Updates on the performances of GKR

(Work with Olivier Bégassat, Nicolas Liochon and Azam Soleimanian)

This post is follow-up to this post and this post.

Since the previous posts, we have been working on GKR to assess its security and improve its performances. The updates includes (non-exhaustively): algorithmic optimizations of GKR drawn from Zhang et al, implementation/parallelization improvements, updates of the techniques to update GKR inside a Groth16 circuit, fix for the initial randomness technique. We thank Dan Boneh for helpful discussions, for pointing out mistakes in an earlier version of the protocol and helping us fix it.

The full description of the protocol will be out in a paper soon (including security analysis). For now, we share results for our implementation.

Update 08/21/2022

The paper is out on eprint,

https://eprint.iacr.org/2022/1072

Benchmarks

The new benchmarks have been made on a different machine than the first one : an AWS hpc6a (96 physical core + 384 Gb of memory). This machine is cheaper than the one before despite having largely superior computing power. We compare the runtime of our prover and the baseline for parallel MiMC permutations of random values. Our prover consists in proving the hashes using GKR, and using our construction

to embed it inside a Groth16 circuit. As in the previous post, the results for the baseline are extrapoled and for the same reason : cannot go beyond a certain number of constraints due to the 2-adicity of the bn254 scalar field. Since, the machine has more memory, we have been able to run the benchmark for 2^{24}

hashes (as opposed to a maximum of 2^{23}

previously).

The implementation can be found here.

Nb Hashes

Initial Randomness

Gkr Prover

Groth16 Proof

Total

Baseline (extrapoled)

Hash Per second

524288

120 (ms)

3.4(s)

4.0 (s)

7.6 (s)

76 (s)

68 463

1048576

196 (ms)

5.4 (s)

6.4 (s)

12.0 (s)

153 (s)

86 795

2097152

412 (ms)

13.1 (s)

7.1 (s)

20.7 (s)

306 (s)

101 214

4194304

756 (ms)

19.7 (s)

12.5 (s)

33.0 (s)

612 (s)

126 892

8388608

1293 (ms)

29.1 (s)

21.3 (s)

51.7 (s)

1224 (s)

162 149

16777216

2504 (ms)

51.8 (s)

24.5 (s)

78.9 (s)

2449 (s)

212 572

In other words, we have a ~x31 speed-up in using GKR over directly Groth16 for the 2^{24}

instance, achieving a proving speed of 212 572 hashes per second.