Generic constructions of PoRs from codes and instantiations

Julien Lavauzelle, Françoise Levy-Dit-Vehel

Generic constructions of PoRs from codes and instantiations

Julien Lavauzelle [†] Françoise Levy-dit-Vehel [‡]

Abstract

In this paper, we show how to construct, from any linear code, a Proof of Retrievability (PoR) which features very low computation complexity on both the client (Verifier) and server (Prover) side, as well as small client storage (typically 512 bits). We adapt the security model initiated by Juels and Kaliski [JK07] to fit into the framework of Paterson *et al.* [PSU13], from which our construction evolves. We thus provide a rigorous treatment of the security of our generic design; more precisely, we sharply bound the extraction failure of our protocol according to this security model. Next, we instantiate our formal construction with codes built from tensor-products as well as with Reed-Muller codes and lifted codes [GKS13], yielding PoRs with moderate communication complexity and (server) storage overhead, in addition to the aforementioned features.

1 Introduction

1.1 Motivation

Cloud computing and storage has evolved quite spectacularly over the past decade. Especially, data outsourcing allows users and companies to lighten their storage burden and maintenance cost. Though, it raises several issues: for example, how can someone check efficiently that he can retrieve without any loss a massive file that he had uploaded on a distant server and erased from his personal system?

Proofs of retrievability (PoRs) address this issue. They are cryptographic protocols involving two parts: a client (or a verifier) and a server (or a prover). PoRs usually consist in the following phases. First, a *key generation* process creates secret material related to the file, meant to be kept by the client only. Then the file is *initialised*, that is, it is encoded and/or encrypted according to the secret data held by the client. This processed file is uploaded to the server. In order to check retrievability, the client can run a *verification procedure*, which is the core of the PoR. Finally, if the client is convinced that the server still holds his file, the client can proceed at any time to the *extraction* of the file.

Several parameters must be taken into account. Plainly, the verification process has to feature a low communication complexity, as the main goal is to avoid downloading a large part of the file to only *check* its extractability. Second, the storage overhead induced by the protocol must be low, as large server overhead would imply high fees for the customer. Third, the computation cost of the verification procedure must be low, both for the client (which is likely to own a lightweight device) and the server (whose computation work could also be expensive for the client).

*This paper appears in: Journal of Mathematical Cryptology, 2019, DOI:10.1515/jmc-2018-0018

[†]LIX, École Polytechnique, Inria & CNRS UMR 7161, Université Paris-Saclay, 1 rue Honoré d'Estienne d'Orves, Bâtiment Alan Turing, 91120, Palaiseau, France, julien.lavauzelle@inria.fr

[‡]ENSTA ParisTech, Inria & LIX, 828 boulevard des Maréchaux, 91762 Palaiseau, France, levy@ensta.fr

Notice that *proofs of data possession* (PDP) represent protocols close to what is needed in PoRs. However, in PDPs one does not require the client to be able to extract the file from the server. Instances of PDPs are given by Ateniese *et al.* [ABC+11]. Besides, protocols of Lillibridge *et al.* [LEB+03] and Naor and Rothblum [NR09] are very often seen as precursors for PoRs. For instance, the work of Naor and Rothblum [NR09] considers a setting in which the client directly accesses the file stored by the prover/server (while the actual PoR definition uses "an arbitrary program as opposed to a simple memory layout and this program may answer these questions in an arbitrary manner" [SW13]).

1.2 Previous work

Juels and Kaliski [JK07] gave the first formal definition of PoRs. They also proposed a first construction based on so-called *sentinels* (namely, random parts of the file to be checked during the verification step) the client keeps secretly on his device. Additionally, an erasure code ensures the integrity of the file to be extracted. This seminal work also raised several interesting points. On the one hand, it revealed that (i) the client must store secret data to be used in the verification step and (ii) coding is needed in order to retrieve the file without erasures or errors. On the other hand, in JK's construction the verification step can only be performed a finite number of times, since sentinels cannot be reused endlessly.

As a consequence, Shacham and Waters proposed to consider *unbounded-use* PoRs in [SW13], where they built two kinds of PoRs. The first one is based on linear combinations of authenticators produced *via* pseudo-random functions; its security was proved using cryptographic tools such as unforgeable MAC scheme, semantically secure symmetric encryption and secure PRFs. The second one is a publicly verifiable scheme based on the Diffie-Hellman problem in bilinear groups.

Bowers, Juels and Oprea [BJO09] adopted a coding-theoretic approach (*inner code, outer code*) to compare variants of SW and JK schemes. They focused on the efficiency of the schemes, and proved that, despite bounded-use, new variants of JK construction are highly competitive compared to other existing schemes.

In [PSU13], Paterson *et al.* provide a general framework for PoRs in the unconditional security model. They show that retrievability of the file can be expressed as error-correction of a so-called *response code*. That allows them to precisely quantify the extraction success as a function of the success probability of a proving algorithm: indeed, in this setting, extraction can be naturally seen as nearest-neighbour decoding in the response code. They notably apply their framework to prove the security of a modified version of SW scheme. Also notice that, prior to [PSU13], Dodis, Vahan and Wichs [DVW09] proposed another coding-theoretic model for PoRs that allowed them to build efficient bounded-use and unbounded-use PoR schemes.

With practicality in mind, other features have been deployed on PoRs. For instance, Wang *et al.* [WWR+11] presented a PoR construction based on Merkle hash trees, which allows efficient *file updates* on the server. Their scheme is provably secure under cryptographic assumptions (hardness of Diffie-Hellman in bilinear groups, unforgeable signatures, etc.), and has been improved by Mo, Zhou and Chen [MZC12] in order to prevent unbalanced trees. More recently, other features have been proposed for PoRs, such as multi-prover PoRs (see [PSU18]) or public verifiability (for instance in [SR16]).

1.3 Our approach

As we remarked before, most PoR schemes rely on two techniques: (i) the client locally stores secret data in order to check the integrity of the file and (ii) the client encodes the file in order to

repair a small number of erasures and errors that could have been missed during the verification step.

In this work, we propose to build PoR schemes using codes that fulfil the two previous goals, when equipped with a suitable family of efficiently computable random permutations. More precisely, our idea is the following. Given a file F, a code $\mathcal{C}$ and a family of random permutations σ_K, the client sends to the server an encoded and scrambled version $\sigma_K(\mathcal{C}(F))$ of his file. Then, the verification step consists in checking "short" relations among descrambled symbols of $w = \mathcal{C}(F)$, which come for instance from low-weight parity-check equations for $\mathcal{C}$. Moreover, during the extraction step, the code $\mathcal{C}$ provides the redundancy necessary to repair erasures and potential unnoticed errors.

In the present work we develop a seminal idea that appeared in [LL16] where the authors proposed a construction of PoRs based on lifted codes. We here provide a more generic construction, and give a deeper analysis of its security.

While our scheme does not feature updatability nor public verifiability, we emphasize the genericity of our construction, which is based on well-studied algebraic and combinatorial structures, namely codes and their parity-check equations. Moreover, since the code $\mathcal{C}$ is public, the client must only store the secret material associated to the random permutations σ_K, which consist in a few bytes. Besides, an honest server simply needs to read pieces of w during the verification step, and therefore has very low computational burden compared to many other PoR schemes.

1.4 Organisation

Section 2 is devoted to the definition and security model of proofs of retrievability. Despite the great disparity of models in PoR literature, we try to keep close to the definitions given in [JK07, PSU13], for the sake of uniformity.

Section 3 presents our construction of PoR. Precisely, in Subsection 3.1, we introduce objects called *verification structures for a code* $\mathcal{C}$ that will be used in the definition of our PoR scheme (Subsection 3.2). A rigorous analysis of our scheme is the purpose of the remainder of that section.

The performance of our generic construction is given in Section 4. We then provide several instances in Section 5, proving the practicality of our PoR schemes for some classes of codes.

2 Proofs of retrievability

2.1 Definition of underlying protocols

We recall that in proofs of retrievability, a user wants to estimate if a message m can be retrieved from a encoded version w of the message stored on a server. In all what follows, the user will be known as the Verifier (wants to verify the retrievability of the message) while the server is the Prover (aims at proving the retrievability). The *message space* is denoted by $\mathcal{M}$ while $\mathcal{W}$, the *(server) file space*, is the set of encoded versions of the messages. We also denote by $\mathcal{K}$ the set of *secret values* (or *keys*) kept by the Verifier, and by $\mathcal{R}$ the space of responses to challenges.

Throughout the paper, symbols $\leftarrow_{\mathrm{R}}$ and $\leftarrow$ respectively denote the output of randomised and deterministic algorithms.

Definition 2.1. A keyed *proof of retrievability* (PoR) is a tuple of algorithms (KeyGen, Init, Verify, Extract) running as follows:

1. The *key generation* algorithm KeyGen generates uniformly at random a key $\kappa \leftarrow_{\mathrm{R}} \mathcal{K}$. The key κ is secretly kept by the Verifier.
2. The *initialisation algorithm* Init is a deterministic algorithm which takes as input a message $m \in \mathcal{M}$ and a key $\kappa \in \mathcal{K}$, and outputs a file $w \in \mathcal{W}$. Init is run by the Verifier which initially holds the message m. After the process, the file w is sent to the Prover and the message m is erased on Verifier's side. Upon receipt of w, the Prover sets a deterministic algorithm $\mathsf{P}^{(w)}$ that will be run during the verification procedure.
3. The *verification algorithm* Verify is a randomised algorithm initiated by the Verifier which needs a secret key $\kappa \in \mathcal{K}$ and interacts with the Prover. Verify is depicted in Figure 1 and works as follows:
   (i) the Verifier runs a random query generator that outputs a challenge $u \leftarrow_{\mathrm{R}} \mathcal{Q}$ (the set $\mathcal{Q}$ being the so-called *query set*);
   (ii) the challenge u is sent to the Prover;
   (iii) the Prover outputs a response $r_u \leftarrow \mathsf{P}^{(w)}(u) \in \mathcal{R}$;
   (iv) the Verifier checks the validity of r_u according to u and κ: the algorithm Verify finally outputs the boolean value $\mathsf{Check}(u, r_u, \kappa)$.
4. The *extraction algorithm* Extract is run by the Verifier. It takes as input κ and $r = (r_u : u \in \mathcal{Q}) \in \mathcal{R}^{\mathcal{Q}}$, and outputs either a message $m' \in \mathcal{M}$, or a failure symbol $\perp$. We say that extraction *succeeds* if $\mathsf{Extract}(r, \kappa) = m$.

The vector $r = (r_u \leftarrow \mathsf{P}^{(w)}(u))_{u \in \mathcal{Q}} \in \mathcal{R}^{\mathcal{Q}}$ is called the *response word* associated to $\mathsf{P}^{(w)}$.

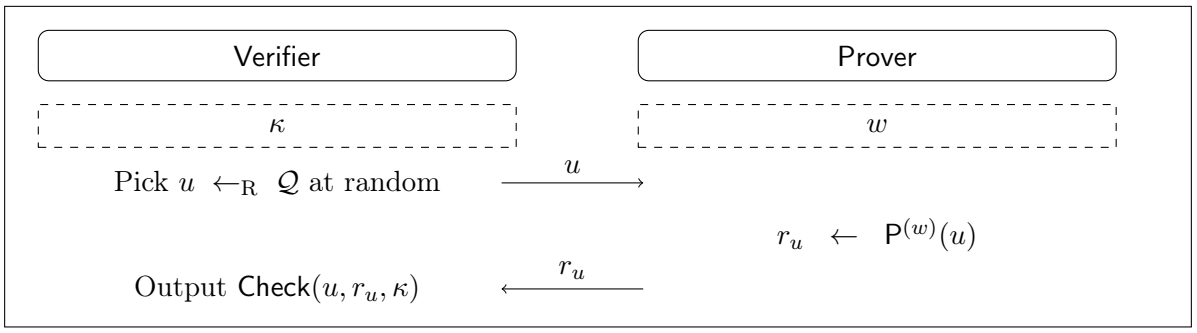

Figure 1: Definition of algorithm Verify

Note that, in assuming that the response algorithm $\mathsf{P}^{(w)}$ is deterministic and non-adaptive[1] we follow the work of Paterson *et al.* [PSU13]. The authors justify determinism of response algorithms by the fact that any probabilistic prover can be replaced by a deterministic prover whose success probability is at least as good as the probabilistic one.

In Definition 2.1, we can see that a deterministic algorithm $\mathsf{P}^{(w)}$ can be represented by the vector of its outputs $r = (\mathsf{P}^{(w)}(u))_{u \in \mathcal{Q}}$, called the response word of $\mathsf{P}^{(w)}$. Therefore, we can assume that before the verification step, the Prover produces a word $r^{(w)} \in \mathcal{R}^{\mathcal{Q}}$ related to the file w he holds. In other words, we model provers as algorithms P which, given as input w, return a word $r \in \mathcal{R}^{\mathcal{Q}}$.

Following [PSU13], we also assume in this chapter that the extraction algorithm Extract is deterministic, though in general it can be randomised. Finally, notice that proofs of retrievability aim at *proving* the extractability of a file. The extraction algorithm is therefore a tool to retrieve the whole file. Hence, its computational efficiency is not a crucial feature.

[1] in the sense that its behaviour only depends on the values of challenges u, and not on previous calls to the verification procedure

The following table summarises the information held by each entity after the initialisation step:

Verifier	Prover
κ	w

Let us also report the inputs and outputs of the algorithms involved in a PoR:

algorithm	KeyGen	Init	Verify	Check	Extract
input	1^λ	m, κ	r, κ	u, r_u, κ	r, κ
output	κ	w	`True` or `False`	`True` or `False`	m' or $\perp$

2.2 Security models

One should first notice that, despite many efforts, proofs of retrievability lack a general agreement on the definition of their security model. Nevertheless, our definitions remain very close to the ones given in the original work of Juels and Kaliski [JK07].

For a response word $r \in \mathcal{R}^{\mathcal{Q}}$ given by the Prover and a key $\kappa \in \mathcal{K}$ kept by the Verifier, we first define the *success of r according to κ* as:

$$\mathrm{succ}(r, \kappa) \coloneqq \Pr_u \left(\mathsf{Check}(u, r_u, \kappa) = \texttt{True} \right),$$

where the probability is taken over the internal randomness of Verify. A first security model can be defined as follows.

Definition 2.2 (security model, strong version)**.** Let $\varepsilon, \tau \in [0, 1]$. A proof of retrievability $(\mathsf{KeyGen}, \mathsf{Init}, \mathsf{Verify}, \mathsf{Extract})$ is *strongly (ε, τ)-sound* if, for every initial file $m \in \mathcal{M}$, every uploaded file $w \in \mathcal{W}$ and every prover $\mathsf{P} : \mathcal{W} \to \mathcal{R}^{\mathcal{Q}}$ we have:

$$\Pr \left(\begin{array}{c} \mathsf{Extract}(r, \kappa) \neq m \\ \text{and} \\ \mathrm{succ}(r, \kappa) \geq 1 - \varepsilon \end{array} \middle| \begin{array}{c} \kappa \leftarrow_{\mathrm{R}} \mathsf{KeyGen}(1^\lambda) \\ w \leftarrow \mathsf{Init}(m, \kappa) \\ r \leftarrow \mathsf{P}(w) \end{array} \right) \leq \tau, \tag{1}$$

the probability being taken over the internal randomness of KeyGen under the constraint that $w = \mathsf{Init}(m, \kappa)$.

A remark concerning parameters ε and τ. In proofs of retrievability, we aim at making the extraction of the desired file m as sure as possible when the audit succeeds. Hence, it is desirable to have τ small. On the other hand, the parameter ε measures the rate of unsuccessful audits which leads the Verifier to believe the extraction will fail. Therefore, one does not necessarily need to look for large values of ε, though in practice, large ε afford more flexibility, for instance if communication errors occur between the Prover and the Verifier during the verification procedure.

Definition 2.2 provides a strong security model, in the sense that (i) it does not require any bound on the response algorithms given by the Prover (ii) the probability in (1) is taken over fixed messages m (informally, it means the Prover knows m).

However, keyed proofs of retrievability are usually insecure according to the security model given in Definition 2.2. For instance, in [PSU13] Paterson *et al.* noticed that in the Shacham-Waters scheme [SW13], given the knowledge of m and w, an unbounded Prover may be able to

1. compute (or at least randomly guess) a key κ such that $\mathsf{Init}(m, \kappa) = w$,
2. build $m' \neq m$ such that $\mathsf{Init}(m', \kappa) = w'$, and
3. set $\mathsf{P}^{(w')} = r'$ which (a) successfully passes every audit and (b) leads to the extraction of $m' \neq m$.

Hence, we choose to use a weaker but still realistic security model, where informally, the Prover only knows what he stores (that is, w) and has no information on the initial message m. The following security model thus remains conform with the one given by Paterson *et al.* [PSU13].

Definition 2.3 (security model, weak version)**.** Let $\varepsilon, \tau \in [0,1]$. A proof of retrievability $(\mathsf{KeyGen}, \mathsf{Init}, \mathsf{Verify}, \mathsf{Extract})$ is *weakly* (ε,τ)*-sound* (or simply (ε,τ)-sound) if, for every polynomial-time prover $\mathsf{P} : \mathcal{W} \to \mathcal{R}^{\mathcal{Q}}$ and every uploaded file $w \in \mathcal{W}$, we have:

$$\Pr\left(\begin{array}{c|c} \mathsf{Extract}(r,\kappa) \neq m \\ \text{and} \\ \mathrm{succ}(r,\kappa) \geq 1-\varepsilon \end{array} \middle| \begin{array}{c} m \leftarrow_{\mathrm{R}} \mathcal{M} \\ \kappa \leftarrow_{\mathrm{R}} \mathsf{KeyGen}(1^\lambda) \\ w \leftarrow \mathsf{Init}(m,\kappa) \\ r \leftarrow \mathsf{P}(w) \end{array}\right) \leq \tau\,. \tag{2}$$

In Equation (2), the randomness comes from pairs $(m,\kappa) \in \mathcal{M} \times \mathcal{K}$ picked uniformly at random among those satisfying $w = \mathsf{Init}(m,\kappa)$.

Since we deal with values of τ very close to 0, we also say that a strongly (ε,τ)-sound PoR admits $\lambda = -\log_2(\tau)$ bits of security against ε-adversaries.

Informally, saying that a PoR is *not* weakly sound amounts to finding a polynomial-time deterministic algorithm P which

- takes as input a file $w \in \mathcal{W}$ and outputs a response word $r \in \mathcal{R}^{\mathcal{Q}}$,
- makes the extraction fail with non-negligible probability (over messages m and keys κ such that the corresponding response words are successfully audited).

3 Our generic construction

Schematically, in the initialisation phase of our construction, the Verifier

(i) encodes his file according to a code $\mathcal{C}$;
(ii) scrambles the resulting codeword using a tuple of permutations over the base field;
(iii) uploads the result to the Prover.

As we explained in the introduction, the verification step then consists in checking that the server is still able to give answers that, once descrambled, satisfy low-weight parity-check equations for $\mathcal{C}$.

For this purpose, we next introduce objects called *verification structures* for codes, which will be used in the definition of our generic PoR scheme.

3.1 Verification structures: a tool for our PoR scheme

We here consider $\mathbb{F}_q$, the finite field with q elements. From well-known coding theory terminology, the *support* of a word $w \in \mathbb{F}_q^n$ is $\mathrm{supp}(w) \coloneqq \{i \in [1,n], w_i \neq 0\}$, and its *weight* is $\mathrm{wt}(w) \coloneqq |\mathrm{supp}(w)|$.

In this work, we need to consider codes whose alphabets are finite dimensional spaces $\mathcal{R}$ over $\mathbb{F}_q$, typically $\mathcal{R} = \mathbb{F}_q^s$. Precisely, a code $\mathcal{C}$ of length n over $\mathcal{R}$ is a subset of $\mathcal{R}^n$. A code $\mathcal{C} \subseteq \mathcal{R}^n$ is $\mathbb{F}_q$*-linear* if $\mathcal{C}$ is a vector space over $\mathbb{F}_q$. When $\mathcal{R} = \mathbb{F}_q$, we get the usual definition of linear codes over finite fields. Unless stated otherwise, we only consider $\mathbb{F}_q$-linear codes, that we will refer to as *codes*.

We usually denote by k the dimension over $\mathbb{F}_q$ of a code $\mathcal{C}$. Its *minimum distance* $d_{\min}(\mathcal{C})$ is the smallest Hamming distance between two distinct codewords. If n is the length of $\mathcal{C}$, then

$d_{\min}(\mathcal{C})/n \in [0,1]$ is the *relative minimum distance* of the code $\mathcal{C}$, while k/n represents its *rate*. If $\mathcal{C} \subseteq \mathbb{F}_q^n$, its dual code $\mathcal{C}^\perp$ is defined as $\{h \in \mathbb{F}_q^n, \sum_{i=1}^n h_i c_i = 0, \forall c \in \mathcal{C}\}$. Codewords in $\mathcal{C}^\perp$ are also called *parity-check equations for* $\mathcal{C}$.

Definition 3.1 (Verification structure)**.** Let $1 \leq \ell \leq n$ and $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a code. Let also $\mathcal{Q}$ be a non-empty set of ℓ-subsets of $[1,n]$. Set $\mathcal{R} = \mathbb{F}_q^\ell$. We define the *restriction map* R associated to $\mathcal{Q}$ as:

$$\begin{array}{rccc} R: & \mathcal{Q} \times \mathbb{F}_q^n & \to & \mathcal{R} \\ & (u, w) & \mapsto & w_{|u} \end{array}$$

Given an integer $s \geq 1$ and a map $V : \mathcal{Q} \times \mathcal{R} \to \mathbb{F}_q^s$, we say that $(\mathcal{Q}, V)$ is a *verification structure* for $\mathcal{C}$ if the following holds:

1. for all $i \in [1,n]$, there exists $u \in \mathcal{Q}$ such that $i \in u$;
2. for all $u \in \mathcal{Q}$, the map $\mathbb{F}_q^n \to \mathbb{F}_q^s$ given by $a \mapsto V(u, R(u,a))$ is surjective and vanishes on the code $\mathcal{C}$. Explicitly,
$$\forall c \in \mathcal{C}, V(u, R(u,c)) = 0\,.$$

The map V is then called a *verification map* for $\mathcal{C}$, and the set $\mathcal{Q}$ a *query set* for $\mathcal{C}$. By convention, for $w \in \mathbb{F}_q^n$ and $r \in \mathcal{R}^\mathcal{Q}$, we define

$$\begin{aligned} R(w) &\coloneqq (R(u,w) : u \in \mathcal{Q}) \in \mathcal{R}^\mathcal{Q}, \\ V(r) &\coloneqq (V(u, r_u) : u \in \mathcal{Q}) \in (\mathbb{F}_q^s)^\mathcal{Q}. \end{aligned}$$

Finally, the code $R(\mathcal{C}) \coloneqq \{R(c), c \in \mathcal{C}\}$ is called the *response code* of $\mathcal{C}$.

Example 3.2 (**Fundamental example**)**.** Let $\mathcal{C}$ be a code, and let $\mathcal{H}$ be a set of parity-check equations for $\mathcal{C}$ of Hamming weight ℓ, whose supports are pairwise distinct. Define the query set $\mathcal{Q} = \{\mathrm{supp}(h), h \in \mathcal{H}\}$, and for any $u \in \mathcal{Q}$, $h(u)$ to be the unique parity-check equation in $\mathcal{H}$ whose support is u. Finally, we define a map V by:

$$\begin{array}{rccc} V: & \mathcal{Q} \times \mathcal{R} & \to & \mathbb{F}_q \\ & (u, r) & \mapsto & \sum_{i=1}^\ell h(u)_{u_i} r_i \end{array}$$

Notice that we set $s = 1$ here. By construction, it is clear that $(\mathcal{Q}, V)$ is a verification structure for $\mathcal{C}$.

Example 3.3 (toy example)**.** Let $\mathcal{C} \subseteq \mathbb{F}_2^7$ be a binary Hadamard code of length $n = 7$ and dimension $k = 3$. In other words, $\mathcal{C}$ is defined by a parity-check matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

According to Example 3.2, we define $\mathcal{Q}$ to be the set of supports of rows of H. In other words,

$$\mathcal{Q} = \{\{1,2,3\}, \{1,4,5\}, \{1,6,7\}, \{2,5,6\}, \{2,4,7\}, \{3,4,6\}, \{3,5,7\}\}\,.$$

Then, the verification map $V : \mathcal{Q} \times \mathbb{F}_2^3 \to \mathbb{F}_2$ can be defined as follows. If $u = \{u_1, u_2, u_3\} \in \mathcal{Q}$ and $b \in \mathbb{F}_2^u$ is indexed according to u, then we define

$$V(u, b) = \sum_{i=1}^3 b_{u_i}\,.$$

Now, let $m = (m_1, m_2, m_3) \in \mathbb{F}_2^3$. The message m can be encoded into

$$c = (m_1,\ m_2,\ m_1 + m_2,\ m_3,\ m_1 + m_3,\ m_1 + m_2 + m_3,\ m_2 + m_3) \in \mathcal{C}\,.$$

Hence, the word $r = R(c) \in (\mathbb{F}_2^3)^7$ is:

$$\begin{aligned} r &= \left(\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}, \begin{pmatrix} c_1 \\ c_4 \\ c_5 \end{pmatrix}, \begin{pmatrix} c_1 \\ c_6 \\ c_7 \end{pmatrix}, \begin{pmatrix} c_2 \\ c_5 \\ c_6 \end{pmatrix}, \begin{pmatrix} c_2 \\ c_4 \\ c_7 \end{pmatrix}, \begin{pmatrix} c_3 \\ c_4 \\ c_6 \end{pmatrix}, \begin{pmatrix} c_3 \\ c_5 \\ c_7 \end{pmatrix} \right) \\ &= \left(\begin{pmatrix} m_1 \\ m_2 \\ m_1 + m_2 \end{pmatrix}, \begin{pmatrix} m_1 \\ m_3 \\ m_1 + m_3 \end{pmatrix}, \begin{pmatrix} m_1 \\ m_1 + m_2 + m_3 \\ m_2 + m_3 \end{pmatrix}, \begin{pmatrix} m_2 \\ m_1 + m_3 \\ m_1 + m_2 + m_3 \end{pmatrix}, \right. \\ &\qquad \left. \begin{pmatrix} m_2 \\ m_3 \\ m_2 + m_3 \end{pmatrix}, \begin{pmatrix} m_1 + m_2 \\ m_3 \\ m_1 + m_2 + m_3 \end{pmatrix}, \begin{pmatrix} m_1 + m_2 \\ m_1 + m_3 \\ m_2 + m_3 \end{pmatrix} \right) \end{aligned}$$

For each vector-coordinate $b \in \mathbb{F}_2^3$ of $r = R(c)$, one can now check that $\sum_j b_j = 0$. Hence, we get $V(R(c)) = 0$, as expected.

From now on, we denote by $N = |\mathcal{Q}|$ the length of the response code $R(\mathcal{C})$ of a code $\mathcal{C}$ equipped with a verification structure $(\mathcal{Q}, V)$.

3.2 Definition of our PoR scheme

Let $(\mathcal{Q}, V)$ be a verification structure for $\mathcal{C} \subseteq \mathbb{F}_q^n$, and let $\sigma \in \mathfrak{S}(\mathbb{F}_q)^n$, where $\mathfrak{S}(\mathbb{F}_q)$ denotes the set of permutations over $\mathbb{F}_q$. Any n-tuple of permutations $\sigma = (\sigma_1, \ldots, \sigma_n) \in \mathfrak{S}(\mathbb{F}_q)^n$ naturally acts on $c \in \mathbb{F}_q^n$ by:

$$\sigma(c) \mapsto (\sigma_1(c_1), \ldots, \sigma_n(c_n))\,,$$

and we define $\sigma(\mathcal{C}) = \{\sigma(c), c \in \mathcal{C}\}$. Let finally

$$\begin{array}{rccc} V^\sigma : & \mathcal{Q} \times \mathbb{F}_q^\ell & \to & \mathbb{F}_q^s \\ & (u, y) & \mapsto & V(u, \sigma_{|u}^{-1}(y)) \end{array}$$

where $\sigma_{|u}^{-1}(y) = (\sigma_{u_1}^{-1}(y_1), \ldots, \sigma_{u_\ell}^{-1}(y_\ell))$. The map V^σ has been defined in order to satisfy

$$V^\sigma(u, R(u, \sigma(c))) = V(u, R(u, c))$$

for every $(c, u) \in \mathcal{C} \times \mathcal{Q}$.

Based on this, our PoR construction is given in Figure 2.

3.3 Analysis

3.3.1 Preliminary results

We first give results concerning verification structures and response codes. The following two lemmata are straightforward to prove.

Lemma 3.4. *Let $(\mathcal{Q}, V)$ be a verification structure for a code $\mathcal{C} \subseteq \mathbb{F}_q^n$. Then $(\mathcal{Q}, V^\sigma)$ is a verification structure for $\sigma(\mathcal{C})$.*

Lemma 3.5. *Let $\mathcal{Q}$ be any query-set for a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ whose elements have cardinality $\ell \geq 1$. Then its response code $R(\mathcal{C})$ is an $\mathbb{F}_q$-linear code over the alphabet $\mathcal{R} \simeq \mathbb{F}_q^\ell$.*

The code $\mathcal{C}$ and the verification structure $(\mathcal{Q}, V)$ for $\mathcal{C}$ are public parameters. We assume that $\mathcal{C}$ is linear, and set $N = |\mathcal{Q}|$. We recall that $\mathcal{R} = \mathbb{F}_q^\ell$ and $\mathcal{W} = \mathbb{F}_q^n$.

• **Key generation:** The Verifier generates uniformly at random an n-tuple of permutations

$$(\sigma_1, \ldots, \sigma_n) = \sigma \leftarrow_{\mathrm{R}} \mathfrak{S}(\mathbb{F}_q)^n .$$

• **Initialisation:** The Verifier first encodes his file $m \in \mathbb{F}_q^k$ into a codeword $c \in \mathcal{C}$ with a systematic encoding algorithm for $\mathcal{C}$. Then, the Verifier scrambles each coordinate c_i using the permutation σ_i:

$$w_i = \sigma_i(c_i), \quad 1 \leq i \leq n .$$

Finally, $w \in \mathcal{W}$ is sent to the Prover, and m is erased by the Verifier. To sum up, the deterministic algorithm Init is defined by

$$\mathsf{Init}(m, \sigma) \coloneqq w = \sigma(\mathcal{C}(m)) \in \mathcal{W} .$$

Based on his knowledge of w and public parameters, the Prover produces a word $r \leftarrow \mathsf{P}(w)$, $r \in \mathcal{R}^{\mathcal{Q}}$, which corresponds to the vector of outputs of the deterministic proving algorithm P on input w.

• **Verification:**

1. The Verifier picks uniformly at random $u = (u_1, \ldots, u_\ell) \leftarrow_{\mathrm{R}} \mathcal{Q}$. Then, the Verifier sends u to the Prover, meaning the Prover is asked to send back $R(u, w) = w_{|u} \in \mathbb{F}_q^\ell$ to the Verifier.
2. The Prover sends back the u-th coordinate $r_u \in \mathcal{R}$ of his response word r to the Verifier.
3. On input $r_u \in \mathcal{R}$, the Verifier runs $V^\sigma(u, r_u)$ and outputs the result. Here we mean that:

$$\mathsf{Check}(u, r_u, \sigma) \coloneqq \begin{cases} \texttt{True} & \text{if } V^\sigma(u, r_u) = 0 \\ \texttt{False} & \text{otherwise.} \end{cases}$$

• **Extraction:** The Verifier first collects $r = (\mathsf{P}(w)_u : u \in \mathcal{Q}) \in \mathcal{R}^{\mathcal{Q}}$. Then, he runs the extraction procedure given in Figure 3, on input σ and r, and he outputs his result.

Figure 2: Definition of our PoR scheme

Input: $\sigma \in \mathfrak{S}(\mathbb{F}_q)^n$ and $r \in \mathcal{R}^{\mathcal{Q}}$.
Output: $m \in \mathbb{F}_q^k$ or a failure symbol $\perp$.

1. Define $r' = \sigma^{-1}(r)$.
2. On challenges $u \in \mathcal{Q}$ such that $V(u, r'_u) \neq 0$, assign $r'_u \leftarrow \perp$.
3. Run a bounded-distance error-and-erasure decoding algorithm for $R(\mathcal{C})$ with input $r' \in (\mathcal{R} \cup \{\perp\})^{\mathcal{Q}}$. It outputs either a word $m' \in \mathbb{F}_q^k$, or the failure symbol $\perp$.
4. Return this output.

Figure 3: Our extraction procedure $\mathsf{Extract}(r, \sigma)$.

Remark 3.6. By considering $\sigma(\mathcal{C})$ instead of $\mathcal{C}$, we loose the $\mathbb{F}_q$-linearity, but one can check that verification structures still make sense and provide the result claimed in Lemma 3.4.

The next result states that the map $\mathcal{C} \mapsto \sigma(\mathcal{C})$ does not modify the distance between codewords.

Lemma 3.7. *Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a linear code, $(\mathcal{Q}, V)$ a verification structure for $\mathcal{C}$, and $\sigma \in \mathfrak{S}(\mathbb{F}_q)^n$. Then it holds that:*

- *the distribution of distances in $\mathcal{C}$ and $\sigma(\mathcal{C})$ are the same,*

- *the distribution of distances in $R(\mathcal{C})$ and $R(\sigma(\mathcal{C}))$ are the same.*

Proof. Since every σ_i is one-to-one, for any $c, c' \in \mathcal{C}$ we get

$$\begin{aligned} d(c, c') &= |\{i \in [1, n], c_i \neq c'_i\}| \\ &= |\{i \in [1, n], \sigma_i(c_i) \neq \sigma_i(c'_i)\}| \\ &= d(\sigma(c), \sigma(c'))\,. \end{aligned}$$

The proof for response codes relies on the same argument. □

Remark these results imply that, if $\mathcal{C}$ is linear, then the minimum distance of $R(\sigma(\mathcal{C}))$ is the minimum weight of $R(\mathcal{C})$.

Definition 3.8. Let $\varepsilon \in [0, 1]$ and $(\mathcal{Q}, V)$ be a verification structure for a code $\mathcal{C} \subseteq \mathbb{F}_q^n$. We say $r \in \mathcal{R}^{\mathcal{Q}}$ is *ε-close* to $(\mathcal{Q}, V)$ if:

$$\mathrm{wt}(V(r)) \coloneqq |\{u \in \mathcal{Q}, V(u, r_u) \neq 0\}| \leq \varepsilon N\,.$$

Let now $c \in \mathcal{C}$ and $\beta \in [0, 1]$. We say that $r \in \mathcal{R}^{\mathcal{Q}}$ is a *β-liar* for $(\mathcal{Q}, V, c)$ if:

$$|\{u \in \mathcal{Q}, V(u, r_u) = 0 \text{ and } r_u \neq R(u, c)\}| \leq \beta N\,.$$

Bounded-distance error-and-erasure decoder. Let $\mathcal{A} \subseteq \mathbb{F}_q^n$ be any code of minimum distance d, and let $a \in \mathcal{A}$ be corrupted with b errors and e erasures, resulting in a word $r' \in (\mathbb{F}_q \cup \{\perp\})^n$. Then, it is well-known that, as long as $2b + e < d$, it is possible to retrieve a from r' thanks to a so-called *bounded-distance error-and-erasure decoding algorithm*. This is precisely the decoding algorithm that we employ in Figure 3 on the code $\mathcal{A} = R(\mathcal{C})$.

Our framework allows us to reformulate the extraction success in terms of a probability to decode corrupted codewords. More precisely:

Proposition 3.9. *Let $\sigma \in \mathfrak{S}(\mathbb{F}_q)^n$, $m \in \mathbb{F}_q^k$ and denote by d the minimum distance of $R(\mathcal{C})$, of length N. Let also $r \in \mathcal{R}^{\mathcal{Q}}$ be the response word, output of a proving algorithm P taking as input $w = \sigma(\mathcal{C}(m))$. Finally, assume that r is ε-close to $(\mathcal{Q}, V^\sigma)$ and a β-liar for $(\mathcal{Q}, V^\sigma, w)$, with $(\varepsilon + 2\beta)N < d$. Then, $\mathsf{Extract}(r, \sigma) = m$, where $\mathsf{Extract}(r, \sigma)$ is defined in Figure 3.*

Proof. Recall that $r' \in (\mathcal{R} \cup \{\perp\})^{\mathcal{Q}}$ represents the word we get from r after the second step of the algorithm given in Figure 3. Let us now translate our assumptions on r in coding-theoretic terminology:

- r is ε-close to $(\mathcal{Q}, V^\sigma)$ means that there are at most εN challenges $u \in \mathcal{Q}$ for which we *know* that the coordinate r'_u is not authentic. This justifies that we assign erasure symbols to these coordinates.
- r is a β-liar for $(\mathcal{Q}, V, c)$ means that there are at most βN other corrupted values r'_u, but we *cannot identify* them. Therefore we can assimilate these coordinates to errors.

To sum up, we see r' as a corruption of $R(\mathcal{C}(m))$ with at most εN erasures and at most βN errors, where $N = |\mathcal{Q}|$. Since we assume that $(\varepsilon + 2\beta)N < d$, we know from the previous discussion that the decoding succeeds to retrieve m. □

3.3.2 Bounding the extraction failure

According to Definition 2.3, our PoR scheme is weakly (ε, τ)-sound if for every polynomial-time algorithm P outputting a response word $r^{(w)}$ from a file w, we have

$$\Pr_{\sigma,m}\left(\begin{array}{c|c} \text{decoding } r^{(w)} \text{ into } m \text{ fails} & m \leftarrow_{\mathrm{R}} \mathbb{F}_q^k \\ \text{and} & \sigma \leftarrow_{\mathrm{R}} \mathfrak{S}(\mathbb{F}_q)^n \\ \mathrm{wt}(V^\sigma(r^{(w)})) \leq \varepsilon N & w = \sigma(\mathcal{C}(m)) \end{array}\right) \leq \tau\,.$$

Using Proposition 3.9, the security analysis of our PoR scheme reduces to measuring the ability of the Prover to produce a response word r which is ε-close to $(\mathcal{Q}, V^\sigma)$ and a β-liar for $(\mathcal{Q}, V^\sigma, w)$, with $(\varepsilon + 2\beta)N \geq d$.

For fixed $r \in \mathcal{R}^{\mathcal{Q}}$, $\sigma \in \mathfrak{S}(\mathbb{F}_q)^n$ and $w = \sigma(\mathcal{C}(m))$ the authentic file given to the prover, we define three subsets of $\mathcal{Q}$:

- $\mathcal{D}(r, w) \coloneqq \{u \in \mathcal{Q}, r_u \neq R(w)_u\}$ and $D(r, w) \coloneqq |\mathcal{D}(r, w)| = \mathrm{wt}(r - R(w))$. This represents challenges u on which the response word r differs from the authentic one $R(w)$.
- $\mathcal{E}(r, \sigma) \coloneqq \{u \in \mathcal{Q}, V^\sigma(u, r_u) \neq 0\}$ and $E(r, \sigma) \coloneqq |\mathcal{E}(r, \sigma)| = \mathrm{wt}(V^\sigma(r))$. These are challenges u on which the associated coordinate r_u is not accepted by the verification map (it corresponds to erasures in the decoding process).
- $\mathcal{B}(r, \sigma, w) \coloneqq \{u \in \mathcal{Q}, r_u \neq R(w)_u \text{ and } V^\sigma(u, r_u) = 0\}$ and $B(r, \sigma, m) \coloneqq |\mathcal{B}(r, \sigma, m)|$. These are the challenges u on which the associated coordinate r_u is accepted by the verification map, but differs from the authentic response s_u (it corresponds to errors in the decoding process).

One can easily check that, for every σ, the sets $\mathcal{E}(r, \sigma)$ and $\mathcal{B}(r, \sigma, w)$ define a partition of $\mathcal{D}(r, w)$. The probability of extraction failure can thus be written as:

$$\Pr\left(\begin{array}{c|c} 2D(r, w) - E(r, \sigma) \geq d_{\min}(R(\mathcal{C})) & m \leftarrow_{\mathrm{R}} \mathbb{F}_q^k \\ \text{and} & \sigma \leftarrow_{\mathrm{R}} \mathfrak{S}(\mathbb{F}_q)^n \\ E(r, \sigma) \leq \varepsilon N & w = \sigma(\mathcal{C}(m)) \end{array}\right). \tag{3}$$

For $w \in \mathbb{F}_q^n$, let us define the set of *admissible* permutations and messages:

$$\Phi_w \coloneqq \{(\sigma, m) \in \mathfrak{S}(\mathbb{F}_q)^n \times \mathbb{F}_q^k,\ w = \sigma(\mathcal{C}(m))\}\,,$$

so that Equation (3) rewrites:

$$\Pr\left(\begin{array}{c|c} 2D(r, w) - E(r, \sigma) \geq d_{\min}(R(\mathcal{C})) & (\sigma, m) \leftarrow_{\mathrm{R}} \Phi_w \\ E(r, \sigma) \leq \varepsilon N & \end{array}\right).$$

Later on, we will use the notation $\Pr_{\Phi_w}$ to refer to the fact that (σ, m) is uniformly drawn from Φ_w. Similarly we will use notation $\mathbb{E}_{\Phi_w}$ for the expectancy and Var_{Φ_w} for the variance.

Given $r \in \mathcal{R}^{\mathcal{Q}}$, we also define

$$\alpha(r, w) \coloneqq \max_{u \in \mathcal{D}(r,w)} \Pr_{\Phi_w}(V^\sigma(u, r_u) = 0)$$

and $\alpha \coloneqq \max_{(r,w)} \alpha(r, w)$ where (r, w) are such that $D(r, w) \neq 0$. The parameter $\alpha \in (0, 1)$ is called the *bias* of the verification structure $(\mathcal{Q}, V)$ for $\mathcal{C}$. It corresponds to the maximum probability that a response is accepted but not authentic.

Lemma 3.10. *For all $r \in \mathcal{R}^{\mathcal{Q}}$ and $w \in \mathbb{F}_q^n$, we have:*

$$\mathbb{E}_{\Phi_w}(E(r, \sigma)) \geq (1 - \alpha) D(r, w)\,.$$

Proof. A simple computation shows:

$$\begin{aligned}\mathbb{E}_{\Phi_w}(E(r,\sigma)) &= \mathbb{E}_{\Phi_w}\Big(\sum_{u\in\mathcal{D}(r,w)} \mathbb{1}_{V^\sigma(u,r_u)\neq 0}\Big)\\ &= \sum_{u\in\mathcal{D}(r,w)} \Pr_{\Phi_w}(V^\sigma(u,r_u)\neq 0)\\ &\geq \sum_{u\in\mathcal{D}(r,w)} (1-\alpha)\\ &\geq (1-\alpha)D(r,w)\,.\end{aligned}$$

□

Lemma 3.10 essentially means that, if an adversary to our PoR scheme wants its response word to be (in average) ε-close to the verification structure, then he should modify at most $D(r,w) \leq \frac{\varepsilon N}{1-\alpha}$ responses. Below we take advantage of this result and we measure the probability of an extraction failure.

First, for $\delta, \varepsilon \in (0,1)$, let

$$\begin{aligned}p(r,w;\varepsilon,\delta) &\coloneqq \Pr_{\Phi_w}(2D(r,w) - E(r,\sigma) \geq \delta N \text{ and } E(r,\sigma) \leq \varepsilon N)\\ &= \Pr_{\Phi_w}(E(r,\sigma) \leq \min\{\varepsilon N, 2D(r,w) - \delta N\})\,.\end{aligned}$$

The probability $p(r,w;\varepsilon,\delta)$ represents the probability that the extraction fails for a response code of relative distance δ and an adversarial response word r associated to w, which is ε-close to the verification structure. Let us bound $p(r,w;\varepsilon,\delta)$.

Proposition 3.11. *Let $\delta, \varepsilon \in (0,1)$ such that $\delta\frac{1-\alpha}{1+\alpha} > \varepsilon$. Let also $r \in \mathcal{R}^{\mathcal{Q}}$ and $w \in \mathbb{F}_q^n$. Then we have:*

$$p(r,w;\varepsilon,\delta) \leq \frac{\mathrm{Var}_{\Phi_w}(E(r,\sigma))}{\left(\frac{1+\alpha}{2}\left(\delta\frac{1-\alpha}{1+\alpha} - \varepsilon\right)\right)^2 N^2}\,.$$

Proof. We distinguish three cases.

1. $2D(r,w) - \delta N < 0$. The event $E(r,\sigma) \leq \min\{\varepsilon N, 2D(r,w) - \delta N\}$ never occurs since $E(r,\sigma) \geq 0$. Hence $p(r,w;\varepsilon,\delta) = 0$.
2. $\varepsilon N \leq 2D(r,w) - \delta N$. The inequality $E(r,\sigma) \leq \varepsilon N$ implies

$$\begin{aligned}E(r,\sigma) - \mathbb{E}_{\Phi_w}(E) &\leq \varepsilon N - (1-\alpha)D(r,w)\\ &\leq \varepsilon N - (1-\alpha)\frac{\varepsilon+\delta}{2}N\\ &\leq -\frac{1+\alpha}{2}\left(\delta\frac{1-\alpha}{1+\alpha} - \varepsilon\right)N\,.\end{aligned}$$

Hence, using Chebychev's inequality,

$$\begin{aligned}p(r,w;\varepsilon,\delta) &= \Pr_{\Phi_w}(E(r,\sigma) \leq \varepsilon N)\\ &\leq \Pr_{\Phi_w}\left(|E(r,\sigma) - \mathbb{E}_{\Phi_w}(E)| \geq \frac{1+\alpha}{2}\left(\delta\frac{1-\alpha}{1+\alpha} - \varepsilon\right)N\right)\\ &\leq \frac{\mathrm{Var}_{\Phi_w}(E(r,\sigma))}{\left(\frac{1+\alpha}{2}\left(\delta\frac{1-\alpha}{1+\alpha} - \varepsilon\right)\right)^2 N^2}\,.\end{aligned}$$

3. $0 \leq 2D(r,w) - \delta N < \varepsilon N$. In this case, $E(r,\sigma) \leq 2D(r,w) - \delta N$ implies

$$\begin{aligned} E(r,\sigma) - \mathbb{E}_{\Phi_w}(E) &\leq (1+\alpha)D(r,w) - \delta N \\ &\leq (1+\alpha)\frac{\varepsilon+\delta}{2}N - \delta N \\ &\leq -\frac{1+\alpha}{2}\left(\delta\frac{1-\alpha}{1+\alpha} - \varepsilon\right)N. \end{aligned}$$

Therefore, similarly to the previous case, we obtain the claimed result. □

For any $u \in \mathcal{D}(r,w)$, denote by X_u the $\{0,1\}$-random variable "$\mathbb{1}_{V^\sigma(u,r_u)=0}$" when σ is uniformly drawn from Φ_w. It holds that $E(r,\sigma) = \sum_{u\in\mathcal{D}(r,w)}(1-X_u)$.

Recall that two real random variables Y, Z are uncorrelated if $\mathbb{E}(YZ) = \mathbb{E}(Y)\mathbb{E}(Z)$. For instance, two independent random variables are uncorrelated.

Lemma 3.12. *Let $r \in \mathcal{R}^{\mathcal{Q}}$ and $w \in \mathbb{F}_q^n$. If the random variables $\{X_u\}_{u\in\mathcal{D}(r,w)}$ are pairwise uncorrelated, then:*

$$\mathrm{Var}_{\Phi_w}(E(r,\sigma)) \leq D(r,w)\,.$$

Proof. By assumption, $\{X_u\}_{u\in\mathcal{D}(r,w)}$ are pairwise uncorrelated, hence $\mathrm{Var}_{\Phi_w}(E(r,\sigma)) = \sum_{u\in\mathcal{D}(r,w)}\mathrm{Var}_{\Phi_w}(1-X_u)$. The trivial bound $\mathrm{Var}_{\Phi_w}(1-X_u) \leq 1$ gives the result. □

As a corollary of Proposition 3.11 and Lemma 3.12, under the same hypothesis and assuming $\delta\frac{1-\alpha}{1+\alpha} > \varepsilon$, we get

$$p(r,w;\varepsilon,\delta) \leq \frac{4}{N\left((1-\alpha)\delta - (1+\alpha)\varepsilon\right)^2}$$

since $D(r,w) \leq N$. Moreover, if $\lim_{N\to\infty}\delta > 0$ and $\lim_{N\to\infty}\alpha = 0$, then $p(r,w;\varepsilon,\delta) = \mathcal{O}(1/N)$.

Therefore, we end up with the following theorem.

Theorem 3.13. *Let $(\mathcal{Q},V)$ be a verification structure for $\mathcal{C}$ with bias α. Denote by $N = |\mathcal{Q}|$ and $\delta = d_{\min}(R(\mathcal{C}))/N$ the relative distance of the associated response code. Finally, assume that, for any $r \in \mathcal{R}^{\mathcal{Q}}$ and any $w \in \mathbb{F}_q^n$ the variables $\{X_u\}_{u\in\mathcal{D}(r,w)}$ are pairwise uncorrelated. Then, for any $\varepsilon < \delta\frac{1-\alpha}{1+\alpha}$, the* PoR *scheme associated to $\mathcal{C}$ and $(\mathcal{Q},V)$ is (ε,τ)-sound, where*

$$\tau = \frac{4}{N\left((1-\alpha)\delta - (1+\alpha)\varepsilon\right)^2}\,.$$

For asymptotically small α, a code $\mathcal{C}$ equipped with a verification structure satisfying the conditions of Theorem 3.13 thus gives an (ε,τ)-sound PoR scheme for every $\varepsilon < (1+o(1))\delta$ and $\tau = \mathcal{O}(1/N)$.

According to Theorem 3.13, we thus need to look for (sequences of) codes $\mathcal{C}$ and associated verification structures $(\mathcal{Q},V)$ such that:

1. the response code $R(\mathcal{C})$ admits a good relative distance $\delta = d_{\min}(R(\mathcal{C}))/N$
2. the bias α is small,
3. random variables $\{X_u\}_{u\in\mathcal{D}(r,w)}$ are pairwise uncorrelated.

Subsections 3.4 and 3.5 characterize conditions under which the last two points are fulfilled. Then, in Section 5 we discuss which response codes can achieve good relative distance.

3.4 Estimating α

In this section we prove that, assuming Φ_w approximates the uniform distribution over $\mathfrak{S}(\mathbb{F}_q)^n$ in a sense that we make precise later, the bias α can be bounded according to parameters of the verification structure.

Let us fix $r \in \mathcal{R}^{\mathcal{Q}}$, $w \in \mathbb{F}_q^n$ and $u \in \mathcal{Q}$. We recall that α is defined by:

$$\alpha = \max_{r,w} \max_{u \in \mathcal{D}(r,w)} \Pr_{\Phi_w}(V^\sigma(u, r_u) = 0)$$

where randomness comes from $\sigma \leftarrow_{\mathrm{R}} \Phi_w = \{(\sigma, m) \in \mathfrak{S}(\mathbb{F}_q)^n \times \mathbb{F}_q^k, w = \sigma(\mathcal{C}(m))\}$. We notice that this is equivalent to write $\sigma \leftarrow_{\mathrm{R}} \{\sigma \in \mathfrak{S}(\mathbb{F}_q)^n, \sigma^{-1}(w) \in \mathcal{C}\}$.

For convenience, we will view $r_u \in \mathcal{R} = \mathbb{F}_q^\ell$ as a vector indexed by $u = (u_1, \ldots, u_\ell)$, so that we can easily denote by $r_u[u_j] \in \mathbb{F}_q$ its j-th coordinate, $1 \leq j \leq \ell$. We define the code $K_u \coloneqq \ker V(u, \cdot) \subseteq \mathbb{F}_q^\ell$, and up to re-indexing coordinates, $\mathcal{C}_{|u} \subseteq K_u$. This allows us to write that for every σ, we have $V^\sigma(u, r_u) = 0$ if and only if $\sigma_u^{-1}(r_u) \in K_u$. Finally, we denote by $Z_u \coloneqq \{i \in u, r_u[i] \neq R(w)_u[i]\}$ the set of coordinates of r_u that are not authentic.

Let $Y_u(\sigma)$ represent the event "$\sigma_u^{-1}(r_u) \in K_u \mid \mathrm{supp}(\sigma_u^{-1}(r_u)) = Z_u$". Informally, the reason why we consider an event $Y_u(\sigma)$ conditioned by $\mathrm{supp}(\sigma_u^{-1}(r_u)) = Z_u$ is that the Prover is free to choose any support Z_u on which he can modify the original file. More formally, this constraint will help us to bound the probability $\Pr_{\Phi_w}(V^\sigma(u, r_u) = 0)$ in Lemma 3.14.

We say that Φ_w is *sufficiently uniform* if, for every $u \in \mathcal{Q}$, we have:

$$\gamma_u \coloneqq \frac{\Pr\left[Y_u(\sigma) \mid \sigma \leftarrow_{\mathrm{R}} \Phi_w\right] - \Pr\left[Y_u(\sigma) \mid \sigma \leftarrow_{\mathrm{R}} \mathfrak{S}(\mathbb{F}_q)^n\right]}{\Pr\left[Y_u(\sigma) \mid \sigma \leftarrow_{\mathrm{R}} \mathfrak{S}(\mathbb{F}_q)^n\right]} = o(1)$$

when the file size $n \log q \to \infty$. In other words, Φ_w is *sufficiently uniform* if it is a good approximation of the whole set of n-tuples of permutations, when considering the probability that $Y_u(\sigma)$ happens.

Lemma 3.14. *Let r, w, u and Z_u be defined as above. Let also $A_u = |\{x \in K_u, \mathrm{supp}(x) = Z_u\}|$. Then*

$$\Pr_{\Phi_w}(V^\sigma(u, r_u) = 0) \leq \frac{(1 + \gamma_u) A_u}{(q-1)^{|Z_u|}}\,.$$

Proof. For every σ such that $(\sigma, m) \in \Phi_w$, we know that $\sigma_u^{-1}(R(w)_u) \in K_u$, and we recall that $V^\sigma(u, r_u) = 0$ if and only if $\sigma_u^{-1}(r_u) \in K_u$. Since K_u is linear , and up to considering $\sigma_u^{-1}(R(w)_u - r_u)$ instead, we can assume without loss of generality that $\sigma_u^{-1}(r_u)[i] = 0$ for every $i \in u \setminus Z_u$. In other words we assume that $\mathrm{supp}(\sigma_u^{-1}(r_u)) = Z_u$.

Remark that

$$\begin{aligned}
\Pr_{\sigma \leftarrow_{\mathrm{R}} \mathfrak{S}(\mathbb{F}_q)^n}\left[\sigma_u^{-1}(r_u) \in K_u \mid \mathrm{supp}(\sigma_u^{-1}(r_u)) = Z_u\right] & \\
&= \Pr_{x \leftarrow_{\mathrm{R}} \mathbb{F}_q^\ell}\left[x \in K_u \mid \mathrm{supp}(x) = Z_u\right] \\
&= \Pr_{x \leftarrow_{\mathrm{R}} \mathbb{F}_q^\ell}\left[x \in K_u \mid \mathrm{supp}(x) = Z_u\right] \\
&= \frac{A_u}{(q-1)^{|Z_u|}}\,,
\end{aligned}$$

since A_u counts the number of codewords in K_u whose support is Z_u.

Therefore we get

$$\begin{aligned}\Pr_{\Phi_w}(V^\sigma(u,r_u)=0) &\le \Pr_{\Phi_w}\left[V^\sigma(u,r_u)=0 \mid \mathrm{supp}(\sigma_u^{-1}(r_u))=Z_u\right]\\ &= (1+\gamma_u)\Pr_{\mathfrak{S}(\mathbb{F}_q)^n}\left[V^\sigma(u,r_u)=0 \mid \mathrm{supp}(\sigma_u^{-1}(r_u))=Z_u\right]\\ &= (1+\gamma_u)\Pr_{x\leftarrow_{\mathrm{R}}\mathbb{F}_q^\ell}\left[x\in K_u \mid \mathrm{supp}(x)=Z_u\right]\\ &= \frac{(1+\gamma_u)A_u}{(q-1)^{|Z_u|}}\,.\end{aligned}$$

□

Lemma 3.15. *Let S_u be the $\mathbb{F}_q$-vector space $\langle\{x\in K_u, \mathrm{supp}(x)=Z_u\}\rangle$ and assume that $S_u\neq\{0\}$. We have:*

$$A_u \le q^{|Z_u|-d_{\min}(S_u)+1}\,.$$

Proof. We prove that, if $A_u > q^e$ for some integer $e\ge 0$, then $d_{\min}(S_u)\le |Z_u|-e$, which clearly induces our result. If $A_u > q^e$, then $\dim S_u > e$ since $|S_u|\ge A_u$. The Singleton bound then provides:

$$d_{\min}(S_u)\le |Z_u|-\dim S_u+1\le |Z_u|-e\,.$$ □

Finally, we get the following upper bound on α.

Proposition 3.16. *Let $\Delta=\min\{d_{\min}(K_u), u\in\mathcal{Q}\}$. Then*

$$\alpha\le(1+\gamma)(1+\tfrac{1}{q-1})^\ell q^{-\Delta+1}\,,$$

where $\gamma_u=\max\gamma_u$.

Proof. Remark that S_u, defined in previous lemma, is a subcode of K_u shortened on $u\setminus Z_u$. Hence $d_{\min}(K_u)\le d_{\min}(S_u)$, and we can apply previous results and obtain the desired bound:

$$\alpha\le\max_{u,r}(1+\gamma_u)\left(\frac{q}{q-1}\right)^{|Z_u|}q^{-d_{\min}(K_u)+1}\le(1+\gamma)(1+\tfrac{1}{q-1})^\ell q^{-\Delta+1}$$

where $\gamma=\max_u\gamma_u$. □

If every Φ_w is sufficiently uniform, then by definition we have $\gamma=o(1)$ when the file size $n\log q\to\infty$. This assumption is significant since we desire to have a small bias α, which is deeply linked to the soundness of PoRs (see Theorem 3.13). In Appendix A we present experimental estimates of α, validating that the assumption that Φ_w is sufficiently uniform.

3.5 Pairwise uncorrelation of $\{X_u\}_{u\in\mathcal{D}}$

This section is devoted to proving that variables $\{X_u\}_{u\in\mathcal{D}(r,w)}$ are pairwise uncorrelated if the supports of challenges $u\in\mathcal{D}(r,w)$ have small pairwise intersection. For this purpose, let us recall that for fixed $r\in\mathcal{R}^{\mathcal{Q}}$, w and $u\in\mathcal{D}(r,w)$, the random variable X_u represents $\mathbb{1}_{V^\sigma(u,r_u)=0}$, when σ is uniformly picked in Φ_w.

We first state a technical lemma that will be useful to prove Proposition 3.18 below. For clarity we denote by $d^\perp(\mathcal{C})$ the minimum distance of the dual code $\mathcal{C}^\perp$ of a linear code $\mathcal{C}$.

Lemma 3.17. *Let $\mathcal{C}\subseteq\mathbb{F}_q^n$ be a linear code and $T\subset[1,n]$, $|T|=t$ where $t<d^\perp(\mathcal{C})$. For $a\in\mathbb{F}_q^T$, we define $\mathcal{V}_a=\{c\in\mathcal{C}, c_{|T}=a\}$, and $N_a=|\mathcal{V}_a|$. Then,*

1. *$\mathcal{V}_0=\{v\in\mathcal{C}, v_{|T}=0\}$ is a linear subcode of $\mathcal{C}$;*

2. *for every non-zero* $a \in \mathbb{F}_q^T$*, there exists a non-zero* $c^{(a)} \in \mathcal{C}$ *such that*

$$\mathcal{V}_a = \mathcal{V}_0 + \{c^{(a)}\}\,;$$

3. *for every* $a \in \mathbb{F}_q^T$*,* $N_a = q^{k-t}$ *where* $k = \dim \mathcal{C}$*.*

Proof.

1. The fact that $\mathcal{V}_0 = \{v \in \mathbb{F}_q^X, v_{|T} = 0\}$ is actually the well-known definition of the shortening of a code. It is easy to prove that it defines a linear code.
2. Let $a \in \mathbb{F}_q^T$ be non-zero, and let us first prove that there exists $c^{(a)} \in \mathcal{C}$ such that $c^{(a)}_{|T} = a$. If it were not the case, then by definition we would have $\mathcal{C}_{|T} \neq \mathbb{F}_q^t$. But this is impossible since $\mathcal{C}^\perp$ contains no non-zero codeword of weight less that t. It is then easy to check that $\mathcal{V}_a = \mathcal{V}_0 + \{c^{(a)}\}$.
3. First notice that $\mathcal{V}_a \cap \mathcal{V}_b = \varnothing$ if $a \neq b$. Since

$$\mathcal{C} = \bigcup_{a \in \mathbb{F}_q^t} \mathcal{V}_a\,,$$

we get the expected result. □

Proposition 3.18. *If* $\max\{|u \cap v|, u \neq v \in \mathcal{Q}\} < \min\{d^\perp(\mathcal{C}_{|u}), u \in \mathcal{Q}\}$*, then the random variables* $\{X_u\}_{u \in \mathcal{Q}}$ *are pairwise uncorrelated.*

Proof. Recall that $K_u = \ker V(u, \cdot)$, and that by definition of a verification structure, we have $\mathcal{C}_{|u} \subseteq K_u$. For $u \neq v \in \mathcal{Q}$, let us prove that $\mathbb{E}(X_u X_v) = \mathbb{E}(X_u)\mathbb{E}(X_v)$. First,

$$\begin{aligned}\mathbb{E}(X_u X_v) &= \Pr\left(V^\sigma(u, r_u) = 0 \text{ and } V^\sigma(v, r_v) = 0\right)\\ &= \Pr\left(\sigma^{-1}(r_u)_{|u} \in K_u \text{ and } \sigma^{-1}(r_v)_{|v} \in K_v\right)\,.\end{aligned}$$

Set $t = |u \cap v|$ and let $(\mathbf{a}, \mathbf{b}) \in (\mathbb{F}_q^t)^2$. We denote by $Z(\sigma, \mathbf{a}, \mathbf{b})$ the event

$$\sigma^{-1}(r_u)_{|u\cap v} = \mathbf{a} \;\text{ and }\; \sigma^{-1}(r_v)_{|u\cap v} = \mathbf{b}\,.$$

We first notice that $\{\sigma^{-1}_{|u\cap v}, \sigma \in \Phi_w\} = \mathfrak{S}(\mathbb{F}_q)^t$. Indeed, we can here use an argument similar to the proof of Lemma 3.17: the constraint $\sigma^{-1}(w) \in \mathcal{C}$ is ineffective on $\sigma^{-1}_{|u\cap v}$, since $|u \cap v| \leq t < d^\perp(\mathcal{C}_{|z})$ for every $z \in \mathcal{Q}$. Therefore, for every $(\mathbf{a}, \mathbf{b}) \in (\mathbb{F}_q^t)^2$, we have

$$\Pr(Z(\sigma, \mathbf{a}, \mathbf{b})) = q^{-2t}\,,$$

and it follows that:

$$\mathbb{E}(X_u X_v) = \frac{1}{q^{2t}} \sum_{\mathbf{a},\mathbf{b} \in (\mathbb{F}_q^t)^2} \Pr\left(\sigma^{-1}(r_u)_{|u} \in K_u \text{ and } \sigma^{-1}(r_v)_{|v} \in K_v \mid Z(\sigma, \mathbf{a}, \mathbf{b})\right).$$

Recall now that $t < \min\{d^\perp(\mathcal{C}_{|u}), u \in \mathcal{Q}\} \leq \min\{d^\perp(K_u), u \in \mathcal{Q}\}$. Hence, for fixed $\mathbf{a}$ and $\mathbf{b}$, the variables $\sigma^{-1}(r_u)_{|u} \in K_u \mid Z(\sigma, \mathbf{a}, \mathbf{b})$ and $\sigma^{-1}(r_v)_{|v} \in K_v \mid Z(\sigma, \mathbf{a}, \mathbf{b})$ are independent (once again it is a consequence of the structure results of Lemma 3.17). Therefore:

$$\begin{aligned}\mathbb{E}(X_u X_v) = \frac{1}{q^{2t}} \sum_{\mathbf{a},\mathbf{b} \in (\mathbb{F}_q^t)^2} &\Pr\left(\sigma^{-1}(r_u)_{|u} \in K_u \mid Z(\sigma, \mathbf{a}, \mathbf{b})\right)\\ &\times \Pr\left(\sigma^{-1}(r_v)_{|v} \in K_v \mid Z(\sigma, \mathbf{a}, \mathbf{b})\right).\end{aligned}$$

Then,

$$\mathbb{E}(X_u X_v) = \frac{1}{q^{2t}} \sum_{\mathbf{a},\mathbf{b}\in(\mathbb{F}_q^t)^2} \Pr(\sigma^{-1}(r_u)_{|u} \in K_u \mid \sigma^{-1}(r_u)_{|u\cap v} = \mathbf{a}) \\ \times \Pr(\sigma^{-1}(r_v)_{|v} \in K_v \mid \sigma^{-1}(r_v)_{|u\cap v} = \mathbf{b})\,.$$

and we conclude since

$$\mathbb{E}(X_u) = q^{-t} \sum_{\mathbf{a}\in\mathbb{F}_q^t} \Pr(\sigma^{-1}(r_u)_{|u} \in K_u \mid \sigma^{-1}(r_u)_{|u\cap v} = \mathbf{a})\,.$$

□

4 Performance

4.1 Efficient scrambling of the encoded file

In the PoR scheme we propose, the storage cost of an n-tuple of permutations in $\mathfrak{S}(\mathbb{F}_q)^n$ is excessive, since it is superlinear in the original file size. In this subsection, we propose a storage-efficient way to scramble the codeword $c \in \mathcal{C}$ produced by the Verifier.

Precisely we want to define a family of maps $(\sigma^{(\kappa)})_\kappa$, where $\sigma^{(\kappa)} : \mathcal{C} \to \mathbb{F}_q^n$, $c \mapsto w \in \mathbb{F}_q^n$, with the following requirements:

- for every κ, the map $\sigma^{(\kappa)}$ is efficiently computable and requires a low storage,
- for every κ and every $c \in \mathcal{C}$, if $w = \sigma^{(\kappa)}(c)$, then for every $i \in [1, n]$ the local inverse map $w_i \mapsto c_i$ is efficiently computable,
- if κ is randomly generated but unknown, then given the knowledge of $w = \sigma^{(\kappa)}(c)$ and $\mathcal{C}$, it is hard to produce a response word $r \in \mathcal{R}^{\mathcal{Q}}$ such that, for many $u \in \mathcal{Q}$, both $V^{\sigma^{(\kappa)}}(u, r_u) = 0$ and $r_u \neq w_{|u}$ hold. To be more specific, and in light of the security analysis of Section 3.3, we require that it is hard to distinguish $\sigma^{(\kappa)}(c)$ from a random $(z_1, \ldots, z_n) \in \mathbb{F}_q^n$, where symbols z_i are picked independently and uniformly at random.

We here propose to derive $\sigma^{(\kappa)}$ from a suitable block cipher, yielding the explicit construction given below. Of course, other proposals can be envisioned.

The construction. Let IV denote a random initialisation vector for AES in CTR mode (IV could be a nonce concatenated with a random value). Vector IV is kept secret by the Verifier, as well as a randomly chosen key κ for the cipher. Let also f be a permutation polynomial over $\mathbb{F}_q$ of degree $d > 1$. For instance one could choose $f(x) = x^d$ with $\gcd(d, q-1) = 1$. Notice that polynomial f can be made public.

Let $s = \left\lfloor \frac{256}{\lceil \log_2 q \rceil} \right\rfloor$ be the number of $\mathbb{F}_q$-symbols one can store in a 256-bit word[2]. Up to appending a few random bits to c, we assume that $s \mid n$, and we define $t = n/s$. Let us fix a partition of $[1, n]$ into s-tuples $i = (i_1, \ldots, i_s)$; it can be for instance $(1, \ldots, s)$, $(s+1, \ldots, 2t), \ldots, ((t-1)s+1, \ldots, n)$. Notice that this partition does not need to be chosen at random. Given $c = (c_1, \ldots, c_n) \in \mathcal{C}$ and i an element of the above partition, we now define

$$b_i = \big(f(c_{i_1}) \mid \cdots \mid f(c_{i_s})\big) \oplus \mathrm{AES}_\kappa(IV \oplus i) \in \{0,1\}^{256}.$$

If $\log_2 q \nmid 256$, trailing zeroes can be added to evaluations of f. Finally, the pseudo-random permutation σ is defined by:

$$\sigma(c) \coloneqq (b_1, \ldots, b_t)\,.$$

[2] in the scheme we propose, we will always have $\log(q) < 256$

Design rationale. AES is a natural choice when one needs a (secret)-keyed pseudo-random permutation. Also notice that with this construction, one only needs to store the key κ and the vector IV, since the other objects (the polynomial f, the partition) are made public. Hence our objectives in terms of storage are met.

We now point out the necessity to use i as a part of the input of the AES cipher. Assume that we do not. Then, the local permutation σ_j, $1 \leq j \leq n$, would not depend on j. As a consequence, for certain class of codes the local verification map $r_u \mapsto V^{\sigma}(u, r_u)$ would not depend on u, and a malicious Prover would then be able to produce accepted answers while storing only a small piece of the file w (*e.g.* $w_{|u}$ for only one $u \in \mathcal{Q}$).

Another mandatory feature is the non-linearity of the permutation polynomial f. Indeed, assume for instance that $f = \mathrm{id}$. Then, given the knowledge of $w = \sigma(c)$, it would be very easy for a malicious Prover to produce a word $w' \neq w$, such that $r' = R(w')$ is always accepted by the Verifier. Simply, the Prover defines $w' = w + c'$, where c' is any non-zero codeword of $\mathcal{C}$. Hence, one sees that the polynomial f must be non-linear in order to prevent such kind of attacks.

4.2 Parameters

We here consider a PoR built upon a code $\mathcal{C} \subseteq \mathbb{F}_q^n$, with verification structure $(\mathcal{Q}, V)$ satisfying $\mathcal{R} = \mathbb{F}_q^{\ell}$ and $V(\mathcal{R}) = \mathbb{F}_q^s$. We also assume that we use an n-tuple of pseudo-random permutations as described in the previous subsection.

Communication complexity. At each verification step, the client sends an ℓ-tuple of coordinates $(u_1, \ldots, u_\ell)$, $u_i \in [1, n]$. The server then answers with corresponding symbols $w_{u_i} \in \mathbb{F}_q$. Therefore the upload communication cost is $\ell \log_2 n$ bits, while the download communication cost is $\ell \log_2 q$, thus a total of $\ell(\log_2 n + \log_2 q)$ bits.

Computation complexity. In the initialisation phase, following the encryption described in section 4.1, the client essentially has:

- to compute the codeword $c \in \mathcal{C}$ associated to its message,
- to make n evaluations of the permutation polynomial f over $\mathbb{F}_q$, and
- to compute $t = \frac{n \log_2 q}{256}$ AES ciphertexts to produce the word w to be sent to the server.

Given a generator matrix of $\mathcal{C}$, the codeword c can be computed in $\mathcal{O}(kn)$ operations over $\mathbb{F}_q$ with a matrix-vector product. Notice that quasi-linear-time encoding algorithms exist for some classes of codes. Besides, if a monomial or a sparse permutation polynomial is used, then the cost of each evaluation is $\mathcal{O}((\log_2 q)^3)$. If we denote by c the bitcost of an AES encryption, we get a total bitcost of $\mathcal{O}(nk(\log_2 q)^2 + n(\log_2 q)^3 + cn \log_2 q)$ for the initialisation phase. Recall this is a worst-case scenario in which the encoding process is inefficient.

At each verification step, an honest server only needs to read ℓ symbols from the file it stores. Hence its computation complexity is $\mathcal{O}(\ell)$. The client has to compute a matrix-vector product over $\mathbb{F}_q$, where the matrix has size $s \times \ell$ and the vector has size ℓ, thus a computation cost of $\mathcal{O}(\ell s)$ operations over $\mathbb{F}_q$.

Storage needs. The client stores 2×256 bits for secret material κ and IV to use in AES. The server storage overhead exactly corresponds to the redundancy of the linear code $\mathcal{C}$, that is $(n - \dim \mathcal{C}) \log_2 q$ bits.

Other features. Our PoR scheme is *unbounded-use*, since every challenge reveals nothing about the secret data held by the client. It does not feature dynamic updates of files. Though, we must emphasize that the file w the client produces can be split among several servers, and the verification step remains possible even if the servers do not communicate with each other. Indeed, computing a response to a challenge does not require to mix distinct symbols w_i of the uploaded file. Therefore, our scheme is well-suited for the storage of *large static distributed* databases.

Client storage	512 bits
Server total storage	$n \log_2 q$ bits
Communication complexity (verif.)	$\ell \log_2(nq)$ bits
Client computation complexity (verif.)	ℓ decryptions, ℓs operations over $\mathbb{F}_q$
Server computation complexity (verif.)	ℓ reads, no computation

Figure 4: Summary of parameters of our PoR construction, for an original file of size $k \log_2 q$ bits, and a code $\mathcal{C}$ of dimension k over $\mathbb{F}_q$ equipped with a verification structure $(\mathcal{Q}, V)$ such that $|u| = \ell$, and $\operatorname{rank} V(u, \cdot) \leq s$ for all $u \in \mathcal{Q}$.

5 Instantiations

In this section we present several instantiations of our PoR construction. We first recall basics and notation from coding theory.

The code $\mathrm{Rep}(\ell) \subseteq \mathbb{F}_q^\ell$ denotes the repetition code $\langle (1, \ldots, 1) \rangle$. We recall that $\mathrm{Rep}(\ell)^\perp$ is the parity code $\mathrm{Par}(\ell) \coloneqq \{c \in \mathbb{F}_q^\ell, \sum_{i=1}^\ell c_i = 0\}$. Let $\mathcal{C}, \mathcal{C}'$ be two linear codes over $\mathbb{F}_q$ of respective parameters $[n, k, d]$ and $[n, k', d']$. Their tensor product $\mathcal{C} \otimes \mathcal{C}'$ is the $\mathbb{F}_q$-linear code generated by words $(c_i c'_j : 1 \leq i \leq n, 1 \leq j \leq n') \in \mathbb{F}_q^{nn'}$. It has dimension kk' and minimum distance dd'. We also denote by

$$\mathcal{C}^{\otimes s} \coloneqq \underbrace{\mathcal{C} \otimes \cdots \otimes \mathcal{C}}_{s \text{ times}} \subseteq \mathbb{F}_q^{n^s}$$

the s-fold tensor product of $\mathcal{C}$ with itself.

5.1 Tensor-product codes

The upcoming subsection illustrates our construction with a non practical but simple instance. The next ones lead to practical PoR instances.

5.1.1 A simple but non-practical instance

Let $n = N\ell$, and $\mathcal{Q} = \{u_i = \{i\ell + 1, i\ell + 2, \ldots, (i+1)\ell\}, i \in [0, N-1]\}$. The set $\mathcal{Q}$ defines a partition of $[1, n]$. We define the code

$$\mathcal{C} = \{c \in \mathbb{F}_q^n, \sum_{j \in u} c_j = 0, \forall u \in \mathcal{Q}\} \subseteq \mathbb{F}_q^n$$

In other words, $\mathcal{C} = \mathrm{Par}(\ell) \otimes \mathbb{F}_q^N$, and a parity-check matrix H for $\mathcal{C}$ is given by:

$$H = \begin{pmatrix} 1 & \cdots & 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & 0 & 1 & \cdots & 1 & \ddots & \vdots & \vdots & \vdots \\ \vdots & & & \ddots & & & \ddots & & & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 1 & \cdots & 1 \end{pmatrix}.$$

The verification map $V : \mathcal{Q} \times \mathbb{F}_q^\ell \to \mathbb{F}_q$ is defined by $V(u, b) \coloneqq \sum_{j=1}^{\ell} b_{u_j}$, for all $(u, b) \in \mathcal{Q} \times \mathbb{F}_q^\ell$. By construction (see the fundamental Example 3.2), the pair $(\mathcal{Q}, V)$ defines a verification structure for $\mathcal{C}$.

Lemma 5.1. *Let $\mathcal{C} = \mathrm{Par}(\ell) \otimes \mathbb{F}_q^N$ as above. Then the response code $R(\mathcal{C})$ has minimum distance 1.*

Proof. We see that the restriction map R sends the codeword $(1, -1, 0, 0, \ldots, 0) \in \mathcal{C}$ to a word of weight 1. Besides, R is injective so $d_{\min}(R(\mathcal{C})) > 0$. □

Since $\delta = d_{\min}(R(\mathcal{C}))/N = 1/N \to 0$ when N goes to infinity, an attempt to build a PoR scheme from $\mathcal{C}$ cannot be practical.

5.1.2 Higher order tensor-product codes

Let $\mathcal{A} \subseteq \mathbb{F}_q^\ell$ be a non-degenerate $[\ell, k_\mathcal{A}, d_\mathcal{A}]_q$-linear code, and define $\mathcal{C} = \mathcal{A}^{\otimes s} \subseteq \mathbb{F}_q^n$ where $n = \ell^s$. Notice it will be more convenient to see coordinates of words $w \in \mathbb{F}_q^n$ as elements of $[1, \ell]^s$.

For $\mathbf{a} \in [1, \ell]^s$ and $1 \leq i \leq s$, we define $L_{i,\mathbf{a}} \subset [1, \ell]^s$, the "$i$-th axis-parallel line with basis $\mathbf{a}$", as

$$L_{i,\mathbf{a}} \coloneqq \{\mathbf{x} \in [1, \ell]^s \text{ such that } x_j = a_j, \forall j \neq i\}\,.$$

By definition of $\mathcal{C}$, a word c lies in $\mathcal{C}$ if and only if, for every $L = L_{i,\mathbf{a}}$, the restriction $c_{|L} \in \mathcal{A}$. This means that we can define:

- a set of queries $\mathcal{Q} = \{L_{i,\mathbf{a}}, i \in [1, s], \mathbf{a} \in [1, \ell]^s\}$;
- a verification map

$$\begin{array}{rccc} V: & \mathcal{Q} \times \mathcal{R} & \to & \mathbb{F}_q^{\ell - k_\mathcal{A}} \\ & (L, r) & \mapsto & Hr \end{array}$$

  where H is a parity-check matrix for $\mathcal{A}$ whose columns are ordered according to the line L.

By the previous discussion, it is clear that $c \in \mathcal{C}$ implies that $V(L, c_{|L}) = 0$ for every $L \in \mathcal{Q}$ (in fact these two assertions are equivalent). Hence, $(\mathcal{Q}, V)$ defines a verification structure for $\mathcal{C}$, and we have $N = |\mathcal{Q}| = s\ell^{s-1}$.

Lemma 5.2. *Let $\mathcal{C} = \mathcal{A}^{\otimes s}$ as above. Then, $R(\mathcal{C})$ has minimum distance $s \cdot d_\mathcal{A}^{s-1}$.*

Proof. Let us first prove that the minimum distance of $R(\mathcal{C})$ is larger than $s \cdot d_\mathcal{A}^{s-1}$. Let $r = R(c) \in R(\mathcal{C})$, and assume $r \neq 0$. Then, there exists $L \in \mathcal{Q}$ such that $0 \neq r_L = c_{|L} \in \mathcal{A}$. Therefore $c_\mathbf{x} \neq 0$ for some $\mathbf{x} \in L \subset [1, \ell]^s$. Consider the set

$$S_{i,\mathbf{x}} = \{\mathbf{y} \in [1, \ell]^s, y_i = x_i\}\,.$$

Very informally, the set $S_{i,\mathbf{x}}$ corresponds to the hyperplane passing through $\mathbf{x}$ and "orthogonal" to the i-th axis. By definition of $\mathcal{C} = \mathcal{A}^{\otimes s}$, we know that $c_{|S_{i,\mathbf{x}}} \in \mathcal{A}^{\otimes(s-1)} \setminus \{0\}$ for every $1 \leq i \leq s$. Denote by $U_i = \mathrm{supp}(c_{|S_{i,\mathbf{x}}}) = \{\mathbf{u}^{(i,1)}, \ldots, \mathbf{u}^{(i,t_i)}\}$, with $t_i \geq d_{\min}(\mathcal{A}^{\otimes(s-1)}) = (d_\mathcal{A})^{s-1}$. Every $\mathbf{u}^{(i,j)} \in U_i$ defines a line $L_{i,\mathbf{u}^{(i,j)}}$ on which $c_{|L_{i,\mathbf{u}^{(i,j)}}}$ is a non-zero codeword of $\mathcal{A}$. Equivalently, r is non-zero on index $L_{i,\mathbf{u}^{(i,j)}} \in \mathcal{Q}$. Therefore,

$$\mathrm{wt}(r) = |\{L \in \mathcal{Q}, r_L \neq 0\}| \geq \left| \bigcup_{i=1}^{s} \left\{ L_{i,\mathbf{u}^{(i,j)}}, 1 \leq j \leq t_i \right\} \right| \geq \sum_{i=1}^{s} t_i \geq s(d_\mathcal{A})^{s-1}\,.$$

Let us now build a word $r \in R(\mathcal{C})$ of weight $s(d_{\mathcal{A}})^{s-1}$. Let $w \in \mathcal{A} \setminus \{0\}$ be a minimum-weight codeword of $\mathcal{A}$, and define $W \coloneqq \mathrm{supp}(w) \subseteq A$. Define $c = w^{\otimes s} \in \mathcal{C}$; then $\mathrm{supp}(c) = W^s$. Let finally $r = R(c)$. We see that $r_{L_{i,\mathbf{x}}} \neq 0$ if and only if $\mathbf{x} \in W^s$. Hence we get

$$\mathrm{wt}(r) = |\{L \in \mathcal{Q}, r_L \neq 0\}| = \left| \bigcup_{i=1}^{s} \{L_{i,\mathbf{x}}, \mathbf{x} \in W^s\} \right| = s \cdot d_{\mathcal{A}}^{s-1} .$$

since each line $L_{i,\mathbf{x}}$ is counted $d_{\mathcal{A}}$ times when $\mathbf{x}$ runs over W^s. □

Proposition 5.3. *Let $\delta > 0$ and $\mathcal{A}$ be an $[\ell, \ell(1-\delta)+1, \ell\delta]_q$ MDS code. Define $\mathcal{C} = \mathcal{A}^{\otimes s}$ and $(\mathcal{Q}, V)$ as above. If every Φ_w is sufficiently uniform, then the* PoR *scheme associated to $\mathcal{C}$ and $(\mathcal{Q}, V)$ is (ε, τ)-sound for $\tau = \mathcal{O}(\frac{1}{(\delta\ell)^s s})$ and every $\varepsilon < \varepsilon_0$ where $\varepsilon_0 = (1 + \mathcal{O}(q^{-\delta\ell+1}))\delta^s$, when $\ell \to \infty$.*

Proof. First, the relative distance of $R(\mathcal{C})$ is δ^s according to Lemma 5.2. Then, the random variables $\{X_u\}_{u \in \mathcal{D}}$ are pairwise uncorrelated because the inequality

$$\max_{u \neq v \in \mathcal{Q}^2} |u \cap v| = 1 < \ell(1-\delta) + 2 = \min_{u \in \mathcal{Q}} d_{\min}((\mathcal{C}_{|u})^{\perp})$$

allows us to apply Proposition 3.18. Besides, if every Φ_w is sufficiently uniform, then the bias α satisfies $\alpha = \mathcal{O}(q^{-\delta\ell+1})$; hence $\frac{1-\alpha}{1+\alpha} = 1 + \mathcal{O}(q^{-\delta\ell+1})$. Therefore we can use Theorem 3.13 and we get the desired result. □

Parameters. We mainly focus on the download communication complexity in the verification step and on the server storage overhead, since these are the most crucial parameters which depend on the family of codes $\mathcal{C}$ we use. Besides, we consider that it is more relevant to analyse the ratio between these quantities and the file size than their absolute values.

Here, for an initial file of size $|F| = ((1-\delta)q+1)^s \log_2 q$ bits, we get

- a redundancy rate $\frac{n \log_2 q}{|F|} = \left(\frac{q}{(1-\delta)q+1}\right)^s \leq \frac{1}{(1-\delta)^s}$;
- a communication complexity rate $\frac{\ell \log_2 q}{|F|} = \frac{q}{((1-\delta)q+1)^s} \leq \frac{1}{(1-\delta)^s} q^{1-s}$.

Example 5.4. We present various parameters of PoR instances admitting $0.10 \leq \varepsilon_0 \leq 0.16$, for files of size approaching 10^4, 10^6 and 10^9 bits. Here $\mathcal{A}$ is a $[q, (1-\delta)q+1, \delta q]_q$ MDS code (*e.g.* a Reed-Solomon code), and $\mathcal{C} = \mathcal{A}^{\otimes s}$.

q	δq	s	file size (bits)	comm. rate	redundancy rate	ε_0
16	10	4	9,604	6.664×10^{-3}	27.3	0.153
25	13	3	10,985	1.138×10^{-2}	7.112	0.141
64	24	2	10,086	3.807×10^{-2}	2.437	0.141
32	21	5	1,244,160	1.286×10^{-4}	134.8	0.122
47	28	4	960,000	2.938×10^{-4}	30.5	0.126
101	47	3	1,164,625	6.071×10^{-4}	6.193	0.101
512	180	2	998,001	4.617×10^{-3}	2.364	0.124
128	85	5	1,154,413,568	7.762×10^{-7}	208.3	0.129
256	150	4	1,048,636,808	1.953×10^{-6}	32.77	0.118
1024	550	3	1,071,718,750	9.555×10^{-6}	10.02	0.155
12167	3900	2	957,037,536	1.78×10^{-4}	2.166	0.103
16384	5500	2	1,658,765,150	1.383×10^{-4}	2.266	0.113

The previous example shows that, while the communication rate is reasonable for these PoR instances over large files, the storage needs remain large.

5.2 Reed-Muller and related codes

Low-degree Reed-Muller codes are known to admit many distinct low-weight parity-check equations, whose supports correspond to affine subspaces of the ambient space. Therefore they seem naturally adapted to our construction. Let us first consider the plane (or bivariate) Reed-Muller code case.

5.2.1 The plane Reed-Muller code $\mathrm{RM}_q(2, q-2)$

Let $\mathcal{C}$ be the Reed-Muller code

$$\mathcal{C} = \mathrm{RM}_q(2, q-2) \coloneqq \{(f(x,y))_{(x,y)\in\mathbb{F}_q^2}, f \in \mathbb{F}_q[X,Y], \deg f \leq q-2\}\,.$$

It is well known that $\mathcal{C}$ has length q^2 and dimension $(q-1)(q-2)/2$. Besides, for every line $L = \{\mathbf{x} = (at+b, ct+d), t \in \mathbb{F}_q\} \subset \mathbb{F}_q^2$ and every $c \in \mathcal{C}$, we can check that $\sum_{\mathbf{x}\in L} c_{\mathbf{x}} = 0$. Indeed, let $f \in \mathbb{F}_q[X,Y]$, $\deg f = a \leq q-2$. The restriction of f on an affine line L can be interpolated as a univariate polynomial $f_{|L}$ of degree at most a. Our claim follows since $\sum_{z\in\mathbb{F}_q} z^i = 0$ for every $i \leq q-2$.

Therefore, we can define $\mathcal{Q}$ as the set of affine lines L of $\mathbb{F}_q^2$, and $V(L,r) = \sum_{j=1}^{\ell} r_j \in \mathbb{F}_q$. From the previous discussion we see that $(\mathcal{Q}, V)$ is a verification structure for $\mathcal{C}$. Also notice there are $q(q+1)$ distinct affine lines in $\mathbb{F}_q^2$, hence $N = q(q+1)$.

Lemma 5.5. *Let $\mathcal{C} = \mathrm{RM}_q(2, q-2)$ equipped with its verification structure defined as above. Then, the response code $R(\mathcal{C})$ has minimum distance q^2+2.*

Proof. Any non-zero codeword $c \in \mathcal{C}$ consists in the evaluation of a non-zero polynomial $f(X,Y) \in \mathbb{F}_q[X,Y]$ of degree at most $q-2$. Denote by $L_1, \ldots, L_a \subset \mathbb{F}_q^2$, the affine lines on which f vanishes; *i.e.* $f(P) = 0$ for every $P \in L_i$, $1 \leq i \leq a$. We claim that $a \leq q-2$. Indeed, since f has total degree $< q-1$, it also vanishes on *closed* lines $\overline{L_1}, \ldots, \overline{L_a}$, considered as affine lines in $\overline{\mathbb{F}_q}^2$, where $\overline{\mathbb{F}_q}$ denotes the algebraic closure of $\mathbb{F}_q$. Denote by $g_i \in \mathbb{F}_q[X,Y]$ the monic polynomial of degree 1 which defines $\overline{L_i}$. From Hilbert's *Nullstellensatz*, there exists $r > 0$ such that $(\prod_{i=1}^{a} g_i) | f^r$. Since the g_i's have degree 1 and are distinct, we get $a \leq \deg f \leq q-2$. Hence, the affine lines different from $L_1, \ldots, L_a$ correspond to non-zero coordinates of $R(c)$. There are $q(q+1) - a \geq q^2+2$ such lines so $d_{\min}(R(\mathcal{C})) \geq q^2+2$.

Now we claim there exists a word $r \in R(\mathcal{C})$ of weight $N - q + 2 = q^2+2$. Let $L^{(0)}$ and $L^{(1)}$ be two distinct parallel affine lines respectively defined by $X = 0$ and $X = 1$. We build the word c which is -1 on coordinates corresponding to points in $L^{(0)}$, 1 on those corresponding to points in $L^{(1)}$ and 0 elsewhere. One can check that $c \in \mathcal{C}$; indeed c corresponds to the evaluation of $\prod_{z\in\mathbb{F}_q\setminus\{0,1\}}(z - X)$. Now, if we want to compute $\mathrm{wt}(R(c))$, we only need to count the number of lines which do not intersect $L^{(0)}$ nor $L^{(1)}$. Clearly there are only $q-2$ such lines. Hence $\mathrm{wt}(R(c)) = q(q+1) - (q-2)$ and this concludes the proof. □

Proposition 5.6. *Let $\mathcal{C} = \mathrm{RM}(2, q-2)$, and let $(\mathcal{Q}, V)$ be its associated verification structure. If every Φ_w is sufficiently uniform, then the* PoR *scheme associated to $\mathcal{C}$ and $(\mathcal{Q}, V)$ is (ε, τ)-sound for $\varepsilon = 1 - o(1)$ and $\tau = \mathcal{O}(\frac{1}{(1-\varepsilon)q^2})$, when $q \to \infty$.*

Proof. One can check that the random variables $\{X_u\}_{u\in\mathcal{D}}$ are pairwise uncorrelated since

$$\max_{u\neq v\in\mathcal{Q}^2} |u \cap v| = 1 < \ell(1-\delta) + 2 = \min_{u\in\mathcal{Q}} d_{\min}((\mathcal{C}_{|u})^{\perp})\,.$$

Besides, the relative distance of $R(\mathcal{C})$ is $\frac{q^2+2}{q(q+1)} \to 1$ according to Lemma 5.5. If every Φ_w is sufficiently uniform, the bias α satisfies $\alpha \in \mathcal{O}(1/q)$; hence $\frac{1-\alpha}{1+\alpha} = 1 + \mathcal{O}(1/q)$. Therefore we can use Theorem 3.13 and we get the desired result. □

Parameters. For an initial file of size $|F| = \frac{1}{2}(q-1)(q-2)\log_2 q$ bits, we get

- a redundancy rate $\frac{q^2 \log_2 q}{|F|} = \frac{2}{(1-1/q)(1-2/q)} \to 2$;
- a communication complexity rate $\frac{q \log_2 q}{|F|} = \frac{2}{q} \frac{1}{(1-1/q)(1-2/q)} = \mathcal{O}(1/q)$.

5.2.2 Storage improvements *via* lifted codes

The redundancy rate of Reed-Muller codes presented above stays stuck above 2. Affine lifted codes, introduced by Guo, Kopparty and Sudan [GKS13], allow to break this barrier while keeping the same verification structure. Generically, they are defined as follows:

$$\begin{aligned}\mathrm{Lift}(m,d) \coloneqq \{(f(\mathbf{P}))_{\mathbf{P}\in\mathbb{F}_q^m}, f \in \mathbb{F}_q[X_1,\ldots,X_m],\\ \forall \text{ affine line } L \subset \mathbb{F}_q^m, (f(\mathbf{Q}))_{\mathbf{Q}\in L} \in \mathrm{RS}_q(d+1)\}\,.\end{aligned}$$

We refer to [GKS13] for more details about the construction. Here we focus on $\mathrm{Lift}(2, q-2)$, since it can be compared to $\mathrm{RM}(2, q-2)$. Indeed, one sees that

$$\mathrm{RM}(2,q-2) \subseteq \mathrm{Lift}(2,q-2) \tag{4}$$

and Equation (4) turns into a proper inclusion as long as q is not a prime. Besides, by definition of lifted codes, $\mathrm{Lift}(2, q-2)$ admits the same verification structure as the one presented previously for $\mathrm{RM}(2, q-2)$.

Lemma 5.7. *The response code of* $\mathrm{Lift}(2, q-2)$ *has minimum distance at least* $q^2 - q + 2$.

Proof. The *rationale* is similar to the proof of Lemma 5.5. Let $0 \neq c \in \mathcal{C}$, $c = (f(\mathbf{P}))_{\mathbf{P}\in\mathbb{F}_q^2}$, $f \in \mathbb{F}_q[X,Y]$, and denote by $L_1,\ldots,L_a \subset \mathbb{F}_q^2$ the lines on which f vanishes. The restriction of f along L_i can be interpolated as a univariate polynomial $f_{|L_i}(T)$ of degree at most $q-2$, since $(f(\mathbf{Q}))_{\mathbf{Q}\in L_i}$ lies in the Reed-Solomon code $\mathrm{RS}_q(q-1)$, by definition of lifted codes. Therefore $f_{|L_i}(T) = 0$, and f vanishes on $\overline{L_i}$. Repeating arguments in the proof of Lemma 5.5, we get $a \leq \deg f \leq 2q-2$, and $d_{\min}(\mathcal{R}(\mathrm{Lift}(2,q-2))) \geq q^2 + q - 2q + 2 = q^2 - q + 2$. □

We believe the bound given in Lemma 5.7 is not tight, but it is sufficient to have $d_{\min}(\mathcal{R}(\mathrm{Lift}(2,q-2)))/N \to 1$. Similarly to Proposition 5.6, we can then prove that practical PoRs can be constructed with the family of lifted codes $\mathrm{Lift}(2, q-2)$.

Proposition 5.8. *Let* $\mathcal{C} = \mathrm{Lift}(2, q-2)$*, and* $(\mathcal{Q}, V)$ *its associated verification structure. If every* Φ_w *is sufficiently uniform, then the* PoR *scheme associated to* $\mathcal{C}$ *and* $(\mathcal{Q}, V)$ *is* (ε,τ)*-sound for every* $\varepsilon < 1$ *and* $\tau = \mathcal{O}(\frac{1}{(1-\varepsilon)q^2})$*, when* $q \to \infty$

The crucial improvement is that lifted codes potentially have much higher dimension than Reed-Muller codes. For $q = 2^e$, the dimension of $\mathrm{Lift}(2, q-2)$ can be proved to equal $4^e - 3^e$ [GKS13].

Example 5.9. We present parameters of PoRs based on Reed-Muller codes and lifted codes, using files of size approaching 10^4, 10^6 and 10^9 bits.

code	q	file size	comm. rate	redundancy rate
Lift	32	$3,905$	4.097×10^{-2}	1.311
RM	64	$11,718$	3.277×10^{-2}	2.097
Lift	64	$20,202$	1.901×10^{-2}	1.217
Lift	256	$471,800$	4.341×10^{-3}	1.111
RM	512	1172745	3.929×10^{-3}	2.012
Lift	512	$2,182,149$	2.112×10^{-3}	1.081
Lift	8192	$851,689,033$	1.25×10^{-4}	1.024
RM	16384	$1,878,704,142$	1.221×10^{-4}	2.000
Lift	16384	$3,691,134,818$	6.214×10^{-5}	1.018

Note that this family of codes has been used in the PoR proposal of [LL16].

5.2.3 On more generic families of codes

We have presented two rather small families of codes producing practical instances of PoR. Let us give a short summary of approximate lower bounds on crucial PoR parameters that have been shown in previous sections.

Family of codes over $\mathbb{F}_q$	redundancy rate	communication complexity rate
s-fold tensor product (Sec. 5.1.2)	$(1-\delta)^{-s}$	$q^{-(s-1)}(1-\delta)^{-s}$
plane RM (Sec. 5.2.1)	2	$2q^{-1}$
plane lifted code (Sec. 5.2.2)	$1 + q^{\log_2(3)-2}$	$q^{-1} + q^{\log_2(3)-3}$

Now we quickly mention other families of codes that could be interesting to consider.

Multi-variate generalisation. We have only presented Reed-Muller and lifted codes embedded into the affine *plane* $\mathbb{F}_q^2$. One could of course consider a broader ambient space $\mathbb{F}_q^m$, $m > 2$. Lines would have smaller *relative weight* compared to the ambient space, and thus we would decrease the communication complexity of our PoR schemes. We must however care about the storage overhead which can drastically increase if m gets large: for instance, any Reed-Muller code $\mathrm{RM}_q(m, q-2)$ has rate $\leq 1/m!$.

Lower degree generalisation. In order to increase the soundness of our PoR schemes, one could consider Reed-Muller codes $\mathrm{RM}_q(2, d)$ (as well as related lifted codes) with a lower degree $d < q-2$. The communication complexity remains unchanged; however we could observe overwhelming storage overhead if d is too small.

Combinatorial generalisation. Codes $\mathrm{Lift}(2, q-2)$ can be viewed as *codes from designs* (see [AK92] for more details), where the underlying *block design* is the classical affine plane. Considering designs with smaller block size would lead to PoRs with smaller communication complexity. But once again, this could be expensive in terms of storage, since only a few designs produce high dimensional codes.

6 Conclusion

We have proposed a security model for PoRs in line of previous work, together with a generic code-based framework. We have then sharply quantified the extraction failure of our PoR construction

as a function of code parameters. Specialising this construction for particular families of codes, we provided instances with practical parameters. We hope our work will be an incentive for further proposals of code instances, aiming at better PoR parameters.

Acknowledgements

This work is partially funded by French ANR-15-CE39-0013-01 "Manta". The authors would like to thank Daniel Augot who shared fruitful discussions on the definition of proofs-of-retrievability, as well as Alain Couvreur for his suggestion leading to the proof of Lemma 5.7.

References

[ABC+11] Giuseppe Ateniese, Randal C. Burns, Reza Curtmola, Joseph Herring, Osama Khan, Lea Kissner, Zachary N. J. Peterson, and Dawn Song. Remote Data Checking using Provable Data Possession. *ACM Trans. Inf. Syst. Secur.*, 14(1):12:1–12:34, 2011.

[AK92] Edward F. Assmus and Jennifer D. Key. *Designs and Their Codes.* Cambridge Tracts in Mathematics. Cambridge University Press, 1992.

[BJO09] Kevin D. Bowers, Ari Juels, and Alina Oprea. Proofs of Retrievability: Theory and Implementation. In Radu Sion and Dawn Song, editors, *Proceedings of the first ACM Cloud Computing Security Workshop, CCSW 2009, Chicago, IL, USA, November 13, 2009*, pages 43–54. ACM, 2009.

[DVW09] Yevgeniy Dodis, Salil P. Vadhan, and Daniel Wichs. Proofs of Retrievability via Hardness Amplification. In Omer Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 109–127. Springer, 2009.

[GKS13] Alan Guo, Swastik Kopparty, and Madhu Sudan. New Affine-Invariant Codes from Lifting. In Robert D. Kleinberg, editor, *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 529–540. ACM, 2013.

[JK07] Ari Juels and Burton S. Kaliski, Jr. PORs: Proofs of Retrievability for Large Files. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*, pages 584–597. ACM, 2007.

[LEB+03] Mark Lillibridge, Sameh Elnikety, Andrew Birrell, Michael Burrows, and Michael Isard. A Cooperative Internet Backup Scheme. In *USENIX Annual Technical Conference, General Track*, pages 29–41. USENIX, 2003.

[LL16] Julien Lavauzelle and Françoise Levy-dit-Vehel. New Proofs of Retrievability using Locally Decodable Codes. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 1809–1813. IEEE, 2016.

[MZC12] Zhen Mo, Yian Zhou, and Shigang Chen. A Dynamic Proof of Retrievability (PoR) Scheme with $O(\log n)$ Complexity. In *Proceedings of IEEE International Conference*

on Communications, ICC 2012, Ottawa, ON, Canada, June 10-15, 2012, pages 912–916. IEEE, 2012.

[NR09] Moni Naor and Guy N. Rothblum. The Complexity of Online Memory Checking. *J. ACM*, 56(1):2:1–2:46, 2009.

[PSU13] Maura B. Paterson, Douglas R. Stinson, and Jalaj Upadhyay. A Coding Theory Foundation for the Analysis of General Unconditionally Secure Proof-of-Retrievability Schemes for Cloud Storage. *J. Mathematical Cryptology*, 7(3):183–216, 2013.

[PSU18] Maura B. Paterson, Douglas R. Stinson, and Jalaj Upadhyay. *J. Mathematical Cryptology*, 12(4):203–220, 2018.

[SR16] Binanda Sengupta and Sushmita Ruj. Efficient Proofs of Retrievability with Public Verifiability for Dynamic Cloud Storage. *CoRR*, abs/1611.03982, 2016.

[SW13] Hovav Shacham and Brent Waters. Compact Proofs of Retrievability. *J. Cryptology*, 26(3):442–483, 2013.

[WWR+11] Qian Wang, Cong Wang, Kui Ren, Wenjing Lou, and Jin Li. Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. *IEEE Trans. Parallel Distrib. Syst.*, 22(5):847–859, 2011.

A Experimental estimate of the bias α

We here confirm our heuristic on the fact that Φ_w is sufficiently uniform, by providing experimental estimates of α.

Setup. We consider PoR schemes using Reed-Muller codes $\mathcal{C} = \mathrm{RM}_q(2, q-2)$, as presented in Section 5.2.1. We also fix the word $w \in \mathbb{F}_q^n$ uploaded on the server during the initialisation step. Remark that, for varying w, all Φ_w are equivalently distributed. Indeed, if $\psi \in \mathfrak{S}(\mathbb{F}_q)^n$ satisfies $\psi(w) = w'$, then the distribution of permutations picked from $\Phi_{w'}$ can be obtained by applying ψ to permutations picked from Φ_w. Hence, without loss of generality we assume that $w = 0$. Proposition 3.16 claims that in this context, α should be $\mathcal{O}(1/q)$ since $\Delta = 2$ and $\ell \leq q$. For convenience, we denote by $p_\Phi \coloneqq \mathbb{P}_{\Phi_w}(V^\sigma(u, r_u) = 0)$, and we recall that α is an upper bound on p_Φ (for varying u and r).

We proceed to three kinds of tests in order to estimate α:

- **Test 1.** We sample N challenges u, and for each sample, we fix $t \leq \ell$ and r_u in $\{x \in \mathbb{F}_q^\ell, |Z_u| = t\}$. Then, we estimate p_Φ by running M trials and computing the average number of times $V^\sigma(u, r_u) = 0$ occurs. We denote by $\xi_M(p_\Phi)$ this *estimator*. We then collect the maximum value of $\xi_M(p_\Phi)$ among the N samples of u.
- **Test 2.** A challenge u is fixed, and for several values of t, we pick N responses r_u randomly in $\{x \in \mathbb{F}_q^\ell, |Z_u| = t\}$. For every r_u, we estimate p_Φ with M samples. We collect the maximum value of $\xi_M(p_\Phi)$ among the N values of r_u that have been picked.
- **Test 3.** A challenge u is fixed, as well as a response r_u to this challenge which satisfies $|Z_u| = t$ for several values of $t \in [2, \ell]$. We then run M trials and collect $\xi_M(p_\Phi)$.

Influence of M and the chosen test on the estimator. At the end of the document, Figures 5, 6 and 7 confirm that, for fixed N and q, and for any Test i we use, $i \in \{1, 2, 3\}$, our estimator $\xi_M(p_\Phi)$ converges to a value close to $1/(q-1)$.

Influence of N on the estimator. Table 1 shows experimentally that, for M large enough and fixed q, the number N has few influence on the estimator (N being respectively the number

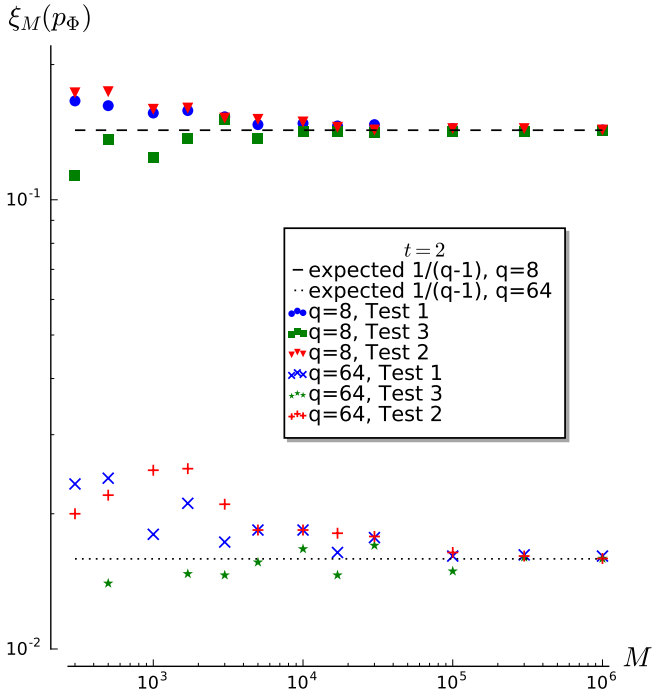

Figure 5: Estimators for various values of $M \in [10^3, 10^6]$, of $q \in \{8, 64\}$, and of Test i, $i \in \{1, 2, 3\}$. Support size $t = 2$ is fixed. For Tests 1 and 2, the parameter N is set to 10. Black horizontal lines represent the expected value of α.

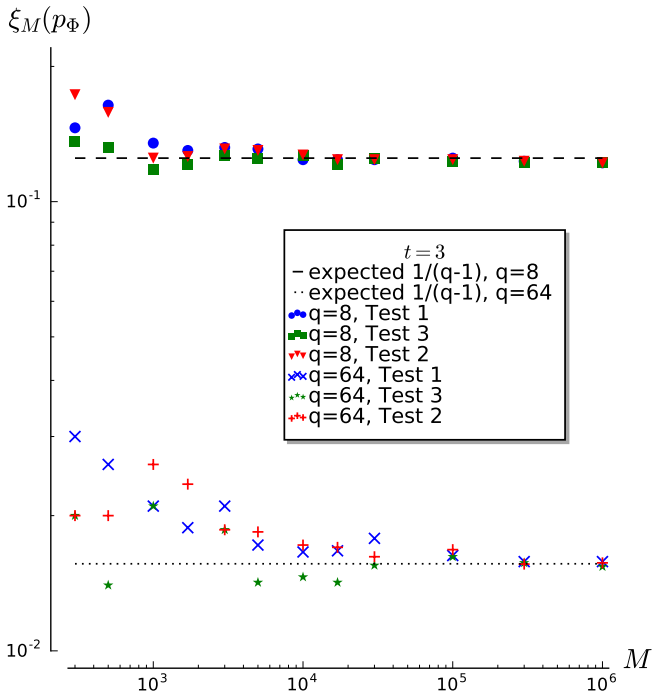

Figure 6: Estimators for various values of $M \in [10^3, 10^6]$, of $q \in \{8, 64\}$, and of Test i, $i \in \{1, 2, 3\}$. Support size $t = 3$ is fixed. For Tests 1 and 2, the parameter N is set to 10. Black horizontal lines represent the expected value of α.

of responses r_u sampled in Test 2, and the number of challenges u sampled in Test 1). The minor increase of the values can be thought as a standard deviation due to the fact that the number of samples $M = 100,000$ is finite.

N	$q = 8$	$q = 64$	N	$q = 8$	$q = 64$
1	0.1418	0.0152	1	0.1414	0.0158
5	0.1433	0.0163	5	0.1431	0.0162
10	0.1443	0.0165	10	0.1452	0.0166
50	0.1455	0.0169	50	0.1450	0.0168
100	0.1452	0.0167	100	0.1458	0.0168
500	0.1464	0.0169	500	0.1470	0.0168
$1/(q-1) =$	0.1429	0.01587	$1/(q-1) =$	0.1429	0.01587

Table 1: Estimators using Test 1 (on the left) and Test 2 (on the right) with $M = 100,000$ and $t = 2$, for $q \in \{8, 64\}$ and various values of N. The quantity $1/(q-1)$ represents an estimated upper bound on α that $\xi_M(p_\Phi)$ should approximate.

Influence of q on the estimator. In Table 2, we show that estimator $\xi_M(p_\Phi)$ converges to an expected value $1/(q-1)$, for any value of q.

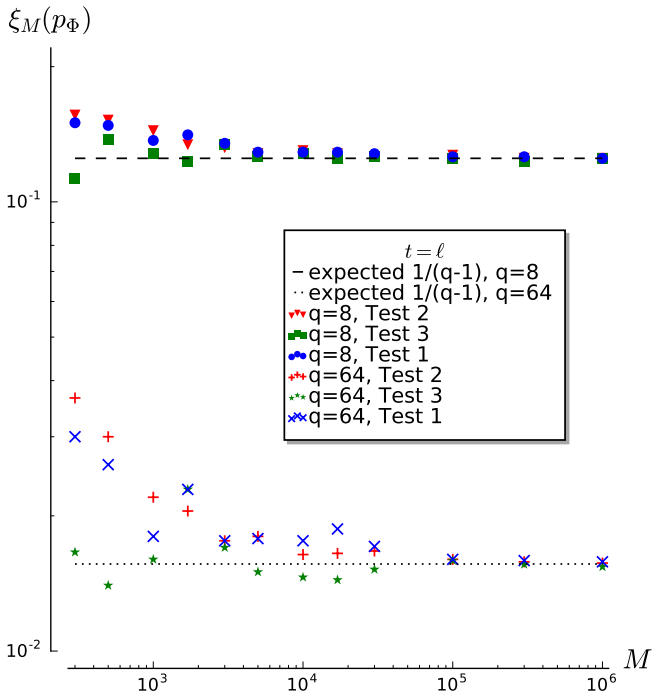

Figure 7: Estimators for various values of $M \in [10^3, 10^6]$, of $q \in \{8, 64\}$, and of Test i, $i \in \{1, 2, 3\}$. Support size $t = \ell$ is fixed. For Tests 1 and 2, the parameter N is set to 10. Black horizontal lines represent the expected value of α.

q	$\xi_M(p_\Phi)$	$1/(q-1)$
4	0.333	0.3333
8	0.143	0.1429
16	0.0665	0.06667
32	0.032	0.03226
64	0.0161	0.01587
128	0.00791	0.007874
256	0.00382	0.003922

q	$\xi_M(p_\Phi)$	$1/(q-1)$
7	0.166	0.1667
17	0.0627	0.0625
31	0.0335	0.03333
257	0.00398	0.004000

Table 2: Estimators using Test 3 with $M = 1,000,000$ and $t = 2$, for various values of prime powers q. The quantity $1/(q-1)$ represents an estimated upper bound on α that $\xi_M(p_\Phi)$ should approximate.