data”, in Proc. Information Hiding’99, Dresden, Germany, 1999, pp.134-145.

[HLLT98] H. D. L. Hollman, J. H. van Lint, J. P. Linnartz , and L. M. G. M. Tolhui-zen , “On Codes with the Identifiable Parent Property,” Journal of Combinatorial Theory, pp. 121-133, Series 82, 1998.

[HP85] D. R. Hughes and F.C. Piper, “Design Theory”, Cambridge University Press, 1985.

[IMK99] H. Inoue, A. Miyazaki, and T. Katsura, “An Image Watermarking Method Based On The Wavelet Transform”, in Proc. ICIP, New York, 1999, pp.296-300.

[KP00] S. Katzenbeisser and F. A. P. Petitcolas (ed.), “Information Hiding Techniques for Steganography and Digital Watermarking”, Artech House, 2000.

[KY02] A. Kiayias and M. Yung, “Cryptographic Hardness based on the Decoding of Reed-Solomon Codes”, in Proc. of ICALP 2002, Málaga, Spain, 2002, pp 232-243.

[KV00] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes,” in Proc. of the 38th Annual Allerton Conference on Communication, Control and Computing, pp.625-635, 2000

[KD98] K. Kurosawa and Y. Desmedt, “Optimum traitor tracing and asymmetric schemes,” in Proc. EuroCrypt'98, Helsinki, Finland, 1998, pp.145-157.

[TL01] T. Lindkvist, “Fingerprinting of Digital documents,” Ph.D thesis, Linköping University, Sep 2001.

[LLS02] T. Lindkvist, J. Löfvenberg, and M. Svanström "A class of traceability codes", IEEE Transaction on Information Theory, Vol. 48, Number 7, pp. 2094-2096, 2002.

[LL00] J. Löfvenberg and T. Lindkvist, “A General Description of Pirate Strategies in a Binary Fingerprinting System”, 2000, available at http://www.it.isy.liu.se/publikationer/LiTH-ISY-R-2259.pdf

[LW98] Jacob Löfvenberg and Niclas Wiberg, "Random Codes for Digital Fingerprinting," 1998, Report LiTH-ISY-R-2059, ISSN 1400-3902, Presented at ISIT'98.

[MAT03] "MatLab™ tool kits", available at www.matlab.com

[MS77] F. J. McWilliams and N. J. Sloane, "The Theory of Error-Correcting Codes", North-Holland, Amsterdam 1977.

[NP00] M. Naor and B. Pinkas, "Efficient Trace and Revoke Schemes", Financial Crypto '2000, Anguilla, 2000.

[NIST00] "NIST FIPS 186-2 Standard", available at www.csrc.nist.gov

[PSS03] C. Peikert, A. Shelat, and A. Smith, "Lower bounds for Collusion-Secure Fingerprinting," In Proc. SODA'03, Baltimore, 2003, pp.472-479.

[PW72] W. W. Peterson and E.J. Weldon, "Error Correcting Codes" MIT Press, Cambridge, M.A., 1972.

[PS96] B. Pfitzmann and M. Schunter, "Assymetric fingerprinting,"in Proc. of Eurocrypt '96, Zaragoza, Spain, 1996, pp. 84-95.

[PS97] B. Pfitzmann and M. Schunte, "Asymmetric Fingerprinting for large collusions", In proc. of 4th ACM conference on computer communications security, pp. 151-160, 1997

[PW97] B. Pfitzmann and M. Waidner, "Annonymous fingerprinting", in Advances in Cryptology, EUROCRYPT 97, Zaragoza, Spain, 1997, pp 88-102.

[JP97] J. S. Plank, "A tutorial on Reed-Solomon coding for fault tolerance in RAID-like systems", Software Practice and Experience (SPE), 27(9), 995-1012, 1997.

[RS60] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," J. Soc. Industrial Appl. Math., vol. 8, pp. 300–304, 1960.

[Roth04] R. Roth, "Lecture notes in Coding theory", available on Internet

[SW03] R. Safavi-Naini and Y. Wang, "Sequential traitor tracing", IEEE Transactions on Information Theory, Vol. 49, No. 5, pp. 1319-1326, 2003.

[SW01] R. Safavi-Naini and Y. Wang, "New results on Frameproof codes and Traceability schemes," IEEE Transactions on Information Theory, Vol. 47, No. 7, pp. 3029-3033, 2001.

[SW02a] R. Safavi-Naini and Y. Wang, "Collusion secure q-ary fingerprinting for perceptual content", in Proc. Security and Privacy in Digital Rights Management (SPDRM'01), Philadelphia, PA, 2002, pp. 57-75.

[SW02b] R. Safavi-Naini and Y. Wang, "Traitor Tracing for Shortened and Corrupted Fingerprints", in Digital Rights Management Workshop (DRM'02), Washington DC, 2002, pp. 81-100.

[SS01] P. Sarkar and D. R. Stinson, "Frameproof and IPP Codes", in Indocrypt'01, Springer-Verlag, pp.117-126, 2001.

[SSW01] J. N. Staddon, D. R. Stinson and R.Wei, "Combinatorial properties of frameproof and traceability codes", IEEE Trans. Inform. Theory, Vol 47, pp.1042-1049, 2001.

[SW99] D. R. Stinson and R. Wei , "Key preassigned Traceability Schemes for Broadcast Encryption," in Selected Areas in Cryptology - SAC '98, Ontario, Canada, 1999, pp. 144-156.

[SW98] D. R. Stinson, R. Wei, "Combinatorial properties and constructions of traceability schemes and frameproof codes," SIAM Journal of Discrete Mathematics 11, pp. 41-53, 1998.

[SWZ00] D. R. Stinson, R. Wei, and L. Zhu, "New constructions for perfect hash families and related structures using combinatorial designs and codes", Journal of combinatorial designs, Volume 8, Number 3 pp. 189-200, 2000.

[MS99] M. Svanström, "Ternary Codes with Weight Constraints," Linköping Studies in Science and Technology, Dissertation No. 572, 1999.

[GT03] G. Tardos, Optimal probabilistic fingerprint codes, in Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC'03), pp.116 - 125, 2003.

[VSG01] Ravi Sankar Veerubhotla, Ashutosh Saxena, and V. P. Gulati, "Digital Certificates with Biometrics", in Proc. of International conference on Information Technology (CIT '01), pp. 178-184, 2001.

[VSGP04a] Ravi Sankar Veerubhotla, Ashutosh Saxena, V.P. Gulati, and A.K.Pujari, “On Gossip codes and Traceability Schemes” in Proc. of the International Conference on Information Technology: Coding and Computing (ITCC 2004), Las Vegas, Nevada, pp.772-777, 2004.

[VSGP04b] Ravi Sankar Veerubhotla, Ashutosh Saxena, V.P. Gulati, and A.K.Pujari, "Gossip Codes for Fingerprinting: Construction, Erasure Analysis and Pirate Tracing", in Journal of Universal Computer Science for a special issue on Information Assurance and Security, Vol. 11, issue 1, pp.122-149, 2005

[VSG02] Ravi Sankar Veerubhotla, Ashutosh Saxena, and V. P. Gulati: "Reed Solomon Codes for Digital Fingerprinting", in Proc of Indocrypt '02, Springer-Verlag, pp. 163-175, 2002.

[XBA97] X. G. Xia, C. G. Boncelet, and G. R. Arce, "A multiresolution watermark for digital images," in Proc. IEEE Int. Conf. Image Processing, Santa Barbara, CA, 1997, pp. 548-551.

Annexure

Published and communicated work

1. Ravi Sankar Veerubhotla, Ashutosh Saxena, V.P. Gulati, and A.K.Pujari, "Gossip Codes for Fingerprinting: Construction, Erasure Analysis and Pirate Tracing", in Journal of Universal Computer Science for a special issue on Information Assurance and Security, Vol. 11, issue 1, pp.122-149, 2005
2. Ravi Sankar Veerubhotla, Ashutosh Saxena, V.P. Gulati, and A.K.Pujari, "On Gossip codes and Traceability Schemes" in Proc. of International Conference on Information Technology: Coding and Computing (ITCC 2004), Las Vegas, Nevada, pp.772-777, 2004.
3. Ravi Sankar Veerubhotla, "Techniques for Broadcast Security", in Proc. of National Conference on Hardware and Software Solutions for Secure Networks (HsecNet'04), Hyderabad, India, pp. 1-9, 2004
4. Ravi Sankar Veerubhotla, Ashutosh Saxena, and V.P.Gulati, "On Copyright Protection Techniques: Watermarking and Digital Fingerprinting" in Proc. of National conference on Information security, New Delhi, India, pp.179-188, 2003.
5. Ravi Sankar Veerubhotla, Ashutosh Saxena, and V. P. Gulati: "Reed Solomon Codes for Digital Fingerprinting", in Proc. of Indocrypt '02, Springer-Verlag, pp. 163-175, 2002.
6. Ravi Sankar Veerubhotla, Ashutosh Saxena, and V. P. Gulati: "Digital Certificates with Biometrics", in Proc. of International conference on Information Technology (CIT '01), pp. 178-184, 2001
7. Ravi Sankar Veerubhotla, Ashutosh Saxena, and V.P.Gulati, "An Elliptic Curve Public Key Traitor Tracing Scheme" Presented at National Workshop on Cryptology, Oct'01, Indian Statistical Institute, Kolkata
8. Ashutosh Saxena, Ravi Sankar Veerubhotla, and V.P.Gulati, "A Time Stamped Multiparty Signature Scheme over Elliptic Curves" Presented at National Workshop on Cryptology, Oct'01, Indian Statistical Institute, Kolkata