On the Concrete Security of Non-interactive FRI

Alexander R. Block[1] [0000-0002-2632-763X] and Pratyush Ranjan Tiwari[2]

[1] Georgetown University and University of Maryland. alexander.r.block@gmail.com
[2] Johns Hopkins University. pratyush@cs.jhu.edu

Abstract. FRI is a cryptographic protocol widely deployed today as a building block of many efficient SNARKs that help secure transactions of hundreds of millions of dollars per day. The Fiat-Shamir security of FRI—vital for understanding the security of FRI-based SNARKs—has only recently been formalized and established by Block et al. (ASIACRYPT '23).
In this work, we complement the result of Block et al. by providing a thorough concrete security analysis of non-interactive FRI under various parameter settings from protocols deploying (or soon to be deploying) FRI today. We find that these parameters nearly achieve their desired security targets (being at most 1-bit less secure than their targets) for non-interactive FRI with respect to a certain security conjecture about the FRI Protocol. However, in all but one set of parameters, we find that the provable security of non-interactive FRI under these parameters is severely lacking, being anywhere between 21- and 63-bits less secure than the conjectured security. The conjectured security of FRI assumes that known attacks are optimal, the security of these systems would be severely compromised should a better attack be discovered. In light of this, we present parameter guidelines for achieving 100-bits of provable security for non-interactive FRI along with a methodology for tuning these parameters to suit the needs of protocol designers.

1 Introduction

Fast verification of computation and statements from untrusted parties is a key problem at the heart of proof systems and complexity theory. Given the vast amount of high-impact applications in verifiable computing, there has been a Cambrian explosion of research on Succinct Non-interactive ARguments of Knowledge (SNARKs) and their applications. A common blueprint for SNARK construction is: (1) construct a public-coin[3] interactive protocol; and (2) remove interaction via the Fiat-Shamir (FS) transformation [19].

However, it is well-known that the FS transformation is *not* secure [3,20,11] when applied to many-round (i.e., super-constant) protocols, even in the random oracle model. When considering an r-round protocol with λ-bits of *interactive security*, the non-interactive protocol built from the FS transformation of this interactive protocol often has roughly $\lambda - r$ bits of *non-interactive security*—a significant loss.[4] Moreover, the number of rounds r can be large enough to rival λ, potentially resulting in a completely insecure non-interactive protocol. Therefore, it does not suffice to prove and provide

[3] An interactive protocol is *public-coin* if all messages sent by the verifier are sampled uniformly at random and are independent of all prior protocol messages.
[4] See Remark 2 for a discussion on the difference between these notions.

security analysis only for interactive versions of protocols and then disregard the impact of Fiat-Shamir, as many protocols have, including *all* FRI-based SNARKs. A non-interactive security analysis is necessary before the deployment of such protocols in settings where protocol failure can have significant financial impacts.

Once a protocol's non-interactive security has been determined, a number of additional key considerations need to be made before deployment, including adversarial capabilities, outcomes, and motivations for attacking the protocol. In financial applications specifically, this can often be analyzed as follows: does breaking the protocol's security cost more than the monetary benefit possible from doing so? A rational actor analysis would always want to answer this question in the affirmative. However, with the growing complexity of protocols and the reduced cost of computational resources, concrete security analysis of protocols requires more attention.

A recent work of Block et al. [12] has begun to address this gap (a similar analysis for ethSTARK [28] was also published concurrently and independently) for the FRI (**F**ast **R**eed-Solomon **I**nteractive Oracle Proof of Proximity) Protocol [6] and related SNARKs by analyzing FS security of FRI. This was an important step forward since FRI serves as a low-degree proximity testing subroutine for many practical proof systems deployed today, including several that secure millions of dollars of transactions per day [24].

However, current protocols deploying FRI determine parameters based on interactive security. Moreover, parameters are selected based on *conjectured (information theoretic) security* of FRI [28], rather than the best known provable (information-theoretic) security bounds for FRI [7]. The conjectured security of FRI boils down to the assumption that the best known attacks on an information-theoretic proof system related to FRI are optimal [28]. If this conjecture holds, then FRI parameters under this conjecture have many real-world performance benefits: (1) much smaller proof sizes in practice; and (2) smaller field size requirements, increasing prover efficiency. These are the main reasons many deployed implementations assume the conjecture to be true. However, our work highlights the difference between conjectured and provable security. Though no known attacks contradict this conjecture, there is a very large gap between this conjectured security and the provable security of FRI. In particular, if an attack is ever found against the conjectured security, there is a huge loss in security when reverting to provable security guarantees of FRI.

There is another less aggressive security conjecture about the soundness of FRI given in [7]. This conjecture is weaker than the conjecture assumed by practitioners today. However, if the aggressive conjecture used today is proven false, one may still hope to prove stronger bounds from a conjecture presented in [7]. This less aggressive conjecture still has the benefits described above (though to a lesser extent) when compared with provable security. However, the same problem remains if an attack was found against this conjecture; i.e., there is still a huge loss in security between this less aggressive conjecture and provable security guarantees of FRI.

1.1 Our Results

We build on and complement Block et al.'s results by analyzing the concrete security of FRI-based protocols under a variety of parameter settings that are *adopted in practice* by real-world systems that deploy (or will soon deploy) FRI in various Layer-2 Ethereum projects [30,26]. We examine these parameter settings under both the best *provable security* bounds for FRI [7] along with a certain *conjectured information theoretic security* bound for FRI given in [28]. We find that most of these protocols have identified their deployment security parameters based purely on an analysis of the *interactive security* of the chosen protocol, without accounting for the soundness loss inherent in the non-interactive variant. We believe this happened because no formal treatment of FRI's concrete security has been put forward, which makes it challenging for practitioners to determine the exact security of their protocols. The result is that the incentive for attackers to improve on the best-known attacks is close to a billion dollars (Tab. 1).

As we note in this work, such an analysis is challenging for several reasons: (1) multiple parameter choices (such as the field size, Reed-Solomon code parameters, number of queries, etc.) all drastically affect the soundness of the resulting protocol and the size of the resulting non-interactive proofs; (2) some (information theoretic) soundness conjectures are poorly understood; and (3) many implementations incorporate engineering optimizations that may have unpredictable impacts on protocol soundness. We are the first work to address all of these issues. Our analysis aims to provide practitioners with a ready reckoner to set their FRI parameters according to their desired level of security.

Summary of Results. We summarize our results on the current state of deployed non-interactive FRI protocols, with respect to their target security level, provable non-interactive security level, and conjectured non-interactive security level, in Tab. 1. For *all* settings of parameters in the projects we examine, the non-interactive *conjectured security* of FRI achieved by these parameters is at most 1-bit below their security target. In contrast, we found that only *a single deployed project* includes parameter settings that achieve the designers' desired security targets with respect to *provable security* (or are at most 1-bit less than their target); *all other systems* have a gap between provable and conjectured security that can range from 21 to 63-bits of security, in some cases placing the security of the entire system at risk from realistic attackers. Following is a list of applications where these protocols are used:

- **zkEVMs/Rollups.** FRI-based SNARKs are currently used in various Layer-2 Ethereum projects [30,26] to help secure hundreds of millions of dollars of assets [24]. These protocols improve scalability by executing/processing multiple transactions/smart contract logic off-chain and then submit an updated state as well as a proof of correctness for the update. A malicious prover that can forge a proof for this application means that *invalid transaction/smart contract logic can be executed* and invalid state updates are accepted. This can result in a **loss of funds**.
- **Non-Custodial Exchanges.** Here, the protocol uses cryptographic proofs to attest to the validity of a batch of transactions and updates a commitment to the state of the exchange on-chain. A malicious prover that can forge a proof for this application means that *invalid transaction/asset trading logic can be executed* and invalid state updates are accepted. This can result in a **loss of funds**.
- **Verifiable Computation & Delegation.** In general, SNARKs allow one to prove the correct execution of any computation. Some projects, such as RISC Zero [31], aim to deploy FRI-based SNARKs to prove the correct execution of arbitrary computation,

such as any application logic. A malicious prover that can forge a proof for this application means that an *invalid execution of computation is accepted*. The impact of this depends on the particular application.

Table 1. Security assessment of Various FRI-based SNARK Projects. The difference in the targeted and provable security of non-interactive FRI is highlighted with bold text.

Org	Repo	Target FRI Security Bits	Provable FS-FRI Security	Conjectured FS-FRI Security	Impact
Polygon	Plonky2	**100**	**38**	99	used by 100+ repositories
Starkware	stone-prover	**96**	**54**	99	\$220M (ImmutableX + Sorare)★
	SHARP Verifier	**96**	**59**	95	Smart contract execution
dYdX	dYdX Protocol	**80**	**52**	79	\$318M on exchange★
Polygon Miden	Miden-VM	**96/128**	**45/67**	96/128	zkVM in testing
Lambda Class	lambda works	**80/100/128**	**81/99/127**	81/101/129	Library for SNARKs
RISC Zero	RISC Zero	**100**	**37**	99	zkVM
Matter Labs	era-boojum	**100**	**50**	99	To replace zkSync-Era (\$841M)★

★ As of May 6, 2024, 00:00 UTC.

Bits of Security. The number of bits of security for a protocol indicates the amount of computational work/cost for an adversary to break the security. Usually, λ-bit security implies that the expected number of operations for an adversary to break soundness is 2^λ for proof systems.[5] The protocols that utilize FRI are usually IOPs compiled to non-interactive SNARKs using cryptographic hash functions as random oracles. Given this, our analysis also highlights the explicit trade-off between adversarial query/hashing power and soundness error of the non-interactive FRI protocol.

Limitations of Our Analysis. Given that FRI is deployed as a sub-routine in the various systems we examine in this work (and many other systems as well), our analysis indicates the security level *only* for the FRI sub-routine. All parts of the protocol need to be similarly analyzed to obtain an end-to-end concrete security analysis. We hope our work encourages end-to-end concrete security analyses of non-interactive protocols deployed in practice. We also do not make any progress on settling the conjectures on the security of FRI, but hope that this work serves as a clear indication for why these conjectures serve as honeypots to potential attackers. If an attacker comes up with a better attack, they could potentially attack multiple deployed projects presented in Tab. 1.

[5] A similar definition for bits of security is given in the ethSTARK documentation [28].

1.2 Related Work

The work most related to ours is the updated ethSTARK documentation [28]. They show both interactive and Fiat-Shamir security of FRI and their ethSTARK protocol, along with various parameter settings to achieve a variety security levels. Next would be the work of Block et al. [12], who give a formal analysis of the Fiat-Shamir of the FRI protocol and some discussion on concrete parameter choices; our work expand greatly upon thier discussion. Haböck [21] also gives an extensive parameter analysis for FRI, but in the context of interactive security and the DEEP-FRI method [9]. Ben-Sasson et al. [4,5] investigate the concrete security of probablistically checkable proofs (PCPs) and PCPs of proximity, protocols that are closely related to interactive oracle proofs [8] and IOPs of Proximity [6], with the goal of improving their concrete efficiency. Other works examining the concrete security of cryptographic protcols include [23,22,18].

The Fiat-Shamir (FS) transformation [19] has been studied and used extensively in the context of removing interaction from interactive protocols. It is known that the FS transformation is secure when applied to secure protocols with a constant number of interaction rounds in the random oracle model (ROM) [19,25,1]; however, it is also well-known that there exist protocols which are FS-secure in the ROM but insecure for *any* concrete instantiation of the random oracle [3,20,11]. Moreover, there are several natural classes of secure interactive protocols that become insecure after applying the FS transformation (e.g., sequential repetition of a protocol and parallel repetition of certain classes of protocols) [14,33,2]. It is also often the case that real-world implementations of the FS transformation are done incorrectly, leading to vulnerabilities of deployed systems [10,17]. Nonetheless, the FS transformation is widely deployed and a critical piece of the majority of SNARG or SNARK constructions.

2 Preliminaries

Throughout this work, we let $\mathbb{F}$ denote a finite field of prime p elements and let $\mathbb{F}^\times := \mathbb{F} \setminus \{0\}$ denote the multiplicative group of $\mathbb{F}$ and we let $\mathbb{F}[X]$ denote the set of all univariate polynomials with coefficients in $\mathbb{F}$ and indeterminate X, and we let $\mathbb{F}^{\leqslant k}[X]$ (resp., $\mathbb{F}^{<k}[X]$) denote the set of all univariate polynomials of degree at most k (resp., degree less than k). We say that $L \subseteq \mathbb{F}^\times$ is a multiplicative subgroup if it is closed under multiplication; i.e., if $x, y \in L$ then $x \cdot y \in L$. For any finite set S, we let $s \xleftarrow{\$} S$ denote the process of sampling an element of S uniformly and independently at random.

For two vectors $u, v \in \mathbb{F}^n$, we let $\Delta(u, v)$ denote the *relative Hamming distance* between u and v, defined as $\Delta(u, v) := |\{u_i \neq v_i \mid i \in [n]\}| / n$. Moreover, for a set of vectors $S \subset \mathbb{F}^n$ and any vector $u \in \mathbb{F}^n$, we define $\Delta(u, S) = \Delta(S, u) := \min_{v \in S}\{\Delta(u, v)\}$. For $\delta \in (0, 1)$, we say that u is *δ-far from S* if $\Delta(u, S) \geqslant \delta$; otherwise, we say that u is *δ-close to S*. Equivalently, u is δ-far from S if $\Delta(u, v) \geqslant \delta$ for all $v \in S$, and u is δ-close to S if there exists $v^* \in S$ such that $\Delta(u, v^*) < \delta$.

2.1 Reed-Solomon Codes

Reed-Solomon (RS) codes [27] are a well-studied and widely used class of linear error correcting codes. In this work, we consider RS codes parameterized by a finite field $\mathbb{F}$, a multiplicative subgroup $L \subseteq \mathbb{F}^{\times}$, and a degree bound $d \in \mathbb{N}$. The code $\mathsf{RS}[\mathbb{F}, L, d]$ is defined as $\mathsf{RS}[\mathbb{F}, L, d] := \{(f(z))_{z \in L} \in \mathbb{F}^{|L|} \mid f \in \mathbb{F}^{<k}[X]\}$. Equivalently, $\mathsf{RS}[\mathbb{F}, L, d]$ is d-dimensional subspace of $\mathbb{F}^{|L|}$, where each vector is uniquely defined as the evaluation of a polynomial in $\mathbb{F}^{<d}[X]$ at all points in L (in some canonical order). We assume that $|L| = 2^n$ and $d = 2^k$ are integer powers of two with $k \leqslant n/2$. The *rate* of the RS code is defined as $\rho := d/|L| = 2^{-(n-k)}$. For our purposes, we require some additional structure on the subgroup L: $x \in L \iff \omega_n^i \cdot x \in L$ for all $i \in [n]$ where ω_n is a primitive n-th root of unity in $\mathbb{F}$. We refer to such a subgroup as a *smooth multiplicative subgroup*.

2.2 Non-interactive Proofs and Concrete Security

In this work, we focus extensively on analyzing the concrete security of *non-interactive (random oracle) proofs*, defined below.

Definition 1 (Non-interactive Random Oracle Proofs [8]). *A* non-interactive random oracle proof *(NIROP) for relation* $\mathbf{R}$ *with soundness error* ε *is a tuple* $\Pi = (\mathcal{P}, \mathcal{V})$ *of probabilistic oracle algorithms such that the following two properties hold.*

1. ***Completeness.*** *For every* $(x; w) \in \mathbf{R}$ *and* $\kappa \in \mathbb{N}$, $\Pr[\mathcal{V}^H(x, \pi) = 1 \mid H \xleftarrow{\$} \mathcal{U}(\kappa), \pi \leftarrow \mathcal{P}^H(x, w)] = 1$.
2. ***Soundness.*** *For every* $x \notin \mathcal{L}(\mathbf{R})$, *$Q$-query* $\widetilde{\mathcal{P}}$, *and* $\kappa \in \mathbb{N}$, $\Pr[\mathcal{V}^H(x, \widetilde{\pi}) = 1 \mid H \xleftarrow{\$} \mathcal{U}(\kappa), \widetilde{\pi} \leftarrow \widetilde{\mathcal{P}}^H(x)] \leqslant \varepsilon(|x|, Q, \kappa)$, *where* $\mathcal{L}(\mathbf{R})$ *is the language corresponding to the relation* $\mathbf{R}$.

Remark 1. In practice, the random oracle H is heuristically replaced with a concrete hash function (e.g., SHA256). However, for analysis purposes, we work in the random oracle model.

In this work, we measure the concrete security of NIROPs using *bits of security*.

Definition 2 (Bits of Security [28]). *Let* Π *be a non-interactive random oracle proof with soundness error* $\varepsilon(Q, \kappa)$ *relative to a random oracle* $\mathcal{H} \colon \{0,1\}^* \to \{0,1\}^{\kappa}$. *Then* Π *has* λ bits of security *if for any* $Q \in \mathbb{N}$ *we have* $Q/\varepsilon(Q, \kappa) \geqslant 2^{\lambda}$.

Remark 2. Intuitively, the bits of security of a system is (logarithm) of the expected amount of work an attacker has to perform to break a system with probability close to 1. In other words, in the random oracle model, a system has λ-bits of security if any adversary making $\log(Q) = o(\lambda)$ query succeeds in breaking the system. When analyzing the Fiat-Shamir security (i.e., our setting of non-interactive security), this intuition and Def. 2 happen to coincide, but this is not always the case. For example, when analyzing the bits of security of hash functions, any hash function achieving λ-bits of security must at least have 2λ-bit digests due to Birthday attacks.

Moreover, this notion is different from bits of security for *interactive protocols*. Roughly speaking, an interactive protocol has λ-bits of security if the success probability of *any valid attacks* (e.g., computationally unbounded or bounded) is at most $2^{-\lambda}$. This is another implicit assumption made in practice, that λ-bits of interactive security are preserved under Fiat-Shamir, a statement that is not true in general.

3 Non-interactive Security of the FRI Protocol

The FRI protocol is a logarithmic round *interactive oracle proof of proximity* [6] in which a (computationally unbounded) prover P convinces a (computationally weak) verifier V that a function $g \in \mathbb{F}^N$ known by P is δ-close to $\mathsf{RS}[\mathbb{F}, L, d]$, where $|L| = N$. FRI is widely deployed as a sub-protocol in many SNARKs (cf. Tab. 1) due to its poly-logarithmic sized proofs (once compiled into a SNARK) and plausible post-quantum security. For a complete description of the FRI protocol (taken from [12]), see the full version of our work.

We are interested in the non-interactive security of various real-world deployments of the FRI protocol. By non-interactive security, we mean specifically the bits of security (Def. 2) achieved by various deployments of FRI. In practice, FRI transformed into a non-interactive proof via a variant of the Fiat-Shamir transformation, known as the *BCS transformation* [8]. Prior to the work of Block et al. [12], it was not known whether the FRI protocol remained secure after applying the BCS transformation due to the fact that FRI has a super-constant number of rounds (with respect to the length of the input). Block et al. proved that the FRI protocol satisfied a stronger notion of security known as *round-by-round soundness* [14], which implies non-interactive security of FRI after applying the BCS transformation [16], as given by the following lemma.

Lemma 1 ([16]). *Let* (P, V) *be an interactive protocol and let* $(\mathcal{P}, \mathcal{V})$ *be the non-interactive random oracle proof obtained by applying the BCS transformation to* (P, V)*. If* (P, V) *has round-by-round (RBR) soundness error* ε*, then for any query upper bound* Q *and random oracle output length* $\kappa \in \mathbb{N}$*,* $(\mathcal{P}, \mathcal{V})$ *has (adaptive) soundness error* $\varepsilon' = Q\varepsilon + 3(Q^2 + 1)/2^\kappa$*.*

The exact definition of round-by-round soundness is not essential to understanding this work; however, we include its definition (as well as a more thorough treatment of interactive protocols) in the full version of our work.

One of the key contributions of Block et al. [12] was establishing bounds on the RBR soundness error of the FRI protocol. In particular, the bounds they obtain vary depending on two key contexts, which we discuss next. First, they establish bounds utilizing both the best provable security analysis FRI (due to [7]), and the conjectured security of FRI (due to [28]). Second, in the context of the *batched FRI* protocol (discussed in this section), they obtain bounds that depend on the type of batching used.

3.1 Provable & Conjectured Security of FRI

The RBR soundness error of FRI, which we denote as $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$, changes drastically depending on whether one utilizes the best-known *provable* soundness guarantees for the FRI protocol, or if one utilizes the *conjectured* security of FRI. As such, we state two separate theorems bounding $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$ and use them in our security analyses. We begin with the best provable bounds.

Theorem 1 ([12]). *Let* $\mathbb{F}$ *be a finite field,* $L_0 \subset \mathbb{F}^\times$ *be a smooth multiplicative subgroup of size* 2^n*,* $d_0 = 2^k$*,* $\rho = d_0/|L_0| = 2^{-(n-k)}$*, and* $\ell \in \mathbb{N}$*. For any integer* $m \geqslant 3$*,*

$\eta \in (0, \sqrt{\rho}/(2m))$, and $\delta \in (0, 1-\sqrt{\rho}-\eta)$, with respect to $\mathsf{RS}[\mathbb{F}, L_0, d_0]$, the FRI protocol has round-by-round soundness error

$$\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})} = \max\left\{\frac{(m+1/2)^7 \cdot |L_0|^2}{3\rho^{3/2} \cdot |\mathbb{F}|}, (1-\delta)^{\ell}\right\}. \tag{1}$$

Next, we state the RBR soundness error of FRI with respect to the ethSTARK conjecture about the security of FRI.

Conjecture 1 ([28]). Let $\mathbb{F}$ be a finite field, $L_0 \subset \mathbb{F}^{\times}$ a smooth multiplicative subgroup of size 2^n. For any $\rho \in (0, 1-1/|\mathbb{F}|)$, $d_0 = \rho \cdot |L_0|$, $\ell \in \mathbb{N}$, and any $\delta \leqslant 1-\rho$, for any function $G_0 \colon L_0 \to \mathbb{F}$ that is δ-far from $\mathsf{RS}[\mathbb{F}, L_0, d_0]$, the FRI protocol has round-by-round soundness error

$$\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})} = \max\left\{1/|\mathbb{F}|, (1-\delta)^{\ell}\right\}. \tag{2}$$

Remark 3. The actual ethSTARK conjecture [28] is related to another protocol and "Toy Problem" related to FRI, and not the actual FRI protocol itself. However, the parameters of FRI in various deployments are with respect to this specific conjecture (whether these deployments actually use the variant of FRI the conjecture relates to). As such, [12] directly adapt the ethSTARK conjecture to the RBR soundness analysis of the FRI protocol. For completeness, we include the discussion from [12] about the actual ethSTARK conjecture in the full version of our work.

With respect to Thm. 1, Conj. 1 differs in two key ways. First, the proximity parameter δ in Conj. 1 can be as large as $1-\rho$, in contrast to $\delta < 1-\sqrt{\rho}$ in the provable security setting. In every system we examined, δ is taken to be exactly $1-\rho$, which decreases proof sizes by allowing for fewer verifier queries to reach targeted bits of security. Second, the error term $1/|\mathbb{F}|$ is orders of magnitude less than the provable security error term $(m+1/2)^7|L_0|^2/(3\rho^{3/2}|\mathbb{F}|)$.

3.2 Grinding

A common proof-size optimization technique used with FRI is called *grinding*. Grinding is the process of having an honest prover perform a (mildly hard) proof of work during any round of the protocol, which is captured by a grinding parameter z. In the FRI protocol, grinding adds one more round of interaction just before the final round of the protocol. During this round, the prover sends some message m_{pow} to the verifier, who then responds with a random string $c_{\mathsf{pow}} \in \{0,1\}^z$. Then the prover and verifier proceed with the remainder of the protocol, and the verifier now accepts if and only if (1) the original FRI verifier accepts; and (2) $c_{\mathsf{pow}} = 0^z$ (i.e., is the all-zeros string). Now when compiling the protocol into a non-interactive argument via the Fiat-Shamir/BCS transformation, this grinding corresponds to the prover computing multiple values of m_{pow} and querying the random oracle at m_{pow} until the first z-bits of the random oracle output are identically 0. See [28, Section 6.3] for more details.

As shown by ethSTARK [28], performing z-bits of grinding in round i of an interactive protocol reduces the round-by-round soundness error of that round by 2^{-z}; in the context of FRI, this reduces the round-by-round soundness error of the query phase from $(1-\delta)^\ell$ down to $(1-\delta)^\ell \cdot 2^{-z}$ (cf. [28, Theorem 6]). We use this fact in our analysis since nearly all works we examined incorporate grinding into their implementation of FRI because grinding reduces the number of verifier queries ℓ that need to be performed, directly reducing the proof sizes of non-interactive FRI. We refer the reader to [28] for an extensive analysis of how grinding affects the soundness of interactive protocols.

3.3 Other FRI Optimizations

There are three optimization techniques which are widely used in practice that we do not consider in our analysis, but mention here for completeness.

Batched FRI. Often in practice, one needs to run FRI on many different functions $F_1, \ldots, F_t \in \mathbb{F}[X]$ and prove their proximity simultaneously. Batched FRI allows one to prove simultaneously that all t functions F_i are δ-close to the same RS code (e.g., see [7,28]), rather than run t independent instances of FRI. The Batched FRI protocol essentially adds one more round to the FRI protocol where the verifier will send uniformly random challenges $\alpha_i \in \mathbb{F}$ and then run the FRI protocol on the new function $G = \sum_i \alpha_i F_i$. A common way to save on communication cost/proof size is to sample a single α and set the rest of the randomness as $\alpha_i = \alpha^i$.

In our analysis, we do not consider the Batched FRI protocol in order to provide both a simple and best-case analysis, since:

- the number of functions t batched in different projects widely varies;
- for both Batched FRI variants (α_i vs α^i), batching only increases the proof sizes by (roughly) an additive factor proportional to t (i.e., Batched FRI amortizes well); and
- communication-saving Batched FRI (α^i) has roughly $\log(t)$-bits less security than plain FRI or regular Batched FRI [7,28,12].

Note that the above security loss is present in any project that uses Batched FRI with a single random challenge, and occurs for both the provable and conjectured security bounds for FRI. Thus, to simplify our analysis, as well as present the best-case bits of security, we omit analyzing Batched FRI in systems currently deploying it.

Larger FRI Folds. In the FRI protocol, every round essentially reduces a claim about a degree d polynomial to one about a degree $d/2$ polynomial; this continues for a logarithmic number of rounds until the degree is constant (i.e., a constant polynomial). This degree reduction can be thought of as "folding together" the even and odd halves of a polynomial via a random linear combination specified by the verifier. However, one can incorporate larger folds; for example, many configurations will reduce degree d to degree $d/4$ or $d/8$. Asymptotically, these folds still result in a logarithmic round protocol, but concretely reduce the proof sizes by large factors (e.g., 2× or 4× smaller). Unfortunately, these savings also result in some losses in security. In particular, larger folding looks like the communication-saving batching we discussed previously; i.e., for reducing d to $d/4$, a function f is divided into 4 parts, say $f_1, \ldots, f_4$, and then are "folded" as $f' = \sum_i \beta^i f_i$

for β randomly sampled by the verifier (see [21]). So by performing a FRI fold of size $k \geqslant 3$, there is roughly a $\log(k)$ loss in security. Thus, we omit this from our analysis to present the best-case bits of security of parameters for FRI in practice. Moreover, every project specifies different fold sizes, so omitting this greatly simplifies our analysis.

Early Protocol Termination. The original FRI protocol specifies termination once the degree bound has been reduced to 1 (i.e., when the polynomial is a constant). At this point in the protocol, the final prover message is a single constant $C \in \mathbb{F}$ corresponding to this polynomial. However, in practice, the protocol is often terminated at a larger (but still constant) degree bound; e.g., $d = 2^8$ is a choice in many systems. In this case, the protocol remains secure so long as the prover sends the 2^8 coefficients directly to the verifier. For non-interactive FRI, this translates to less prover and verifier computation, but the same/comparable proof sizes, as when terminating at degree $d = 1$. Moreover, this does not affect the soundness and/or bits of security of the overall protocol; therefore, we omit this optimization in our analysis.

4 Concrete Security Analysis of Non-interactive FRI

We now turn to analyzing the concrete security achieved by compiling the FRI protocol with the BCS transformation [8] in the random oracle model; for ease of presentation, we refer to this protocol as FS-FRI. Our analysis will determine both the bits of security given by various parameter settings of FS-FRI in practice, as well as give a more fine-grained trade-off analysis between the non-interactive soundness error of FS-FRI and the adversarial computational power (measured in number of random oracle queries). In particular, we provide a methodology for determining FRI parameters (i.e., field size, message length, code rate, etc.) such that soundness error $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{fs})} = 2^{-\nu}$ is achieved when an attacker can make Q queries to the random oracle (i.e., roughly the number of hashes an attacker can perform). This more fine-grained analysis can capture the following scenario that is not always captured by bits of security. Consider a system with 80-bits of security that versus a $Q = 2^{60}$ query adversary has non-interactive soundness error 2^{20}. Then, in this case, a group of 2^{20} parties with $Q = 2^{60}$ hashing power could expect to break the system (i.e., at least one of them is expected to be successful in an attack if they coordinate properly). With real-world hashing power continuing to rise, we believe that considering these types of attacks will be important in the future and is something that bits of security alone do not capture.

Let $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$ be the RBR soundness error of FRI (given in Thm. 1 or Conj. 1). Given $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$, the query parameter Q, and target non-interactive soundness error ν, we analyze $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{fs})}$ under the following constraints (induced by Lem. 1):

$$Q\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})} \leqslant 2^{-(\nu+1)} \qquad (3) \qquad\qquad 3(Q^2+1)/2^{\kappa} \leqslant 2^{-(\nu+1)}. \qquad (4)$$

So long as the above constraints are satisfied, then we have $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{fs})} \leqslant 2^{-\nu}$.

Remark 4. In general, one can choose $\alpha, \beta \in (0, 1)$ such that $Q\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})} \cdot \alpha + 3(Q^2+1)/2^\kappa \cdot \beta \leqslant 2^{-\nu}$. Then, intuitively, α determines how heavily the adversary tries to break the RBR soundness of FRI, and β determines how heavily the adversary tries to find a collision in the random oracle. Note that above, we are (a) putting equal weight on both cases by setting $\alpha = \beta = 1/2$; and (b) assuming that finding even a *single collision* in the random oracle completely breaks security of the entire protocol. However, we are unaware of any attack against FRI that can exploit a single random oracle collision, and leave it as an open question to determine if such attacks exist.

We break down our analysis into two parts. In the first part (Section 4.1), we examine FRI parameters $\mathbb{F}$, k, ρ, and δ used today in several projects (Tab. 1) and analyze what values of ν are achievable under certain fixed values of Q with respect to Thm. 1 and Conj. 1; we also give the bits of security (Def. 2) achieved by these projects under provable and conjectured security. In the second part (Section 4.3), we set targets for the soundness error ν and the FRI parameters $\mathbb{F}$, k, and ρ and analyze: (1) the upper bound on the number of RO queries Q given by Eq. (4); and (2) for various settings of Q less than or equal to the computed upper bound, whether or not the constraint in Eq. (3) is satisfiable or not. The satisfiability of Eq. (3) for fixed values of ν, $\mathbb{F}$, k, ρ, and Q depends on how $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$ is computed (i.e., either via provable security or conjectured security). For both parts of our analysis, we also analyze the size of the non-interactive proofs produced under these parameters, given in kilobytes or megabytes.

Notation. As a reminder, we use the following notation in the remainder of this section. $\mathbb{F}$ denotes a finite field; ρ, $d_0 = 2^k$, and $N = 2^n$ are the rate, degree bound (i.e., message length), and domain size of the RS code; Q is an upper bound on the number of random oracle queries made by an adversary; ℓ is the number of verifier queries; z is the number of grinding bits.

Sagemath Code. All calculations performed in the remainder of this section were done in Sagemath version 9.5 using Python 3.10.12, running on Pop!_OS 22.04 LTS with Kernel 6.8.0-76060800daily20240311-g on a laptop with an Intel i7-1165G7 (4 cores, 8 threads) and 16GB of RAM. The code used can be found at the following link: https://github.com/alexander-r-block/FRI-Parameter-Testing-Sagemath.

4.1 Non-interactive Security of FRI Parameters in Practice

We turn towards understanding the achievable non-interactive security of FRI under parameters being deployed *today* by various projects. We will examine the trade-offs between the adversarial power Q and the non-interactive soundness error ν (i.e., soundness error $2^{-\nu}$). Examining Eqs. (3) and (4), we see that ν has the following constraints:

$$\nu \leqslant \left\lfloor \log\left(\frac{1}{2 \cdot Q \cdot \varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}}\right)\right\rfloor \quad (5) \qquad \nu \leqslant \left\lfloor \log\left(\frac{2^{\kappa-1}}{3 \cdot (Q^2+1)}\right)\right\rfloor. \quad (6)$$

Therefore in what follows, we always set $\nu = \min\{\text{Eq. (5), Eq. (6)}\}$.

Recall that $\varepsilon_{(\mathrm{rbr})}^{(\mathrm{FRI})}$ has two different values depending on whether we use provable (Thm. 1) or conjectured (Conj. 1) security. By incorporating z-bits of grinding at the end of the FRI protocol, combining Eq. (5) with Eqs. (1) and (2) gives us the following constraints for ν:

$$\text{(provable) } \nu \leqslant \left\lfloor \log\left(\frac{1}{2Q}\cdot\min\left\{\frac{3\rho^{3/2}|\mathbb{F}|}{(m+1/2)^7\cdot|L_0|^2}, \frac{2^z}{(1-\delta)^\ell}\right\}\right)\right\rfloor; \tag{7}$$

$$\text{(conjectured) } \nu \leqslant \left\lfloor \log\left(\frac{1}{2Q}\cdot\min\left\{|\mathbb{F}|, \frac{2^z}{\rho^\ell}\right\}\right)\right\rfloor. \tag{8}$$

We use the constraints Eqs. (6) to (8) in our analysis (i.e., in the Sagemath code).

Bits of Security. For bits of security, rewriting Def. 2, to achieve λ-bits of security we analyze the bound $\varepsilon_{(\mathrm{fs})}^{(\mathrm{FRI})}/Q \leqslant 2^{-\lambda}$. Expanding the left-hand side, we have

$$\varepsilon_{(\mathrm{fs})}^{(\mathrm{FRI})}/Q = [Q\varepsilon_{(\mathrm{rbr})}^{(\mathrm{FRI})} + 3(Q^2+1)/2^\kappa]/Q \leqslant \varepsilon_{(\mathrm{rbr})}^{(\mathrm{FRI})} + 3(Q+1)/2^\kappa \leqslant 2^{-\lambda}.$$

Thus, to achieve λ-bits of security, it suffices to choose parameters satisfying

$$\varepsilon_{(\mathrm{rbr})}^{(\mathrm{FRI})} \leqslant 2^{-(\lambda+1)} \tag{9}$$

$$3(Q+1)/2^\kappa \leqslant 2^{-(\lambda+1)}. \tag{10}$$

Random Oracle Output Length and Queries. Eq. (10) depends on the output length of the random oracle. To simplify the analysis, we would like to choose κ such that Eq. (10) is not the bottleneck to soundness; i.e., for values of λ we consider, we want Eq. (9) $\geqslant$ Eq. (10). Analyzing this equation, we have

$$3(Q+1)/2^\kappa \leqslant 4Q/2^\kappa \leqslant 2^{-(\lambda+1)} \implies Q \leqslant 2^{\kappa-\lambda-3} \implies \log(Q) \leqslant \kappa-\lambda-3.$$

Choosing $\kappa = 256$ gives us the constraint $\log(Q) \leqslant 253-\lambda$. In our parameter analysis for the remainder of this section, the largest value of λ considered is $\lambda = 128$, giving $\log(Q) \leqslant 125$. Thus, for the remainder of this section and our analysis, we consider $\log(Q) \in \{20, 40, 60, 80, 100, 120\}$ and $\kappa = 256$ to ensure that Eq. (9) is the bottleneck for bits of security.

Proximity Parameter δ Selection. In all of our tests, we choose δ maximally in order to satisfy the corresponding theorems. In particular, for provable security, we choose $\delta = 1 - 7\sqrt{\rho}/6$ (given by setting $m = 3$ in Thm. 1); for conjectured security, we choose $\delta = 1-\rho$ (given in Conj. 1).[6]

Proof Sizes. We also include an upper bound on the proof sizes achieved by the FS-FRI proofs of every system we consider. For a detailed description of how these proof sizes are calculated, see the full version of our work.

[6] To the best of our knowledge, *all projects* using this conjecture set $\delta = 1-\rho$.

Remark 5. As noted in [32], there are numerous ways that practitioners attempt to reduce the size of FS-FRI proofs. These techniques are applicable to both provable and conjectured security. Moreover, the application of these techniques varies widely from project to project, so for simplicity we simply give the worst-case sizes of FS-FRI proofs under our parameters.

4.2 Parameter Analysis

We now turn to our parameter analysis. In Tab. 2, we give an overview of all the parameters we tested in our experiments. Parameters were either relatively easy to find in the source code of these projects, or they required extensive code tracing due to lack of documentation. We discuss how we found difficult to extract parameters in the full version of our work.

Given these parameters, we present graphs analyzing the trade-offs between a Q-query adversary attacking the system and the achievable soundness error $2^{-\nu}$. Included in each graph is: (1) trade-offs between provable and conjectured security, with and without grinding; (2) the proximity parameter δ achieved by the parameters; (3) the estimated proof sizes (with grinding); and (4) the achievable bits of security of the stated parameters (indicated as bold points that intersect the Q-axis).

Table 2. Parameters we assess in our analysis of various systems running FRI. $|\mathbb{F}|$ is the size of the finite field, ρ is the rate, $k = \log(d_0)$ for degree bound d_0, ℓ is the number of verifier queries, z is the number of grinding bits, and Sec. Target is the target security level. $p_{sta} = 2^{251} + 17 \cdot 2^{192} + 1$ is one of Starkware's 252-bit primes, $p_{gol} = 2^{64} - 2^{32} + 1$ is the Goldilocks prime, and $p_{bb} = 15 \cdot 2^{27} + 1$ is the BabyBear prime. All parameters are current as of May 6, 2024, 19:30 UTC.

	$\lvert\mathbb{F}\rvert$	ρ	k	ℓ	z	Sec. Target
Plonky2	p_{gol}^2	1/2	31	84	16	100
		1/8	29	28		
stone-prover	p_{sta}	1/16	30	18	24	96
SHARP	p_{sta}	1/32	26	13	31	96
dYdX	p_{sta}	1/16	31	12	32	80
Midden VM	p_{gol}^2	1/8	30	27	16	96
	p_{gol}^3	1/16			21	128
lambda-works (provable)	p_{sta}	1/4	40	80	20	80
				104		100
				140		128
lambda-works (conj.)	p_{sta}	1/4	40	31	20	80
				41		100
				55		128
RISC Zero	p_{gol}^2	1/4	24	50	0	100
	p_{bb}^4					
era-boojum	p_{gol}^2	1/4	19	40	20	100
		1/8	30	34	0	

Due to space constraints, we show all of our findings for each system in Tab. 2 in Fig. 1. We also briefly discuss where/how we found the parameters given in Tab. 2.

Plonky2 Paramters. Configuration 1 (Params. 1) is taken from here and Configuration 2 (Params. 2) is taken from here. Information on their finite fields can be found here. Note that the PolygonZero zkEVM uses Configuration 1 above (see here).

stone-prover Parameters. We use the example configuration given here. The field they use is one of Starkware's 252-bit prime fields; information here.

Sharp Verifier & dYdX Parameters. Distilling the SHARP Verifier and dYdX parameters required extensive code analysis, beginning by looking at posted transactions on etherscan.io then examining solidity code from Starkware. We distil both the SHARP parameters and the dYdX paramters in Tab. 2 the following transactions on Etherscan.

- **SHARP.** Etherscan Address. Transactions we use: 1, 2, 3, 4, 5.
- **dYdX.** L2 On-Chain Etherscan Address. Transactions we use: 1, 2, 3, 4, 5.

Miden-VM Parameters. We use the configurations found here. Information on what these parameters mean is found here.

lambdaworks Parameters. All parameters we use are defined in this file.

RISC Zero Parameters. We use the configuration found here. Information on the fields they use is found here.

era-boojum Parameters. We use the default configuration found here. The second configuration we use is found here.

Overall, our analysis finds significant gaps between the provable and conjectured security achieved by all systems, except for lambdaworks, under their current FRI parameters; these findings are higlighted in Tab. 1. These gaps (unsurprisingly) held regardless of grinding. Note that if one is comfortable with Conj. 1, then our results serve to show that current systems today are achieving their security targets (or are at most 1-bit away from their targets). However, we remark that Conj. 1 is not a *cryptographic* conjecture; i.e., it is not a conjecture about some problem (e.g., factoring) being difficult for *computationally bounded adversaries*. It is an *information theoretic* conjecture, positing that the best-known attack against an *information theoretically secure* protocol is optimal. That is, there do not exist any adversarial strategies, regardless of adversarial run-time, that can beat this attack. Our work, in this light, can be viewed as highlighting how heavily FRI-based systems are relying on this conjecture. Unsurprisingly, the reliance is very heavy (all in order to reduce the proof sizes of FRI). Thus, if others (apart from ourselves) are hesitant to make aggressive information theoretic conjectures about the security of FRI, our work highlights how lacking most FRI-based systems are in provable security, serving as a warning to those uncomfortable with these aggressive conjectures.

4.3 Parameter Suggestions for FS-FRI

We now switch gears and turn towards giving parameter suggestions for achieving various levels of security. That is, we set various targets for v, fix the parameters $\mathbb{F}$, k, and ρ of FRI, and analyze both the upper bound on Q given by Eq. (4) and whether or not Eq. (3) is satisfiable for certain fixing of Q within the upper bound. We perform this analysis with respect to provable security (Thm. 1) and conjectured security (Conj. 1).

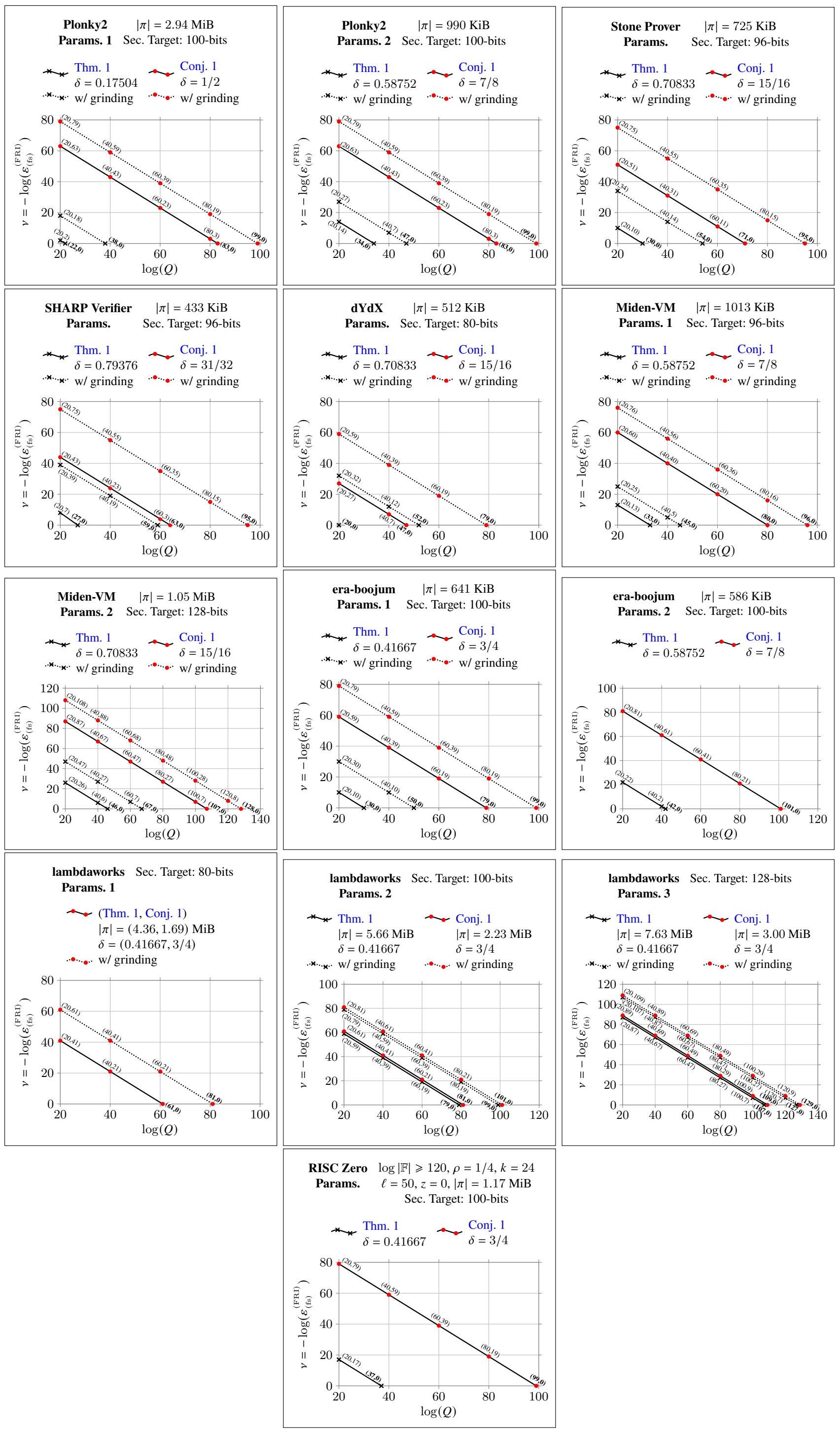

Fig. 1. Graphs plotting the trade-off between Q and ν for the FRI parameters taken from Tab. 2. $|\pi|$ indicates the proof size and δ the proximity parameter. Bold points on the graph indicate the achieved bits of security by these parameters.

Now notice that the constraint given by Eq. (4) is independent of the round-by-round soundness. Thus, under this constraint, we can upper bound the number of queries Q as

$$3(Q^2+1)/2^\kappa \leqslant 2^{-(\nu+1)} \quad \Longrightarrow \quad Q \leqslant \left\lfloor \left(2^{\kappa-\nu-1}/3-1\right)^{1/2} \right\rfloor . \tag{11}$$

Thus Eq. (11) is an upper bound on the number of RO queries any adversary is allowed to make when attacking the non-interactive proof system. Note we take the floor as the upper bound since the number of queries is an integer. Given this upper bound on Q, we can turn to analyzing the other constraint given by Eq. (3), which we rewrite as:

$$\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})} \leqslant 1/(Q \cdot 2^{\nu+1}). \tag{12}$$

Now Eq. (12) give us an upper bound for $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$.

Given these constraints, we can now proceed with setting various values of ν, $\mathbb{F}$, k, and ρ, and analyzing Q and $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$. For the remainder of this section, we consider the following parameters:

- Hashes of length $\kappa = 256$;
- Finite fields $\mathbb{F}$ of bit length $\log(|\mathbb{F}|) \in \{128, 192, 256\}$;
- Message lengths $d_0 = 2^k$ such that $k \in \{10, 15, 20, 25\}$;
- Rates $\rho \in \{1/2, 1/4, 1/8, 1/16\}$; and
- $\nu = 20$ (i.e., soundness error at most $2^{-\nu}$ versus a Q-query adversary).

Due to space constraints, we only analyze $\nu = 20$ and, as we will see, settings for Q that roughly translate to 80-bits and 100-bits of security.

Remark 6. In what follows, we are only discussing the feasibility of parameters with respect to the parameters outlined above. It is entirely possible to find other feasible parameters for different settings of k, $\log|\mathbb{F}|$, and ρ. We do not exhaustively consider all such feasible parameters and only consider our setting of parameters as a guide to get a quick grasp on how certain settings of parameters behave.

Moreover, one can additionally modify Eq. (14) to include z-bits of grinding during the final round of FRI. In this case, the left-hand side of this equation becomes $(1-\delta)^\ell/2^z$. For simplicity, we do not perform our analysis with grinding but note that one could suitably reduce ℓ by roughly z-bits if z-bits of grinding were included in the analysis.

Provable Security. We first analyze the above parameters in the context of the provable RBR soundness of FRI given by Thm. 1. Before we begin, we first obtain constraints on $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$ under provable security guarantees. Recall that $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})}$ is given in Eq. (1). Combining this equation with Eq. (12), obtain the constraints:

$$\frac{(m+1/2)^7|L_0|^2}{3\rho^{3/2}|\mathbb{F}|} \leqslant \frac{1}{Q2^{\nu+1}} \quad (13) \qquad\qquad (1-\delta)^\ell \leqslant \frac{1}{Q2^{\nu+1}}. \quad (14)$$

Recall that in FRI, we have $|L_0| = 2^k/\rho$, so under our fixing of k and ρ, the parameter $|L_0|$ is fixed as well. For provable security, we always consider maximal δ, i.e., we set $\delta = 1 - \sqrt{\rho}(1 + 1/(2m))$ for $m = 3$.

Table 3. Numerical calculations for non-interactive soundness error $\nu = 20$ via Thm. 1, where $\log |\mathbb{F}|$ represents the bit-length of the field size, $d_0 = 2^k$, $|L_0| = d_0/\rho$, ℓ is the number of repetitions during the Query Phase, Q is the allowed number of random oracle queries made by a malicious prover, and $|\pi|$ is the proof size. The table for $Q = 2^{60}$ roughly represents 80-bits of security, and the table for $Q = 2^{80}$ roughly represents 100-bits of security.

$\nu = 20$, $Q = 2^{60}$		Thm. 1			
		$\rho = 1/2$	$\rho = 1/4$	$\rho = 1/8$	$\rho = 1/16$
192	10	(0.293, 163, 840 KiB)	(0.500, 82, 474 KiB)	(0.646, 55, 352 KiB)	(0.750, 41, 288 KiB)
	15	(0.293, 163, 1.60 MiB)	(0.500, 82, 903 KiB)	(0.646, 55, 657 KiB)	(0.750, 41, 529 KiB)
	20	(0.293, 163, 2.64 MiB)	(0.500, 82, 1.43 MiB)	(0.646, 55, 1.02 MiB)	(0.750, 41, 833 KiB)
	25	(0.292, 163, 3.92 MiB)	(0.499, 82, 2.10 MiB)	(0.645, 55, 1.49 MiB)	(0.749, 41, 1.17 MiB)
256	10	(0.293, 163, 866 KiB)	(0.500, 82, 487 KiB)	(0.646, 55, 361 KiB)	(0.750, 41, 295 KiB)
	15	(0.293, 163, 1.64 MiB)	(0.500, 82, 923 KiB)	(0.646, 55, 670 KiB)	(0.750, 41, 538 KiB)
	20	(0.293, 163, 2.69 MiB)	(0.500, 82, 1.45 MiB)	(0.646, 55, 1.04 MiB)	(0.750, 41, 846 KiB)
	25	(0.293, 163, 3.98 MiB)	(0.500, 82, 2.13 MiB)	(0.646, 55, 1.51 MiB)	(0.750, 41, 1.19 MiB)
$\nu = 20$, $Q = 2^{80}$					
		$\rho = 1/2$	$\rho = 1/4$	$\rho = 1/8$	$\rho = 1/16$
192	10	(0.293, 203, 1.02 MiB)	(0.500, 102, 590 KiB)	(0.646, 68, 435 KiB)	(0.750, 51, 358 KiB)
	15	(0.292, 203, 2.00 MiB)	(0.499, 102, 1.10 MiB)	(0.645, 68, 813 KiB)	(0.749, 51, 657 KiB)
	20	(0.290, 205, 3.32 MiB)	(0.497, 102, 1.77 MiB)	(0.644, 68, 1.27 MiB)	(0.747, 51, 1.01 MiB)
	25	(0.285, 209, 5.02 MiB)	(0.492, 104, 2.66 MiB)	(0.639, 69, 1.87 MiB)	(0.742, 52, 1.49 MiB)
256	10	(0.293, 203, 1.05 MiB)	(0.500, 102, 605 KiB)	(0.646, 68, 446 KiB)	(0.750, 51, 366 KiB)
	15	(0.293, 203, 2.04 MiB)	(0.500, 102, 1.12 MiB)	(0.646, 68, 829 KiB)	(0.750, 51, 669 KiB)
	20	(0.293, 203, 3.35 MiB)	(0.500, 102, 1.81 MiB)	(0.646, 68, 1.29 MiB)	(0.750, 51, 1.03 MiB)
	25	(0.293, 203, 4.96 MiB)	(0.500, 102, 2.65 MiB)	(0.646, 68, 1.87 MiB)	(0.750, 51, 1.48 MiB)
$\log \lvert\mathbb{F}\rvert$	k	$(\delta, \ell, \lvert\pi\rvert)$			

Notice that for any fixed values of Q and ν, the constraint of Eq. (14) is always satisfiable by taking $\ell = \lceil \log(1/(Q \cdot 2^{\nu+1}))/\log(1-\delta) \rceil$. Notice also that for fixed values of Q and ν, the constraint of Eq. (13) gives an upper bound on the Johnson proximity parameter m. Given that $m \geqslant 3$ is required and Eq. (13) gives an upper bound on m, it is possible that under certain parameters $\mathbb{F}$, k, ρ, Q, and ν we have $m < 3$. In this case, we conclude that these parameters are *infeasible*.

Given the above constraints, we can now turn to analyzing what provable security guarantees are achievable under the setting of parameters we are considering. As stated before, we fix $\nu = 20$, analyze the query upper bound via Eq. (11), then analyze whether or not Eq. (13) is satisfiable under certain fixings of Q. For $\nu = 20$ and the constraint of Eq. (11) along with $\kappa = 256$ tells us that the upper bound of Q lies in the interval $[2^{116}, 2^{117}]$ (represented as powers of two for ease of presentation). Thus, in our parameter analysis below, we consider two cases: $Q = 2^{60}$ and $Q = 2^{80}$ since for $\nu = 20$, $Q = 2^{60}$ roughly corresponds to 80-bits of security and $Q = 2^{80}$ roughly corresponds to 100-bits of security. We present our parameter analysis in Tab. 3.

Examining the given table, we can conclude that for target soundness error $\nu = 20$ and query range $Q \in \{2^{60}, 2^{80}\}$, all the parameters we consider are feasible for both 192-bit and 256-bit finite fields, whereas none of the parameters we consider were feasible in the case of 128-bit finite fields. Our analysis highlights that larger fields, smaller messages, and larger rates (i.e., ρ is larger) are necessary for the feasibility of parameters under $\nu = 20$. None of the parameters we considered were feasible for 128-bit finite fields, though there may be finite fields between 128 and 192-bits in size where parameters are feasible (similarly for between 192 and 256-bit finite fields).

Conjectured Security. We conclude our analysis of the parameters we consider in the context of the conjectured RBR soundness given by Conj. 1. To begin, under this conjecture, the RBR soundness error of FRI is given by $\varepsilon^{(\mathrm{FRI})}_{(\mathrm{rbr})} = \max\{1/|\mathbb{F}|, \rho^{\ell}\}$, where $\delta = 1 - \rho$ is the FRI proximity parameter. Combining this expression with Eq. (3), we have the constraints

$$Q \cdot 2^{\nu+1} \leqslant |\mathbb{F}| \qquad (15) \qquad\qquad \rho^{\ell} \leqslant {}^{1}/_{Q2^{\nu+1}}. \qquad (16)$$

Again here we set $|L_0| = 2^k/\rho$, so fixing k and ρ fixes $|L_0|$ as well.

Table 4. Numerical calculations for non-interactive soundness error $\nu = 20$ via Conj. 1, where $d_0 = 2^k$, $|L_0| = d_0/\rho$, ℓ is the number of repetitions during the Query Phase, Q is the allowed number of random oracle queries made by a malicious prover, and $|\pi|$ is the proof size. The table for $Q = 2^{60}$ roughly represents 80-bits of security, and the table for $Q = 2^{80}$ roughly represents 100-bits of security.

$\nu = 20$, $Q = 2^{60}$		Conj. 1			
		$\rho = 1/2$	$\rho = 1/4$	$\rho = 1/8$	$\rho = 1/16$
128	10	(0.500, 81, 405 KiB)	(0.750, 41, 230 KiB)	(0.875, 27, 169 KiB)	(0.938, 21, 144 KiB)
	15	(0.500, 81, 797 KiB)	(0.750, 41, 442 KiB)	(0.875, 27, 316 KiB)	(0.938, 21, 266 KiB)
	20	(0.500, 81, 1.29 MiB)	(0.750, 41, 718 KiB)	(0.875, 27, 506 KiB)	(0.938, 21, 420 KiB)
	25	(0.500, 81, 1.92 MiB)	(0.750, 41, 1.03 MiB)	(0.875, 27, 739 KiB)	(0.938, 21, 607 KiB)
192	10	(0.500, 81, 417 KiB)	(0.750, 41, 237 KiB)	(0.875, 27, 173 KiB)	(0.938, 21, 147 KiB)
	15	(0.500, 81, 816 KiB)	(0.750, 41, 452 KiB)	(0.875, 27, 323 KiB)	(0.938, 21, 271 KiB)
	20	(0.500, 81, 1.31 MiB)	(0.750, 41, 730 KiB)	(0.875, 27, 515 KiB)	(0.938, 21, 427 KiB)
	25	(0.500, 81, 1.95 MiB)	(0.750, 41, 1.05 MiB)	(0.875, 27, 749 KiB)	(0.938, 21, 616 KiB)
256	10	(0.500, 81, 430 KiB)	(0.750, 41, 243 KiB)	(0.875, 27, 177 KiB)	(0.938, 21, 151 KiB)
	15	(0.500, 81, 835 KiB)	(0.750, 41, 461 KiB)	(0.875, 27, 329 KiB)	(0.938, 21, 276 KiB)
	20	(0.500, 81, 1.34 MiB)	(0.750, 41, 743 KiB)	(0.875, 27, 523 KiB)	(0.938, 21, 433 KiB)
	25	(0.500, 81, 1.98 MiB)	(0.750, 41, 1.06 MiB)	(0.875, 27, 760 KiB)	(0.938, 21, 624 KiB)
$\nu = 20$, $Q = 2^{80}$		$\rho = 1/2$	$\rho = 1/4$	$\rho = 1/8$	$\rho = 1/16$
128	10	(0.500, 101, 505 KiB)	(0.750, 51, 287 KiB)	(0.875, 34, 212 KiB)	(0.938, 26, 179 KiB)
	15	(0.500, 101, 994 KiB)	(0.750, 51, 550 KiB)	(0.875, 34, 398 KiB)	(0.938, 26, 329 KiB)
	20	(0.500, 101, 1.60 MiB)	(0.750, 51, 893 KiB)	(0.875, 34, 638 KiB)	(0.938, 26, 520 KiB)
	25	(0.500, 101, 2.39 MiB)	(0.750, 51, 1.28 MiB)	(0.875, 34, 930 KiB)	(0.938, 26, 752 KiB)
192	10	(0.500, 101, 521 KiB)	(0.750, 51, 295 KiB)	(0.875, 34, 218 KiB)	(0.938, 26, 183 KiB)
	15	(0.500, 101, 1018 KiB)	(0.750, 51, 562 KiB)	(0.875, 34, 406 KiB)	(0.938, 26, 335 KiB)
	20	(0.500, 101, 1.63 MiB)	(0.750, 51, 909 KiB)	(0.875, 34, 648 KiB)	(0.938, 26, 528 KiB)
	25	(0.500, 101, 2.43 MiB)	(0.750, 51, 1.30 MiB)	(0.875, 34, 943 KiB)	(0.938, 26, 762 KiB)
256	10	(0.500, 101, 536 KiB)	(0.750, 51, 303 KiB)	(0.875, 34, 223 KiB)	(0.938, 26, 187 KiB)
	15	(0.500, 101, 1.02 MiB)	(0.750, 51, 574 KiB)	(0.875, 34, 414 KiB)	(0.938, 26, 341 KiB)
	20	(0.500, 101, 1.67 MiB)	(0.750, 51, 925 KiB)	(0.875, 34, 659 KiB)	(0.938, 26, 536 KiB)
	25	(0.500, 101, 2.47 MiB)	(0.750, 51, 1.32 MiB)	(0.875, 34, 957 KiB)	(0.938, 26, 772 KiB)
$\log \lvert\mathbb{F}\rvert$	k	$(\delta, \ell, \lvert\pi\rvert)$			

As in the case of prior sections, for any fixed Q and ν, the constraint of Eq. (16) is always satisfied by taking $\ell = \lceil \log(1/(Q \cdot 2^{\nu+1}))/\log(\rho) \rceil$. Given these constraints, we can now turn to analyzing what conjectured security guarantees are achievable under the setting of parameters we are considering. As stated before, we fix $\nu = 20$, analyze the query upper bound via Eq. (11), then analyze whether or not Eq. (13) is satisfiable under certain fixings of Q. For $\nu = 20$ and the constraint of Eq. (11) along with $\kappa = 256$ tells us that the upper bound of Q lies in the interval $[2^{116}, 2^{117}]$ (represented as powers of two for ease of presentation). As before, we choose to consider $Q = 2^{60}$ and $Q = 2^{80}$ in our calculations, with for $\nu = 20$ roughly corresponds to 80-bits and 100-bits of security, respectively. Note that we additionally have the constraint $Q \leqslant |\mathbb{F}|/2^{\nu+1}$, which may be smaller than 2^{60} or 2^{80}. In this case, we consider the parameters as being infeasible and indicate them appropriately. We present our analysis in Tab. 4. The constraint of Eq. (15) gives us a lower bound on $|\mathbb{F}|$ of $Q \cdot 2^{\nu+1}$, so feasibility only depends on having a sufficiently large field. Notice also that Eq. (15) gives an additional upper bound on Q as well, along with the constraint of Eq. (11).

Examining the table, all parameter settings are feasible for all finite field sizes, for both $Q = 2^{60}$ and $Q = 2^{80}$ query bounds. Under the conjectured security of FRI given in Conj. 1, our analysis highlights that larger fields, smaller messages, and larger rates lead to the feasibility of parameters for $\nu = 20$. In contrast with both provable security given under Thm. 1, the more aggressive conjecture of Conj. 1 allows us to have feasible parameters for 128-bit finite fields for $\nu = 20$.

Acknowledgements

The authors would like to thank Matthew Green for all the helpful presentation suggestions of this work. Alexander R. Block was supported by DARPA under Contract Nos. HR00112020022 and HR00112020025. Pratyush Ranjan Tiwari was supported by the Ethereum Foundation's grant FY23-1087, and partly supported by a Security & Privacy research gift from Google, a research gift from Cisco. The views, opinions, findings, conclusions, and/or recommendations expressed in this material are those of the authors and should not be interpreted as reflecting the position or policy of DARPA or the United States Government, and no official endorsement should be inferred.

References

1. Abdalla, M., An, J.H., Bellare, M., Namprempre, C.: From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 418–433. Springer, Heidelberg (Apr / May 2002).
2. Attema, T., Fehr, S., Klooß, M.: Fiat-shamir transformation of multi-round interactive proofs. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I. LNCS, vol. 13747, pp. 113–142. Springer, Heidelberg (Nov 2022).
3. Barak, B.: How to go beyond the black-box simulation barrier. In: 42nd FOCS. pp. 106–115. IEEE Computer Society Press (Oct 2001).
4. Ben-Sasson, E., Bentov, I., Gabizon, A., Riabzev, M.: Improved concrete efficiency and security analysis of reed-solomon pcpps. Electron. Colloquium Comput. Complex. **TR16-073** (2016)
5. Ben-Sasson, E., Bentov, I., Gabizon, A., Riabzev, M.: A security analysis of probabilistically checkable proofs. Electron. Colloquium Comput. Complex. **TR16-149** (2016)
6. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Fast reed-solomon interactive oracle proofs of proximity. In: Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D. (eds.) ICALP 2018. LIPIcs, vol. 107, pp. 14:1–14:17. Schloss Dagstuhl (Jul 2018).

7. Ben-Sasson, E., Carmon, D., Ishai, Y., Kopparty, S., Saraf, S.: Proximity gaps for reed-solomon codes. In: 61st FOCS. pp. 900–909. IEEE Computer Society Press (Nov 2020).
8. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 31–60. Springer, Heidelberg (Oct / Nov 2016).
9. Ben-Sasson, E., Goldberg, L., Kopparty, S., Saraf, S.: DEEP-FRI: Sampling outside the box improves soundness. In: Vidick, T. (ed.) ITCS 2020. vol. 151, pp. 5:1–5:32. LIPIcs (Jan 2020).
10. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to Helios. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 626–643. Springer, Heidelberg (Dec 2012).
11. Bitansky, N., Dachman-Soled, D., Garg, S., Jain, A., Kalai, Y.T., López-Alt, A., Wichs, D.: Why "Fiat-Shamir for proofs" lacks a proof. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 182–201. Springer, Heidelberg (Mar 2013).
12. Block, A.R., Garreta, A., Katz, J., Thaler, J., Tiwari, P.R., Zajac, M.: Fiat-shamir security of FRI and related SNARKs. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023, Part II. LNCS, vol. 14439, pp. 3–40. Springer, Heidelberg (Dec 2023).
13. Block, A.R., Garreta, A., Tiwari, P.R., Zajac, M.: On soundness notions for interactive oracle proofs. IACR Cryptol. ePrint Arch. p. 1256 (2023)
14. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D., Wichs, D.: Fiat-Shamir: from practice to theory. In: Charikar, M., Cohen, E. (eds.) 51st ACM STOC. pp. 1082–1090. ACM Press (Jun 2019).
15. Chiesa, A., Manohar, P., Spooner, N.: Succinct arguments in the quantum random oracle model. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019, Part II. LNCS, vol. 11892, pp. 1–29. Springer, Heidelberg (Dec 2019).
16. Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 769–793. Springer, Heidelberg (May 2020).
17. Dao, Q., Miller, J., Wright, O., Grubbs, P.: Weak fiat-shamir attacks on modern proof systems. In: 2023 IEEE Symposium on Security and Privacy. pp. 199–216. IEEE Computer Society Press (May 2023).
18. Davis, H., Diemert, D., Günther, F., Jager, T.: On the concrete security of TLS 1.3 PSK mode. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 876–906. Springer, Heidelberg (May / Jun 2022).
19. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987).
20. Goldwasser, S., Kalai, Y.T.: On the (in)security of the Fiat-Shamir paradigm. In: 44th FOCS. pp. 102–115. IEEE Computer Society Press (Oct 2003).
21. Haböck, U.: A summary on the FRI low degree test. Cryptology ePrint Archive, Report 2022/1216 (2022), https://eprint.iacr.org/2022/1216
22. Hopwood, D., Bowe, S., Hornby, T., Wilcox, N.: Zcash protocol specification. https://zips.z.cash/protocol/protocol.pdf, version 2022.3.8 [NU5]
23. Jaeger, J., Tessaro, S.: Expected-time cryptography: Generic techniques and applications to concrete soundness. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 414–443. Springer, Heidelberg (Nov 2020).
24. L2BEAT: L2beat total value locked. https://l2beat.com/scaling/tvl
25. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 387–398. Springer, Heidelberg (May 1996).
26. Polygon Labs: Fri verification procedures. https://wiki.polygon.technology/docs/miden/user_docs/stdlib/crypto/fri/

27. Reed, I.S., Solomon, G.: Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics **8**(2), 300–304 (1960). , https://doi.org/10.1137/0108018
28. StarkWare: ethSTARK documentation. Cryptology ePrint Archive, Report 2021/582 (2021), https://eprint.iacr.org/2021/582
29. StarkWare: ethstark documentation - version 1.2. Cryptology ePrint Archive, Paper 2021/582 (2021), https://eprint.iacr.org/2021/582, https://eprint.iacr.org/2021/582, updated in July, 2023
30. StarkWare Industries: Starkex documentation: Customers and their deployment contract addresses. https://docs.starkware.co/starkex/deployments-addresses.html
31. Team, R.Z.: Risc zero's proof system for a zkvm (2023), https://github.com/risc0/risc0
32. Thaler, J.: Snark security and performance. https://a16zcrypto.com (2019), https://a16zcrypto.com/snark-security-and-performance/
33. Wikström, D.: Special soundness in the random oracle model. Cryptology ePrint Archive, Report 2021/1265 (2021), https://eprint.iacr.org/2021/1265

Appendix

A Additional Preliminaries

In this section, we add some additional preliminaries for completeness. A *relation* $\mathbf{R}$ is a subset of pairs $(\mathbb{x}, \mathbb{w}) \in \{0,1\}^* \times \{0,1\}^*$. The strings $\mathbb{x}$ are called *inputs* (these are often called also *statements* or *instances*), and the strings $\mathbb{w}$ are called *witnesses*. To each relation $\mathbf{R}$ there corresponds a language $\mathcal{L}_{\mathbf{R}} \subseteq \{0,1\}^*$ consisting of all statements $\mathbb{x}$ such that $(\mathbb{x}, \mathbb{w}) \in \mathbf{R}$ for some $\mathbb{w}$; i.e., $\mathcal{L}_{\mathbf{R}} := \{\mathbb{x} \colon \exists \mathbb{w} \text{ s.t. } (\mathbb{x}, \mathbb{w}) \in \mathbf{R}\}$. When $(\mathbb{x}, \mathbb{w}) \in \mathbf{R}$, we say that $\mathbb{w}$ is a *valid witness for* $\mathbb{x}$. We assume our relations to be in the class $\mathbf{NP}$.

A.1 Interactive Oracle Proofs

Definition 3 (Interactive Proofs (IP)). *A μ-round* interactive proof *for a relation $\mathbf{R}$ is a pair of interactive algorithms $\Pi = (\mathsf{P}, \mathsf{V})$ such that:*

- *For $\mathbb{x} \in L_{\mathbf{R}}$ and $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w}) \in \mathbf{R}$, before the start of the protocol, P receives both $(\mathbb{x}, \mathbb{w})$ as input and V receives $\mathbb{x}$ as input.*
- *$\mathsf{P}(\mathbb{x}, \mathbb{w})$ and $\mathsf{V}(\mathbb{x})$ exchange $2\mu(|\mathbb{x}|) + 1$ messages, where P sends the first and last message, and during any round of interaction P sends message m_i to V. After P sends $m_{\mu(|\mathbb{x}|)+1}$, V either accepts (outputs 1) or rejects (outputs 0).*

We require the following properties to hold:

- ***Completeness:** for all $(\mathbb{x}, \mathbb{w}) \in \mathbf{R}$, we have*

$$\Pr\left[\langle \mathsf{P}(\mathbb{w}), \mathsf{V}\rangle(\mathbb{x}) = 1\right] = 1,$$

  where $\langle \mathsf{P}(\mathbb{w}), \mathsf{V}\rangle(\mathbb{x})$ denotes the output of P and V interacting on common input $\mathbb{x}$ where P is additionally given $\mathbb{w}$ as input, and the above probability holds over the random coins of V.
- ***ϵ-Soundness:** for any $\mathbb{x} \notin \mathcal{L}_{\mathbf{R}}$ and any unbounded interactive algorithm P^*, we have*

$$\Pr\left[\langle \mathsf{P}^*, \mathsf{V}\rangle(\mathbb{x}) = 1\right] \leqslant \epsilon,$$

  where the probability is taken over the random coins of V.

We say that Π is public-coin *if all messages sent by V are independent uniform random strings of some bounded length and the output of V does not depend on any secret state.*

Definition 4 (Interactive Oracle Proof). *An Interactive Oracle Proof (IOP) for a relation $\mathbf{R}$ is a μ-round IP (P, V) for $\mathbf{R}$ such that for all $\mathbb{x}$, at the start of each round of interaction $i \in [\mu(|\mathbb{x}|)]$, P sends m_i and V receives oracle access to m_i. Crucially, at the end of the interactive phase, V does not necessarily need to read the whole m_i in order to decide whether to accept or reject.*

A.2 Round-by-round Soundness

Round-by-round soundness [14] is a stronger notion of interactive protocol soundness that is sufficient for arguing Fiat-Shamir security of interactive protocols [14,15,16]. Informally, RBR soundness captures the notion that a cheating prover has to get "lucky in a single round" in an IOP: if the protocol is initiated in a state with a false statement and should be rejected by the verifier (i.e., a "doomed state"), then no matter how cleverly the prover responds in subsequent rounds, the protocol will "forever remain doomed" (except with negligible probability). More formally, invoking the following (generalized) RBR soundness definition from [13].

Definition 5 (Round-by-round Soundness). *An IOP* $\Pi = (\mathsf{P}, \mathsf{V})$ *for a relation* $\mathbf{R}$ *has* round-by-round soundness *with error* ε *if there exists a (not necessarily efficiently computable) "doomed set"* $\mathcal{D}$ *of partial transcripts such that the following hold:*

1. *If* $\mathbb{x} \notin \mathcal{L}_{\mathbf{R}}$*, then* $(\mathbb{x}, \emptyset) \in \mathcal{D}$*, where* $\emptyset$ *denotes the empty transcript.*
2. *For any* $\mathbb{x}$ *and any* μ*-round partial transcript* $(\mathbb{x}, \tau)$ *and any last prover message* $m \in \mathcal{M}_{\mu+1}$*, if* $(\mathbb{x}, \tau) \in \mathcal{D}$ *then* $V(\mathbb{x}, \tau, m) = 0$*.*
3. *If* $(\mathbb{x}, \tau)$ *is a* $(i-1)$*-round partial transcript for some* $i \in [\mu]$ *and* $(\mathbb{x}, \tau) \in \mathcal{D}$*, then for all* $m \in \mathcal{M}_i$ *we have* $\Pr_{c \xleftarrow{\$} C_i}[(\tau, m, c) \notin \mathcal{D}] \leqslant \varepsilon$*, where* $\mathcal{M}_i$ *and* C_i *are the prover and verifier message spaces in round i, respectively.*

A.3 The BCS Transformation

The BCS transformation [8] turns any IOP $\Pi = (\mathsf{P}, \mathsf{V})$ into a non-interactive argument. Informally speaking, this is achieved by giving access to P and V to a random oracle ρ. Then, every time that P would send to V oracle access to a map $m : S \to \{0,1\}^*$, instead it sends the Merkle tree root of the vector $(m(s))_{s \in S}$, using ρ as the "hash function". Then, when V wants to query m at some point $s \in S$, the prover sends the value $m(s)$ to V, along with a Merkle tree path certifying that $m(s)$ is a correct opening. Finally, to protocol is made non-interactive via the Fiat-Shamir transformation using the random oracle ρ. We refer to [8] for a formal description of this transformation.

It turns out that applying the BCS construction to a round-by-round sound IOP Π with round-by-round knowledge soundness yields a SNARK; i.e., a succinct non-interactive argument of knowledge [15]. Moreover, if Π is zero-knowledge, then so is the resulting SNARK. Specifically, BCS compiles round-by-round sound IOPs into *non-interactive random oracle proofs*.

Below we state a "meta" theorem from [12] which captures all the relevant properties of the BCS transformation used in our analysis.

Theorem 2 ([8,15,16]). *There exists a polynomial-time transformation* BCS *such that for every relation* $\mathbf{R}$*, random oracle* $\mathcal{H} : \{0,1\}^* \to \{0,1\}^\kappa$*, and random oracle query bound* $Q \in \mathbb{N}$*, if* (P, V) *is a public-coin IOP for* $\mathcal{R}$ *with*

– *proofs of length* $\ell(\mathbb{x})$*;*
– *verifier query complexity* $q(\mathbb{x})$*; and*
– *round-by-round soundness error* $\varepsilon_{(rbr)}(\mathbb{x})$*,*

then $(\mathbb{P}, \mathbb{V}) := \mathsf{BCS}^{\mathcal{H}}(\mathsf{P}, \mathsf{V})$ *is a non-interactive random oracle proof system for* $\mathbf{R}$ *with adaptive soundness error* $\varepsilon_{(fs)}(\mathbb{x}, Q, \kappa) = Q\varepsilon_{(rbr)}(\mathbb{x}) + 3(Q^2+1)/2^\kappa$ *against* Q*-query adversaries.*

B Formal Description of the FRI Protocol

Algorithm 1: FRI-IOPP

```
Input: Finite field $\mathbb{F}$, smooth multiplicative subgroup $L_0 \subset \mathbb{F}^\times$ of size $2^n$, degree bound
       $d_0 = 2^k$, and $\ell \in \mathbb{N}$.
$P$ has function $G_0 \colon L_0 \to \mathbb{F}$ and $V$ has oracle $(G_0(z))_{z \in L_0}$.
Output: The verifier $V$ outputs accept or reject.
1  foreach $i \in [k]$ do // Fold Phase
2      $V$ sends $x_{i-1} \xleftarrow{\$} \mathbb{F}$ to $P$.
3      $P$ and $V$ set $d_i := d_{i-1}/2$ and $L_i := \{z^2 : z \in L_{i-1}\}$.
4      $P$ computes unique bi-variate polynomial $Q_{i-1}(X,Y)$ such that
         1. $\deg_X(Q_{i-1}) = 1$;
         2. $\deg_Y(Q_{i-1}) < d_i$; and
         3. $G_{i-1}(r) = Q_{i-1}(r, r^2)$ for all $r \in L_{i-1}$.
5      $P$ defines $G_i(Y) := Q_{i-1}(x_{i-1}, Y)$.
6      if $i = k$ then $P$ sends $G_k = C \in \mathbb{F}$ to $V$.
7      else $P$ sends oracle $(G_i(z))_{z \in L_i}$ to $V$.
8  forall $j \in [\ell]$ do // Query Phase; processed in parallel
9      $V$ samples $s_{0,j} \xleftarrow{\$} L_0$.
10     foreach $i \in [k]$ do
11         $V$ computes $s_{i,j} = (s_{i-1,j})^2$ and $s'_{i-1,j} \neq s_{i-1,j}$ such that $(s'_{i-1,j})^2 = s_{i,j}$.
12         $V$ queries and obtains $q_{i-1,j} = G_{i-1}(s_{i-1,j})$ and $q'_{i-1,j} = G_{i-1}(s'_{i-1,j})$.
13         $V$ computes linear polynomial $\widetilde{Q}_{i-1,j}(X)$ via Lagrange interpolation on the set
             $\{(s_{i-1,j}, q_{i-1,j}), (s'_{i-1,j}, q'_{i-1,j})\}$.
15         if $G_i(s_{i,j}) \neq \widetilde{Q}_{i-1,j}(x_{i-1})$ then $V$ outputs reject.
16 $V$ outputs accept.
```

C Proof Sizes of FS-FRI

Here, we give a detailed overview on how we calculate the FS-FRI proof sizes. As a reminder, FS-FRI is the non-interactive protocol obtained by compiling the FRI IOPP with the BCS transformation. We remark that this version of FS-FRI may or may not be the non-interactive version of FRI used in practice, as there are many optimizations that can be made to reduce concrete proof sizes. We do not take into consideration any of these optimizations; for example, Merkle capping,which shortens the length of Merkle authentication paths at the cost of increasing commitment size (i.e., sending multiple Merkle roots per round), or terminating FRI early and sending the coefficients of a constant-degree polynomial rather than a single field element, which reduces the number of Merkle authentication paths needed to be sent by the prover at the cost of verifier complexity. Thus our proof size analyses are overestimates; we believe that the proof sizes here can be reduced by roughly 33% via optimizations used in practice [32].

We briefly discuss how Alg. 1 is transformed into FS-FRI via the BCS transformation [8]. The BCS transformation incorporates a random oracle $\mathcal{H}$. During every round of the IOPP where the prover sends an oracle G_i, this oracle is instead replaced with a Merkle commitment M_i over the vector $G_i(L_i)$, where L_i is the domain used during round i. Next, instead of querying oracle G_i at some point z, the verifier instead sends z to the prover, and the prover responds with the value $G_i(z)$ along with a Merkle authentication path π_i that is consistent with Merkle commitment (or root) M_i. Finally, this above interaction is compressed via the classic Fiat-Shamir transformation; i.e., every message sent by the verifier is instead replaced with the hash (i.e., random oracle output) of all prior prover messages and the input instance ($\mathbb{x}$) (to the underlying relation).

Let κ be the output length of the random oracle $\mathcal{H}$, let $\mathbb{F}$ be a finite field, let L_0 be the evaluation domain of size 2^n, let $d_0 = 2^k$ be the degree bound, and let ℓ be the number of times the verifier repeats the Query Phase of FRI. Now first consider an intermediate version of FS-FRI, where instead of using the random oracle to compute verifier challenges and queries, there is still interaction with the verifier for its challenges. In particular, the prover sends Merkle roots of its various oracles to the verifier and responds to verifier queries to these oracles with Merkle authentication paths. We refer to this protocol as FRI-ARG. In this case, the transcript of the interaction consists of the following.

– During the Folding Phase:
  - Merkle roots $M_0, M_1, \ldots, M_{k-1}$ and field element G_k sent by the prover; and
  - Field elements $x_0, x_1, \ldots, x_{k-1}$ sent by the verifier.
– During the Query Phase:
  - Field elements $s_{0,1}, \ldots, s_{0,\ell}$ sent by the verifier; and
  - Field elements $q_{i,j}, q'_{i,j}$ and Merkle authentication paths $\pi_{i,j}, \pi'_{i,j}$ sent by the prover for $i \in \{0, 1, \ldots, k-1\}$ and $j \in [\ell]$.

The size of this transcript is the proof size. From the above derivation, we have a transcript that consists of

– $k + \ell + 2k\ell + 1$ field elements;
– k hashes of size κ; and
– $2k\ell$ Merkle authentication paths.

The size of the Merkle authentication paths differs for each $i \in \{0, 1, \ldots, k-1\}$. In particular, Merkle root M_i is constructed with $|L_i| = 2^{n-i}$ many leaves, and thus the Merkle authentication paths $\pi_{i,j}$ and $\pi'_{i,j}$ consist of $n - i + 1$ hashes of size κ for all $i \in \{0, 1, \ldots, k-1\}$ and $j \in [\ell]$. Thus the total number of hashes given by all the Merkle authentication paths is

$$2\ell\sum_{i=0}^{k-1} n-i+1 = 2\ell\sum_{i=n-k+2}^{n+1} i = 2\ell\left(\sum_{i=1}^{n+1} i - \sum_{i=1}^{n-k+1} i\right)$$
$$= 2\ell\left(\frac{(n+1)(n+2)}{2} - \frac{(n-k+1)(n-k+2)}{2}\right)$$
$$= \ell\big((n+1)(n+2) - (n-k+1)(n-k+2)\big).$$

Thus our total proof size for FRI-ARG is

- $k+\ell+2k\ell+1$ field elements; and
- $k+\ell\cdot\big((n+1)(n+2)-(n-k+1)(n-k+2)\big)$ hashes of size κ.

Now when we compile FRI-ARG into a non-interactive argument via Fiat-Shamir, the proof only consists of messages sent by the prover, and the verifier messages are obtained via the random oracle. This means that the $k+\ell$ verifier messages in $\mathbb{F}$ are not included in the non-interactive proof. Again assuming a random oracle $\mathcal{H}$ with κ bits of output, the non-interactive proof consists of

- $2k\ell+1$ field elements (the prover's responses to the verifier queries and G_k); and
- $k+\ell\cdot\big((n+1)(n+2)-(n-k+1)(n-k+2)\big)$ hashes of size κ.

Remark 7 (Grinding and Proof Sizes). When incorporating grinding (see Section 3.2) into FS-FRI, it allows the verifier query complexity ℓ to become smaller while achieving the same level of security. This directly reduces the proof size since smaller ℓ directly reduces the number of field elements and hashes needed for the non-interactive proof.

D ethSTARK Conjecture

For completeness, we restate the overview of the ethSTARK conjecture that was originally presented in [12]. The ethSTARK conjecture about a variant of FRI is actually a conjecture about a "Toy Problem Protocol". This Toy Problem Protocol operates as follows. Fix ρ to be a positive constant and fix L to be a multiplicative subgroup of a finite field $\mathbb{F}$ of size $2^k/\rho$, where $k \geqslant 0$. Then this toy protocol operates as follows:

- First a prover sends oracle access to some function $f\colon L_0 \to \mathbb{F}$ (e.g., purported to be an RS codeword).
- Next the verifier samples $\alpha \xleftarrow{\$} \mathbb{F}$ and sends it to the prover.
- The prover and verifier run FRI with respect to the new function $g(x) = (f(x)-\alpha)/x$.

The actual ethSTARK conjecture relates this toy problem to the ethSTARK IOP, which invokes FRI. The actual conjecture is informally stated as follows.

Conjecture 2 (ethSTARK Conjecture, Informal). If a T-time malicious prover attacks the toy problem over finite field $\mathbb{F}$, rate ρ, and $k \geqslant 0$, and succeeds with probability ϵ, then the ethSTARK IOP invoking FRI over $\mathbb{F}$, ρ, and k can be attacked in time T with success probability ϵ.

Conversely, if a T-time malicious prover attacks the ethSTARK IOP using FRI over finite field $\mathbb{F}$, rate ρ, and $k \geqslant 0$, and succeeds with probability ϵ, then the toy problem with finite field $\mathbb{F}$, rate ρ, and k can be attacked in time T with success probability ϵ.

A key observation regarding the above conjecture and the toy problem is that FRI is not being applied to the function f but rather a function g derived from f in a randomized manner by the verifier. Moreover, this also occurs in the case of Batched FRI as well: the prover sends multiple $f_1, \ldots, f_t$, and the verifier sends $\alpha_1, \ldots, \alpha_t$, and the prover and verifier engage in Batched FRI on the functions $g_i(x) = (f_i(x) - \alpha_i)/x$.

The above conjecture posits that the soundness error of the toy problem characterizes the soundness error of commonly deployed FRI-based SNARKs. The following conjecture essentially states that known attacks on the toy problem are optimal. A conjecture in this vein is implicit in [29].

E Extracting SHARP and dYdX Parameters

Here, we document the process of extracting parameters from the SHARP Verifier and the dYdX L2 On-Chain Operator. Examining either of their addresses on Etherscan gives you a list of transactions to examine. The ones we are interested in are transactions with the labels `Verify Proof and Register` under the "Method" column. Examining any of these transactions (by clicking on the transaction hash), one then needs to scroll down to the row labeled "More Details" and press "+ Click to show more". This opens a row with the label "Input Data" with data about a function `verifyProofAndRegister` and information about its input in hexadecimal. Just below the box with this information, one can click "Decode Input Data" to get human-readable parameters.

Turning towards the function `verifyProofAndRegister`, this function comes from the StarkEx Contracts, and has 5 parameters: `proofParams`, `proof`, `taskMetadata`, `cairoAuxInput`, and `cairoVerifierId`. We are interested in `proofParams` and `cairoAuxInput`. The input `proofParams` is a list of integers, and within `verifyProofAndRegister`, the function `verifyProofExternal` is called on `proofParams`. The function `verifyProofExternal` then calls `verifyProof`. All of this leads us to what the integers in `proofParams` represents; they are defined here. In particular, we find that

- `proofParams`[0] is the number ℓ of verifier queries;
- `proofParams`[1] is $-\log(\rho)$ for code rate ρ;
- `proofParams`[2] is the number of grinding bits z;
- `proofParams`[3] is the $\log(d^*)$, where d^* represents the degree of the polynomial sent during the final round of the FRI folding phase (just before the Query Phase; in plain FRI, $d^* = 1$, but in practice many systems set $d^* = 2^8$);
- `proofParams`[4] specifies the number of folding rounds for the protocol; and
- `proofParams`[$5 + i$] specifies the degree reduction factor in round i of the protocol.

The contract defines the bits of security of their system here, which is identical to Conj. 1 (i.e., they are using conjectured security).

Finally, we turn to the input `cairoAuxInput`. here it tells you that the log of trace length = `logNSteps` + 4 (the second term is constant). `logNSteps` is the 0th index of the public parameters. The public parameters are exactly everything in `cairoAuxInput` except the last 2 entries. So the log of the trace length (and thus the input degree of FRI) is given by `cairoAuxInput`[0] + 4.

With all of the above information, one can examine transactions posted on-chain under the `verifyProofAndRegister` method to extract the parameters being used by both the SHARP Verifier and the dYdX L2 On-chain operator.