Hi, want to bring attention to the analysis of ethSTARK security in the random oracle model given in the latest version of the ethSTARK documentation, which is also explained at a higher level in thismedium blog post, which I'll quote the very start of here. Happy to discuss further.

TL;DR

- Non-interactive STARKs start as Interactive

Oracle Proofs

(IOPs), compiled into non-interactive ones in the random Oracle model.

- This post explains the recent update to theethSTARK documentation, which gives a full and concrete analysis of the security of the ethSTARK protocol in the random oracle model.

STARK Security Explained

A STARK proof system (Scalable Transparent Argument of Knowledge) is a powerful tool for computational integrity: it allows verifying the correctness of computations performed on public data in a trustless manner. In this blog post, we delve into the security provided by STARK proofs, defining it and exploring techniques to prove scheme security.

(Read Section 6 in the ethSTARK documentation (version 1.2) for full details and the important and comprehensive independent work of Block et al. on the topic.)

What are we trying to achieve with our security analysis? We would like to prevent a "successful attack" on the STARK system, which is given by a false statement and a STARK proof accepted by the STARK verifier for this (false) statement. Since false statements are dangerous and they can come in all sizes and shapes, we want to be secure against all

false statements. Any false statement, even as trivial as 1+1=3, combined with a STARK proof accepted by a STARK verifier for this statement, is considered a successful attack on the system. (Those with a cryptographic background may be interested to know that STARKs also satisfy stronger security notions such as [knowledge soundness

](https://eprint.iacr.org/2016/116.pdf), but for simplicity, this post focuses on the simpler case of soundness.)

How do we formally define the security of a STARK system? We do so by analyzing the "soundness error" which roughly measures the expected "cost" that an attacker would need to spend to construct a successful attack (i.e., find a STARK proof for a false statement that nevertheless is accepted by the STARK verifier). Mathematically speaking, the soundness error is a function e

(t

) that gets as input a time parameter "t"

, representing the amount of computation time an attacker is willing to spend to mount the attack and outputs the success probability of the attacker in succeeding with the attack (finding a convincing proof of a false statement). As the "cost" t

that the attacker is willing to spend grows, his success probability increases.

Thus far, we have defined the security of STARKs as a function e(t),

which is not the way you naturally discuss security, say, on crypto Twitter. There, you probably heard statements of the form "The scheme has 96 bits of security". How does such a statement translate to our security definition? There is no one answer to this, as people have slightly different interpretations of "x

bits of security":

- A very strict translation would mean that for any t

between 1 and 2^{96}, the soundness error is e

(t

) ≤ 2^{96}

. This means that any attacker running time at most 2^{96}

has a tiny probability of success, smaller than $1/2^{96}$

, which is smaller than one in a billion times a billion times a billion.

- A more relaxed, and perhaps more common, translation is that 96 bits of security means that for any t

, it holds that t

/e

(t

) ≥ 2^{96}

. This means that the success probability is (inverse) linear to the running time. For example, if an attacker has a running time 2^{86}, its success probability is at most $1/2^{10}$.

Read the rest here.