Dynamic Frameproof Codes

Maura Paterson

Technical Report
RHUL-MA-2005-12
13 December 2005

Department of Mathematics
Royal Holloway, University of London
Egham, Surrey TW20 0EX, England
http://www.rhul.ac.uk/mathematics/techreports

DYNAMIC FRAMEPROOF CODES

Maura Beth Paterson

Royal Holloway and Bedford New college,
University of London

Thesis submitted to

The University of London

for the degree of

Doctor of Philosophy

2005.

Abstract

There are many schemes in the literature for protecting digital data from piracy by the use of digital fingerprinting, such as *frameproof codes*, which prevent traitorous users from colluding to frame an innocent user, and *traitor-tracing schemes*, which enable the identification of users involved in piracy. The concept of traitor tracing has been applied to a digital broadcast setting in the form of *dynamic traitor-tracing schemes* and *sequential traitor-tracing schemes*, which could be used to combat piracy of pay-TV broadcasts, for example. In this thesis we explore the possibility of extending the properties of frameproof codes to this dynamic model.

We investigate the construction of *l-sequential c-frameproof codes*, which prevent framing without requiring information obtained from a pirate broadcast. We show that they are closely related to the ordinary frameproof codes, which enables us to construct examples of these schemes and to establish bounds on the number of users they support.

We then define *l-dynamic c-frameproof codes* that can prevent framing more efficiently than the sequential codes through the use of the pirate broadcast information. We give constructions for schemes supporting an optimal number of users in the cases where the number c of users colluding in piracy satisfies $c \geq 2$ or $c = 1$.

Finally we consider *sliding-window l-dynamic frameproof codes* that provide ongoing protection against framing by making use of the pirate broadcast. We provide constructions of such schemes and establish bounds on the number of users they support. In the case of a binary alphabet we use geometric structures to describe constructions, and provide new bounds. We then go on to provide two families of constructions based on particular parameters, and we show that some of these constructions are optimal for the given parameters.

Acknowledgements

I wish to thank to my supervisors Prof. Simon Blackburn and Prof. Peter Wild for their many helpful suggestions, their patient reading of my work and their generosity with their time.

Thanks to Alex, Amy, Chris, Geraint, Illana, James, Laurence, Livi, Paula, Roger and Su-Jeong for all the super-happy-fun, and especially to Thomas for his kindness and support, and his assistance with proofreading.

Thanks to Dad, Ailsa, and Fiona for their inspiration and encouragement, and for proofreading a draft.

I am very grateful to the Commonwealth Scholarship Commission whose financial assistance enabled me to undertake this research.

Contents

Chapter 1

Introduction

1.1 Digital Data Requires Piracy Protection

Over the past few decades there have been significant advances in the capacity to store and transmit digital data, leading to a substantial increase in the amount of material that is being provided in this form. Many types of data, including movies, music, images and text are stored in a digital format and are made available to consumers through media such as CDs, DVDs and the internet. Such data are potentially worth a lot of money to their owners, with an increasing number of consumers being willing to pay for access to data.

Storing data in a digital format has the advantage that the data can readily be reproduced or transmitted without any degradation or loss of information. However, the ease with which digital information can be copied makes it potentially vulnerable to piracy, the production of illicit copies for which the owner of the content has not been paid. Suppliers are interested in ways in which they can protect their intellectual property from being stolen; there is thus a desire for schemes that can discourage piracy of digital data.

1.2 Outline of the Thesis

In this section we give a synopsis of the thesis, describing the contents and the main results of each chapter. We also discuss its aims and the motivation behind the problems investigated.

Chapters 2 and 3 of this thesis contain a survey of certain aspects of the theory of piracy prevention present in the literature. Two commonly discussed mechanisms for implementing piracy-prevention schemes, the use of *digital fingerprinting* and the strategic distribution of keys in *decoder boxes*, are described in Chapter 2. This is followed by an examination of two types of codes designed for use with such systems: *c-traceability codes*, which allow coalitions of up to c traitors involved in piracy to be traced and incriminated, and *c-frameproof codes*, which prevent coalitions of at most c traitors from falsely incriminating innocent users. We give precise definitions of these codes as well as examples of how they are constructed, and we discuss known bounds on the number of users they can support.

In Chapter 3 we introduce the dynamic setting in which information is continually broadcast to its recipients; this setting models such real-world scenarios as pay-TV broadcasting. We then describe *sequential traitor-tracing schemes* and *dynamic traitor-tracing schemes*, which result from applying the concepts of fingerprinting and traitor tracing in the dynamic setting. Again we give examples of such schemes, and discuss bounds on the time taken to implement them relative to the number of users that they support, and we mention an observation of Fiat and Tassa [12] that leads to a limitation on when such schemes can be applied.

The aim of this thesis is to examine the possibility of extending the properties of frameproof codes to the dynamic broadcast setting. Doing this would

afford a degree of piracy protection in cases when the full strength of a traitor-tracing scheme is not required, or when there are insufficient resources for the implementation of such a scheme. In Chapter 4 we consider the sequential model, in which access to the pirate's broadcast is not available. We propose a definition of *l-sequential c-frameproof codes*, schemes that ensure that coalitions of up to c traitors cannot falsely incriminate an innocent user over any l consecutive time segments. We make some observations on how to simplify the construction of such schemes and we show that l-sequential c-frameproof codes are closely related to the standard c-frameproof codes. These schemes are influenced by several parameters, namely q, the number of different ways in which a segment of the data can be marked, the number of users, denoted by n, the number c of traitorous users, and l, the size of the windows of adjacent time segments over which framing is to be prevented. We use the connection between sequential frameproof codes and frameproof codes to establish bounds on some of these parameters, and to describe constructions that are optimal with respect to these bounds in the case where $c = n - 1$. We are particularly interested in finding bounds for certain parameters such as l or n once the values of the other parameters have been fixed. Since we wish our schemes to be as efficient as possible in terms of implementation time and cost it is natural to seek the maximum number of users that a scheme can support, or the minimum time required for its implementation using a given quantity of resources. The ultimate aim is to reconcile the upper and lower bounds for these extremal quantities and to find ways to construct schemes meeting these bounds, as such constructions would result in schemes of optimal efficiency. In the case of c-sequential frameproof codes, however, we are constrained by our incomplete knowledge of the equivalent bounds for c-frameproof codes.

In Chapter 5 we consider the situation when feedback from the pirate broadcast is available. In this chapter we investigate schemes that require l time segments in order to ensure no innocent user is framed, which we refer to as *l-dynamic frameproof codes*. We prove a bound on the maximum number of users supported by these schemes and provide a method of constructing schemes that meet the bound. Dynamic frameproof codes differ from sequential frameproof codes in that they are applied once to prevent framing over a particular choice of l consecutive time segments, whereas the sequential codes are applied continuously and prevent framing over any l consecutive segments.

This difference between the sequential and dynamic frameproof codes leads to the definition of the *sliding-window model* of framing prevention, discussed at the start of Chapter 6. The rest of the chapter is devoted to making use of the dynamic setting to construct *sliding-window dynamic frameproof codes*; as the name suggests these schemes are applied in the dynamic setting with access to information from the pirate broadcast, but have the same sliding-window frameproof properties as sequential frameproof codes. The use of the pirate broadcast information results in these schemes being more efficient than the corresponding sequential schemes, protecting a given number of users over much shorter windows. Examples of these schemes are given, and bounds on some of their parameters are established, but a discrepancy remains between the known upper and lower bounds. In an effort to reduce this gap we proceed to concentrate on the case where $q = 2$. We give a sufficient condition for the existence of sliding-window l-dynamic frameproof code and find that a bound on the possible window-lengths arises from that condition. A construction of a code meeting that bound is given.

We then turn our attention to the use of geometric concepts in constructing

new sliding-window dynamic frameproof codes. We reinterpret some of our previous results in a geometric notation, and give a lower bound on the possible window lengths of codes arising from geometric constructions that is very close to the upper bounds resulting from previously-discussed constructions.

In attempting to generalise these geometric constructions to make use of alphabets of size $q > 2$, however, we soon discover that the resulting constructions are not optimal. In Chapter 7 we attempt to find both better constructions for $q > 2$ and ways of comparing the different constructions we have discussed previously. These goals are achieved by the construction of a new family of codes that provides a common description for all the sliding-window dynamic frameproof codes constructed so far, and which provides parameters that can be selected in order to optimise the number of users supported by the resulting code. We then go on to describe a new construction that yields more-efficient schemes than the above family of constructions when the same parameters are used. We show that this new construction protects a number of users that is optimal for the given parameters. We then consider some asymptotic bounds on the number of users protected when the alphabet size becomes large. The thesis concludes with a discussion of future possibilities.

Chapter 2

Schemes for the Prevention of Piracy

A great deal of valuable intellectual property, such as software, music and so on, is stored in digital form. Digital data can easily be copied and transmitted, and this facilitates the distribution of material by its owners. Unfortunately it has the further implication that unscrupulous people can readily make illicit copies of the data, something which the owners naturally wish to prevent. Many different methods for discouraging such piracy have been proposed. In this chapter we describe two of the most commonly discussed mechanisms through which piracy prevention measures can be delivered. We then consider two types of codes that can be applied in either setting, namely *traceability codes* and *frameproof codes*, which each have slightly different piracy-deterring properties. We give examples of how such codes are constructed and discuss known bounds on their sizes, and we consider the piracy protection afforded by these codes.

2.1 Mechanisms for Discouraging Piracy

The methods used to prevent piracy will naturally depend on the form in which the protected data is stored and distributed. The first mechanism we

discuss is used in the case where the data is transmitted to the recipient in a single instance. This could include CDs carrying software or music, or DVD movies. The second setting we discuss involves decoder boxes, as used by pay-TV stations to enable legitimate users to decrypt encrypted broadcasts. In this case the goal is to deter those who have not paid to watch the broadcast from constructing illicit "pirate" decoder boxes. Most of the piracy prevention schemes discussed in the literature are designed to fit one of these two scenarios (apart from the dynamic schemes, which we will discuss in the next chapter).

2.1.1 Digital Fingerprinting

When data is distributed on CDs or DVDs little action can be taken to stop pirates from making illegal copies. If, however, the source of the illegal copies can be detected then action can be taken against the pirates. This in itself will act as a deterrent to pirates as it increases the risk of being caught, and makes piracy less attractive as a result. The idea behind digital fingerprinting is somehow to mark the data that is given to each user so that an illegally-made copy will contain information about the user who made the copy, much as ordinary fingerprints can be used to identify the person who left them. The actual method for embedding marks in the data lies beyond the scope of this thesis; some proposals are discussed in [17]. We require that the marks satisfy two properties; the first, in the terminology of Fiat and Tassa [12], is that of *similarity* , namely that the presence of the mark should not affect the functioning of the data. For instance, if the data consists of a movie then a person watching the movie should be unable to distinguish between a marked and an unmarked version. The second property that we require, termed *robustness* [12], is that it should be impossible for the pirate to alter or

remove a mark without causing noticeable damage to the actual data. There is some debate, however, over whether it is possible to create marks of this type. For the purposes of what follows we assume it can be done; this is known as the *watermarking assumption* (see [17] for a discussion of issues associated with watermarking).

If every user receives a copy of the data that has been uniquely marked then any pirate copies can be traced back to the user responsible. It may be infeasible to produce this many different variations of the data, however. One solution is to split the data into a finite number of sections, each of which is marked. If there are l different sections and you produce q differently-marked variants of each section then that enables you to produce q^l different versions of the entire data set. We will think of the q different versions of a segment as corresponding to the letters of an alphabet Q of size q; each version of the data is then associated with a word of Q^l. The set $C \subseteq Q^l$ of words used to mark the data is then a length l, q-ary code.

If a pirate makes a copy of a single version of the data then it can be traced to the user who owned that version. If a pirate has access to more than one version, however, it can combine different sections from different versions to try and escape being identified. Recall that we are assuming the pirate is unable to remove or change any given mark, but there is nothing to stop it replacing a section in one version with a section from another version.

Example 2.1 Suppose the data is marked using words of length 3 and an alphabet $Q = \{0, 1, 2\}$. If a pirate possesses versions marked by the words $(0, 0, 1)$ and $(1, 0, 2)$ then by combining the various sections it can produce any of the words in the set $\{(0, 0, 1), (0, 0, 2), (1, 0, 1), (1, 0, 2)\}$. ■

If a pirate has access to a particular set S of marking words then the set of

new words it can form in such a manner is referred to as the *set of descendents of* S.

Definition 2.1. *Suppose* $S \subseteq Q^l$. *We define the* set of descendents of S, *denoted* $\mathrm{desc}(S)$, *by*

$$\mathrm{desc}(S) = \{x \in Q^l | \forall\ i = 1, 2, \ldots, l\ \exists y \in S \text{ with } x_i = y_i\}.$$

We wish to be able to deter piracy even when users collude by combining their different versions in this manner. This can be achieved by allocating marking words from a code $C \subseteq Q^l$ with particular properties. Two such types of code are described later in this chapter.

2.1.2 Decoder Boxes for Encrypted Broadcasts

Pay-TV stations encrypt their broadcasts so that only paying users are able to view their programs. This is frequently done by providing paying users with decoder boxes that attach to their televisions and decrypt the data once it is received. In what follows we assume that the encryption used is sufficiently secure that pirates cannot break the cryptosystem unless they have access to the decryption keys contained within the boxes. A pirate with access to a decoder box could make a replica of the box, however, thus enabling someone who is not a paying user to decrypt the programs. In order to prevent this we wish to be able to trace any pirate decoder box back to the legitimate box that was copied in its construction. One way of doing this is to place a unique set of l keys $k_1, k_2, \ldots, k_l$ in each box. If each key k_i comes from a set of q possible keys there are q^l possible boxes. The program to be broadcast is then split into l sections, b_1 to b_l, each of which is encrypted using a temporary key $s_1, s_2, \ldots, s_l$. The temporary key s_1 is then encrypted with each of the q

possible keys k_1^1 to k_1^q, the key s_2 is encrypted with each of the q possibilities for k_2^i and so on. The encryptions of all the s_i are placed together in an *enabling block*, which is transmitted before the encrypted program is broadcast (Fig. 2.1). When a paying user receives the broadcast they can then use their

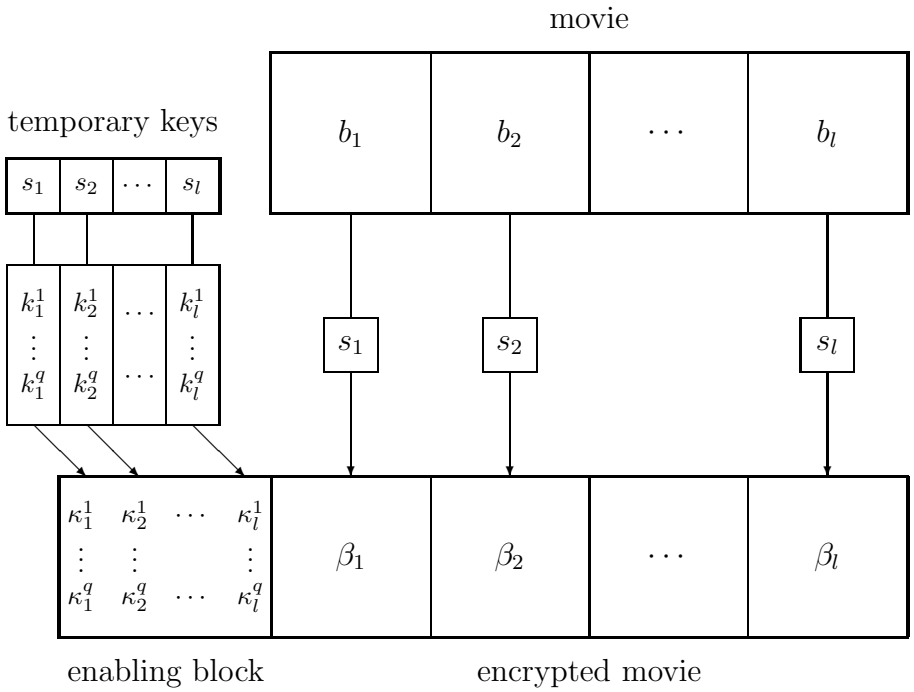

Figure 2.1: Encryption of a movie for broadcast on pay-TV

key k_i for $i = 1, 2, \ldots, l$ to retrieve the keys s_i from the enabling block, which allows them to decrypt the program (Fig. 2.2). The set of keys for each decoder box can be thought of as a word of length l over an alphabet Q of size q. The set of words $C \subseteq Q^l$ that are used in the boxes is thus a q-ary, length l code.

As we saw in the case of digital fingerprinting, if the pirate copies a single box it is easy to see which user is responsible. If the pirate combines keys from boxes whose keys correspond to words from a set $S \subseteq Q^l$, however, then it is capable of reproducing any box whose keys correspond to a word from the set

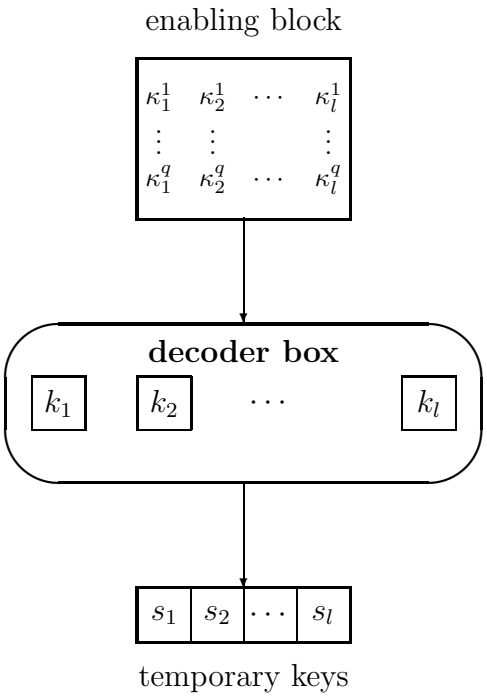

Figure 2.2: Retrieving temporary keys from the enabling block

$\mathrm{desc}(S) \subseteq Q^l$. In the following section we describe codes that can be applied in this situation, as well as in a fingerprinting context, in an attempt to limit the pirate's ability to evade detection by combining keys in this manner.

2.2 Traitor Tracing

Ideally, when a pirate decoder box or a pirated CD or DVD is discovered, we would like to be able to identify at least one of the users, referred to as *traitors*, who have contributed to its manufacture. This concept is known as *traitor tracing* and was first introduced by Chor, Fiat and Naor in [9]. There are many different definitions of traitor tracing in the literature, and an abundance of schemes have been proposed ([9, 10, 12, 7, 15, 2, 18, 3, 19, 8, 14]),

which have widely varying properties. For example some support *black-box traitor tracing*, in which the tracing process uses the pirate decoder box as a "black box" without requiring a knowledge of the details of its construction (see [9, 10]). Some schemes use public key encryption ([7]), others, known as *threshold schemes* have a certain probability of failure ([9, 10, 2]). Chor, Fiat, Naor and Pinkas [10], working in the decoder-box setting, define a *traitor-tracing scheme* to consist of a *user initialisation scheme*, an *encryption scheme* and a *traitor-tracing algorithm*. The user intialisation scheme determines how the keys are distributed to the users. The encryption scheme controls how the data is encrypted, and the traitor-tracing algorithm takes the keys from a pirate decoder box as input, and outputs at least one of the users responsible for the piracy. They define a *fully c-resilient tracing scheme* to be one in which a pirate box made by c or fewer traitors can be used as a black box to trace at least one of the traitors, provided that the encryption scheme is not broken. In this section we will assume that the encryption is secure, and focus on the code used to distribute the keys, and on corresponding algorithms that will trace a traitor given the codeword from the pirate decoder. We note that these codes and tracing algorithms can also be applied in the digital fingerprinting setting.

2.2.1 Traceability Codes

If a pirate possesses a set S of codewords then it can produce any of the words in the set $\mathrm{desc}(S)$. In order to limit the pirate's ability to evade detection, it is natural to use codes where there is some sort of limitation on the potential descendants of sets of codewords. An example of such a code is a *c-traceability code*.

Definition 2.2. *A set $C \subseteq Q^l$ is a* c-traceability code *(c-TA code) if given any subset $S \subset C$ with $|S| \leq c$ and any $x \in \mathrm{desc}(S)$ it follows that when a codeword $y \in C$ satisfies $d(x, y) \leq d(x, z)$ for all $z \in C$ then y lies in S. (Here $d(x, y) = |\{i \in 1, 2, \ldots, l | x_i \neq y_i\}|$ is the Hamming distance.)*

This means that if a word $x \in Q^l$ is a descendent of a particular set of size at most c then the codewords nearest to x (with respect to the Hamming metric) are members of that set. Therefore if we use a c-traceability code (either to fingerprint data or to distribute keys in decoder boxes) then when a pirate CD or decoder box is found we can examine the corresponding word and we know that the nearest codewords correspond to some of the users responsible for the piracy.

Example 2.2 The length 3 code C over the alphabet $Q = \{0, 1, 2\}$ whose words are $(0, 0, 0)$, $(1, 1, 1)$, and $(2, 2, 2)$ is a 2-TA code. This is trivially true since any descendent of two codewords must contain two letters from one of those words. This descendent will have a distance of at most 1 from this parent, a distance of at least 2 from the other parent, and a distance of 3 from any other codeword. Hence the closest codeword is necessarily a member of the original parent set. ■

The concept of a traceability code first appeared in Chor *et al.* [9, 10] under the name of an open one-level k-resilient traceability scheme, although an explicit construction was not given. Stinson and Wei [15] studied slightly more general structures, which they refer to as *traceability schemes*; here we restrict our attention to traceability codes as they relate more closely to the schemes we will discuss later.

Staddon *et al.* [14] compare TA codes with frameproof codes and other related structures, and provide constructions as well as bounds on the sizes of

such codes. In particular they give the following bound.

Theorem 2.3. *[14] Let C be a q-ary, length l, c-TA code. Then*

$$|C| \leq q^{\lceil \frac{l}{c} \rceil} + 2c - 2.$$

Proof. Suppose C is a q-ary, length l, c-TA code with $|C| > q^{\lceil \frac{l}{c} \rceil} + 2c - 2$. Let $A = (a_{ij})$ be the $|C| \times l$ matrix whose rows are the codewords of C and divide A into c submatrices by letting A_1 be the $|C| \times \lceil \frac{l}{c} \rceil$ matrix consisting of the first $\lceil \frac{l}{c} \rceil$ columns of A, letting A_2 consist of the next $\lceil \frac{l}{c} \rceil$ columns and so on, with $A_1, \ldots, A_t$ having size $|C| \times \lceil \frac{l}{c} \rceil$ and $A_{t+1}, \ldots, A_c$ having size $|C| \times \lfloor \frac{l}{c} \rfloor$ where $t \in \{1, 2, \ldots, c\}$ satisfies $t \equiv l \pmod{c}$. Now A_1 has more than $q^{\lceil \frac{l}{c} \rceil} + 2c - 2$ rows, hence there exist two rows of A, say i_1 and j_1, with the corresponding rows of A_1 being identical. Let A_2' be the matrix obtained from A_2 by removing rows i_1 and j_1. Then A_2' has more than $q^{\lceil \frac{l}{c} \rceil} + 2c - 4$ rows, hence there exist two rows i_2 and j_2 identical in A_2', and these are distinct from i_1 and j_1. Continue this procedure, at each step finding identical rows i_α and j_α in the matrix A_α', the matrix obtained by deleting rows $i_1, i_2, \ldots, \mathrm{i}_{\alpha-1}$ and $j_1, j_2, \ldots, \mathrm{j}_{\alpha-1}$ from matrix A_α. Once this has been performed for matrices $A_1, \ldots, A_c$ the result is two disjoint sets of rows (corresponding to codewords) $I = \{i_1, i_2, \ldots, i_c\}$ and $J = \{j_1, j_2, \ldots, j_c\}$ with i_1 and j_1 agreeing in the first $\lceil \frac{l}{c} \rceil$ positions, i_2 and j_2 agreeing in the next $\lceil \frac{l}{c} \rceil$ positions, and so on. Let $x \in Q^l$ be the word whose first $\lceil \frac{l}{c} \rceil$ coordinates match those of i_1 and j_1, and whose next $\lceil \frac{l}{c} \rceil$ coordinates match those of i_2 and j_2 and so on. Then x is a descendent of both I and J. By the c-TA property if y is the closest codeword to x then y lies in each parent set of size at most c, hence $y \in I$ and $y \in J$. But I and J are disjoint by construction, which leads to a contradiction. □

One way to construct c-TA codes is by using error-correcting codes. This

is made possible by the following observation of Chor *et al.*

Theorem 2.4. *[9][10] Let C be a q-ary, length l code containing n codewords whose minimum (Hamming) distance d satisfies $d > l - \frac{l}{c^2}$ for some positive integer c. Then C is a c-TA code.*

Proof. Suppose C is such a code. The distance between any two codewords being greater than $l - \frac{l}{c^2}$ it follows that any two codewords agree in fewer than $\frac{l}{c^2}$ coordinates. Let $x \in Q^l$, and suppose $x \in \text{desc}(S)$ for some $S \subset C$ with $|S| \leq c$. Let $z \in C$ be a closest codeword to x, so that $d(x, z) \leq d(x, w)$ for all $w \in C$. Then x and z have at least $\frac{l}{c}$ coordinates in common; denote the set of coordinates in which they agree by I. As S is a parent set of x containing c codewords there exists some codeword $y \in S$ agreeing with x in at least $\frac{|I|}{c} > \frac{l}{c^2}$ of the coordinates in I. But y also agrees with z in these coordinates; because of the minimum distance this implies that $y = z$, so $z \in S$. Hence C satisfies the c-TA condition. □

Blackburn [5] observes that this result is still true in the case where the minimum distance satisfies $d > l - \lceil \frac{l}{c^2} \rceil$. Examples of codes with an appropriate minimum distance are provided by *Reed-Solomon codes* (see [5]), as in the following construction.

Construction 2.5. *Let q be a prime power, and l be a positive integer with $l \leq q$. Let $\alpha_1, \alpha_2, \ldots, \alpha_l$ be distinct elements of the field $F = GF(q)$. We define a code C by setting*

$$C = \left\{ \big(f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_l)\big) \,\middle|\, f \in F[x],\ \deg f < \left\lceil \frac{l}{c^2} \right\rceil \right\}.$$

(This is a specific example of the Reed-Solomon construction.)

The number of distinct polynomials over f with degree less than $\lceil \frac{l}{c^2} \rceil$ is

$q^{\lceil \frac{l}{c^2} \rceil}$; *as* $l \geq \lceil \frac{l}{c^2} \rceil$ *each choice of polynomial will give rise to a distinct codeword, hence* $|C| = q^{\lceil \frac{l}{c^2} \rceil}$. *We observe that no two distinct codewords agree in* $\lceil \frac{l}{c^2} \rceil$ *or more positions, for by interpolation the corresponding polynomials would then be equal. Thus the minimum distance of* C *is at least* $l - \lceil \frac{l}{c^2} \rceil$. *By Theorem 2.4 we have that* C *is a c-traceability code.*

Note that as for Construction 2 of [6] the condition that $l \leq q$ can be weakened to $l \leq q+1$ by allowing the α_i to be distinct elements of $GF(q) \cup \{\infty\}$, where $f(\infty)$ is defined to be the coefficient of $x^{\lceil \frac{l}{c^2} \rceil - 1}$ in f.

The code described in Example 2.2 is in fact a Reed-Solomon code. Here we give a less-trivial example of this construction.

Example 2.3 Let $l = 5$, $c = 2$ and $q = 5$; then $\lceil \frac{l}{c^2} \rceil = 2$. Let $\alpha_1 = 0$, $\alpha_2 = 1$, $\alpha_3 = 2$, $\alpha_4 = 3$ and $\alpha_5 = 4$. There are 25 polynomials over $GF(5)$ of degree less than two, so the code resulting from Construction 2.5 contains 25 words. For instance, the polynomials 3, x, $x + 1$ and $2x + 3$ give rise to the words $(3, 3, 3, 3, 3)$, $(0, 1, 2, 3, 4)$, $(1, 2, 3, 4, 0)$ and $(3, 0, 2, 4, 1)$ respectively. This code has minimum distance $5 - 2 = 3$, and is a 2-TA code. ■

Blackburn [5] poses the open problem of whether this Reed-Solomon construction provides optimal c-TA codes for large alphabet sizes.

2.3 Frameproof Codes

It may be the case that two or more users involved in piracy attempt to avoid detection by combining their codewords to produce a word corresponding to another user, in the hope that the innocent user will be blamed for the piracy. *Frameproof codes* were designed to prevent framing of this nature; they were first proposed by Boneh and Shaw [8]. We use the following definition of a

frameproof code, which appears in [12]. Note that this is slightly different from the original Boneh and Shaw definition, as that definition allows pirates with the ability to completely efface marks, which are prohibited by our robustness requirement.

Definition 2.6. *A code $C \subseteq Q^l$ is a* c-frameproof code *if every set $S \subset C$ with $|S| \leq c$ satisfies* $\text{desc}(S) \cap C = S$.

Thus when a frameproof code is used to fingerprint data (or to assign keys to decoder boxes) no set of c or fewer traitors can collude to frame a user outside of that set.

Example 2.4 The binary length 3 code C whose words are listed below has the property that any two distinct words agree in at most one position.

$$(0,0,0)$$
$$(1,1,0)$$
$$(0,1,1)$$
$$(1,0,1)$$

This implies that it is a 2-frameproof code. For, if $x \in \mathcal{C}$ is framed by $S = \{y_1, y_2\} \subset \mathcal{C}$ then $x \neq y_1$ means x and y_1 agree in at most one coordinate. This implies that x and y_2 must agree in the remaining two coordinates, which in turn implies that $x = y_2$, so x is framed by S if and only if $x \in S$. ■

In fact we have already seen several examples of c-frameproof codes, as shown by the following theorem.

Theorem 2.7. *[14] If $C \subset Q^l$ is a c-TA code, then it is also a c-frameproof code.*

Proof. Let $C \subset Q^l$ be a c-TA code, but suppose that it is not c-frameproof. Then there exists some set $S \subset C$ with $|S| \leq c$ and some $x \in C \setminus S$ with $x \in \text{desc}(S)$. But $x \in C$ and $d(x,x) = 0$ so $d(x,x) \leq d(x,z)$ for all $z \in C$, yet

$x \notin S$ thus the c-TA property is contradicted. Hence we conclude that every c-TA code must be c-frameproof. $\square$

Therefore we see that the c-TA property is a stronger condition than the c-frameproof property, and all the c-TA codes described in the previous section are also c-frameproof codes.

Given values of q, l and c we would like to know the largest possible size of a q-ary, length l, c-frameproof code. The literature contains many bounds on this quantity, some of which we state below.

Theorem 2.8. *Let $l \geq 2$ and $c \geq 2$ be integers, and let $t \in \{1, 2, \ldots, c\}$ satisfy $t \equiv l \pmod{c}$, so that $l = t\lceil \frac{l}{c} \rceil + (c-t)\lfloor \frac{l}{c} \rfloor$. If C is a q-ary, length l, c-frameproof code then*

1. *[6]* $\quad |C| \leq \max\left\{ q^{\lceil \frac{l}{c} \rceil}, t\left(q^{\lceil \frac{l}{c} \rceil} - 1\right) + (c-t)\left(q^{\lfloor \frac{l}{c} \rfloor} - 1\right) \right\}$;
2. *[16, 6]* $\quad |C| \leq \left(\frac{l}{l-(t-1)\lceil \frac{l}{c} \rceil} \right) q^{\lceil \frac{l}{c} \rceil} + \binom{l}{\lceil \frac{l}{c} \rceil - 1} q^{\lceil \frac{l}{c} \rceil - 1}$.

Proof. (1) Let C be a q-ary, length l, c-frameproof code. With each subset $S \subseteq \{1, 2, \ldots, l\}$ we associate a subset $U_S \subseteq C$ by setting

$$U_S = \{x \in C | \nexists y \in C \text{ with } x_i = y_i \text{ for all } i \in S\}.$$

Thus every word x in U_S is determined uniquely by its components x_i with $i \in S$, so $|U_S| \leq q^{|S|}$ as there are at most $q^{|S|}$ possible choices for these x_i. If the set $C \setminus U_S$ is non-empty then at least one of these choices belongs to words in $C \setminus U_S$ and hence does not correspond to a word of U_S. Thus if $|C| > q^{|S|}$ then $|U_S|$ is at most $q^{|S|} - 1$.

Partition the set $\{1, 2, \ldots, l\}$ into c disjoint subsets $S_1, S_2, \ldots, S_c$ where $|S_j| = \lceil \frac{l}{c} \rceil$ when $j \leq t$ and $|S_i| = \lfloor \frac{l}{c} \rfloor$ when $j > t$. Then

$$\left| \bigcup_{j=1}^{c} U_{S_j} \right| \leq t\left(q^{\lceil \frac{l}{c} \rceil} - 1\right) + (c-t)\left(q^{\lfloor \frac{l}{c} \rfloor} - 1\right).$$

The proof of the theorem results from the fact that $C = \bigcup_{i=1}^{c} U_{S_i}$. Suppose that there exists a word $x \in C \setminus \bigcup_{i=1}^{c} U_{S_i}$. Then for each $j = 1, 2, \ldots, c$ there exists another word $y^j \neq x$ in C that agrees with x in all the positions $i \in S_j$, as $x \notin U_{S_j}$. From this it follows that $x \in \mathrm{desc}(\{y^1, y^2, \ldots y^c\})$, in contradiction of the c-frameproof property of C. Hence we conclude that $C = \bigcup_{i=1}^{c} U_{S_i}$, so $|C| \leq t\Big(q^{\lceil \frac{l}{c} \rceil} - 1\Big) + (c-t)\Big(q^{\lfloor \frac{l}{c} \rfloor} - 1\Big)$.

(2) From the proof of (1) we know that if $S_1, S_2, \ldots S_c$ are non-empty sets partitioning $\{1, 2, \ldots, l\}$ then $C = \bigcup_{j=1}^{c} U_{S_j}$. Let $k = \lceil \frac{l}{c} \rceil$ and let $D \subseteq C$ be the set of codewords that are not defined uniquely by any choice of $k-1$ of their coordinates, so if $\mathcal{S} = \{S \subseteq \{1, 2, \ldots, l\} || S| = k-1\}$ then

$$D = \{x \in C | \forall S \in \mathcal{S}\ \exists y \in C \setminus \{x\} \text{ with } x_i = y_i \text{ for all } i \in S\}.$$

From the definition we see that $D = C \setminus \bigcup_{S \in \mathcal{S}} U_S$; together with the fact that $|\mathcal{S}| = \binom{l}{k-1}$ this gives us $|D| \geq |C| - \binom{l}{k-1} q^{k-1}$. Thus any bound on the size of D translates into a bound on the size of C; we now proceed to establish such a bound.

For any set $S \subseteq \{1, 2, \ldots, l\}$ we define $V_S = U_S \cap D$, and we define the set $\mathcal{V}$ by $\mathcal{V} = \{V_S | S \subseteq \{1, 2, \ldots, l\},\ |S| = k\}$. A bound on the size of D can be derived by counting in two ways the number of pairs (x, S) for which $V_S \in \mathcal{V}$ and $x \in V_S$. Since $|S| = k$ we have $|V_S| \leq |U_S| \leq q^k$, so for a given S there are at most q^k possible $x \in V_S$, and there are $\binom{l}{k}$ possible choices of S with $|S| = k$. Hence the number of such pairs is at most $\binom{l}{k} q^k$.

There are $|D|$ possible choices for x; let χ be the maximum possible size of a family $\mathcal{F}$ of sets $V_S \in \mathcal{V}$ not containing x. It follows that the number of sets $V_s \in \mathcal{V}$ with $x \in V_S$ is at least $|\mathcal{V}| - \chi = \binom{l}{k} - \chi$, hence the number of possible pairs (s, V_S) with $x \in V_s$ is at least $|D|\big(\binom{l}{k} - \chi\big)$.

Combining these two bounds implies that

$$\binom{l}{k} q^k \geq |D| \left(\binom{l}{k} - \chi \right),$$

therefore

$$|D| \leq q^k \left(\frac{1}{1 - \chi / \binom{l}{k}} \right).$$

We observe that if $S_1, S_2, \ldots, S_t$ are pairwise disjoint k-subsets of $\{1, 2, \ldots, l\}$ then $D = \bigcap_{j=1}^{t} V_{S_j}$. For, as the S_j are pairwise disjoint we have

$$|\{1, 2, \ldots, l\} \setminus \bigcup_{j=1}^{t} S_j| = l - tk = (c - t)(k - 1).$$

Thus it is possible to construct $(c-t)$ disjoint $(k-1)$-subsets $S_{t+1}, S_{t+2}, \ldots S_c$ partitioning $\{1, 2, \ldots, l\} \setminus \bigcup_{j=1}^{t} S_j$, so $C = \bigcup_{j=1}^{c} U_{S_j}$. However for $t+1 \leq j \leq c$ we have $V_{S_j} = D \cap U_{S_j} = \emptyset$ by definition of D as $|S_j| = k - 1$. Hence

$$\begin{aligned} \bigcup_{j=1}^{t} V_{S_j} &= \bigcup_{j=1}^{c} V_{S_j} \\ &= D \cap \bigcup_{j=1}^{c} U_{S_j} \\ &= D \cap C \\ &= D. \end{aligned}$$

This implies that there do not exist t pairwise-disjoint subsets in $\mathcal{F}$, for the union of any t disjoint sets in $\mathcal{V}$ is D, which implies that x is contained in at least one of them. A family of subsets of a set is said to be *t-colliding* if there do not exist t pairwise-disjoint subsets in the family; $\mathcal{F}$ is therefore t-colliding. The following lemma, due to Blackburn [6] gives a bound on the size of t-colliding families.

Lemma 2.9. *[6] Let l,t and k be positive integers with $tk \leq l$. A t-colliding family of of k-subsets of $\{1, 2, \ldots, l\}$ contains at most $\binom{l}{k} \frac{(t-1)k}{l}$ elements.*

Thus $\chi \leq \binom{l}{k}\frac{(t-1)k}{l}$ and $|D| \leq q^k\left(\frac{1}{1-\frac{(t-1)k}{l}}\right) = q^k\left(\frac{l}{l-(t-1)k}\right)$. From this we deduce $|C| \leq q^k\left(\frac{l}{l-(t-1)k}\right) + \binom{l}{k-1}q^{k-1}$ hence proving the result. □

A lower bound on the maximum size of a frameproof code is provided, in the case where q is a prime power, by a result due to Cohen and Encheva [11].

Theorem 2.10. *[11] Let $Q = GF(q)$. A length l linear code C (i.e. a code whose codewords form a vector space over Q) of dimension $\lceil \frac{l}{c} \rceil$ and minimum distance $l - \lceil \frac{l}{c} \rceil + 1$ is a c-frameproof code.*

Proof. Let C be such a code. As C has minimum distance $l - \lceil \frac{l}{c} \rceil + 1$ it follows that any two codewords agree in at most $\lceil \frac{l}{c} \rceil - 1$ positions. This implies that no c codewords can frame any other, since c codewords could contribute to at most $c\big(\lceil \frac{l}{c} \rceil - 1\big) \leq l - 1$ positions of a further codeword, thus at least $c + 1$ codewords would be required for framing. □

Linear codes with minimum distance d, dimension k and length l that satisfy $d = l - k + 1$ are known as MDS codes; Reed-Solomon codes provide an example of codes known to meet this bound. Therefore, as in the case of c-TA codes, it is possible to construct c-frameproof codes from Reed-Solomon codes of suitable parameters. Taking Construction 2.5 but requiring that the polynomials have degree less than $\lceil \frac{l}{c} \rceil$ instead of $\lceil \frac{l}{c^2} \rceil$ will give codes of the required parameters; this technique appears as Construction 2 of [6]. These codes contain $q^{\lceil \frac{l}{c} \rceil}$ codewords instead of the the $q^{\lceil \frac{l}{c^2} \rceil}$ codewords of the corresponding c-TA code, demonstrating that the weaker c-frameproof condition enables the construction of much larger codes than are possible with the c-TA condition.

We therefore have the following lower bound on the size of c-frameproof codes.

Corollary 2.11. *Let q be a prime power, and let $l \leq q+1$. Then there exists a q-ary, length l, c-frameproof code of size $q^{\lceil \frac{l}{c} \rceil}$.*

The following example is the c-frameproof parallel of Example 2.3.

Example 2.5 Let $l = 5$, $c = 2$ and $q = 5$; then $\lceil \frac{l}{c} \rceil = 3$. There are 125 polynomials over $GF(5)$ of degree less than three, so the 2-frameproof code resulting from the modified Construction 2.5 contains 125 words. This is five times as many words as in the 5-ary, length 5, 2-TA code in Example 2.3. ■

There are other constructions in the literature that yield larger codes in certain cases [6, 16], but most of these are concerned with achieving good asymptotic results as q goes to infinity, and there are still discrepancies between the absolute upper and lower bounds for most choices of parameter. In the case where $l \leq c$, however, precise results are known.

Theorem 2.12. *[6] There exist q-ary, length l, c-frameproof codes of size $l(q-1)$ when $q \geq 2$ and $2 \leq l \leq c$, but no larger code with these parameters is possible.*

An upper bound on the size of such codes is provided by Theorem 2.8. The lower bound required to prove this theorem comes from the following construction, which yields codes that are c-frameproof for any $c \geq 2$, regardless of their length.

Construction 2.13. *Let $Q = \{0, 1, \ldots, q-1\}$. For $i = 1, 2, \ldots, l$ define the set C_i by $C_i = \{x \in Q^l | x_i \neq 0$ and $x_j = 0$ for $j \neq i\}$. Let $C = \bigcup_{i=1}^{l} C_i$. The sets C_i are disjoint and have size $(q-1)$, therefore the size of C is $l(q-1)$. We claim that C is a c-frameproof code for any c with $2 \leq c \leq l(q-1)-1$.*

Proof. This construction appears as Construction 1 in Blackburn [6]. A word

$x \in C_i$ has a symbol in position i that is shared by no other word in C. Therefore no combination of other words can frame x, since no descendent of any set of other words matches x in position i. As each word lies in C_i for some i it follows that no word can be framed by a set of other words, no matter what the size. Hence C is indeed c-frameproof for any $c \geq 2$. □

Example 2.6 Let $Q = \{0, 1, 2, 3\}$. Construction 2.13 can be used to obtain the following length 4 code of size $4(4 - 1) = 12$.

$$
\begin{array}{c}
(1, 0, 0, 0) \\
(2, 0, 0, 0) \\
(3, 0, 0, 0) \\
(0, 1, 0, 0) \\
(0, 2, 0, 0) \\
(0, 3, 0, 0) \\
(0, 0, 1, 0) \\
(0, 0, 2, 0) \\
(0, 0, 3, 0) \\
(0, 0, 0, 1) \\
(0, 0, 0, 2) \\
(0, 0, 0, 3)
\end{array}
$$

■

We have now seen how traceability codes and frameproof codes can be applied in an effort to thwart piracy. In the following chapters we will see situations in which these schemes cannot be applied successfully; new approaches will have to be adopted as a result. Nevertheless we will see that some of the new schemes relate in substantial ways to those we have seen in this chapter.

Chapter 3

Piracy Prevention in a Dynamic Setting

In the previous chapter we discussed schemes for preventing piracy of pay-TV broadcasts. By controlling how decryption keys are allocated to users we are able to trace pirate decoder boxes back to the users responsible, thereby discouraging traitors from giving away copies of their keys. A traitorous user may decide, however, to use his or her keys to decrypt the program and then rebroadcast it in the clear instead, essentially acting as a pirate TV station. If this occurs then we can no longer use the key distribution as a means of tracing the culprits. Ideally we would like to be able to gain information about the traitors from the pirate broadcast: this suggests that it might be useful to adapt the fingerprinting techniques discussed previously for use in a broadcast situation. This idea was first proposed by Fiat and Tassa in [12]. In the following sections we describe this dynamic model and discuss schemes for tracing traitors within this model. We will refer to previously-discussed scheme as *static* schemes to distinguish them from the dynamic ones.

3.1 The Dynamic Model

The basic scenario in which we are interested involves valuable digital data, such as the output of a pay-TV station, being broadcast continuously to a set of users $U = \{u_1, u_2, \ldots, u_n\}$ who pay to receive it. A *pirate* is a set $T = \{t_1, t_2, \ldots, t_c\} \subset U$ of users, known as *traitors*, that illegally rebroadcasts the data. Fiat and Tassa, in their paper on dynamic traitor tracing [12], were the first to apply fingerprinting techniques similar to those discussed in Section 2.1.1 in such a setting. As in the static case the content, such as a movie or TV program, can be split into sections each of which might correspond to a few minutes of the program. We assume that it is possible to mark each segment in a robust way, with the presence of the marks being undetectable by the viewer. If q differently-marked variants of each segment are produced we can think of them as corresponding to letters of an alphabet of size q. We then wish to distribute differently-marked variants to different users; in the following sections we discuss schemes for determining how to distribute the marks. As for the distribution itself, Fiat and Tassa suggest a couple of ways in which this might be carried out. Their first suggestion requires that each user share a unique symmetric key with the broadcaster. For segment t of the program, version i of that segment is encrypted with a randomly-chosen key k_t^i, for each $i = 1, 2, \ldots, q$. The key k_t^i is then individually encrypted and sent to each user who is to receive version i of this segment, then at the appropriate time all q differently-encrypted versions of the segment are broadcast. This ensures that each user can only decrypt the marked version that is intended for them. Fiat and Tassa mention that broadcast encryption could be used to send the appropriate keys to the users, thereby eliminating the need to send individual messages to each user; see [12] for other suggestions in making this

distribution process more efficient.

We have a set $U = \{u_1, u_2, \ldots, u_n\}$ of n users, and a marking alphabet Q of size q. During time segment j user u_i is allocated a segment with the mark $m_{ij} \in Q$. At this time the pirate can only broadcast a version of segment j marked with $m_{t_j j}$ for some $t_j \in T$.

Definition 3.1. *A* pirate broadcast sequence *corresponding to a pirate T is a sequence $\{\xi_j\}_{j=1}^{i}$ where for each $j \in \{1, 2, \ldots, i\}$ we have $\xi_j = m_{t_j j}$ for some $t_j \in T$.*

Such a sequence represents a possible output by the pirate between times 1 and i.

Example 3.1 Suppose we have four users $\{u_1, u_2, u_3, u_4\}$ and an alphabet $Q = \{a, b, c\}$, and that users u_3 and u_4 decide to collude in piracy. If the marks on the first three segments are distributed as in the following table:

	1	2	3
u_1	a	b	c
u_2	a	c	a
u_3	a	b	b
u_4	b	a	b

then the possible pirate broadcast sequences corresponding to $T = \{u_3, u_4\}$ are:

a, b, b
a, a, b
b, b, b
$b, a, b.$

■

A mark that is received by precisely one user at some time is referred to as a *unique mark*. If the symbol broadcast by the pirate at any time is a unique mark then we can deduce the identity of one of the traitors, as only one user is capable of contributing that mark to the pirate broadcast.

Producing and then broadcasting a large number of different variants is very expensive, so ideally q is kept as small as possible. The number of segments required to implement a particular scheme relates directly to the time that it takes to run, so schemes which require fewer segments are more efficient. In the following sections we consider specific traitor-tracing schemes that work in this dynamic setting, and we discuss their efficiency based on these considerations.

3.2 Sequential Traitor Tracing

In a static traitor-tracing scheme marks are distributed to the users, then the marks from pirate copies are used to trace the traitors. In the dynamic model there is the potential for feedback from the pirate broadcast to be used in determining the mark distribution, as well as in the tracing. The pirate could attempt to thwart this, however, by delaying the start of the pirate broadcast until the legitimate broadcast containing the mark distribution was completed. This is referred to as the *delayed rebroadcast attack*, and *sequential traitor-tracing schemes* were proposed by Safavi-Naini and Wang to counteract it [13]. In their conception of sequential traitor-tracing schemes the marks are distributed according to a predetermined *mark allocation table*, and tracing commences as soon as information from the pirate broadcast is received; here we restrict our attention to the case in which this occurs after the mark distribution is complete. In constructing sequential traitor-tracing schemes we assume that the pirate coalition has size at most c for some known integer c.

The mark allocation table is an $n \times l$ array $M = \left(m_{ij}\right)$ with entries from the mark alphabet Q, where n is the number of users, and l is the number of time segments needed to implement the scheme. The marks are then distributed

to the users according to this table, with user i receiving the symbol m_{ij} at time j. The rows of the matrix M can be thought of as the words of a q-ary, length l code C and the corresponding pirate broadcast will be a word in Q^l that is a descendent of the set of at most c words belonging to the traitors. In order for a matrix to be the mark allocation table of a sequential traitor-tracing scheme we require that there exists a deterministic algorithm taking as input a pirate broadcast sequence coming from a set of c or fewer traitors and outputting the identity of at least one of those traitors. (Note that in general it is only possible to guarantee the identification of one of the traitors, since the pirate set can always elect to broadcast the versions corresponding to a single traitorous user.) This requirement places a condition on the code C that is equivalent to C being a *c-IPP code*.

Definition 3.2. *Let $C \subset Q^l$ be a code. A set $S \in C$ is a* parent set *of a word $x \in Q^l$ if $x \in \mathrm{desc}(S)$. Denote the set of all parent sets of x of size less than or equal to c by $\mathcal{H}_c(x)$; thus $\mathcal{H}_c(x) = \{S \subseteq C || S| \leq c,\ x \in \mathrm{desc}(S)\}$. We say that the code C is a c-IPP code if for all words $x \in Q^l$ either $\mathcal{H}_c(x) = \emptyset$, or*

$$\bigcap_{S \in \mathcal{H}_c(x)} S \neq \emptyset.$$

If a mark allocation table is a c-IPP code then given any pirate broadcast Ξ there exists at least one user u who is a member of every set of c or fewer users capable of producing Ξ, which implies that u is necessarily a traitor. If, however, the code does not have the c-IPP property then there exists a potential pirate broadcast $x \in Q^l$ that will not allow us to trace any traitor, since for each user u capable of having contributed to x there exists a pirate set $T \subset U \setminus \{u\}$ of size at most c that could also have produced x.

We have already seen examples of c-IPP codes:

Theorem 3.3. *[14] Every c-TA code is a c-IPP code.*

Proof. Let $C \subset Q^l$ be a c-TA code. If $x \in \text{desc}(S)$ for some $S \subset C$ containing at most c words (i.e. if $\mathcal{H}_c(x) \neq \emptyset$) then the (nonempty) set Y of codewords closest to x is contained in S. As this is true for any parent S of x, so Y is contained in the intersection of all parent sets S with $|S| \leq c$, hence this intersection is nonempty. Therefore C satisfies the c-IPP property. □

There exist c-IPP codes that are not c-TA codes (see, for example, [14]). What the c-TA property provides, however, is a natural algorithm for finding a parent: simply take the closest codeword(s). Using a c-TA code as a mark allocation table thus yields a scheme capable of tracing at least one traitor.

Once one traitor has been traced and disconnected, a similar scheme assuming the existence of at most $c - 1$ traitors can be run in order to catch another traitor and so on in sequence, hence the name sequential.

As the mark allocation table of a sequential traitor-tracing scheme corresponds to a c-IPP code any bounds on the number of words in such a code will serve to bound the number of users that can be accommodated by a sequential traitor-tracing scheme. The following bound is due to Alon and Stav [1].

Theorem 3.4. *[1] Define*

$$s(c) = \begin{cases} \frac{c^2}{4} + c & \text{when } c \text{ is even,} \\ \frac{c^2}{4} + c - \frac{1}{4} & \text{when } c \text{ is odd.} \end{cases}$$

A q-ary, length l, c-IPP code has at most $s(c)q^{\lceil \frac{l}{s(c)} \rceil}$ codewords.

This allows us to bound the minimum l for which there exists a sequential traitor-tracing scheme that supports n users and uses l segments.

Corollary 3.5. *A sequential traitor-tracing scheme that supports n users and assumes the existence of at most c traitors requires the use of at least*

$s(c)\log_q \frac{n}{s(c)} - s(c) + 1$ *segments for its implementation.*

Proof. We know that the words of the mark allocation table form a c-IPP code, and therefore by Theorem 3.4 we have $n \leq s(c)q^{\lceil \frac{l}{s(c)} \rceil}$. From this we deduce that

$$\log_q \frac{n}{s(c)} \leq \left\lceil \frac{l}{s(c)} \right\rceil.$$

Therefore

$$\begin{aligned} s(c)\log_q \frac{n}{s(c)} &\leq s(c) \left\lceil \frac{l}{s(c)} \right\rceil \\ &\leq l + s(c) - 1. \end{aligned}$$

The result follows directly. □

We see therefore that to implement a sequential traitor-tracing scheme requires around $\frac{c^2}{4}\log_q \frac{4n}{c^2}$ or more segments to catch one traitor; when $c = 2$, for instance, this is approximately $\log_q n$.

3.3 Dynamic Traitor Tracing

It is through schemes such as dynamic traitor-tracing schemes that the dynamic model really comes into its own. In these schemes the feedback from the pirate broadcast sequence is used not only for traitor tracing but also for determining the mark allocation, which is altered on the fly in response to the pirate's broadcast. This extra information means that dynamic traitor tracing can be carried out in a much shorter time than sequential tracing. Furthermore, unlike the sequential schemes these schemes do not require prior knowledge of the number of traitors. It is with such schemes in mind that the dynamic model was first proposed by Fiat and Tassa [12]. They define a watermarking scheme to be *deterministic* if it traces all traitors without falsely

incriminating any innocent users; the following example of such a scheme comes from their paper [12].

Construction 3.6. *[12] This algorithm requires the use of $c+1$ different variants and runs in time at most $\binom{n}{c}+2\sum_{t=0}^{c-1}\binom{n}{t}$. Note that prior knowledge of the value of c is not assumed.*

Suppose there exists a set U of n users and a mark alphabet $Q \subseteq \{1,2,\ldots,n\}$. Proceed as follows.

1. *Set $t=1$.*

2. *(a) Choose a set $W=\{w_1,w_2,\ldots,w_t\} \subset U$ of t users, give user w_i the mark i, and all users in $U \setminus W$ the mark 0. Unless situation 2b occurs, choose another set of t users and distribute the marks similarly. Repeat until each t-subset of users has been chosen once, then increase t by one and proceed as before.*

   (b) If the pirate broadcasts a unique mark then the corresponding user is guilty. Disconnect that user, subtract one from the value of t and continue from step 2a.

3. *When the final traitor is eliminated the value of t becomes 0 and piracy ceases. The algorithm terminates at this point.*

The goal of this algorithm is to assign unique marks to all the traitors, thus forcing one of them to incriminate his or herself. The parameter t acts as a lower bound on the number of traitors: initially set to 1 it is gradually increased until a traitor is caught, at which point it decreases by 1. We note that the value of t never exceeds c, for if $t=c$ one of the c-subsets of U is the pirate, T. When this subset is chosen, each traitor receives a unique mark,

hence the pirate broadcast is necessarily a unique mark that incriminates one of the traitors, and t decreases to $c-1$. As only $c-1$ traitors remain so one of the choices of $(c-1)$-subset will result in all remaining traitors receiving unique marks, and so on. Thus this algorithm traces all traitors yet never requires more than $c+1$ differently-marked variants.

Example 3.2 The following table demonstrates a possible run of this algorithm for four users $u_1, u_2, \ldots, u_4$, where users u_1 and u_3 are pirates.

	1	2	3	4	5	6	7	8	9
u_1	1	0	0	0	1	**1**			
u_2	0	1	0	0	2	0	1	0	
u_3	0	0	1	0	0	2	0	**1**	
u_4	0	0	0	1	0	0	0	0	
T	0	0	0	0	0	**1**	0	**1**	
t	1	1	1	1	2	2	1	1	0

During the first four time segments the value of t is one; as no traitor is uncovered in that time t is increased to two, and sets of two users are given unique marks at each time. At time 6 user u_1 is incriminated and subsequently disconnected; the value of t is decreased to one. The remaining traitor u_3 is incriminated at time 8. ■

In devising dynamic traitor-tracing schemes we wish to minimise both the number of marks needed and the implementation time. The above scheme catches c traitors with the aid of $c+1$ marks; in fact it is impossible to reduce this figure any further.

Theorem 3.7. *[12] A deterministic watermarking scheme that can trace a set of c colluding traitors requires the use of a marking alphabet of size at least $c+1$.*

Proof. If every member of a pirate coalition is assigned a unique mark at some time then the pirate has no choice but to broadcast one of those marks and the

corresponding traitor is thus incriminated. If, however, some traitor receives a mark that is shared with another user then by broadcasting this mark the pirate can avoid incrimination. If there are c colluding traitors but only c symbols available then, by the pigeon-hole principle, at each time some traitor receives a symbol that is also received by another user. By broadcasting this symbol the pirate can avoid detection in each time segment. Hence a watermarking scheme using c or fewer symbols cannot be deterministic. □

The scheme described in Construction 3.6 demonstrates the existence of dynamic traitor-tracing schemes using the minimal number $c + 1$ of symbols; its running time, however, is exponential in c. In [12] Fiat and Tassa present two additional schemes, one running in time $O(c \log n)$ but requiring $2c + 1$ symbols, and one using $c+1$ symbols with a running time of $O(3^c c \log n)$: still exponential in c, but an improvement on the initial scheme. They pose the problem of whether there exists a deterministic scheme using $c+1$ symbols that runs in a time polynomial in c. This question is answered in the affirmative by Berkman, Parnas and Sgall in [4], who provide an algorithm using $c + 1$ variants and requiring time $\Theta(c^2 + c \log n)$. They show that this running time is optimal.

We have seen that it is possible to find algorithms for tracing traitors that are polynomial in the number c of traitors provided that at least $c + 1$ differently-marked variants are used. If the number of traitors is high, however, the broadcaster may not have sufficient resources to implement such a scheme. Nevertheless, the broadcaster may still wish to make use of dynamic digital fingerprinting in some fashion. We have seen how static c-TA codes relate to sequential traitor-tracing schemes, and how dynamic schemes can be used to provide even more-efficient traitor tracing. In the static case we

have also seen how c-frameproof codes can be used to ensure innocent users are not falsely incriminated by pirates, without actually having to trace the traitors responsible. In the following chapters we will see how similar ideas can be translated into the dynamic setting, thus providing protection for innocent users in situations where there are insufficient resources to implement a deterministic traitor-tracing scheme.

Chapter 4

Sequential Frameproof Codes

In Chapter 3 we saw how the concept of traitor tracing applied in a dynamic context lead to the development of sequential and dynamic traitor-tracing schemes. If you are lacking the resources to trace traitors but still wish to prevent pirates from framing innocent users you might hope to translate the concept of frameproof codes in a similar fashion to the dynamic setting. In this chapter we explore how to prevent framing without recourse to the information contained in the pirate's broadcast. We begin by proposing a definition for *l-sequential c-frameproof codes*, which prevent framing by up to c traitors over l consecutive time segments, then describe closely-related functions that can be used to simplify their construction. We then show that in fact sequential c-frameproof codes are closely connected to ordinary c-frameproof codes, with the existence of one implying the existence of the other and *vice versa.* Finally we consider sequential frameproof codes that protect users from any number of pirates. We provide a construction of such an l-sequential code that uses an alphabet of size q to protect n users with $l = \left\lceil \frac{n}{q-1} \right\rceil$ and show that this is optimal, in that it is not possible to protect n users when $l < \left\lceil \frac{n}{q-1} \right\rceil$.

4.1 Definitions

In a sequential traitor-tracing scheme code marks are broadcast one at a time and the information contained in the pirate broadcast is used to disconnect traitorous users, hence the set of users decreases with time.

In the context of frameproof codes, however, our goal is somewhat different. In this instance we do not seek to identify the traitors from their broadcast, rather we wish solely to prevent the pirate from broadcasting a sequence of segments corresponding to that allocated to some innocent user. We propose the following definition of a sequential frameproof code:

Definition 4.1. *An* l-sequential c-frameproof code *is a function* $\mathcal{M}$ *mapping* $\mathbb{N}^+ \times U$ *to* Q *with* $(j, u) \mapsto M_j(u)$*, such that for any pirate* T *with* $|T| \leq c$*, and for any sequence of marks* $\{\xi_j\}_{j=i}^{i+l-1}$ *broadcast by that pirate over* l *consecutive time segments, there is no legitimate user* $u \in U \setminus T$ *with* $M_j(u) = \xi_j$ *for all* $j = i, i+1, \ldots, i+l-1$*.*

During time section j the function $\mathcal{M}$ assigns to user u the segment marked with mark $M_j(u)$; the sequential c-frameproof property ensures that over the course of any l consecutive time segments the sequence of marked segments broadcast by any pirate T with $|T| \leq c$ will differ from that allocated to any innocent user. We refer to this as the *l-sequential c-frameproof condition.* If $\mathcal{M}$ is an l-sequential c-frameproof code for all $c \geq 1$ then we refer to it simply as an l-sequential frameproof code. The integer l will be known as the *convergence time* of the code.

If at some time t a user u receives a mark received by no other user at that time we refer to this as a *unique* mark. If the pirate broadcast ξ_t is a unique mark then we know that the user who received the mark must be part of the

pirate coalition.

Example 4.1 Let $U = \{u_1, u_2, \ldots u_n\}$ and $Q = \{1, 2, \ldots, n\}$, and define $\mathcal{M}$ by setting $\mathcal{M}(i, u_j) = j$ for all $i \in \mathbb{N}^+$. Then $\mathcal{M}$ is an l-sequential frameproof code for any $l \geq 1$, since the fact that no two users get the same mark at any time means that no user can be framed during any time interval. ■

The code described above is rather trivial, and has a particularly simple description. In general we might expect to have more difficulty in describing the function $\mathcal{M}$. Our task is made somewhat easier, however, by observing that we can construct an appropriate function $\mathcal{M}$ by simply specifying the values that it takes on the first l time segments.

Lemma 4.2. *Suppose there exists a user set U of size n, a mark alphabet Q, and a function $f_l^c \colon \{1, 2, \ldots, l\} \times U \to Q$. Then f_l^c can be thought of as a function for distributing marked segments for the first l time intervals of a broadcast and it is possible to extend it to a function $\mathcal{M} \colon \mathbb{N}^+ \times U \to Q$ as follows. Given f_l^c let $\mathcal{M} \colon \mathbb{N}^+ \times U \to Q$ be defined by setting $\mathcal{M}(i, u) = f_l^c(i', u)$, where i' is the unique element of $\{1, 2, \ldots, l\}$ with $i \equiv i' \pmod{l}$. If f_l^c obeys the l-sequential c-frameproof condition over this time then $\mathcal{M}$ is an l-sequential c-frameproof code.*

Proof. Consider the segments broadcast between times i and $i + l - 1$. Suppose there is a pirate T of size at most c capable of framing some innocent user $u \in U \setminus T$ over this interval, so that for each $j \in \{i, i + 1, \ldots, i + l - 1\}$ there exists $t_j \in T$ such that $m_j(u) = m_j(t_j)$. By the definition of $\mathcal{M}$, we have $m_j(u) = m_{j'}(u)$ whenever $j \equiv j' \pmod{l}$. As such, for each time $j' \in \{1, 2, \ldots l\}$, there exists $j \in \{i, i + 1, \ldots, i + l - 1\}$ for which we have $m_{j'}(u) = m_j(u) = m_j(t_j) = m_{j'}(t_j)$. It follows that the pirate T can frame

user u in the first l time segments, which contradicts the assumption that f_l^c has the sequential c-frameproof property in this time, since for $j \in \{1, 2, \ldots l\}$ we have $m_j(v) = f_l^c(j, v)$ for any $v \in U$. □

Thanks to this lemma we know that to construct l-sequential c-frameproof codes we need only define their behaviour over the first l time segments. We will make use of this fact in subsequent constructions, including our demonstration of how sequential c-frameproof codes can be constructed from ordinary c-frameproof codes.

4.2 The Connection Between c-Frameproof Codes and Sequential c-Frameproof Codes

We have defined sequential c-frameproof codes, but as yet have seen few examples. It turns out, however, that they are essentially familiar objects under a new guise: since neither the set of users, nor the allocation of marked segments are affected by the pirate broadcast the sequential setting is in fact closely related to the static case, as detailed in the following lemmas. First we show that by restricting an l-sequential c-frameproof code to any window of l consecutive time segments we can obtain an ordinary c-frameproof code.

Theorem 4.3. *Suppose $\mathcal{M}$ is an l-sequential c-frameproof code over an alphabet Q of size q protecting a set U of users with $|U| = n$. Fix some integer $j \geq 1$ and associate a word $x_u = \big(M_j(u), M_{j+1}(u), \ldots, M_{j+l-1}(u)\big) \in Q^l$ with each user $u \in U$. Then the set $\Gamma = \{x_u | u \in U\} \subset Q^l$ is a length l c-frameproof code over Q of size n.*

Proof. By definition Γ is a length l code over Q. There is a one-to-one correspondence between users $u \in U$ and words $x_u \in \Gamma$ since if there are two users

$u, v \in U$ with $u \neq v$ but $x_u = x_v$ then for each $i = j, j+1, \ldots, j+l-1$ we have $M_i(u) = M_i(v)$. This implies that u is capable of creating a pirate broadcast with marks $M_i(u) = M_i(v)$ for all $i = j, j+1, \ldots, j+l-1$ and hence framing v over the length l window starting at time j, thereby contradicting the sequential c-frameproof property of $\mathcal{M}$. Thus we conclude that the number of codewords in Γ is n.

We now proceed to prove that Γ is c-frameproof. Suppose there exists a word $x_u \in \Gamma$ and set $S \subseteq \Gamma$ with $|S| \leq c$ such that $x_u \in \mathrm{desc}(S)$. Let $T \subseteq U$ be the set of users in U corresponding to the words of S. For each $i = j, j+1, \ldots, j+l-1$ we know that $x_u^i = x_{t_i}^i$ for some $t_i \in T$, which implies that $M_i(u) = M_i(t_i)$. If the users of T decide to commit piracy, they have the capacity to broadcast the segments with sequence of marks $\{M_i(t_i)\}_{i=1}^l$. As these each agree with the marked segments given to u this contradicts the sequential c-frameproof property of $\mathcal{M}$ unless $u \in T$. Hence we conclude that $u \in T$ and thus $x_u \in S$. It follows that Γ is a c-frameproof code. $\square$

We see therefore that the output of a sequential c-frameproof code over any window of l consecutive time segments gives rise in a natural fashion to an ordinary c-frameproof code. By Lemma 4.2 we know that we can construct a sequential c-frameproof code by specifying its behaviour over the first l segments; this suggests that it may be possible to construct sequential c-frameproof codes based on ordinary c-frameproof codes. Indeed this is the case, and we describe this construction explicitly in the following theorem.

Theorem 4.4. *Suppose $\Gamma = \{x_1, x_2, \ldots x_n\} \subset Q^l$ is a c-frameproof code. We define a function $f_l^c \colon \{1, 2, \ldots, l\} \times U \to Q$, where $U = \{u_1, u_2, \ldots u_n\}$, by setting $f_l^c(i, u_j) = x_j^i$ for $i = 1, 2, \ldots, l$ and $j = 1, 2, \ldots, n$. The function $\mathcal{M} \colon \mathbb{N}^+ \times U \to Q$ obtained by extending f_l^c as in Lemma 4.2 is an l-sequential*

c-frameproof code.

Proof. We prove this lemma by showing that the function f_l^c satisfies the l-sequential c-frameproof condition over the first l time segments; the result then follows by Lemma 4.2. Suppose there exists a pirate set $T \subset U$ with $|T| \leq c$ capable of broadcasting segments with mark sequence $\{\xi_i\}_{i=1}^l$ and an innocent user $u_k \in U \setminus T$ with $f_l^c(i, u_k) = \xi_i$ for all $i = 1, 2, \ldots l$. Let $S = \{x_j \in \Gamma | u_j \in T\}$. For each $i = 1, 2 \ldots l$,

$$\begin{aligned} x_k^i &= f_l^c(i, u_k) \\ &= \xi_i \\ &= f_l^c(i, u_j) \text{ for some } u_j \in T \\ &= x_{u_j}^i \text{ where } x_j \in S. \end{aligned}$$

Hence the set S is capable of framing x_k, which implies that $x_k \in S$, and thus $u_k \in T$, thereby contradicting the assumption that $u_k \in U \setminus T$. Therefore no such T and u_k exist, so f_l^c is sequential c-frameproof on the first l time segments, whence the associated $\mathcal{M}$ is an l-sequential c-frameproof code by Lemma 4.2. □

Thus we see that known examples of c-frameproof codes can be effectively translated into the dynamic setting to yield sequential c-frameproof codes. The following example illustrates how this works in practice.

Example 4.2 Example 2.4 contained a binary, length 3, 2-frameproof code Γ with

$$\Gamma = \{x_0 = (0, 0, 0), x_1 = (1, 1, 0), x_2 = (0, 1, 1), x_3 = (1, 0, 1)\}.$$

As Γ contains four words, it can be turned into a 3-sequential 2-frameproof code for four users. Let $U = \{u_0, u_1, u_2, u_3\}$ and $Q = \{0, 1\}$. Define a

function $f_3^2 \colon \mathbb{N}^+ \times U \to Q$ by setting $f_3^2(i, u_j) = x_j^{i'}$, where $i' \in \{1, 2, 3\}$ and $i' \equiv i \pmod 3$. Using Lemma 4.2 we can extend f_3^2 to a 3-sequential 2-frameproof code $\mathcal{M}$. The following table indicates how the marks would be distributed according to $\mathcal{M}$ over the first nine time segments.

	1	2	3	4	5	6	7	8	9
u_0	0	0	0	0	0	0	0	0	0
u_1	1	1	0	1	1	0	1	1	0
u_2	0	1	1	0	1	1	0	1	1
u_3	1	0	1	1	0	1	1	0	1

If we consider any three consecutive time segments we observe that the marks received by each user correspond to words in Q^3 that are either the words of Γ, or a cyclic shift of those words. Such a shift does not affect the 2-frameproof property that Γ possesses therefore it is not possible for any two colluding users to frame a third user over any length 3 window of consecutive segments. Thus we conclude that $\mathcal{M}$ is indeed a 3-sequential 2-frameproof code. ■

Theorems 4.3 and 4.4 provide us with a description of the structure of an l-sequential c-frameproof code, namely that the sequences of marks distributed to each user correspond to successive c-frameproof codes of length l. In the codes constructed according to Theorem 4.4 the c-frameproof codes derived from the various length l windows of the l-sequential c-frameproof code are essentially all equivalent. This is not always true of the c-frameproof codes arising from a sequential c-frameproof code, as the following example demonstrates; however, sequential codes of this form benefit from being easier to describe and construct.

Example 4.3

	1	2	3	4	5	6	
u_0	0	0	0	0	0	0	$\cdots$
u_1	1	1	1	1	1	1	$\cdots$
u_2	0	1	2	3	3	3	$\cdots$
u_3	3	0	2	4	4	4	$\cdots$

The table above shows the mark distribution over six time segments of a 5-ary, 3-sequential 2-frameproof code protecting four users. Inspection of the symbols received by any two users over any three consecutive segments reveals that at most one of these symbols is common to both users. Hence at least three users are required to frame a fourth over three consecutive segments. The 2-frameproof code corresponding to the first three segments has minimum distance two and thus differs from that corresponding to segments 4, 5 and 6, which has a minimum distance of three. ■

The fact that there is this close connection between sequential frameproof codes and ordinary frameproof codes means that we can apply known results about ordinary frameproof codes to the case of the sequential frameproof codes. In particular we are interested in finding the minimum l for which there exists an l-sequential c-frameproof code protecting n users or, conversely, finding the maximum number of users that can be supported by an l-sequential c-frameproof code. The connection with ordinary frameproof codes means that bounds on the size of these codes lead directly to bounds on the number of users supported by the corresponding sequential frameproof codes.

Example 4.4 Suppose we have an alphabet of size 5, and wish to construct an l-sequential 4-frameproof code protecting sixteen users with as small a value of l as possible. Construction 2.13 yields a q-ary c-frameproof code of cardinality $l(q-1)$ provided that $l \leq c$. Thus it is possible to construct a 5-ary, length 4, 4-frameproof code containing sixteen words, for example. This in turn gives rise to a 5-ary 4-sequential 4-frameproof code protecting sixteen users. Furthermore, by Theorem 2.12 there can be no 5-ary 3-sequential 4-frameproof code protecting sixteen users, as we know that the maximum possible size for a length 3, 4-frameproof code is $3(q-1) = 12$. Thus we see that the

4-sequential 4-frameproof code derived via Construction 2.13 has the minimum possible convergence time for a sequential 4-frameproof code protecting sixteen users. ■

In Section 2.3 we described the known bounds on the sizes of frameproof codes. For many choices of parameter, however, precise bounds are still not known. One case in which more-precise results are available is the case where the number of traitors is not bounded; essentially this is equivalent to c being equal to $n-1$. In the following section we show how to construct l-sequential $(n-1)$-frameproof codes supporting an optimal number of users.

4.3 l-Sequential $(n-1)$-Frameproof Codes

In the case where there is no restriction on the number of traitors (essentially when $c = n-1$) precise results are known about the maximal sizes of c-frameproof codes. In what follows we describe the construction of a $\lceil \frac{n}{q-1} \rceil$-sequential $(n-1)$-frameproof code, and then use known bounds on the size of the related ordinary $(n-1)$-frameproof codes to show that the resulting code has the minimum possible convergence time given the number of users and the alphabet size. We begin with a lemma that will be used to motivate our construction.

Lemma 4.5. *Let $\mathcal{M}\colon \mathbb{N}^+ \times U \to Q,\ (i,u) \mapsto M_i(u)$, let $u \in U$ and fix an integer $j \geq 1$. Suppose that for each $i = j, j+1, \ldots j+l-1$ there exists a user $t_i \in U \setminus \{u\}$ with $M_i(u) = M_i(t_i)$. Then $\mathcal{M}$ is not an l-sequential frameproof code.*

Proof. Suppose $T = \{t_j, t_{j+1}, \ldots, t_{j+l-1}\}$ is a pirate. Then T can broadcast the sequence $\{\xi_i\}_{i=j}^{j+l-1}$ with $\xi_i = M_i(t_i)$. But then for each $i = j, j+1, \ldots, j+l-1$

we have $M_i(u) = \xi_i$; hence T can frame u, and thus $\mathcal{M}$ is not an l-sequential l-frameproof code. This implies that $\mathcal{M}$ is not an l-sequential frameproof code either. □

Thus we see that in order for a mark distribution to be an l-sequential frameproof code it is necessary for each user to receive a unique mark at some time in every length l window, for otherwise they risk being framed. This requirement gives rise to a natural construction of a sequential $(n-1)$-frameproof code.

Construction 4.6. *Let* $Q = \{0, 1, \ldots q-1\}$, *and* $U = \{u_0, u_1, \ldots u_{n-1}\}$.
Define $f_{\lceil \frac{n}{q-1} \rceil} : \{1, 2, \ldots, \lceil \frac{n}{q-1} \rceil\} \times U \to Q$ *by setting*

$$f_{\lceil \frac{n}{q-1} \rceil}(i, u_j) = \begin{cases} j + 1 - (q-1)(i-1) & \text{if } (q-1)(i-1) \leq j \leq (q-1)i - 1, \\ 0 & \text{otherwise.} \end{cases}$$

Extending $f_{\lceil \frac{n}{q-1} \rceil}$ *with the aid of lemma 4.2 to a function* $\mathcal{M}$ *results in a* $\lceil \frac{n}{q-1} \rceil$*-sequential* $(n-1)$*-frameproof code.*

This description of $f_{\lceil \frac{n}{q-1} \rceil}$ is perhaps deceptively complicated; it can be more clearly illustrated by an example.

Example 4.5 Suppose you have ten users $u_0, u_1, \ldots, u_9$ and an alphabet $\{0, 1, 2, 3\}$. Then $\lceil \frac{n}{q-1} \rceil = 4$, and the function f_4 resulting from the above construction distributes marks to the users as in the following table.

	1	2	3	4
u_0	1	0	0	0
u_1	2	0	0	0
u_2	3	0	0	0
u_3	0	1	0	0
u_4	0	2	0	0
u_5	0	3	0	0
u_6	0	0	1	0
u_7	0	0	2	0
u_8	0	0	3	0
u_9	0	0	0	1

We see that each user receives a unique mark in one of these four segments; extending f_4 to $\mathcal{M}$ will thus result in a mark distribution with the property that every user receives a unique mark at some time during every length 4 window. ■

Knowing that each user receives a unique mark in each window of length $\lceil \frac{n}{q-1} \rceil$ we see that $\mathcal{M}$ is indeed a $\lceil \frac{n}{q-1} \rceil$-sequential $(n-1)$-frameproof code.

Proof. The function $f_{\lceil \frac{n}{q-1} \rceil}$ allocates a unique mark to the first $q-1$ users during the first time segment, the next $q-1$ users during the second time segment and so on, until each user has received a unique mark at some time. This procedure takes $\lceil \frac{n}{q-1} \rceil$ time intervals; $f_{\lceil \frac{n}{q-1} \rceil}$ is then extended to a sequential frameproof code as usual. Since each user has been allocated a unique mark at some time during every interval of $\lceil \frac{n}{q-1} \rceil$ time segments no user can be framed by a pirate coalition irrespective of its size. □

The above construction shows that for an l-sequential $(n-1)$-frameproof code protecting n users the minimum possible convergence time is at most $\lceil \frac{n}{q-1} \rceil$. In order determine this minimum value exactly, we turn our consideration to ordinary $(n-1)$-frameproof codes. The $(n-1)$-frameproof code obtained from the sequential $(n-1)$-frameproof code above as in Theorem 4.3 corresponds to a subset of the $(n-1)$-frameproof code given in Example 2.6. Theorem 2.12 shows that there is no larger q-ary, length l, $(n-1)$-frameproof code, which leads us to the following theorem.

Theorem 4.7. *The $\lceil \frac{n}{q-1} \rceil$-sequential $(n-1)$-frameproof code of Construction 4.6 has an optimal convergence time, in the sense that there does not exist a sequential $(n-1)$-frameproof code protecting n users with a convergence time less than $\lceil \frac{n}{q-1} \rceil$.*

Proof. Suppose there exists a $\left(\left\lceil \frac{n}{q-1} \right\rceil - 1\right)$-sequential $(n-1)$-frameproof code protecting n users. The q-ary, length $\left\lceil \frac{n}{q-1} \right\rceil - 1$, $(n-1)$-frameproof code derived from the first $\left\lceil \frac{n}{q-1} \right\rceil - 1$ time segments of the sequential code has size n. According to Theorem 2.12, however, the maximum possible size of a $(n-1)$-frameproof code of these parameters is in fact

$$\begin{aligned}\left(\left\lceil \frac{n}{q-1} \right\rceil - 1\right)(q-1) &= (q-1)\left\lceil \frac{n}{q-1} \right\rceil - (q-1) \\ &< n + (q-1) - (q-1) \\ &= n.\end{aligned}$$

This results in a contradiction, so our original supposition must be false. Thus the smallest possible l for which there exists an l-sequential $(n-1)$-frameproof code is $\left\lceil \frac{n}{q-1} \right\rceil$, hence the code resulting from Construction 4.6 has an optimal convergence time. □

We have now seen several examples of how we can make use of the connection between sequential c-frameproof codes and ordinary c-frameproof codes. Essentially we can regard sequential c-frameproof codes to be a useful new manifestation of c-frameproof codes that enables their framing-preventing properties to be extended to the dynamic model. We have seen examples of choices of parameters q, c and n for which we know the optimal convergence time l, although in the majority of cases there is still a discrepancy between the best known upper and lower bounds for the minimum possible l. In Chapter 3 we saw that dynamic traitor-tracing schemes proved to be considerably more efficient than sequential traitor-tracing schemes, which is to be expected given the extra information available. In the context of framing protection it is not unreasonable, therefore, to expect that dynamic schemes that make use of the information contained in the pirate's broadcast to prevent framing should be

able to achieve shorter convergence times than even the best sequential frame-proof codes. Indeed this is the case; such possibilities are explored in the two subsequent chapters.

Chapter 5

Dynamic Frameproof Codes

In Section 3.3 we saw how dynamic traitor-tracing schemes are capable of catching traitors more quickly than the corresponding sequential traitor-tracing schemes thanks to the ability to use the information provided by the pirate's broadcast in determining the mark distribution. In this chapter we show how this information can be used in the prevention of framing. We give a definition of *l-dynamic frameproof codes* that take l time segments to prevent framing by any number of traitors, and we provide a construction that is optimal with respect to the number of time segments required to implement it. We demonstrate that for n users and an alphabet of size q an l-dynamic frameproof code preventing framing by a coalition of up to c traitors exists for any $c > 1$ if and only if $n \leq q^{l-1}(q-1)$, and that a similar code preventing framing by a single traitor exists if and only if $n \leq q^l$.

5.1 Definitions

The dynamic setting differs from the sequential case in that we wish in this instance to make use of the information present in the pirate's broadcast. This should allow us to find more-efficient ways of distributing marks so as to prevent framing.

Suppose we have a set of users $U = \{u_1, u_2, \dots u_n\}$ and a mark alphabet Q of size q, and suppose there exists a pirate $T = \{t_1, t_2, \dots, t_c\} \subset U$. We wish to distribute marked segments as in the sequential case, only this time our distribution of marks at a particular time may depend on a pirate's previous broadcast. We assume that at any given time α we know the sequence $\{\xi_j\}_{j=1}^{\alpha-1}$ of marks previously broadcast by the pirate and we use this sequence to determine how to allocate marks to the users. The pirate T responds by broadcasting a marked segment received by one of the $t_j \in T$; this mark is then taken into account when distributing the marks at time $\alpha+1$, and so on.

Definition 5.1. *A sequence $\{\xi_j\}_{j=1}^{\alpha}$ is a* valid pirate broadcast sequence *for a particular mark distribution if there exists a set $T \subset U$ (the pirate) such that for all $i = 1, 2, \dots, \alpha$ the mark ξ_i was received by some user $t \in T$ and is not a unique mark.*

If the pirate broadcasts a unique mark we can take action against the user who received that mark and remove them from the set of users, hence weakening the pirate. As such a user is part of the guilty coalition they can not be said to have been framed. In defining dynamic frameproof codes we consider only valid pirate broadcast sequences, which ensures that framing can be prevented even when traitors do not incriminate themselves in this fashion. For the sake of brevity we represent the pirate sequence $\{\xi_j\}_{j=1}^{\alpha-1}$ by the word $\Xi_\alpha = (\xi_1, \xi_2, \dots \xi_{\alpha-1}) \in Q^{\alpha-1}$.

Definition 5.2. *An l-*dynamic *c-*frameproof code *is a finite family of functions $\{D_\alpha\}_{\alpha=1}^{l}$ where $D_1 \colon U \to Q$ and $D_\alpha \colon Q^{\alpha-1} \times U \to Q$ for $\alpha > 1$, with the property that for any valid pirate broadcast sequence $\{\xi_j\}_{j=1}^{l}$ corresponding to a pirate T with $|T| \leq c$ there is no user $u \in U \setminus T$ with $D_j(\Xi_j, u) = \xi_j$ for all*

$j = 1, 2, \ldots, l$.

At time $\alpha > 1$ the sequence of marks Ξ_α previously broadcast by the pirate is used as an input to the function D_α in order to determine how the marked segments are distributed among the users. Recall that at time α the pirate T is capable of broadcasting precisely those marks in the set $\{D_\alpha(\Xi_\alpha, t) | t \in T\}$. An l-dynamic frameproof code guarantees that no matter which of these marks the pirate chooses to broadcast at each time $\alpha \leq l$ once l segments have been broadcast the pirate broadcast sequence will not match the sequence allocated to any innocent user.

Example 5.1 If $\mathcal{M}$ is a q-ary l-sequential c-frameproof code for user set U we can define a q-ary l-dynamic c-frameproof code by setting $D_1(u) = M_1(u)$ and $D_\alpha(\Xi_\alpha, u) = M_\alpha(u)$ for all $u \in U$ and $\alpha = 2, \ldots, l$. ■

An l-sequential c-frameproof code guarantees that no innocent user can be framed over l time segments, and thus can be used as a dynamic frameproof code. The use for the dynamic codes is somewhat different from that of the sequential frameproof codes, however. The latter, if used continuously, prevent framing for any time intervals of length l or greater whereas the dynamic codes are designed to be put into use over a specific interval once there is a suspicion that framing is occurring. At this time the broadcaster starts to distribute marks in a manner corresponding to a dynamic frameproof code with α set to 1. If the broadcaster suspects framing is occuring the use of such a code will prevent that framing from continuing. This is accomplished within a time l which, to be of practical use, is less than that achieved by any sequential frameproof code with the appropriate parameters. In Chapter 6 we go on to

define *sliding-window l-dynamic frameproof codes*, which are perhaps more-closely related to the l-sequential frameproof codes; their construction builds on ideas obtained from the study of l-dynamic frameproof codes, however, much as l-sequential frameproof codes were obtained from the functions f_l^c in Section 4.1.

5.2 Construction of l-Dynamic Frameproof Codes

In the case of sequential frameproof codes improved convergence times can be achieved when the number of traitors is limited. In a dynamic situation, however, we will see that once we suppose there is more than one traitor we do not in fact gain anything by considering limits on the number of traitors: two traitors can do as much damage as $n-1$ traitors.

An l-dynamic frameproof code that uses an alphabet $Q = \{0, 1, \ldots, q-1\}$, and works for users $U = \{u_0, u_1, \ldots, u_{n-1}\}$ and any number $c < n$ of traitors in time $l = \left\lceil \log_q\left(\left\lceil \frac{n}{q-1} \right\rceil\right)\right\rceil + 1 \approx \log_q(n)$ is constructed as follows:

Construction 5.3. *Let $l = \left\lceil \log_q\left(\left\lceil \frac{n}{q-1} \right\rceil\right)\right\rceil + 1$; let $Q = \{0, 1, \ldots, q-1\}$ and $U = \{u_0, u_1, \ldots u_{n-1}\}$. Denote by q_i^j the j^{th} digit in the q-ary expansion of the integer $i \geq 0$. We define functions $D_1 \colon U \to Q$ and $D_j \colon Q^{j-1} \times U \to Q$ for $2 \leq j \leq l-1$ as follows:*

$$D_j(\Xi_j, u_i) = q_{i'}^j,$$

where $i' \in \left\{1, 2, \ldots \left\lceil \frac{n}{q-1} \right\rceil\right\}$ and $i' \equiv i \pmod{\left\lceil \frac{n}{q-1} \right\rceil}$.

This ensures that each q-ary sequence of length $l-1$ will have been received by up to $q-1$ users over the first $l-1$ time segments. At most $q-1$ users will have received sequences of marks matching the pirate broadcast sequence $\Xi_l = \{\xi_i\}_{i=1}^{l-1}$; denote these users (if they exist) by $w_1, w_2, \ldots w_{q-1}$. We then

define

$$D_l(\Xi_l, u_i) = \begin{cases} x & \text{if } u_i = w_x, \\ 0 & \text{otherwise.} \end{cases}$$

This ensures that any user whose sequence matches the pirate's over the first $l-1$ segments receives a unique symbol at time l. This is illustrated by the following example.

Example 5.2 Suppose there are 18 users $u_0, u_1, \ldots, u_{17}$, and a mark alphabet $\{0, 1, 2\}$. The following table shows how the marks are allocated to the users according to the above construction, based on a particular pirate broadcast sequence.

	1	2	3
u_0	0	0	1
u_1	1	0	0
u_2	2	0	0
u_3	0	1	0
u_4	1	1	0
u_5	2	1	0
u_6	0	2	0
u_7	1	2	0
u_8	2	2	0
u_9	0	0	2
u_{10}	1	0	0
u_{11}	2	0	0
u_{12}	0	1	0
u_{13}	1	1	0
u_{14}	2	1	0
u_{15}	0	2	0
u_{16}	1	2	0
u_{17}	2	2	0
T	0	0	0

Users u_0 and u_9 have been framed over the first two segments so at time 3 user u_0 is given the symbol 1 and u_9 the symbol 2 while all other users receive 0. A valid pirate broadcast sequence for this distribution will satisfy $\xi_3 = 0$, therefore in this case the pirate broadcast sequence is $0, 0, 0$, which

does not correspond to the sequence received by any user. Hence framing has not occurred.

This construction does not depend on the potential size of the pirate coalition. If we compare the code of this example with the sequential frameproof code resulting from Construction 4.6 we see that in the latter case $\frac{18}{3-1} = 9$ time segments are required to prevent framing, instead of the $\log_3\left(\frac{18}{3-1}\right) + 1 = 3$ required here. Thus the extra information available in the dynamic setting allows us to prevent framing much more quickly than when using sequential constructions. ■

In the example framing was prevented in time $\left\lceil \log_q\left(\left\lceil \frac{n}{q-1} \right\rceil\right) \right\rceil + 1$; we now show that the construction indeed achieves this in general.

Theorem 5.4. *The functions resulting from Construction 5.3 constitute an l-dynamic c-frameproof code for any $c \geq 0$.*

Proof. By the construction of D_l any user whose sequence matched the pirate broadcast over the first $l-1$ segments (i.e. any user who was framed over that time) is given a unique mark at time l. For any valid pirate broadcast sequence none of these users' received marks matches the pirate's at time l, thus there is no user who is framed over the entire interval. The functions defined in this construction hence constitute a $\left(\left\lceil \log_q\left(\left\lceil \frac{n}{q-1} \right\rceil\right) \right\rceil + 1\right)$-dynamic c-frameproof code for any $c \geq 0$, as the construction is independent of the value of c. □

If the number of users is $n = q^k(q-1)$ then the convergence time of the code yielded by this construction is $l = k+1$. This is in fact optimal for this number of users, a result which is a corollary of the following lemma.

Lemma 5.5. *Let $k \geq 1$, $q \geq 2$. A $(k+1)$-dynamic 2-frameproof code that*

protects n users from framing using an alphabet of size q exists if and only if $n \leq q^k(q-1)$.

Proof. In the case where $n \leq q^k(q-1)$ the existence of a $(k+1)$-dynamic 2-frameproof code is demonstrated by Construction 5.3. To prove the converse we use induction on k.

Let $P(k)$ be the proposition that the existence of a q-ary $(k+1)$-dynamic 2-frameproof code protecting n users implies $n \leq q^k(q-1)$.

We prove $P(1)$ is true: suppose the number of users is greater than $q(q-1)$. By the pigeon-hole principle there exists some set S of at least q users who receive the same marked segment during the first time interval. As we have $q(q-1)$ users but our alphabet only has q marks, some user $u \in S$ will receive the same marked segment as some other user t_2 (not necessarily in S) during the second time interval. Let $t_1 \in S \setminus \{u\}$, but note that t_1 and t_2 are not necessarily distinct. Then the sequence $D_1(t_1), D_2(D_1(t_1), t_2)$ is a valid pirate broadcast sequence for the pirate $T = \{t_1, t_2\}$, which can thus frame $u \notin T$ as $D_1(u) = D_1(t_1)$ and $D_2(D_1(t_1), u) = D_2(D_1(t_1), t_2)$. Therefore no matter how the functions D_1 and D_2 allocate the marks it is impossible to guarantee that framing will not occur when $n > q(q-1)$. Hence $P(1)$ is true.

Assume that $P(j)$ is true for some j, then each $(j+1)$-dynamic 2-frameproof code requires the number of users to be less than or equal to $q^j(q-1)$. Suppose there exists a $(j+2)$-dynamic 2-frameproof code $\{D_\alpha\}_{\alpha=1}^{j+2}$ for more than $q^{j+1}(q-1)$ users and an alphabet of size q. Then there is some subset S of users who will be given the same marked segment m during the first time interval, with $|S| > q^j(q-1)$. After this time interval, define a new family of functions $\{E_\gamma\}_{\gamma=1}^{j+1}$ with $E_1 \colon S \to Q$ and $E_\gamma \colon Q^{\gamma-1} \times S \to U$ by setting $E_\gamma(\bar{\Xi}_\gamma, u) = D_{\gamma+1}|_S(\Xi_{\gamma+1}, u)$, where if $\Xi_{\gamma+1}$ is the sequence $m, \xi_2, \xi_3, \ldots, \xi_{\gamma+1}$

then $\bar{\Xi}_\gamma$ is the sequence $\xi_2, \xi_3, \ldots, \xi_{\gamma+1}$. We claim that the family of functions $\{E_\gamma\}_{\gamma=1}^{j+1}$ is a $(j+1)$-dynamic 2-frameproof code for user set S, for otherwise there exists a valid pirate broadcast sequence $\{\xi_2, \xi_3, \ldots \xi_{j+2}\}$ corresponding to a pirate set T with $|T| \leq 2$, and some user $u \in S \setminus T$ with $E_1(u) = \xi_2$ and $E_i(\xi_2, \xi_3, \ldots, \xi_i, u) = \xi_{i+1}$ for $i = 2, 3, \ldots j+1$. As the members of T are in S then during the first segment they must have received the symbol m, as did user u. Hence the pirate T is capable of framing user u over all time segments from $t = 1$ to $j+2$. This contradicts the assumption that $\{D_\alpha\}_{\alpha=1}^{j+2}$ is $(j+2)$-dynamic 2-frameproof. Thus we conclude that $\{E_\gamma\}_{\gamma=1}^{j+1}$ is in fact $(j+1)$-dynamic 2-frameproof. However S has more than $q^j(q-1)$ members, which contradicts our inductive assumption. Hence $P(j) \Rightarrow P(j+1)$, and therefore $P(k)$ is true for all $k \geq 1$. □

Construction 5.3 demonstrates the existence of a $(k+1)$-dynamic frameproof code protecting $q^k(q-1)$ users from any number $1 \leq c \leq n-1$ of traitors. By the above lemma we see that a $(k+1)$-dynamic c-frameproof code exists only if $n \leq q^{k-1}(q-1)$ for any c with $2 \leq c \leq n-1$. Hence we obtain the following theorem.

Theorem 5.6. *An l-dynamic c-frameproof code protecting n users with $c > 1$ exists if and only if $n \leq q^{l-1}(q-1)$.*

If the number of users exceeds $q^{k-1}(q-1)$ the convergence time must be at least $k+1$, from which we deduce:

Corollary 5.7. *The convergence time of the code resulting from Construction 5.3 is optimal for the given number of users and alphabet size.*

The case where there is known to be precisely one traitor differs slightly. In this situation a pirate is some user t, who can only frame an innocent

user u if they have received the same sequences, so $D_i(t) = D_i(\Xi_i, u)$ for all $i = 1, 2, \ldots l$. From this we deduce the following theorem.

Theorem 5.8. *A q-ary l-dynamic 1-frameproof code protecting n users exists if and only if $n \leq q^l$.*

Proof. If the marked segments are distributed so that each user receives a distinct sequence of l symbols then no user can frame another. There are exactly q^l length l sequences with symbols from an alphabet of size q; if there are q^l users it is therefore possible to allocate a unique sequence to each user. This demonstrates the existence of q-ary l-dynamic 1-frameproof codes protecting q^l users.

Now suppose that the number of users is at least $q^l + 1$. At time 1 suppose the pirate broadcasts the symbol that was received by the largest number of users. Denote the set of users receiving this symbol by S_1. Then $|S_1| \geq \left\lceil \frac{q^l+1}{q} \right\rceil = q^{l-1} + 1$ (for $l \geq 1$). Suppose that at time 2 the pirate broadcasts the symbol received by the largest number of users in set S_1. Denote by S_2 the set of users whose received sequence corresponds with the pirate broadcast over the first two segments. Then $|S_2| \geq q^{l-2} + 1$. Repeating this process, at time $l - 1$ there is a set S_{l-1} of size $q + 1$ whose sequences correspond to the pirate broadcast over the first l time segments. At time l, therefore, two of the users in S_{l-1} receive the same mark. If the pirate broadcasts this mark at this time it follows that either of these users is capable of having produced the entire pirate broadcast, so that one of them may be the pirate who has thus framed the other user. Thus we claim that when the number of users exceeds q^l it is impossible to guarantee that a single traitor will not frame an innocent user. □

Chapter 6

Sliding-Window Dynamic Frameproof Codes

In this chapter we undertake to apply the extra information provided in the dynamic setting to create schemes similar to the sequential frameproof codes but having shorter convergence times; *sliding-window l-dynamic frameproof codes* are the result. We start by observing that such schemes can be constructed from dynamic frameproof codes and provide a bound on the convergence time that is tight for certain numbers of users. After giving a new construction for these schemes we then restrict our attention to schemes using a binary alphabet, and describe another construction which provides a model that is exploited by later constructions. We discuss a sufficient condition for a mark distribution to constitute a sliding-window l-dynamic framproof code and derive a bound on the convergence time of schemes satisfying this condition, showing that it is tight in many cases. This is followed by an examination of a geometric model that facilitates the study of a particular class of these schemes; finally, we determine a bound on the convergence times of schemes constructed geometrically.

6.1 The Sliding-Window Model

The dynamic frameproof codes described in the previous chapter can be used to prevent framing over one particular length l window. Sequential frameproof codes, on the other hand, have the property that they can be applied continuously to ensure protection from framing over any window of l consecutive time segments; we refer to this as the *sliding-window model*. This difference between the sequential and dynamic frameproof codes reflects a difference in how they might be used: a sequential frameproof code can be used throughout the broadcast and the broadcaster can be confident that framing will not occur, whereas a dynamic frameproof code may be applied at a particular time once the broadcaster suspects that framing is occurring. Using a dynamic frameproof code from time t to time $t+l-1$ the broadcaster can reassure each user that he or she will not have been framed over this time interval; this is perhaps not so useful if the pirate simply decides instead to frame some user from time $t+1$ onwards. The main difference in the construction of dynamic frameproof codes compared with schemes that work in the sliding-window model is that in the latter case the broadcaster does not know in advance over which interval the pirate might frame some user. The dynamic frameproof codes described in the previous chapter will no longer protect all users if the first segment over which they are applied is ignored, whereas a sliding-window frameproof code will not only protect users from being framed from time t to $t+l-1$, but also from time $t+1$ to $t+l$ and so on.

Example 6.1 Suppose $U = \{u_0, u_1, u_2, u_3\}$ and $Q = \{0, 1\}$ and let $l = 3$. The following table shows a potential mark distribution and pirate broadcast over

five time segments with a dynamic frameproof code resulting from Construction 5.3 applied over the first three.

	1	2	3	4	5
u_0	0	0	1	0	0
u_1	0	1	0	0	1
u_2	1	0	0	1	0
u_3	1	1	0	1	1
T	0	0	0	1	0

The use of a dynamic frameproof code prevents framing over the first three segments; however, the pirate is able to frame user u_2 over segments 2 to 5, indicating that this mark distribution does not prevent framing in the sliding-window model. ■

So far we have seen sequential frameproof codes that work in the sliding-window model and dynamic frameproof codes that do not. It would be advantageous if we could use the dynamic setting to create schemes offering sliding-window frameproof protection but having shorter convergence times (i.e. having shorter window length l) than sequential codes; it is not immediately clear that this is possible, however. Consider an l-dynamic c-frameproof code. In an analogue to Lemma 4.2 it can be applied repeatedly starting at times $1, l+1, 2l+1$ and so on. This will ensure that framing cannot occur over any $2l-1$ consecutive time segments, since a time interval of that size necessarily includes one complete application of the code. Hence sliding-window functionality is acheived. The following example shows that greatly improved convergence times can be obtained by such methods.

Example 6.2 Suppose there are 2×3^{10} users (slightly more than 10^5 users), and you wish to use an alphabet of size three to prevent framing by any number of traitors. Construction 4.6 results in a ternary l-sequential frameproof code with $l = 3^{10}$. Construction 5.3, however, yields an l-dynamic frameproof

code with $l = 11$. As per the argument above applying this code repeatedly enables the prevention of framing over any intervals of length $2l - 1 = 21$. Thus we see that making use of the information in the pirate broadcast sequence can allow us to continually prevent framing much more efficiently for a given alphabet size. ■

In order to further explore this idea we introduce the concept of a *sliding-window l-dynamic frameproof code*.

Definition 6.1. *A* sliding-window l-dynamic frameproof code *is a countable family of functions* $\{D_\alpha\}_{\alpha=1}^{\infty}$ *where* $D_1 \colon U \to Q$ *and* $D_\alpha \colon Q^{\alpha-1} \times U \to Q$ *for* $\alpha > 1$ *with the property that for any valid pirate broadcast sequence* $\{\xi_j\}_{j=1}^{\alpha}$ *corresponding to a pirate* T *there is no legitimate user* $u \in U \setminus T$ *and time* $i \leq \alpha - l$ *with* $D_j(\Xi_j, u) = \xi_j$ *for all* $j = i, i+1, \ldots i+l-1$.

A sliding-window l-dynamic frameproof code therefore prevents each user from being framed over any window of l consecutive time segments. The code described in the above example, which protects $q^k(q-1)$ users, is thus a sliding-window $(2k+1)$-dynamic frameproof code.

As in the case of sequential codes we refer to l as the *convergence time* of the code. It would be natural to wonder what is the smallest convergence time $l_{q,n}$ for which a sliding-window $l_{q,n}$-dynamic frameproof code exists for an alphabet of size q and user set U with $|U| = n$. The above discussion implies that $l_{q,q^k(q-1)} \leq 2k+1$. Furthermore, since restricting a sliding-window $l_{q,q^k(q-1)}$-dynamic frameproof code to the first $l_{q,q^k(q-1)}$ time intervals gives an ordinary $l_{q,q^k(q-1)}$-dynamic frameproof code, we have that

$$l_{q,q^k(q-1)} \geq k+1.$$

We show in Theorem 6.2 that this bound can be improved to $l_{q,q^k(q-1)} \geq k+2$. This is tight when $k=1$ and hence $l_{q,q(q-1)} = 3$, as shown in Corollary 6.3.

Theorem 6.2. *Let $q \geq 2$. If there exists a q-ary sliding-window l-dynamic frameproof code protecting $q^k(q-1)$ users then*

$$l \geq k+2.$$

Proof. We know that $l \geq k+1$. Suppose $\{D_\alpha\}_{\alpha=1}^{\infty}$ is a sliding-window $(k+1)$-dynamic frameproof code protecting $n = q^k(q-1)$ users. We refer to a segment in which at least one user receives a mark not received by any other user at that time as a *protection segment*. Suppose that at each time t prior to the first protection segment the pirate broadcast consists of the marked segment that ensures that the greatest possible number of users have been framed over segments 1 to t. Then the first protection segment must occur within the first l segments, or else at least one user will have been framed over this time, as in the proof of Lemma 4.5.

Suppose that the first protection segment occurs at some time j_0 and suppose the pirate broadcast at this time is the symbol ξ_{j_0} received by the greatest number of users at this time. Denote the set of users who received this symbol by S_0, and let $h_0 = |S_0| \geq \lceil \frac{n-1}{q-1} \rceil = q^k$. At time j_0+1 there exists a mark ξ_{j_0+1} received by at least $h_1 = \lceil \frac{h_0}{q} \rceil \geq q^{k-1}$ users; denote the set of such users by S_1 and suppose the pirate broadcast at this time consists of this mark. Continuing in this manner we find that at time j_0+k-1 there are $h_{k-1} \geq q$ users who have been framed over the k segments from j_0 to j_0+k-1. As the number of users is greater than k, at least one user $u \in S_{k-1}$ will receive a mark at time j_0+k that has been received by some other user t_1; suppose the pirate broadcast at this time consists of this mark. Also, there exists a

user $t_2 \in S_{k-1} \setminus \{u\}$, not necessarily distinct from t_1. The set $T = \{t_1, t_2\}$ is capable of having produced the pirate broadcast from time j_0 to j_k and thereby framing u over this period. We note further that the set $T' = U \setminus \{u\}$ is capable of having produced the entire broadcast from time 0 to $j_0 + k$, since the fact that no segment prior to j_0 was a protection segment implies that each mark broadcast by the pirate was received by at least one user in T', as $|T'| = n - 1$. Thus for any sliding-window $(k+1)$-dynamic frameproof code protecting $q^k(q-1)$ users there exists a pirate T' capable of framing a user $u \notin T'$ over some window of $k+1$ consecutive time segments, which is a contradiction. Hence we conclude that for any sliding-window l-dynamic frameproof code protecting $q^k(q-1)$ users we require $l \geq k+2$. □

In the case where $k = 1$, Theorem 6.2 allows us to determine the exact value of $l_{q,q^k(q-1)}$.

Corollary 6.3. *Suppose $q > 2$. Then*

$$l_{q,q(q-1)} = 3.$$

Proof. We know that $k + 2 \leq l_{q,q(q-1)} \leq 2k + 1$ by the intial observations of this section and by Theorem 6.2, but for $k = 1$ we have $k + 2 = 2k + 1 = 3$. □

For $k > 1$, however, there is still a discrepancy between the upper and lower bounds for $l_{q,q^k(q-1)}$. The following new construction requires $q > 2$ but yields a convergence time $l_{q,n} = \lceil \log_{q-1}(n) \rceil + 1$, which results in a reduction of the upper bound for $l_{q,q^k(q-1)}$ when $k > \frac{1}{2 - \log_{q-1} q}$, since we have that $\lceil \log_{q-1}(q^k(q-1)) \rceil + 1 < 2k + 1$ if $\lceil \log_{q-1}(q^k) \rceil < 2k - 1$. This holds if $\lceil \log_{q-1} q \rceil < 2 - \frac{1}{k}$, thus $k > \frac{1}{2 - \log_{q-1} q}$.

Now $\log_{q-1} q$ is a decreasing function when $q > 2$ and approaches a limit of 1 as $q \to \infty$; we observe that $\frac{1}{2 - \log_2 3} \approx 2.41$ and that $\frac{1}{2 - \log_{q-1} q}$ approaches

1 as $q \to \infty$. Hence we conclude that for $k > 2$ Construction 6.4 leads to a bound on $l_{q,q^k(q-1)}$ that is an improvement on that arising from Example 6.2.

Construction 6.4. *Suppose $q > 2$ and let $Q = \{0, 1, \ldots q-2\} \cup \{\infty\}$ and $U = \{u_0, u_1, \ldots u_{n-1}\}$. Let $l = \lceil \log_{q-1}(n) \rceil + 1$. Denote by q_i^j the j^{th} digit in the $(q-1)$-ary expansion of the integer $i \geq 0$. We define functions $D_1 \colon U \to Q$ and $D_j \colon Q^{j-1} \times U \to Q$ for $j > 1$ as follows:*

$$D_1(u_i) = q_i^1$$

$$D_j(u_i) = \begin{cases} \infty & \text{if } D_\alpha(\Xi_\alpha, u_i) = \xi_\alpha \text{ for all } \alpha = j-l+1, j-l+2, \ldots j-1, \\ q_i^{j'} & \text{otherwise, where } j' \in \{1, 2, \ldots, l-1\} \\ & \text{and } j' \equiv j \pmod{l-1}. \end{cases}$$

When this construction is used, a user u_i receives the $l-1$ digits of the $(q-1)$-ary expansion of the integer i repeatedly, unless that user has been framed over the $l-1$ previous segments, in which case he or she receives the protection symbol ∞. This is illustrated in the following example.

Example 6.3

	1	2	3	4	5	6	7	8	9
u_0	0	0	0	0	0	0	0	0	0
u_1	0	1	∞	1	0	1	0	1	0
u_2	0	2	0	2	0	2	0	2	0
u_3	0	3	0	3	0	3	0	3	0
u_4	1	0	1	0	1	0	1	0	1
u_5	1	1	1	1	1	1	1	∞	1
u_6	1	2	1	2	1	2	1	2	1
u_7	1	3	1	3	1	3	1	3	1
u_8	2	0	2	0	2	∞	2	0	2
u_9	2	1	2	1	2	1	∞	1	2
u_{10}	2	2	2	2	2	2	2	2	2
u_{11}	2	3	2	3	2	3	2	3	2
u_{12}	3	0	3	0	∞	0	3	0	3
u_{13}	3	1	3	∞	3	1	3	1	3
u_{14}	3	2	3	2	3	2	3	2	3
u_{15}	3	3	3	3	3	3	3	3	3
T	0	1	3	0	2	1	1	1	0

Suppose there are sixteen users $u_0, \ldots u_{15}$, and an alphabet $Q = \{0, 1, 2, 3, \infty\}$. The table above demonstrates how marks are allocated by the functions constructed above over the first nine time intervals based on a particular example of a sequence broadcast by some pirate.

For $q = 5$ and $n = 16$ we have $\lceil \log_{q-1}(n) \rceil + 1 = 3$. The symbol ∞ is broadcast whenever a user has been framed over consecutive time intervals. ■

The functions arising from Construction 6.4 define a q-ary sliding-window $(\lceil \log_{q-1}(n) \rceil + 1)$-dynamic frameproof code. To prove this we need the following lemma.

Lemma 6.5. *If marks* $\{0, 1, \ldots, q-2\} \cup \{\infty\}$ *are allocated to* n *users according to the functions defined in Construction 6.4 then at every time* $i \geq 1$ *at most one user receives the symbol* ∞*.*

Proof. We use strong induction on i. As before, we let $l = \lceil \log_{q-1}(n) \rceil + 1$.

Let $P(i)$ be the proposition that at time i at most one user receives the symbol ∞.

Then $P(i)$ is true for $i = 1, 2, \ldots, l$ since no user receives the symbol ∞ in the first $l-1$ time segments; during this time the user u_i receives the sequence of numbers corresponding to the $(q-1)$-ary expansion of i. As such, no two users have received the same sequence of marks over the first l time intervals. This implies that at most one user has received a sequence corresponding to that broadcast by the pirate; such a user receives ∞ at time l and the others all receive the first number of their $(q-1)$-ary expansion.

Assume $P(h-l+2), P(h-l+3), \ldots, P(h)$ are true. If the pirate broadcasts ∞ at some time between $h - l + 2$ and h no innocent user is framed at that time, hence no user receives ∞ at time $h + 1$.

We now assume that the pirate has not broadcast ∞ at any time during that interval. Any user who has received ∞ during that time can not be framed over time $h-l+2, h-l+3, \ldots, h$ and thus will not receive ∞ at time $h+1$. Any user u_i who did not receive ∞ at any time during that interval received a cyclic shift of the $(q-1)$-ary expansion of i. No two users have the same such expansion, thus at most one user has received a sequence corresponding to that broadcast by the pirate, and hence receives ∞ at time $h+1$.

Thus $P(h-l+2), P(h-l+3), \ldots, P(h) \Rightarrow P(h+1)$, and so $P(i)$ is true for all $i \geq 1$. □

We are now in a position to prove the desired result.

Theorem 6.6. *The functions defined in Construction 6.4 constitute a sliding-window* $(\lceil \log_{q-1}(n) \rceil + 1)$*-dynamic frameproof code.*

Proof. Suppose that between times i and $i + \lceil \log_{q-1}(n) \rceil$ there exists a set S of users who receive sequences of marks equal to those broadcast by the pirate. By the definition of $D_{i+\lceil \log_{q-1}(n) \rceil + 1}$ these users receive symbol ∞ at time $i + \lceil \log_{q-1}(n) \rceil + 1$; by Lemma 6.5 there is at most one user in S. No user in $U \setminus S$ can be framed over the interval from i to $i + \lceil \log_{q-1}(n) \rceil + 1$; assume therefore that there exists a user $u \in S$. At time $i + \lceil \log_{q-1}(n) \rceil + 1$ the user u receives the symbol ∞. As no other user receives this symbol at this time it is impossible for u to be framed by any pirate coalition of which it is not a member over the time interval i to $i + \lceil \log_{q-1}(n) \rceil + 1$. Hence our functions define a sliding-window $(\lceil \log_{q-1}(n) \rceil + 1)$-dynamic frameproof code, as required. □

We now turn our attention to the binary case in an effort to determine some more-precise bounds.

6.2 The Binary Case

When one considers sliding-window dynamic frameproof codes over binary alphabets it becomes apparent that the situation is slightly different from that of general q: Construction 6.4 can no longer be used, for instance. By restricting our attention to this case, however, we obtain new bounds for the minimum convergence time. We also provide examples of constructions whose properties will be exploited by later more-general constructions.

When there are 2^k users the repeated use of a $k+1$-dynamic frameproof code as in Example 6.2 yields a sliding-window $(2k+1)$-dynamic frameproof code. We can, however, do better than this: the following construction provides an example of a sliding-window $2k$-dynamic frameproof code.

Construction 6.7. *Let $Q = \{0,1\}$ and $U = \{u_0, u_1, \ldots u_{2^k-1}\}$, for some $k \geq 4$. Denote by b_i^γ the γ'^{th} bit in the binary expansion of the integer i where $\gamma' \equiv \gamma \pmod{k}$ and $\gamma' \in \{1, 2, \ldots, k\}$. At each time segment marks will be distributed in one of two ways: either at most one user will receive the mark* 1 *and all others will receive* 0*, or exactly half the users will receive* 1 *and half* 0*. A time segment in which precisely one user receives a* 1 *will be known as a* protection segment*, as this distribution protects the user receiving the* 1 *from being framed; the segments in which half the users receive* 1 *will be referred to as* ordinary segments*.*

We define functions $D_1 \colon U \to Q$ and $D_j \colon Q^{j-1} \times U \to Q$ for $j > 1$ by setting

$$D_1(u_i) = b_i^1,$$

and, for $j > 1$, by setting

$$D_j(\Xi_j, u_i) = \begin{cases} b_i^\gamma & \textit{if } (k-1) \nmid j\textit{, where } u_i \textit{ received } b_i^{\gamma-1} \textit{ in the last} \\ & \textit{ordinary segment.} \\ 1 & \textit{if } (k-1) | j \textit{ and } u_i \textit{ has been framed over the} \\ & \textit{previous } k \textit{ ordinary segments.} \\ 0 & \textit{otherwise.} \end{cases}$$

Essentially what happens is that a user u_i repeatedly receives the sequence of 0s and 1s corresponding to the binary representation of i, except that at each time j where j is a multiple of $k-1$ he receives a 1 if he has been framed over the previous k ordinary segments (those segments in which he is receiving the bits of his binary expansion), and a 0 otherwise. As there are 2^k users each user thus corresponds to a unique k-bit binary number, so over the course of k non-protection segments precisely one user will have a sequence corresponding to the pirate broadcast and hence be framed. This ensures that when $k-1$ divides j, exactly one user will receive the bit 1, thus these segments are protection segments.

Example 6.4 Suppose there are sixteen users $u_0, u_1, \ldots, u_{15}$, and an alphabet $Q = \{0, 1\}$. The table below demonstrates how marks are allocated according to the above construction over the first twelve time intervals based on a particular example of a sequence broadcast by some pirate T.

In this example, time segments 6, 9 and 12 are protection segments with users u_4, u_7 and u_3 respectively being protected. The remaining segments are ordinary segments, except for segment 3, in which no user is protected since no user has yet been framed for $k = 4$ sections. The time interval from 1 to 12 includes five windows of length $2k = 8$: those starting at times $1, 2, 3, 4$ and 5. Inspection shows that the pirate sequence over any of these windows does

not match that received by any user.

	1	2	3	4	5	6	7	8	9	10	11	12
u_0	0	0	0	0	0	0	0	0	0	0	0	0
u_1	1	0	0	0	0	0	1	0	0	0	0	0
u_2	0	1	0	0	0	0	0	1	0	0	0	0
u_3	1	1	0	0	0	0	1	1	0	0	0	1
u_4	0	0	0	1	0	1	0	0	0	1	0	0
u_5	1	0	0	1	0	0	1	0	0	1	0	0
u_6	0	1	0	1	0	0	0	1	0	1	0	0
u_7	1	1	0	1	0	0	1	1	1	1	0	0
u_8	0	0	0	0	1	0	0	0	0	0	1	0
u_9	1	0	0	0	1	0	1	0	0	0	1	0
u_{10}	0	1	0	0	1	0	0	1	0	0	1	0
u_{11}	1	1	0	0	1	0	1	1	0	0	1	0
u_{12}	0	0	0	1	1	0	0	0	0	1	1	0
u_{13}	1	0	0	1	1	0	1	0	0	1	1	0
u_{14}	0	1	0	1	1	0	0	1	0	1	1	0
u_{15}	1	1	0	1	1	0	1	1	0	1	1	0
T	0	0	0	1	0	0	1	1	0	0	0	0

■

In fact this property illustrated above is true in general: for any k the code resulting from the above construction is sliding-window $2k$-dynamic frameproof, as shown by the following theorem.

Theorem 6.8. *The functions defined in Construction 6.7 constitute a sliding-window $2k$-dynamic frameproof code.*

Proof. Let I be any time interval of length $2k$. For $k \geq 4$ we have $\lfloor \frac{2k}{k-1} \rfloor = 2$ and $\lceil \frac{2k}{k-1} \rceil = 3$ so there will be either two or three protection segments occurring during time interval I. Denote by t the last protection segment occurring within I. In the case where there are three protection segments, I must contain a sequence of a protection segment followed by $k-2$ non-protection segments, another protection segment, $k-2$ more non-protection segments and then segment t. Since $k \geq 4$ it follows that $2(k-2) \geq k$, so the k non-protection

segments preceding t all lie within I. Similarly, if I contains two protection segments we know that t must occur within the last $k-1$ segments of I and hence is preceded by at least $k+1$ segments of I, at most one of which is a protection segment. Thus in this case too, the k non-protection segments preceding t lie within I. Let S be the set of time segments consisting of t and the k non-protection segments preceding t. During those k segments a user u_i receives a cyclic shift of the bits in the binary expansion of i; each user thus receives a unique sequence over these k segments, and hence one user will have a sequence corresponding to that broadcast by the pirate. That user will be the only user to receive the symbol 1 in segment t, and hence cannot be framed by any coalition of other users at that segment. Therefore it is impossible for a pirate coalition to frame any user on every time segment in S. Since S is a subset of the interval I we can therefore conclude that no user can be framed over the interval I. Since this is true for any length $2k$ time interval I we can conclude that our functions do indeed define a sliding-window $2k$-dynamic frameproof code. □

Construction 6.7 uses regularly-spaced protection segments to construct a sliding-window l-dynamic frameproof code with at least two protection segments in every length l time interval. It is natural to wonder whether a scheme using more protection segments per interval could produce codes with smaller values of l. Indeed this is the case, as will be demonstrated in the subsequent discussion. But first it pays to consider more closely the properties of the above construction. The method of allocating marks leads to each segment being either a protection segment or an ordinary segment. Any dynamic frameproof code must make use of protection segments as otherwise there will exist pirate sets capable of framing some user indefinitely (*cf.* Lemma 4.5).

Ordinary segments do not eliminate framing altogether, rather they can serve to restrict the size of the set of users who are framed over a particular interval. Assigning binary sequences to each user as in Construction 6.7 allows you to halve the size of such a set with each ordinary segment; when a binary alphabet is used this is the best reduction that can be guaranteed in a single segment.

The proof that the code produced in Construction 6.7 is indeed sliding-window $2k$-dynamic frameproof relies on the fact that each length $2k$ time interval contains some protection segment t and the k preceding ordinary segments. This condition is sufficient for constructions of this type to yield sliding-window dynamic frameproof codes. We show in Theorem 6.10 that in order for this condition to hold it is necessary that $l \geq k + \lceil 2\sqrt{k} \rceil$. This theorem is expressed in terms of sequences of zeros and ones, with 1 representing a protection segment, and 0 an ordinary segment. Its proof makes use of the following lemma.

Lemma 6.9. *Let $l, k, b \in \mathbb{N}$ with $k < l < k + 2\sqrt{k}$ and $b < l - k$. Then $\left\lceil \frac{k}{l-k-b} \right\rceil > b$.*

Proof. Consider $b^2 - (l-k)b + k$ to be a quadratic in b. It is positive definite when the discriminant is less than 0, which occurs when we have $(l-k)^2 - 4k = l^2 - 2kl + k^2 - 4k < 0$. Considering the discriminant to be a quadratic in l we find it has zeros $\frac{2k \pm \sqrt{4k^2 - 4(k^2-4k)}}{2} = k \pm 2\sqrt{k}$, thus for $k < l < k + 2\sqrt{k}$ the discriminant is negative. So $b^2 - (l-k)b + k > 0$, and we deduce that $\frac{k}{l-k-b} > b$, as $l - k - b > 0$. □

Theorem 6.10. *Let $\{x_i\}_{i=1}^{\infty}$ where $x_i \in \{0, 1\}$ be a sequence with the property that every window of l consecutive elements contains at least one 1 and at least*

k 0s before the final 1 *in that window. Then* $l \geq k + \lceil 2\sqrt{k} \rceil$.

Proof. Suppose there exists a sequence $S = \{x_i\}_{i=1}^{\infty}$ satisfying the above property for $l \leq k + \lceil 2\sqrt{k} \rceil - 1$ (we note that this implies $l < k + 2\sqrt{k}$, and that trivially $l \geq k+1$). Consider a window $\{x_i, x_{i+1}, \ldots, x_{i+l-1}\}$ with $i > l(\lceil 2\sqrt{k} \rceil - 1)$ that ends with b consecutive 0s for some $b \geq 1$ (such windows necessarily exist). If $b \geq \lceil 2\sqrt{k} \rceil - 1$ there are not enough elements remaining in the window to satisfy the desired property, hence we can assume $b < \lceil 2\sqrt{k} \rceil - 1$. The remaining $l - b$ elements of the window must include at least k further 0s, hence the window contains at most $l - k - b$ 1s. By the pigeon-hole principle one of these 1s must be preceded by at least $b_1 = \left\lceil \frac{k}{l-k-b} \right\rceil$ 0s, hence we can find another window ending in b_1 0s, and $b_1 > b$ by Lemma 6.9. By repeating this process up to $\lceil 2\sqrt{k} \rceil - 2$ times we find there must exist a window ending in $\lceil 2\sqrt{k} \rceil - 1$ 0s; such a window does not have enough remaining elements to allow the desired property to be satisfied, thus our original assumption is contradicted. □

This result implies that in order for every window of a sliding-window l-dynamic frameproof code to contain at least one protection segment with k ordinary segments occurring prior to the final protection segment we require $l \geq k + \lceil 2\sqrt{k} \rceil$.

The following construction, which parallels Construction 6.7 but requires k to be a square, results in a code satisfying the above condition with $l = k + 2\sqrt{k}$, which is thus a sliding-window $(k + 2\sqrt{k})$-dynamic frameproof code. This implies that for k square the codes resulting from this construction are optimal in the sense of having the smallest convergence time possible for codes satisfying that condition; the bound of Theorem 6.10 is therefore tight in this case.

Construction 6.11. *Let k be a square, let $Q = \{0, 1\}$ and let the set of users be $U = \{u_0, u_1, \ldots u_{2^k-1}\}$. As before denote by b_i^γ the γ'^{th} bit in the binary expansion of the integer $i \geq 0$ where $\gamma \equiv \gamma' \pmod{k}$ and $\gamma' \in \{1, 2, \ldots, k\}$. We define functions $D_1 \colon U \to Q$ and $D_j \colon Q^{j-1} \times U \to Q$ for $j > 1$ as follows:*

$$D_1(u_i) = b_i^1$$

$$D_j(\Xi_j, u_i) = \begin{cases} b_i^\gamma & \text{if } (\sqrt{k}+1) \nmid j, \text{ where } u_i \text{ received mark } b_i^{\gamma-1} \\ & \text{in the previous ordinary segment.} \\ 1 & \text{if } (\sqrt{k}+1) \mid j \text{ and } u_i \text{ has been framed over} \\ & \text{the previous } k \text{ ordinary segments.} \\ 0 & \text{otherwise.} \end{cases}$$

In an analogue of Construction 6.7, every $(\sqrt{k}+1)^{th}$ segment will be a protection segment, and all the others will be ordinary segments. In intervals with $\sqrt{k}+1$ protection segments the final protection segment must be preceded by at least $\sqrt{k} \times \sqrt{k} = k$ non-protection segments, and in intervals with $\sqrt{k}$ protection segments the final one must be preceded by at least

$$\begin{aligned}(\sqrt{k}-1)\sqrt{k} + (l - (\sqrt{k}-1)\sqrt{k} - \sqrt{k} - \sqrt{k}) &= (\sqrt{k}-1)\sqrt{k} + \sqrt{k} \\ &= k\end{aligned}$$

non-protection segments. We can then apply the proof of Theorem 6.8 *mutatis mutandis* to show that Construction 6.11 really does afford a sliding-window $(k + 2\sqrt{k})$-dynamic frameproof code.

Example 6.5 In the case where $k = 4$ we have that $\sqrt{k}+1 = k-1$. As such, the code resulting from this construction will be identical to that illustrated in Example 6.4. ■

Hence we now have a sufficient condition for codes to be sliding-window l-dynamic frameproof that implies the bound $l \geq \lceil k + 2\sqrt{k} \rceil$, which is tight

at least when k is a square. It is certainly not a necessary condition, however: for $q = 2$ and $|U| = 2^k$ the code resulting from Construction 4.6 can be thought of as a sliding-window 2^k-dynamic frameproof code, yet for this code every segment is a protection segment and hence the given condition is not fulfilled. Therefore we would like to know whether in general there exist sliding-window l-dynamic frameproof codes not satisfying this condition that achieve $l < k + 2\sqrt{k}$. We begin by turning our attention to a particular class of codes, those given by a geometric construction, and show that for these codes there is not much room for improvement, with l required to be greater than or equal to $k + \lceil 2\sqrt{k} \rceil - 1$ (see Theorem 6.17).

6.2.1 Geometric Constructions

In preceding sections we have given definitions for dynamic and sliding-window dynamic frameproof codes, as well as constructions of such codes. In what follows we consider examples of these codes that can be given by geometric constructions. The geometric setting has the advantage that it is frequently easier to visualise the structures involved, which can lead to simpler arguments. These are not the most general constructions possible but in subsequent sections we will make use of results obtained in studying them. Our aim in this section is to show that geometric codes protecting 2^k users require $l \geq k + \lceil 2\sqrt{k} \rceil - 1$; we first present some results analogous to those of prior sections in order to introduce the geometric notation.

Suppose there is a set U of 2^k users. We can then associate with each user one of the 2^k points of the k-dimensional affine space $AG(k, 2)$ obtained by removing a hyperplane Σ_∞ of the k-dimensional projective space $PG(k, 2)$. Denote by P_u the point of $AG(k, 2)$ associated with the user $u \in U$.

The hyperplanes of $AG(k,2)$ (i.e. the intersections of $AG(k,2)$ with hyperplanes of $PG(k,2)$ other than Σ_∞) fall into $2^k - 1$ parallel classes, with two hyperplanes in each class. Specifying a hyperplane in a particular parallel class partitions the set of users into two subsets of size 2^{k-1}: those users corresponding to points in that hyperplane, and those whose points fall in the other hyperplane of that parallel class.

We can use geometric ideas to construct optimal dynamic frameproof codes as follows:

Construction 6.12. *In $PG(k,2)$ it is possible to find a set of $k+1$ linearly-independent hyperplanes, $\Sigma_\infty, \Sigma_1, \Sigma_2, \ldots, \Sigma_k$ say.*

We define functions D_α for $\alpha = 1, 2, \ldots, k$ by

$$D_\alpha(u) = \begin{cases} 1 & \text{if } u \in \Sigma_\alpha, \\ 0 & \text{otherwise.} \end{cases}$$

We also define a function D_{k+1} that depends on the pirate broadcast sequence $\{\xi_i\}_{i=1}^k$ where $\xi_i \in GF(2)$. Consider the set $S = \{\Sigma'_1, \Sigma'_2, \ldots, \Sigma'_k\}$, where $\Sigma'_i = \Sigma_i$ if $\xi_i = 1$ and $\Sigma'_i = \Sigma_i^{par}$, the hyperplane parallel to Σ_i, if $\xi_i = 0$. For each i we have $\Sigma_i^{par} = \Sigma_i + \Sigma_\infty$ as these three hyperplanes form a pencil about a secundum. Thus the k hyperplanes in S are linearly-independent and will therefore intersect in a unique point P. We then define D_{k+1} by:

$$D_{k+1}(\Xi_{k+1}, u) = \begin{cases} 1 & \text{if } P_u = P, \\ 0 & \text{otherwise.} \end{cases}$$

The user P is the unique user whose allotted marks correspond to the pirate broadcast sequence over the first k segments; as P is the only user to receive symbol 1 at time $k+1$ it is thus impossible for an innocent user to be framed over the entire $k+1$ time segments. Hence the functions $\{D_\alpha\}_{\alpha=1}^{k+1}$ form a $(k+1)$-dynamic frameproof code. By Lemma 5.6 this is optimal when

the number of users is 2^k.

Example 6.6 Suppose you have sixteen users, $u_0, u_1, \ldots, u_{15}$. Construction 6.12 can be implemented as follows. For $k = 4$ we are working in a five-dimensional projective space that has coordinates $(x_0, x_1, x_2, x_3, x_4)$ with the $x_i \in \{0, 1\}$ not all zero. Let Σ_∞ be the hyperplane given by the equation $x_4 = 0$, and represent each user by a point of $PG(4, 2) \setminus \Sigma_\infty$ by setting $u_0 \to (0, 0, 0, 0, 1)$ and $u_1 \to (0, 0, 0, 1, 1)$, $u_2 \to (0, 0, 1, 0, 1)$ and so on up to $u_{15} \to (1, 1, 1, 1, 1)$. We let the hyperplanes $\Sigma_1, \Sigma_2, \Sigma_3, \Sigma_4$ be given by equations $x_3 + x_4 = 0$, $x_2 + x_4 = 0$, $x_1 + x_4 = 0$ and $x_0 + x_4 = 0$ respectively. The point $(0, 0, 0, 0, 1)$ corresponding to user u_0 does not lie in $\Sigma_1 : x_3 + x_4 = 0$, hence at time 1 the user u_0 receives the symbol 0, whereas the point $(0, 0, 0, 1, 1)$ corresponding to u_1 does lie in this hyperplane and thus u_1 receives a 1 at this time. The following table shows the marks received by each user according to Construction 6.12 at times $1, 2, \ldots, k + 1 = 5$ based on a particular choice of pirate broadcast.

	1	2	3	4	5
u_0	0	0	0	0	0
u_1	1	0	0	0	0
u_2	0	1	0	0	1
u_3	1	1	0	0	0
u_4	0	0	1	0	0
u_5	1	0	1	0	0
u_6	0	1	1	0	0
u_7	1	1	1	0	0
u_8	0	0	0	1	0
u_9	1	0	0	1	0
u_{10}	0	1	0	1	0
u_{11}	1	1	0	1	0
u_{12}	0	0	1	1	0
u_{13}	1	0	1	1	0
u_{14}	0	1	1	1	0
u_{15}	1	1	1	1	0
T	0	1	0	0	0

The symbols broadcast by the pirate at times 1 through 4 correspond to the four hyperplanes given by $\Sigma_1' = \Sigma_1 + \Sigma_\infty : x_3 = 0$, $\Sigma_2' = \Sigma_2 : x_2 + x_4 = 0$, $\Sigma_3' = \Sigma_3 + \Sigma_\infty : x_1 = 0$, $\Sigma_4' = \Sigma_4 + \Sigma_\infty : x_0 = 0$. These hyperplanes intersect in the point

$$\left| \begin{pmatrix} x_0 & x_1 & x_2 & x_3 & x_4 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \right| = (0,0,1,0,1) \to u_2.$$

Hence the user u_2 has been framed for the first four segments (as is shown by the table), and therefore receives a 1 at time 5, with all other users receiving 0 at this time. This ensures that no user has been framed over the first $k+1 = 5$ segments. ■

Similar ideas can be used in the construction of sliding-window l-dynamic frameproof codes. Informally, during each time segment the broadcaster selects either a hyperplane or a point of $AG(k,2)$ based on the pirate's previous broadcasts, and transmits segments marked with a 1 to the corresponding users with the other users receiving 0s. This is made more precise in the following construction and lemma.

Construction 6.13. *Let Γ be the union of the set of points of $AG(k,2)$ with the set of hyperplanes of $AG(k,2)$, let G_1 be an element of Γ and suppose there exists a countable family of functions $\{G_\alpha\}_{\alpha=2}^{\infty}$, where $G_\alpha \colon GF(2)^{\alpha-1} \to \Gamma$.*

We construct a countable family of functions $\{D_\alpha\}_{\alpha=1}^{\infty}$ with $D_1 \colon U \to GF(2)$ and $D_\alpha \colon GF(2)^{\alpha-1} \times U \to GF(2)$ for $\alpha > 1$. The sequence broadcast by the pirate prior to time α will be denoted $\Xi_\alpha = \{\xi_i\}_{i=1}^{\alpha-1}$. We set

$$D_1(u) = \begin{cases} 1 & \text{if } u \in G_1 \\ 0 & \text{otherwise,} \end{cases}$$

and, for $j > 1$,

$$D_\alpha(\Xi_\alpha, u) = \begin{cases} 1 & \text{if } u \in G_\alpha(\Xi_\alpha) \\ 0 & \text{otherwise.} \end{cases}$$

This link between functions G_α associating subspaces of $AG(k,2)$ with pirate broadcast sequences and functions D_α assigning marked segments to users allows us to set geometric conditions for $\{D_\alpha\}_{\alpha=1}^{\infty}$ to constitute a sliding-window l-dynamic frameproof code. In order to express these conditions we define $\mathcal{P}_L = \{i \in L \mid G_i(\Xi_i) \text{ is a point of } AG(k,2)\}$ for any time interval L. We will refer to time segments $i \in L \setminus \mathcal{P}_L$, for which $G_i(\Xi_i)$ is a hyperplane, as *ordinary segments*, and we will use the notation $\Sigma_i = G_i(\Xi_i)$. Time segments $i \in \mathcal{P}_L$ will be called *protection segments*, and the point $P_i = G_i(\Xi_i)$ will be said to have been *protected* during segment i, since the corresponding user has received a unique mark and thus cannot be framed by any other users at that time. For $i \in L \setminus \mathcal{P}_L$ set $\Sigma_i' = \Sigma_i$ if $\xi_i = 1$ and $\Sigma_i' = \Sigma_i + \Sigma_\infty$ if $\xi_i = 0$. Then at time $i \in L \setminus \mathcal{P}_L$ the hyperplane Σ_i' contains those users whose received symbol matches that broadcast by the pirate at that time.

Lemma 6.14. *The family of functions* $\{D_\alpha\}_{\alpha=1}^{\infty}$ *resulting from Construction 6.13 is a sliding-window* l*-dynamic frameproof code if and only if for every valid pirate broadcast sequence* $\{\xi_i\}_{i=1}^{\infty}$ *we have that for every length* l *interval* L *the set*

$$S_L = \bigcap_{i \in L \setminus \mathcal{P}_L} \Sigma_i' \setminus \{P_j | j \in \mathcal{P}_L\}$$

is empty.

Proof. Based on the above construction the set S_L consists of those users who have received symbols matching the pirate's broadcast on all ordinary segments of L but who have not been protected in any protection segment, that is precisely of those users who have received a sequence of marks agreeing

with that broadcast by the pirate. It then follows trivially from the definition of a sliding-window l-dynamic frameproof code that the code obtained from the above construction is sliding-window l-dynamic frameproof precisely when this set is empty. □

As an example of a geometric sliding-window l-dynamic frameproof code we describe a code analogous to that of Construction 6.11, recast in the geometric setting.

Example 6.7 Suppose there exists a set U containing 2^k users, where k is square. Choose a set $S = \{\Sigma_\infty, \Sigma_1, \Sigma_2, \ldots, \Sigma_k\}$ of $k+1$ linearly-independent hyperplanes in $PG(k,2)$ and associate each user $u \in U$ with a point P_u in $PG(k,2) \setminus \Sigma_\infty$.

For i with $(\sqrt{k}+1) \nmid i$ define the images of Ξ_i under G_i to be successive hyperplanes from S, so that $G_1(\Xi_1) = \Sigma_1$ and $G_2(\Xi_2) = \Sigma_2$, and then $G_{\sqrt{k}+2}(\Xi_{\sqrt{k}+2}) = \Sigma_{\sqrt{k}+1}$ and so on.

When $i < k + \sqrt{k}$ and $(\sqrt{k}+1) \mid i$ let $G_i(\Xi_i) = P_{u_1}$. (This choice of point is arbitrary.)

For $i \geq k + \sqrt{k}$ with $(\sqrt{k}+1) \mid i$, define

$$G_i(\Xi_i) = \bigcap_{j \in \{i-k-\sqrt{k}+1, \ldots, i-1 \mid (\sqrt{k}+1) \nmid j\}} \Sigma'_j.$$

By construction $G_i(\Xi_i)$ is the intersection of k linearly-independent hyperplanes and is thus a point of $AG(k,2)$.

Therefore $\{G_i\}_{i=1}^{\infty}$ consists of $\sqrt{k}$ ordinary segments followed by a protection segment, and during protection segments occurring after time $k+\sqrt{k}$ the user who has been framed over the previous k ordinary segments is protected.

For instance, in the case $k = 4$ if the coordinates are chosen as in Example 6.6 and the same set of hyperplanes is used then the resulting code

is the same as that described in Example 6.5 and Example 6.4 (comparing the mark distribution tables of Example 6.6 and Example 6.4 reveals that the distributions are the same during the corresponding ordinary segments).

We claim that the G_i thus defined constitute a geometric sliding-window $(k + 2\sqrt{k})$-dynamic frameproof code.

Proof. At each time $i \geq k + \sqrt{k}$ where $(\sqrt{k} + 1)|i$ a user is allocated a unique mark, as for such i we have that $G_i(\Xi_i)$ is a point corresponding to the only user who may have been framed over the previous $k + \sqrt{k} - 1$ time segments. Consider any time interval I of length $k + 2\sqrt{k}$. There will be a time i with $(\sqrt{k} + 1)|i$ within the last $\sqrt{k} + 1$ segments of that interval; the $k + \sqrt{k} - 1$ segments prior to i will be contained in I. These $k + \sqrt{k} - 1$ segments contain precisely k ordinary segments so the corresponding hyperplanes Σ'_i are linearly-independent and thus intersect in a point. Hence over those $k + \sqrt{k}$ segments at most one user has been framed. That user is protected at time i, and thus can not be framed at that time. Therefore we conclude that no user can be framed for the entirety of I, and hence the code defined by our G_i is indeed sliding-window $k + 2\sqrt{k}$-dynamic frameproof. ■

Thus we see that there exist geometric codes with a convergence time of $k + 2\sqrt{k}$. In an effort to bound the smallest possible convergence time that can be achieved we consider here a particular strategy that might be adopted by a pirate; the need to ensure that a code prevents framing by such a pirate will allow us to deduce a bound for this minimal value.

Definition 6.15. *We will use the term* clever pirate *to refer to a set of users forming a pirate coalition that broadcasts marked segments according to the*

strategy described below. (The description being necessarily somewhat convoluted the strategy is perhaps better illustrated by Example 6.8.)

- *At any protection segment broadcast a 0.*
- *If an ordinary segment lies in a block of consecutive ordinary segments contained in a window whose k^{th} ordinary segment falls in that block, and if the point that is the intersection of the k hyperplanes corresponding to the pirate's broadcast in those segments has not been protected during that window then broadcast the symbol pertaining to the hyperplane containing that point.*
- *At an ordinary segment that follows a non-empty block of consecutive protection segments choose a symbol corresponding to a hyperplane containing at most half of the points protected in those segments. In any subsequent ordinary segments broadcast the symbol whose hyperplane contains at most half of those protected points lying in the previously chosen hyperplane, unless you are in the situation mentioned in the previous rule. If the previously chosen hyperplane contains none of those protected points, consider the previous non-empty block of consecutive protections segments and choose the hyperplane containing at most half of those points thus protected that lie in the intersection of the hyperplanes coming from all subsequent ordinary segments. If none of the points lie in this intersection consider the previous block of protected segments and so on, until you reach a block lying at least l segments prior to the current one or until you reach the first time segment; if this happens broadcast a 0.*

The following example shows a clever pirate's broadcast in response to the

distribution of marks from a (supposed) sliding-window 7-dynamic frameproof code for 2^3 users.

Example 6.8

	1	2	3	4	5	6	
u_1	0	0	0	1	0	0	...
u_2	0	0	1	0	0	0	...
u_3	0	1	0	0	0	1	...
u_4	0	1	1	0	0	1	...
u_5	1	0	0	0	1	0	...
u_6	1	0	1	0	1	0	...
u_7	1	1	0	0	1	1	...
u_8	1	1	1	0	1	1	...
T	0	0	0	0	1	0	...

Segment four is a protection segment, the others are all ordinary segments. During the fourth segment the pirate is using the first of the above rules, and in the fifth segment he is using the third rule: broadcasting a 1 here is equivalent to choosing the hyperplane which does not contain the point belonging to u_1, who was protected in segment four. In the sixth time segment rule two is being invoked: segment five is the third (i.e. the k^{th}) ordinary segment of the window commencing at segment two. The pirate broadcast in the three ordinary segments 2, 3 and 5 corresponds to hyperplanes intersecting in u_5's point, so in segment six the pirate broadcasts a 0 (the symbol received by u_5 at this time). This second rule essentially ensures that any user who is framed over the first k ordinary segments of a window continues to be framed unless explicitly protected by a protection segment. ■

We note that for any geometric sliding-window dynamic frameproof code there will always be some set of users capable of acting as a clever pirate as the set of all users is certainly able to do so. Also (provided there are more than two users) any set consisting of all but one user is also able to behave as a clever pirate.

The following lemma describes a consequence of a clever pirate's actions, and will play a role similar to that of Lemma 6.9 in allowing us to prove Theorem 6.10, which gives us a bound for the convergence time of binary geometric sliding-window l-dynamic frameproof codes.

Lemma 6.16. *Let b be a positive integer, and suppose there are 2^k users for some $k \geq b$. Suppose also that there exists a geometric code with the property that when faced with a clever pirate at most b ordinary segments occur consecutively at any time after some time t_0. Then for any interval of $k + \lceil \frac{k}{b} \rceil - 2$ consecutive segments occurring after t_0 there exists some pirate set T capable of behaving as a clever pirate and some user $u \notin T$ who is framed over those segments.*

Proof. Consider any length $k+\lceil \frac{k}{b} \rceil -2$ interval occurring after time t_0. Suppose that $a \geq 0$ ordinary segments occur within the interval. First we observe that $0 < a < k$. If there were no ordinary segments then 2^k protection segments would be required to protect all the users, but $k + \lceil \frac{k}{b} \rceil - 2 < 2k - 2 < 2^k$ for all k. Furthermore, if there were k or more ordinary segments, the fact that they can occur in blocks of at most b segments implies there would have to be at least $\lceil \frac{k}{b} \rceil - 1$ protection segments to separate them, which is impossible in an interval of this size.

We assume therefore that there are $0 < a < k$ ordinary segments within the interval. As these can occur in groups of at most b there must be at least $\lceil \frac{a}{b} \rceil - 1$ protection segments separating them. For each of these separating protection segments there is a corresponding nonempty block of consecutive ordinary segments occurring at a later time within the interval; in the first ordinary segment of each block the clever pirate uses rule three and thus excludes

at least one protected point from the intersection of the pirate's chosen hyperplanes. There are then $k + \lceil \frac{k}{b} \rceil - 2 - a - \lceil \frac{a}{b} \rceil + 1$ further protection segments whose protected points may lie in this intersection.

Now a hyperplanes intersect in a space of dimension $k - a$ containing 2^{k-a} points of which up to $k + \lceil \frac{k}{b} \rceil - 1 - a - \lceil \frac{a}{b} \rceil$ are protected.

We observe that

$$\begin{aligned} 2^{k-a} - (k + \frac{k}{b}) + a + \frac{a}{b} &= 2^{k-a} - (k-a)\frac{b+1}{b} \\ &\geq 2^{k-a} - 2(k-a) \quad (\text{as } b \geq 1) \\ &\geq 0. \end{aligned}$$

Hence $2^{k-a} + a + \frac{a}{b} \geq k + \frac{k}{b}$ and therefore $2^{k-a} + a + \lceil \frac{a}{b} \rceil \geq k + \lceil \frac{k}{b} \rceil$, and $2^{k-a} - (k + \lceil \frac{k}{b} \rceil - 1 - a - \lceil \frac{a}{b} \rceil) \geq 1$. Thus we conclude that there is at least one user who is framed over the entire interval. If we denote this user by u then the set $U \setminus \{u\}$ is capable of behaving as a clever pirate and thus framing u. □

By definition a sliding-window l-dynamic frameproof code must be able to protect every potentially-innocent user from being framed regardless of which users form the pirate coalition and which marks it chooses to broadcast; it must therefore be able to protect all users from a clever pirate. We use this fact to obtain the following result.

Theorem 6.17. *A binary geometric sliding-window l-dynamic frameproof code protecting 2^k users must satisfy $l \geq k + \lceil 2\sqrt{k} \rceil - 1$.*

Proof. The proof of this result is analogous to that of Theorem 6.10. Suppose there exits a binary geometric sliding-window l-dynamic frameproof code with $l \leq k + \lceil 2\sqrt{k} \rceil - 2$. Suppose the code is applied against a clever pirate, and consider a length l window W starting after time $t = l(2\sqrt{k} - 1)$ that ends with

b consecutive ordinary segments, for some $b \geq 1$. As before we can assume $b < 2\sqrt{k}$. Prior to these b segments the window contains $l-b$ segments, ending with a protection segment. However $l-b < k+\lceil 2\sqrt{k}\rceil -1-b$. This is less than $k + \lceil \frac{k}{b} \rceil - 1$ for any k and $b > 0$ since $(\sqrt{k} - b)^2 \geq 0$, implying $k + b^2 \geq 2b\sqrt{k}$ and thus $\frac{k}{b} \geq 2\sqrt{k} - b$. Suppose that these $l - b$ segments have the property that at most b ordinary segments occur consecutively. By Lemma 6.16 at least one user has been framed over those $l - b$ segments. We claim that the clever pirate continues to frame one or more of these users over the remaining b ordinary segments of the window. At the first of these ordinary segments the third clever pirate rule is invoked, so the pirate chooses the hyperplane whose intersection with the previously-chosen hyperplanes contains the least number of points protected by the most recent block of protection segments. Lemma 6.16 implies that at least one point in this intersection is not protected, hence the intersection with the newly-chosen hyperplane must also contain at least one unprotected point. During the remaining segments either the third rule is used again, in which case there is at least one user who continues to be framed as before, or else the second rule is required. In this case the intersection of the first k hyperplanes chosen by the pirate is an unprotected point, which will lie in any further hyperplanes chosen. Hence the intersection of all the hyperplanes chosen by the pirate in the window W contains at least one unprotected point. Thus some user is framed over the entire window, contradicting the fact that the code was sliding-window l-dynamic frameproof. From this we deduce that the first $l - b$ segments of the window must in fact have included some block of $b' > b$ consecutive ordinary segments.

Repeating this process as in the proof of Theorem 6.10 leads to a contradiction as before, thus implying that l is in fact greater than or equal to

$k + \lceil 2\sqrt{k} \rceil - 1$. □

This result yields a much-improved lower bound on l for most values of k. There is still a slight discrepancy between the upper and lower bounds for l, as the best known construction have $l = k + \lceil 2\sqrt{k} \rceil$, one greater than the above lower bound. Thus a problem remains open: *does there exist a binary, geometric, sliding-window l-dynamic frameproof code with $l = k + \lceil 2\sqrt{k} \rceil - 1$?* Also, the above bound applies only to geometric codes. Attempts to extend the concept of a clever pirate to the setting of general binary sliding-window dynamic frameproof codes encounter the problem that whereas in the geometric setting exactly half the users received each symbol during an ordinary segment, in the general case this is not necessarily so. This causes the dilemma that whereas choosing one mark might result in more users being framed in a particular segment, choosing the other might enable the pirate to continue framing a larger group than would otherwise be the case. Thus it is not always immediately clear which choice of broadcast mark most benefits the pirate. (In practice the pirate will be unaware of the mark distribution and will therefore not be consciously making such choices; as we wish to protect against all pirates, however, we need to make sure our schemes are proof against pirates who by chance choose to broadcast those symbols that benefit them most.)

Example 6.9 Suppose the mark distribution and pirate broadcast over four time segments are as follows.

	1	2	3	4
u_0	0	0	0	0
u_1	0	1	0	1
u_2	1	0	0	0
u_3	0	0	1	1
u_4	1	1	1	1
T	0	0	0	?

At time 4, by broadcasting a 1 the pirate can frame three users instead of

two, increasing its potential to continue framing in successive segments. By broadcasting a 0, however, the pirate ensures that two users instead of no users are framed over segments 2 to 4, and similarly ensures user u_0 is framed over all four segments, whereas no user would have been thus framed had 1 been chosen. ■

This leads to a second open problem: *do there exist binary sliding-window dynamic frameproof codes protecting n users with a convergence time l less than that of any geometric binary sliding-window dynamic frameproof code protecting n users?*

We now proceed to describe geometric constructions for sliding-window dynamic frameproof codes using alphabets of size $q > 2$. We will see that when $q > 2$ the geometrically-constructed codes certainly do not achieve the optimum possible l.

6.3 Geometric Constructions for Prime Power Values of q

The geometric constructions of binary sliding-window dynamic frameproof codes in the previous section generalise readily to yield q-ary sliding-window dynamic codes where q is a power of a prime. In this section we describe these generalisations and discuss certain limitations of codes constructed in this manner.

For every prime power q and for every positive integer k there exists a k-dimensional affine space $AG(k, q)$ containing q^k points. The hyperplanes of $AG(k, q)$ are partitioned into parallel classes, each containing q pairwise disjoint hyperplanes. Given a parallel class Δ the hyperplanes within that class can therefore be labelled H_i^Δ where $i \in GF(q)$. We use this fact in

constructing mark distributions from sequences of elements in $\Gamma_{q,k}$ by the following method, which is a generalisation of Construction 6.13.

Construction 6.18. *Let q be a prime power. Suppose there are q^k users; we identify these users with the points of $AG(k,q)$. Define the set $\Gamma_{q,k}$ by setting $\Gamma_{q,k} = \{$points of $AG(k,q)\} \cup \{$parallel classes of hyperplanes of $AG(k,q)\}$.*

Suppose there exists a countable family of functions $\{G_\alpha\}_{\alpha=2}^{\infty}$ where the function $G_\alpha : GF(q)^{\alpha-1} \to \Gamma_{q,k}$, and let G_1 be an element of $\Gamma_{q,k}$.

We construct a family of functions $\{D_\alpha\}_{\alpha=1}^{\infty}$ with $D_1 : U \to GF(q)$ and $D_\alpha : GF(q)^{\alpha-1} \times U \to GF(q)$ for $\alpha > 1$ by setting

$$D_1(u) = \begin{cases} 1 & \text{if } G_1 = u, \\ i & \text{if } u \in H_i^{G_1}, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$D_\alpha(\Xi_\alpha, u) = \begin{cases} 1 & \text{if } G_\alpha(\Xi_\alpha) = u, \\ i & \text{if } u \in H_i^{G_\alpha(\Xi_\alpha)}, \\ 0 & \text{otherwise,} \end{cases}$$

for $\alpha > 1$.

As in the binary case, if $G_\alpha(\Xi_\alpha)$ is a point, then the corresponding user receives a 1 and all others receive a 0; this is a protection segment. If $G_\alpha(\Xi_\alpha)$ is a parallel class then the users are partitioned by the hyperplanes of this class into q pair-wise disjoint sets of q^{k-1} users, with all the users in a set receiving the same symbol. Segments where this occurs are called ordinary segments, as before. The difference from the binary case is simply that in ordinary segments the users are split into q sets instead of 2 by the distribution of the symbols.

As previously, if L is a time interval we define the set of time segments $\mathcal{P}_L = \{i \in L | G_i(\Xi_i)$ is a point of $AG(q,k)\}$ and we denote $G_i(\Xi_i)$ by P_i when $i \in \mathcal{P}_L$. If the pirate broadcast is $\{\xi_i\}_{i=1}^{\infty}$ then for $i \in L \setminus \mathcal{P}_L$ we set

$\Sigma'_i = \Sigma_{\xi_i}^{G_i(\Xi_i)}$; this is the hyperplane in class $G_i(\Xi_i)$ whose points correspond to the users whose marks match the pirate's broadcast at time i. Then we see that Lemma 6.14 remains valid in this context too:

Lemma 6.19. *The family of functions $\{D_\alpha\}_{\alpha=1}^{\infty}$ is a sliding-window l-dynamic frameproof code if and only if for every valid pirate broadcast sequence $\{\xi_i\}_{i=1}^{\infty}$ we have that for every length l interval L the set*

$$S_L = \bigcap_{i \in L \setminus \mathcal{P}_L} \Sigma'_i \setminus \{P_j | j \in \mathcal{P}_L\}$$

is empty.

In the case where q is a prime power and k is a square we can use this lemma to generalise the construction of Example 6.7 to yield a q-ary geometric sliding window $(k + 2\sqrt{k})$-dynamic frameproof code protecting q^k users, as described below.

Example 6.10 Suppose that there are q^k users with q a prime power and k square. Choose a set $S = \{\Sigma_\infty, \Sigma_1, \Sigma_2, \ldots, \Sigma_k\}$ of $k+1$ linearly-independent hyperplanes in $PG(k, q)$, and associate each user $u \in U$ with a point P_u in the space $AG(k, q) = PG(k, q) \setminus \Sigma_\infty$. The hyperplanes of $AG(k, q)$ given by $\Sigma_i \cap AG(k, q)$ for $i \in 1, 2, \ldots, k$ each lie in distinct parallel classes, since the set of hyperplanes $\{\Sigma_i | i = \infty, 1, 2, \ldots, k\}$ is linearly independent. Denote the parallel class containing Σ_i by Δ_i, and let $\mathcal{D} = \{\Delta_i | i = 1, 2, \ldots, k\}$.

For i with $(\sqrt{k} + 1) \nmid i$ define the images of Ξ_i under G_i to be successive parallel classes from $\mathcal{D}$, so that $G_1(\Xi_1) = \Delta_1$ and $G_2(\Xi_2) = \Delta_2$, and then $G_{\sqrt{k}+2}(\Xi_{\sqrt{k}+2}) = \Delta_{\sqrt{k}+1}$ and so on.

When $i < k + \sqrt{k}$ and $(\sqrt{k} + 1) \mid i$ let $G_i(\Xi_i) = P_{u_1}$. (This choice of point

is arbitrary.) For $i \geq k + \sqrt{k}$ with $(\sqrt{k}+1) \mid i$, define

$$G_i(\Xi_i) = \bigcap_{j \in \{i-k-\sqrt{k}+1,\ldots,i-1 | (\sqrt{k}+1) \nmid j\}} \Sigma'_j.$$

By construction $G_i(\Xi_i)$ is the intersection of k hyperplanes from different parallel classes, which are therefore linearly independent, hence this intersection is a point of $AG(k,q)$. The proof that Example 6.7 yields a sliding-window $(k + 2\sqrt{k})$-dynamic frameproof code can now be applied to show that this construction also produces a sliding-window $(k + 2\sqrt{k})$-dynamic frameproof code, except that in this case the alphabet is of size q, and the number of users protected is q^k. ■

It is possible to generalise Construction 6.12 in the same manner, yielding a $(k+1)$-dynamic frameproof code protecting q^k users. This is described in the following example, which may also serve to elucidate Example 6.10.

Example 6.11 Suppose there are nine users and you wish to construct a 3-ary, 3-dynamic frameproof code. In this case $k = 2$ so $PG(k,3)$ is the projective plane of order 3, whose hyperplanes are lines and whose points have coordinates (x,y,z) with $x,y,z \in GF(3)$ not all zero. The lines $\Sigma_\infty : z = 0$, $\Sigma_1 : x = 0$ and $\Sigma_2 : y = 0$ are linearly independent. We choose Σ_∞ to be the line at infinity and associate points of $P_u \in PG(2,3) \setminus \Sigma_\infty$ with each user by setting $u_0 \to (0,0,1)$, $u_1 \to (0,1,1)$, $u_2 \to (0,2,1)$, $u_3 \to (1,0,1)$, $u_4 \to (1,1,1)$, $u_5 \to (1,2,1)$, $u_6 \to (2,0,1)$, $u_7 \to (2,1,1)$ and $u_8 \to (2,2,1)$. The parallel class Δ_1 contains the lines $H_0^{\Delta_1} : x = 0$, $H_1^{\Delta_1} : x + z = 0$ and $H_2^{\Delta_1} : x + 2z = 0$; the parallel class Δ_2 contains lines $H_0^{\Delta_2} : y = 0$, $H_1^{\Delta_2} : y + z = 0$ and $H_2^{\Delta_2} : y + 2z = 0$.

We define functions D_α for $\alpha = 1, 2$ by

$$D_\alpha(\Xi_\alpha, u) = i \text{ when } u \in H_i^{\Delta_\alpha}.$$

We also define a function D_3 by setting

$$D_3(\Xi_3, u) = \begin{cases} 1 & \text{if } u = \Sigma_1' \cap \Sigma_2', \\ 0 & \text{otherwise.} \end{cases}$$

At time 1 the point $(0, 1, 1)$ corresponding to user u_1 lies on the line $H_0^{\Delta_1}$, so this user receives the symbol 0 at this time, and so on. The marks received by the users, given a particular pirate broadcast, are shown in the following table.

	1	2	3
u_0	0	0	0
u_1	0	2	0
u_2	0	1	0
u_3	2	0	0
u_4	2	2	0
u_5	2	1	0
u_6	1	0	0
u_7	1	2	0
u_8	1	1	1
T	1	1	0

The marks broadcast by the pirates at times 1 and 2 correspond to the lines $H_1^{\Delta_1}$ and $H_1^{\Delta_2}$ respectively, which intersect in the point

$$\left| \begin{pmatrix} x & y & z \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \right| = (2, 2, 1) \to u_8,$$

hence user u_8 receives a 1 at time 3, with the other users receiving 0. ■

We have thus seen how geometric constructions yield $(k + 1)$-dynamic frameproof codes protecting q^k users using an alphabet of size q. We know, however, from Theorem 5.6 that this is not optimal when $q > 2$, as there exist q-ary, $(k + 1)$-dynamic frameproof codes protecting up to $q^k(q - 1)$ users. We will see in the following chapter that the sliding-window dynamic frameproof codes of Example 6.10 are also suboptimal in the number of users they support. This points to a weakness in the geometric construction when $q > 2$, which is quickly seen to arise from the protection segments. In the binary

case it was only possible to protect one user in any given segment. When $q > 2$ it is possible to assign up to $q-1$ unique marks and thereby protect up to $q-1$ users in one segment. The geometric constructions described above only permit one user to be protected, however, thereby losing this potential flexibility. One might be tempted to modify the definition of the geometric construction to enable $q-1$ users to be protected in each protection segment. The problem with this, however, is that the set of users framed over a series of γ ordinary segments is a subspace of $AG(k,q)$ and consequently contains $q^{k-\gamma}$ users. Thus if more than one user is in danger at a protection segment then at least q users are, so it is impossible to protect all of them in one segment. The number $q-1$ of possible unique marks in any one segment does not fit efficiently with the fact that the number of users is restricted to being a power of q; consider the fact that l-dynamic frameproof codes can protect at most $q^{l-1}(q-1)$ users. One might consider using a subset of $AG(k,q)$ of an appropriate size, but in doing so one loses the symmetry that made the geometric constructions attractive in the first place.

A restriction to protecting at most one user per segment might not in fact be a problem (see Section 7.3); the real disadvantage of the geometric constructions described above is the fact that in protection segments only two symbols are used, hence $q-2$ potentially-useful symbols are wasted. It is possible to distribute $q-1$ symbols amongst the users and still give a unique mark to one user. Examples of this were seen in Construction 6.4, which yields shorter convergence times than the geometric constructions for certain numbers of users.

Lastly, there is also the objection that the geometric construction is restricted to cases where q is a prime power. In the following chapter we will

investigate q-ary sliding-window dynamic frameproof codes for general q that draw their inspiration from the geometric constructions, but which overcome the abovementioned shortcomings to protect greater numbers of users for a given value of l.

Chapter 7

Improved Constructions of Sliding-Window Dynamic Frameproof Codes

In this chapter we investigate a family of constructions of sliding-window l-dynamic frameproof codes, and show that all our previous constructions can be interpreted as members of this family. We make use of this connection to compare our existing constructions and consider optimal choices of certain parameters. We then demonstrate that it is possible to construct sliding-window dynamic frameproof codes that protect more users than those resulting from previous constructions when the same parameters are used. We show that the schemes arising from these new constructions are optimal for the given parameters. After a brief consideration of the asymptotic properties of these codes as the alphabet size is increased we conclude with a discussion of some unresolved issues.

7.1 A Unifying Family of Constructions

So far we have seen several constructions for q-ary sliding-window dynamic frameproof codes such as Construction 6.4, the construction of Example 6.10,

and the construction arising from the repeated application of dynamic frameproof codes (see Example 6.2). In general it can be hard to say conclusively whether one of these constructions is better than another; quick calculations serve to demonstrate that for certain values of l, q and n one of these constructions is more efficient than the others, whereas other choices of parameter may lead to a less-efficient code resulting from the same construction.

Example 7.1 Suppose there are q^k users, where $q = 3$ and $k = 25$. Then Construction 6.4 yields a sliding-window 41-dynamic frameproof code, whereas Example 6.10 yields a more-efficient sliding-window 35-dynamic frameproof code. If, however, q is increased to 20 then the tables are turned, with the code resulting from Construction 6.4 being sliding-window 27-dynamic, making it more efficient then the sliding-window 35-dynamic frameproof code resulting from Example 6.10. ■

These constructions all have certain elements in common, however, such as their use of unique marks, the necessity of which can be shown by adapting the proof of Lemma 4.5. In fact we will see that these separate constructions can instead be thought of as members of a larger family of constructions.

7.2 A New Family of Constructions

Section 6.3 contained a discussion of some of the limitations inherent in geometric constructions when the alphabet size is greater than two. This forces us to go beyond the restrictions of the geometric setting if we wish to construct more-efficient schemes. In what follows we discuss several heuristic principles that can be used to inform the design of such schemes, and we present a construction based on these principles.

In general we observe that any time segment can be either a protection

segment (in which at least one user receives a unique mark) or an ordinary segment. It would seem desirable in an ordinary segment to divide the q available symbols as evenly as possible among the users, as this ensures that that at most $\lceil \frac{n}{q} \rceil$ users will receive a symbol matching the pirate broadcast at that time.

In a protection segment it is possible to protect up to $q-1$ users, since an alphabet of size q allows at most $q-1$ unique symbols to be distributed. Inflexibility in this respect was one of the limitations of the geometric schemes discussed in Section 6.3. To overcome this, we introduce a parameter a, with $1 \leq a \leq q-1$ being the number of users protected in each protection segment. Choosing to protect the same number of users in each protection segment not only simplifies the description of the scheme but permits us to choose the number a that leads to the most efficient schemes, given particular values of l and q. Once a users are protected $q-a$ symbols remain at the disposal of the broadcaster; as in the case of the ordinary segments these remaining marks will be distributed evenly among the remaining users. Thus at most $\lceil \frac{n-a}{q-a} \rceil$ users can be framed in such a segment.

It would be possible to envisage a scheme in which the distribution of protection segments was determined as a reaction to the pirate broadcast. In the construction described below, however, the distribution of the protection segments will be predetermined, with the information from the pirate broadcast being used instead to determine which users are protected in each such segment. We introduce a second parameter $b \geq 0$, being the number of ordinary segments occurring between successive protection segments. As discussed above this simplifies the description of the schemes and allows an optimal spacing of the protection segments to be determined. We observe that

any ordinary segments occurring after the final protection segment cannot be relied upon for framing prevention. If there are b ordinary segments between protection segments, then a window ends in at most b ordinary segments. In this case the final protection segment is preceded by $l-b-\left\lceil \frac{l-b}{b+1} \right\rceil$ ordinary segments and $\left\lceil \frac{l-b}{b+1} \right\rceil - 1$ protection segments that fall within that window. Every length l window can thus be guaranteed to have at least this many ordinary and protection segments occurring before the final protection segment of the window.

Having decided how many unique marks to allocate in a protection segment, and how many ordinary segments will lie between those segments, it is necessary to decide how to determine the mark distribution within each segment, given the condition that non-unique symbols are evenly distributed among the users. We wish to avoid the case where the same mark distribution is used in adjacent ordinary segments as the pirate could then broadcast the same symbol each time, thus continuing to frame a particular set of users, without yielding any new information that would help prevent this framing.

Example 7.2 Of the tables below, the one on the left illustrates two ordinary segments using the same distribution. Here the pirate frames three users over both segments.

	1	2
u_0	0	0
u_1	0	0
u_2	0	0
u_3	1	1
u_4	1	1
u_5	1	1
u_6	2	2
u_7	2	2
u_8	2	2
T	0	0

	1	2
u_0	0	0
u_1	0	1
u_2	0	2
u_3	1	0
u_4	1	1
u_5	1	2
u_6	2	0
u_7	2	1
u_8	2	2
T	0	0

In the table on the right, however, the pirate is only able to frame one user, no matter what marks it decides to broadcast, as each user has received a different pair of marks. ■

In the geometric constructions we used linearly independent hyperplanes in successive ordinary segments. This ensured not only that each symbol was distributed evenly among the users in each ordinary segment, but also that in every i succesive ordinary segments, for $i = 1, 2, \ldots, k$ all the possible length i sequence of symbols were evenly distributed among the users over that time. We will make use of this property in our subsequent construction.

Taking the above considerations in mind we now present a construction of a family of q-ary sliding-window l-dynamic frameproof codes.

Construction 7.1. *Let $q \geq 2$ and $Q = \{0, 1, \ldots, q-1\}$, with $1 \leq a \leq q-1$, and $b \geq 0$ with $l \geq 2b+1$. We will construct a sliding-window l-dynamic frameproof code supporting $n = q^{l-b-\lceil \frac{l-b}{b+1} \rceil}(q-a)^{\lceil \frac{l-b}{b+1} \rceil - 1}a$ users.*

Consider the set S of all words of length $l-b-1$ whose first $l-b-\lceil \frac{l-b}{b+1} \rceil$ letters come from Q and whose remaining letters are restricted to the set $\{0, 1, \ldots, q-a-1\}$. Then S contains $q^{l-b-\lceil \frac{l-b}{b+1} \rceil}(q-a)^{\lceil \frac{l-b}{b+1} \rceil - 1} = \frac{n}{a}$ words. Define an $n \times (l-b-1)$ array $M = (m_{ij})$ by letting the first a rows of M be given by a particular word from S, the next a rows be given by a different word of S, and so on. We partition M into an array $M_O = (g_{ij})$ of size $n \times \left(l-b-\lceil \frac{l-b}{b+1} \rceil\right)$ and an array $M_P = (h_{ij})$ of size $n \times \left(\lceil \frac{l-b}{b+1} \rceil - 1\right)$ by letting M_O consist of the first $l-b-\lceil \frac{l-b}{b+1} \rceil$ columns of M, and M_P the rest.

$$M = \left(\begin{array}{ccc|ccc} g_{11} & \cdots & g_{1\left(l-b-\lceil \frac{l-b}{b+1} \rceil\right)} & h_{11} & \cdots & h_{1\left(\lceil \frac{l-b}{b+1} \rceil - 1\right)} \\ \vdots & & \vdots & \vdots & & \vdots \\ g_{n1} & \cdots & g_{n\left(l-b-\lceil \frac{l-b}{b+1} \rceil\right)} & h_{n1} & \cdots & h_{n\left(\lceil \frac{l-b}{b+1} \rceil - 1\right)} \end{array} \right)$$

The entries of M_O come from Q, and those of M_P lie in $\{0, 1, \ldots, q-a-1\}$. We will now use these arrays in the construction of the mark distribution for a sliding-window l-dynamic frameproof code.

- *Segments occurring at times j where $(b+1) \nmid j$ will be ordinary segments. We define a function $D_1 : U \to Q$ by*

$$D_1(u_i) = g_{i1}.$$

- *We define the distribution at successive ordinary segments by using successive columns of M_O. So if at some ordinary segment the column $\gamma - 1$ of M_O was used $\left(mod\ l - b - \left\lceil \frac{l-b}{b+1} \right\rceil\right)$ and the next ordinary segment occurs at some time j we define $D_j : Q^{j-1} \times U \to Q$ by*

$$D_j(\Xi_j, u_i) = g_{i\gamma}.$$

- *In protection segments occurring before time $l-b$ we will protect the first a users; this choice is arbitrary (see note below). In the first protection segment, occurring at time $b+1$, we define D_{b+1} by*

$$D_{b+1}(\Xi_j, u_i) = \begin{cases} i + q - a & when\ j = 0, 1, \ldots, a, \\ h_{i1} & otherwise. \end{cases}$$

- *In subsequent protection segments occurring prior to time $l-b$ we continue to protect the first a users, and distribute the rest of the marks according to successive columns of M_P.*

- *At the first protection segment occurring at time $l-b$ or later, we consider the $l-b-1$ previous time segments. We observe that $l-b-\left\lceil \frac{l-b}{b+1} \right\rceil$ of these are ordinary segments, and the remaining $\left\lceil \frac{l-b}{b+1} \right\rceil - 1$ are protection segments. Because of the way the marks have been allocated these segments represent a permutation of the columns of M and the sequences of*

marks received by each user will be such that each possible sequence will have been received by up to a users. As each row in M matched exactly $a-1$ other rows of M we see that the pirate will have framed at most a users (it may be the case that some of the a users whose sequences would otherwise have corresponded to that row of M were actually previously protected during this time). The symbols $q-a, q-a+1, \ldots, q-1$ are allocated to these users as unique marks; the other users have their marks allocated according to the next column of M_P.

- *This same procedure is followed at all future protection segments.*

We note that the choice of users who are protected in the first $l-1$ segments does not affect the functioning of the scheme: there is no danger of a user having been framed over l segments as there aren't enough previous segments. Each window of l consecutive segments must include some sequence of $l-b$ segments ending with a protection segment, due to the fact that the last protection segment of any window is at most b segments from the end of the window. Therefore each window contains a sequence of $l-b$ segments over which no user has been framed; hence the resulting scheme is a sliding-window l-dynamic frameproof code.

Example 7.3 Suppose $q = 4$ and $l = 5$, and that $a = 2$ and $b = 1$.

$$\left(\begin{array}{cc|c} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ \vdots & \vdots & \vdots \\ 3 & 3 & 0 \\ 3 & 3 & 1 \\ 3 & 3 & 1 \end{array}\right)$$

Then the above construction yields a code protecting 64 users; the corresponding array M appears above.

The following table demonstrates how the marks are allocated to some of the 64 users over the first nine segments, based on a particular pirate broadcast.

	1	2	3	4	5	6	7	8	9
u_0	0	0	0	0	0	0	0	0	0
u_1	0	0	0	0	0	0	0	0	0
u_2	0	1	0	1	0	1	0	1	0
u_3	0	1	0	1	0	1	0	1	0
u_4	0	0	1	**2**	0	0	1	0	0
u_5	0	0	1	**3**	0	0	1	0	0
u_6	0	1	1	1	0	**2**	1	1	0
u_7	0	1	1	1	0	**3**	1	1	0
u_8	0	0	2	0	0	0	2	**2**	0
u_9	0	0	2	0	0	0	2	**3**	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
u_{64}	3	1	3	1	3	1	3	1	3
T	0	0	1	1	0	0	2	1	3

Unique marks have been written in bold, the odd-numbered segments are ordinary segments and the even-numbered segments are protection segments. Any user who is framed over the three segments prior to a protection segment receives a unique mark in that segment, which ensures that no user is framed over four segments that commence with an odd-numbered segment. As any length 5 window contains such a sequence of segments this mark distribution corresponds to a sliding-window 5-dynamic frameproof code. ■

In fact all the previously-discussed constructions of sliding-window dynamic frameproof codes, except for the sub-optimal Example 6.10, can be thought of as members of the family of such codes created in the above manner. In Table 7.1 we list all these constructions, as well as the lengths, alphabet sizes and values of a and b that give rise to them.

Construction	q	l	a	b	n
Example 6.2	3	21	2	10	2×3^{10}
Construction 6.4	q	l	1	0	$(q-1)^{l-1}$
Construction 6.7	2	$2k$	1	$k-1$	2^k
Construction 6.11	2	$k+2\sqrt{k}$	1	$\sqrt{k}$	2^k
Example 6.7	2	$k+2\sqrt{k}$	1	$\sqrt{k}$	2^k

Table 7.1: parameters of the known constructions of sliding-window l-dynamic frameproof codes

The fact that all these constructions can be viewed as members of a common family gives us a framework in which to compare them, and they can be related to each other through the parameters a and b. Given particular values of q and l the broadcaster can choose the values of these parameters in order to maximise the number of users supported by the resulting scheme.

In the following sections we look more closely at special cases corresponding to particular choices for the parameters a and b and consider situations in which it is beneficial to use these values.

7.3 The Case where $a = 1$

Sliding-window l-dynamic frameproof codes arising from the above construction can support up to $q^{l-b-\lceil \frac{l-b}{b+1} \rceil}(q-a)^{\lceil \frac{l-b}{b+1} \rceil - 1}a$ users. In the case where $l = 2b+1$ this reduces to $q^b a$, which is maximised with respect to a when $a = q-1$. If $l > 2b+1$, however, we have

$$n(a) = q^{l-b-\lceil \frac{l-b}{b+1} \rceil}(q-a)^{\lceil \frac{l-b}{b+1} \rceil - 1}a,$$

the derivative of which is

$$\begin{aligned}\frac{dn}{da} &= q^{l-b-\lceil \frac{l-b}{b+1} \rceil}\left((q-a)^{\lceil \frac{l-b}{b+1} \rceil - 1} - a\left(\lceil \tfrac{l-b}{b+1} \rceil - 1\right)(q-a)^{\lceil \frac{l-b}{b+1} \rceil - 2}\right), \\ &= q^{l-b-\lceil \frac{l-b}{b+1} \rceil}(q-a)^{\lceil \frac{l-b}{b+1} \rceil - 2}\left(q - \lceil \tfrac{l-b}{b+1} \rceil\, a\right).\end{aligned}$$

The first two factors of this expression are positive, hence the derivative is zero only when $a = \frac{q}{\left\lceil \frac{l-b}{b+1} \right\rceil}$. Therefore $n(a)$ has a single stationary point, which is a local maximum, thus n is maximal when a takes this value. When l is sufficiently large with respect to b and q this quantity is less than one, hence the optimal value for a (which has to be a positive integer) is 1. This is a plausible scenario, since the broadcaster is likely to wish to use a small value of q to reduce costs, which will necessitate a larger value of l. Furthermore, in the case of a binary alphabet a is necessarily equal to 1. When $a = 1$ we have that $n = q^{l-b-\left\lceil \frac{l-b}{b+1} \right\rceil}(q-1)^{\left\lceil \frac{l-b}{b+1} \right\rceil - 1}$. Ignoring the ceilings and treating this as a continuous function we see that

$$\begin{aligned} n(b) &= q^{l-b-\frac{l-b}{b+1}}(q-1)^{\frac{l-b}{b+1}-1}, \\ &= e^{\ln q\left(l-b-\frac{l+1}{b+1}+1\right)+\ln(q-1)\left(\frac{l+1}{b+1}-2\right)}. \end{aligned}$$

Taking the derivative with respect to b yields

$$\begin{aligned} \frac{dn}{db} &= \left(\ln q\left(-1+\frac{l+1}{(b+1)^2}\right) - \ln(q-1)\frac{l+1}{(b+1)^2}\right) q^{l-b-\frac{l-b}{b+1}}(q-1)^{\frac{l-b}{b+1}-1}, \\ &= \left(\frac{l+1}{(b+1)^2}(\ln q - \ln(q-1)) - \ln q\right) q^{l-b-\frac{l-b}{b+1}}(q-1)^{\frac{l-b}{b+1}-1}. \end{aligned}$$

The final two factors in this expression are non-zero, hence $n(b)$ has stationary points when

$$(b+1)^2 = (1-\log_q(q-1))(l+1).$$

This equation has two solutions, one with $b > 0$ that corresponds to a local maximum, the other with $b < 0$. Hence when b is restricted to the region $0 \leq b \leq \frac{l-1}{2}$ the value of n is maximised when b satisfies the above equation. In the binary case this amounts to $(b+1)^2 = l+1$. This accords with what we have seen with Construction 6.11 where a code with minimal $l = k + 2\sqrt{k}$ was found to result from $b = \sqrt{k}$, since $(\sqrt{k}+1)^2 = k + 2\sqrt{k} + 1$.

In practice numerical calculations can be used to determine the optimal value of b for a particular q and l, but this analysis at least gives an idea of the expected results.

7.4 The Case where $b = 0$

The quantity n is of the form $q^{l-b+1} + O(q^{l-b-2})$. The degree of the leading term is maximal when $b = 0$, hence for sufficiently large q choosing a scheme with $b = 0$ will result in the greatest number of users being protected. In this case the slight gain represented by having more ordinary segments with q symbols to divide among the users instead of the $q-a$ available in a protection segment is offset by the fact that there are up to b ordinary segments at the end of a window that cannot guarantee protection, which means there are fewer multiples of $(q - a)$ or q contributing to the value of n.

When $b = 0$ we have $n = (q - a)^{l-1}a$. This is maximised when $a = \frac{q}{l}$, which leads to $n = \frac{(l-1)^{l-1}}{l^l}q^l$.

7.5 Improved Constructions of Sliding-Window Dynamic Frameproof Codes

We have a framework in which to analyse known constructions of sliding-window l-dynamic frameproof codes; we now wish to determine whether there exist constructions that can protect more users for given values of q and l. In fact better constructions do exist, as we will see in the following section. The codes resulting from Construction 7.1, while having many beneficial properties, are still inefficient in certain respects. For example they rely on the last protection segment of each window to protect any previously-framed users, without taking into account any protection resulting from protection segments

occurring earlier in the window. On account of this, in some windows the number of users framed prior to the final protection segment may be less than a, in which case to protect a users at that time is to fail to use the available resources with maximal efficiency. Information from the pirate broadcast is only used to determine which users are protected in the protection segments; in the following section we will see constructions in which this information is used not only for this purpose, but also for determining how non-unique marks are distributed at each time.

7.5.1 Improved Constructions with $b = 0$

In the interest of simplicity we begin by considering the case in which precisely a users receive unique marks in every segment. We find that it is possible to construct sliding-window l-dynamic frameproof codes with this property, that protect up to $a\big((q-a)^{l-1}+(q-a)^{l-2}+\cdots+(q-a)+1\big)$ users. This construction relies on the trivial observation that if the pirate broadcast at time t is ξ_t then any users requiring protection at time $t+1$ must also have received the symbol ξ_t at time t. In later sections we generalise it to the case where $b > 0$.

Construction 7.2. *This construction uses the alphabet $Q = \{0, 1, \ldots, q-1\}$ to protect $n = a\big((q-a)^{l-1} + (q-a)^{l-2} + \cdots + (q-a) + 1\big)$ users over windows of size l, with a users receiving unique marks in each segment.*

- *In the first time segment, give the marks $q-a, q-a+1, \ldots, q-1$ to the first a users. The remaining marks are to be distributed evenly among the remaining $a\big(q-a)^{l-1} + (q-a)^{l-2} + \cdots + (q-a)\big)$ users; we observe that each mark is received by $a\big((q-a)^{l-2} + (q-a)^{l-3} + \cdots + (q-a) + 1\big)$ of those users.*

- *Denote the pirate broadcast at time 1 by* ξ_1*. In the second time segment give the marks* $q-a, q-a+1, \ldots, q-1$ *to the first* a *users who received the mark* ξ_1 *at time 1. Distribute the first* $q-a$ *symbols evenly among the rest of the users who were framed at time 1; each mark is thus received by* $a\big((q-a)^{l-3}+(q-a)^{l-4}+\cdots+(q-a)+1\big)$ *of these users. Then distribute those symbols evenly among the remaining users.*

- *Repeat this process for the first* $l-1$ *segments as follows: at time* t *protect the first* a *users who have been framed over the first* $t-1$ *segments. Distribute the first* $q-a$ *symbols evenly among the remaining users who have been framed over those segments. Then consider the set of users who have been framed over segments* 2 *to* $t-1$*; give the symbols* 0 *to* $q-a-1$ *to any users of this set who have yet to be allocated a mark so that these symbols are distributed evenly among the users in this set other than those protected at time* t*. Note that the number of users framed over this time is* $a\big((q-a)^{l-1-(t-2)}+(q-a)^{l-2-(t-2)}+\cdots+(q-a)+1\big)$ *and* a *of them have been protected at time* t*, hence each of the first* $q-a$ *symbols is received by* $a\big((q-a)^{l-2-(t-2)}+(q-a)^{l-3-(t-2)}+\cdots+(q-a)+1\big)$ *users in this set. Then repeat this process with the set of users who have been framed over segments* 3 *to* $t-1$ *and so on until all users have received marks.*

- *The number of users who have been framed over the first* $l-1$ *segments is* a*; at time* l *give these users the unique marks* $q-a, q-a+1, \ldots, q-1$*. Then consider any remaining users who have been framed over the previous* $l-2$ *segments and allocate the first* $q-a$ *symbols to them so that these symbols are evenly distributed on the set of all unprotected users*

framed over these segments. Then repeat this process with the set of users who have been framed over the previous $l-3$ segments, and so on until all users have received marks.

- *Repeat this process in all subsequent segments.*

This construction ensures that precisely a users are framed over the first $l-1$ segments of each window; these users are then protected in the final segment of the window, hence the resulting mark distribution constitutes a sliding-window l-dynamic frameproof code. This example shows how it works in practice.

Example 7.4 Let $l = 3$ and $q = 4$, with $a = 2$. Then the resulting scheme will protect $2(2^2+2+1) = 14$ users. The following table shows an example of the above construction being applied over 7 segments given a particular choice of pirate broadcast.

	1	2	3	4	5	6	7
u_0	2	0	0	2	0	0	0
u_1	3	0	0	3	0	0	0
u_2	0	2	0	0	2	0	0
u_3	0	3	0	0	3	0	0
u_4	0	0	2	0	0	1	0
u_5	0	0	3	0	0	1	0
u_6	0	1	0	1	0	1	1
u_7	0	1	0	1	0	1	1
u_8	1	0	1	0	1	2	1
u_9	1	0	1	0	1	3	1
u_{10}	1	1	1	1	1	0	1
u_{11}	1	1	1	1	1	0	1
u_{12}	1	1	1	1	1	1	2
u_{13}	1	1	1	1	1	1	3
T	0	0	0	0	1	1	0

We observe that it does not matter how the symbols are distributed among users who have not been framed, except that an overall even distribution of symbols in each segment must be achieved. Hence users u_{10} to u_{13} receive the

same sequence on the first five segments without ill effect, as none of them are framed on any of those segments. ■

This construction is clearly more efficient then Construction 7.1, since with the same parameters it can protect $a\big((q-a)^{l-1}+(q-a)^{l-2}+\cdots+(q-a)+1\big)$ users, compared with $a(q-a)^{l-1}$. In fact no greater number of users can be protected by a scheme in which precisely a users receive unique marks in each segment:

Theorem 7.3. *Suppose there exists a q-ary sliding-window l-dynamic frame-proof code protecting n users, in which a users receive unique marks during each segment, with $1 \leq a \leq q-1$. Then n satisfies*

$$n \leq a\big((q-a)^{l-1}+(q-a)^{l-2}+\cdots+(q-a)+1\big).$$

Proof. Suppose there exists such a code protecting n users where n satisfies $n \geq a\big((q-a)^{l-1}+(q-a)^{l-2}+\cdots+(q-a)+1\big)+1$. Suppose a pirate adopts the strategy over the first l time segments of always broadcasting a symbol that ensures that the greatest possible number of users has been framed over those segments. During the first time segment a users receive unique marks, which leaves $q-a$ marks to be distributed among the remaining users. By the pigeon-hole principle one such mark is received by at least h_1 users, where $h_1 = a\big((q-a)^{l-2}+(q-a)^{l-3}+\cdots+(q-a)+1\big)+1$; hence at least h_1 users are framed at this time. In the second time segment, there exists some symbol that is received by at least h_2 of the users framed at time 1, where h_2 is equal to $\left\lceil \frac{h_1-a}{q-a} \right\rceil = a\big((q-a)^{l-3}+(q-a)^{l-4}+\cdots+(q-a)+1\big)+1$. Thus the pirate broadcasts a symbol ensuring that at least h_2 users have been framed over the first two symbols. Applying this reasoning to the first $l-1$ time segments ensures that over the first i of these segments at least

$h_i = a\big((q-a)^{l-i-1} + (q-a)^{l-i-2} + \cdots + (q-a) + 1\big) + 1$ users have been framed, hence at least $a+1$ users are framed over the first $l-1$ segments. As only a users are protected in segment l, however, there exists at least one user who has been framed over the first $l-1$ segments yet is not protected at time l. If the pirate broadcasts the symbol received by that user then that user is framed over the entire window, contradicting the assumption that the code was sliding-window l-dynamic frameproof. Thus we conclude that for a sliding-window l-dynamic frameproof code of the desired properties we have $n \leq a\big((q-a)^{l-1} + (q-a)^{l-2} + \cdots + (q-a) + 1\big)$. □

7.5.2 General b

It is possible to generalise the above construction to achieve codes that are more efficient than those of Construction 7.1 in the case where $b > 0$.

Construction 7.4. *This construction is a modification of Construction 7.2; it protects* $aq^r\Big(\big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 1} + \big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 2} + \cdots + (q-a)q^b + 1\Big)$ *users, where* $r = \big(l - (b+1)\lceil \frac{l-b}{b+1} \rceil\big)$. *At each time the marks are distributed as in Construction 7.2, with two differences. Firstly, during ordinary segments (those occurring at times* j *where* $(b+1) \nmid j$*) no users are protected. Secondly, a sliding-window* l*-dynamic frameproof code for which* $b > 0$ *will guarantee that every window includes a sequence of* $l-b$ *segments ending in a protection segment, of which* $l-b-\lceil \frac{l-b}{b+1} \rceil$ *are ordinary segments and* $\lceil \frac{l-b}{b+1} \rceil - 1$ *are protection segments occurring prior to the last protection segment. We are interested in preventing framing over these sequences. Such a sequence will start with the* $\big(l - (b+1)\lceil \frac{l-b}{b+1} \rceil\big)^{th}$ *ordinary segment prior to a protection segment; we denote this quantity by* r. *Whereas in the case* $b = 0$ *symbols were allocated evenly to users who had been framed over* $l-1$ *segments, then* $l-2$, $l-3$ *and*

so on, in this case it is only necessary to consider sets of users who have been framed over sequences of up to $l-b$ consecutive segments that commence with the r^{th} ordinary segment prior to a protection segment. This is illustrated in the example below.

In order to show that the scheme resulting from this construction is indeed sliding-window l-dynamic frameproof we observe that each length l window contains an interval of $l-b$ consecutive segments ending in a protection segment, and show that no users can be framed over such an interval.

Consider an interval of $l-b$ consecutive segments ending in a protection segment. Let n_0 be the total number of users, and denote by n_i the number of users who are framed over the first i segments of the interval.

The first r segments are ordinary segments. Therefore in segment i, for $1 \leq i \leq r$ the q symbols are divided evenly among the users who were framed for the first $i-1$ segments (the fact that $q^r \mid n_0$ ensures this is possible). Thus we have $n_i = \frac{n_{i-1}}{q}$, hence $n_r = \frac{n_0}{q^r}$ and

$$n_r = a\Big(\big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 1} + \big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 2} + \cdots + (q-a)q^b + 1\Big).$$

Segment $r+1$ is a protection segment so a of the users who have been framed so far receive unique marks, and the remaining $q-a$ marks are divided evenly among the remaining framed users. Thus $n_{r+1} = \frac{n_r - a}{q-a}$, so

$$n_{r+1} = a\Big(\big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 1} + \big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 2} + \cdots + (q-a)q^b\Big)(q-a)^{-1}.$$

The following b segments are protection segments, and $q^b \mid n_{r+1}$ so in each of them the marks are distributed evenly among the framed users, which implies $n_{r+1+b} = \frac{n_{r+1}}{q^b}$, thus

$$n_{r+b+1} = a\Big(\big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 2} + \big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 3} + \cdots + (q-a)q^b + 1\Big).$$

Each subsequent sequence of $b+1$ segments consists of a protection segment followed by b ordinary segments, so $n_{r+\gamma(b+1)} = \frac{n_{r+(\gamma-1)(b+1)}-a}{(q-a)q^b}$. Thus after the γ^{th} such sequence following the first r segments we find

$$n_{r+\gamma(b+1)} = a\Big(\big((q-a)q^b\big)^{\lceil\frac{l-b}{b+1}\rceil-\gamma-1} + \big((q-a)q^b\big)^{\lceil\frac{l-b}{b+1}\rceil-\gamma-2} + \cdots + (q-a)q^b + 1\Big).$$

The $(l-b-1)^{th}$ segment of the interval occurs after $\lceil\frac{l-b}{b+1}\rceil - 1$ such sequences, therefore $n_{l-b-1} = a\Big(\big((q-a)q^b\big)^{\lceil\frac{l-b}{b+1}\rceil-(\lceil\frac{l-b}{b+1}\rceil-1)-1}\Big) = a$. These remaining a users are protected in the final protection segment, hence no users have been framed, irrespective of which marks the pirate chooses to broadcast.

Therefore we conclude that the code constructed above will indeed protect up to

$$aq^r\Big(\big((q-a)q^b\big)^{\lceil\frac{l-b}{b+1}\rceil-1} + \big((q-a)q^b\big)^{\lceil\frac{l-b}{b+1}\rceil-2} + \cdots + (q-a)q^b + 1\Big)$$

users. Note that in the case $b = 0$ this reduces to

$$a\big((q-a)^{l-1} + (q-a)^{l-2} + \cdots + (q-a) + 1\big)$$

as before.

Example 7.5 Let $l = 5$ and $b = 1$ with $a = 1$ and $q = 3$. Then $\lceil\frac{l-b}{b+1}\rceil = 2$, so the code resulting from the above construction protects 21 users. The table below is an example of a mark distribution that results over six time segments.

In segment 3 each of the three users u_4, u_5 and u_6 who have been framed over the first two segments gets a different symbol and then the symbols are distributed evenly among the rest of the users. In time 4 the unique user u_5 who was framed over the first three segments is protected, the symbols 0 and 1 are distributed evenly among the remaining users u_9 to u_{14} who were framed

in segment 3, then they are distributed evenly among the remaining users.

	1	2	3	4	5	6
u_0	0	2	0	0	0	0
u_1	0	0	0	0	0	0
u_2	0	0	0	0	0	0
u_3	0	0	0	0	0	0
u_4	0	1	0	0	0	0
u_5	0	1	1	2	0	0
u_6	0	1	2	0	1	0
u_7	1	0	0	0	1	1
u_8	1	0	0	1	1	1
u_9	1	0	1	0	1	1
u_{10}	1	0	1	0	1	1
u_{11}	1	0	1	0	1	1
u_{12}	1	0	1	1	0	1
u_{13}	1	0	1	1	1	1
u_{14}	2	1	1	1	2	2
u_{15}	2	1	2	1	2	0
u_{16}	2	1	2	1	2	0
u_{17}	2	1	2	1	2	0
u_{18}	2	1	2	1	2	1
u_{19}	2	1	2	1	2	1
u_{20}	2	1	2	1	2	1
T	0	1	1	1	2	0

■

This construction is in fact optimal for the given parameters: the proof of Theorem 7.3 can be modified to show

Theorem 7.5. *Suppose there exists a q-ary sliding-window l-dynamic frame-proof code in which every $b+1^{th}$ segment is a protection segment where a users receive unique marks and the remaining segments are ordinary segments. If this code supports n users then n satisfies*

$$n \leq aq^{l-(b+1)\lceil \frac{l-b}{b+1} \rceil} \Big(\big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 1} + \big((q-a)q^b\big)^{\lceil \frac{l-b}{b+1} \rceil - 2} + \cdots + (q-a)q^b + 1 \Big).$$

7.6 Asymptotic Results

If we consider the behaviour of the above upper bound as $q \to \infty$ we see that the degree of the leading term is $l - b - 1$, which is maximised when $b = 0$, in which case the leading term reduces to $a(q-a)^{l-1}$. In the case where $q \mid l$ this is maximised by setting $a = \frac{q}{l}$; this results in a leading term of size $\frac{(l-1)^{l-1}}{l^l} q^l$. Hence we have the following.

Theorem 7.6. *A sliding-window l-dynamic frameproof code that uses evenly-spaced protection segments and protects the same number of users in each protection segment with an alphabet of size q can protect at most n users, where*

$$n \leq \frac{(l-1)^{l-1}}{l^l} q^l + O(q^{l-1}),$$

as $q \to \infty$ with l fixed.

Also, since the restriction of a q-ary sliding-window l-dynamic frameproof code to the first l time segments yields a dynamic frameproof code that can support at most $q^{l-1}(q-1)$ users we have the following:

Corollary 7.7. *A sliding-window l-dynamic frameproof code using an alphabet of size q can support at most n users where*

$$n \leq q^l + O(q^{l-1}),$$

as $q \to \infty$ with l fixed.

7.7 Future Possibilities

In order to study sliding-window dynamic frameproof codes of complete generality, it would be necessary to consider codes in which the value of a varied

with each segment. By letting a range between 0 and $q-1$ this would encompass all possible sliding-window dynamic frameproof codes. The discussion of Section 6.2 gives some indication of the difficulties inherent in attempting to bound the sizes of such codes. There remains the open problem *do there exist q-ary sliding-window l-dynamic frameproof codes supporting more users than the ones discussed above?* Also, there is a discrepancy between the two asymptotic results given in the previous section: the degree of the leading term is the same in each case, but the coefficient differs. Thus we have the related question *do there exist q-ary sliding-window l-dynamic frameproof codes of size $cq^l + O(q^{l-1})$ where $c > \frac{(l-1)^{l-1}}{l^l}$?*

In the case where $a = 1$ and $b = 0$ the above construction yields a code protecting $(q-1)^{l-1} + (q-2)^{l-2} + \cdots + (q-1) + 1$ users. When q is one more than a prime power this is equal to the number of points in the $(l-1)$-dimensional projective space $PG(l-1, q-1)$. This raises the question of whether a geometric interpretation of this construction is possible.

The sequential frameproof model assumes complete ignorance of the pirate broadcast, whereas the dynamic model assumes total knowledge of this broadcast. All of the sliding-window dynamic frameproof schemes described so far require the knowledge of the most recent $l-1$ segments of the pirate broadcast. Is it possible to construct schemes that only require knowledge of the most recent m segments for some $m < l-1$? Such schemes would fit between the sequential and dynamic models.

7.8 Conclusion

In this thesis we set out to investigate whether the concept of a frameproof code could be applied in a dynamic setting to prevent pirates from framing

innocent users when fingerprinting is used with digital broadcasts. We have shown that this is indeed possible. We now have a good understanding of the behaviour of sequential frameproof codes, dynamic frameproof codes, and sliding-window l-frameproof codes with regular protection. Further study will be needed to determine the optimal size of a set of users protected by a general q-ary sliding-window l-dynamic frameproof code, and as discussed above, other interesting problems remain open in this area.

Bibliography

[1] Noga Alon and Uri Stav. New bounds on parent-identifying codes: The case of multiple parents. preprint, 2002. available from `www.math.tau.ac.il/~nogaa/`.

[2] Alexander Barg, G. Robert Blakely, and Gregory Kabatiansky. Digital fingerprinting codes: Problems statements, constructions, identification of traitors. Technical Report 2001-52, DIMACS, December 2001. available from `http://dimacs.rutgers.edu.TechnicalReports`.

[3] Alexander Barg and Gregory Kabatiansky. A class of I.P.P. codes with efficient identification. Technical Report 2002-36, DIMACS, September 2002. available from `http://dimacs.rutgers.edu.TechnicalReports`.

[4] Omer Berkman, Michal Parnas, and Jiří Sgall. Efficient dynamic traitor tracing. *SIAM Journal on Computing*, 30:1802–1828, 2001.

[5] Simon R. Blackburn. Combinatorial schemes for protecting digital content. In C. D. Wensley, editor, *Surveys in Combinatorics 2003*, volume 307 of *LMS lecture notes series*, pages 43–78. Cambridge University Press, 2003.

[6] Simon R. Blackburn. Frameproof codes. *SIAM Journal on Discrete Mathematics*, 16(3):499–510, 2003.

[7] Dan Boneh and Matthew Franklin. An efficient public key traitor tracing scheme. In M. Wiener, editor, *Advances in Cryptology -Crypto '99*, volume 1666 of *LNCS*, pages 338–353. Springer-Verlag, 1999.

[8] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, 1998.

[9] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In *Advances in Cryptology -Crypto '94*, volume 839 of *LNCS*, pages 257–270. Springer-Verlag, 1994.

[10] Benny Chor, Amos Fiat, Moni Naor, and Benny Pinkas. Tracing traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, May 2000.

[11] Gérard D. Cohen and Sylvia B. Encheva. Efficient constructions of frameproof codes. *Electronics Letters*, 36:1840–1842, 2000.

[12] Amos Fiat and Tamir Tassa. Dynamic traitor tracing. In *Advances in Cryptology -Crypto '99*, volume 1666 of *LNCS*, pages 354–371. Springer-Verlag, 1999.

[13] Reihaneh Safavi-Naini and Yejing Wang. Sequential traitor tracing. *IEEE Transactions on Information Theory*, 49(5):1319–1326, 2003.

[14] Jessica N. Staddon, Douglas R. Stinson, and Ruizhong Wei. Combinatorial properties of frameproof and traceability codes. *IEEE Transactions on Information Theory*, 47:1042–1049, 2001.

[15] Douglas R. Stinson and Ruizhong Wei. Combinatorial properties and constructions of traceability schemes and frameproof codes. *SIAM Journal on Discrete Mathematics*, 11:41–53, 1998.

[16] Roger James Stockwell. *Frameproof Codes: Combinatorial Properties and Constructions*. PhD thesis, Royal Holloway University of London, 2002.

[17] Mitchell D. Swanson, Mei Kobayashi, and Ahmed H. Tewfik. Multimedia data-embedding and watermarking technologies. *Proceedings of the IEEE*, 86:1064–1087, 1998.

[18] Dongvu Tonien and Reihaneh Safavi-Naini. Recursive constructions of secure codes and hash families using difference function families. Cryptology ePrint Archive, Report 2005/184, 2005. `http://eprint.iacr.org/`.

[19] Tran van Trung and Sosina Martirosyan. On a class of traceability codes. preprint, University of Essen, 2002. available from `www.exp-math.uni-essen.de/~trung/`.