VRF-based mining

Runchao Han, Haoyu Lin and Jiangshan Yu
Monash University and CSIRO-Data61
runchao.han@monash.edu

Abstract

In this paper, we describe VRF-based mining, a simple and effective way of making pooled mining impossible. Instead of using hash functions, VRF-based mining uses Verifiable Random Functions (VRFs) for Proof-of-work (PoW)-based consensus. As VRF binds the authorship with hashes, a pool operator should reveal his private key to outsource the mining process to miners, and miners can easily steal cryptocurrency in the pool operator's wallet anonymously.

In addition, we revisit the definition of non-outsourceability in existing research, and show VRF-based mining achieves stronger non-outsourceability than all existing proposals. Moreover, we describe how to instantiate VRFs for VRF-based mining. Last, we discuss potential problems of having no mining pools, and how to address them.

Introduction

In this work, we introduce *VRF-based mining*, a novel idea of using Verifiable Random Functions (VRFs) rather than hash functions for mining in PoW-based consensus. With VRF, a miner should use his private key to mine blocks, so a pool operator should reveal his private key in order to outsource the mining process to miners. In this way, no one can outsource the mining process, otherwise any miner can steal all cryptocurrency in his wallet anonymously.

Our contributions are as follows:

VRF-based mining We propose the idea of VRF-based mining, and describe its detailed construction. The construction is surprisingly straightforward: we replace the hash function in PoW-based consensus with a VRF, and a block in the blockchain should contain the VRF hash and its proof, but does not need to contain the digital signature. Verifying the VRF hash can prove both the correctness of the hash and the authorship of the block in the same time.

Revisiting non-outsourceable cryptocurrency mining We revisit the definition of non-outsourceability of Miller et al. [5]. In particular, we show non-outsourceability consists of punishing pool operators and the unaccountability of punishing. We then show how VRF-based mining achieves stronger punishment of outsourcing than non-outsourceable scratch-off puzzle of Miller et al. [5].

Instantiating VRFs for VRF-based mining We discuss how to instantiate VRFs for VRF-based mining. In particular, we show a VRF has four tweakable components, namely the elliptic curve and two hash functions between strings and elliptic curve elements, and a normal hash function. Then, we discuss concerns how to choose them for VRF-based mining purpose.

Problems and solutions of having no mining pools We discuss two problems of having no mining pools, namely the high mining reward variance and the high orphan block rate. We then discuss how to address them without mining pools.

Verifiable random functions

Verifiable Random Function (VRF) [4] is a public-key version of cryptographic hash function. In addition to the input string, VRF involves a pair of a secret key and a public key. Given an input string and a secret key, one can compute a hash. Anyone knowing the associated public key can verify the correctness of the hash, and can also verify the hash is generated by the owner of the secret key. A VRF consists of four algorithms: VRFKeyGen, VRFHash, VRFProve and VRFVerify.

- $(sk, pk) \leftarrow \mathsf{VRFKeyGen}(1^\lambda)$: on input a security parameter 1^λ, outputs the secret/public key pair (sk, pk).
- $\beta \leftarrow \mathsf{VRFHash}(sk, \alpha)$: on input sk and an arbitrary-length string α, outputs a fixed-length hash β.
- $\pi \leftarrow \mathsf{VRFProve}(sk, \alpha)$: on input sk and α, outputs the proof π for β.
- $\{0, 1\} \leftarrow \mathsf{VRFVerify}(pk, \alpha, \beta, \pi)$: on input pk, α, β, π, outputs the verification result 0 or 1.

VRF-based mining

Cryptocurrency mining consists of two components, namely mining blocks and verifying blocks. We call the process of mining a block Work, and the process of verifying a block Verify. Algorithm 1 and 2 describe Work and Verify, respectively.

```
Algorithm 1: Work($sk, t, Target$).
Input: The secret key $sk$, the block template $t$, and the difficulty parameter $Target$
Output: The block $blk$, the VRF output $h$, and the VRF proof $\pi$
Initialise $n$, $h$, $blk$                          ▷ Initialise variables
while $n \leftarrow$ NextNonce() do                  ▷ Refresh the nonce
    $blk \leftarrow$ ConstructBlock($t, n$)          ▷ Assemble the block
    $h \leftarrow$ VRFHash($sk, blk$)                ▷ Produce the VRF output
    if $h < Target$ then                             ▷ If meeting difficulty
        break                                        ▷ Mining successful
    end
end
$\pi \leftarrow$ VRFProve($sk, blk$)                 ▷ Produce the proof
return $blk$, $h$, $\pi$                             ▷ Return block, hash and proof
```

```
Algorithm 2: Verify($blk, h, \pi, Target$)
$pk \leftarrow blk.txs[0].scriptPubKey$              ▷ Find pubkey in coinbase tx
require($h < Target$)                                ▷ Hash should meet diff requirement
/* Here VRFVerify($\cdot$) ensures:                                  */
/* 1.  $h$ is generated by the owner of $sk$                         */
/* 2.  $h$ is the valid output of VRFHash($sk, blk$)                 */
require(VRFVerify($pk, blk, h, \pi$))
...  ▷ Verify other fields
...  ▷ Verify transactions
```

Work. Miners run Work to mine new blocks. More specifically, a miner - with his private key sk the block template (a complete block without nonce) t - keeps searching for a nonce n that can make the (VRF) hash h of the block blk to meet the difficulty requirement $Target$. Once finding a valid n, the miner produces the proof π of h, and appends blk (with n), h and π to the blockchain.

Verify. Upon incoming blocks, miners run Verify to verify their validity. While other verifications are the same as in hash-based mining, Verify in VRF-based mining should additionally run $\mathsf{VRFVerify}(\cdot)$ to verify 1) whether h is produced by the owner of sk, and 2) whether h is a valid output of $\mathsf{VRFHash}(sk, blk)$.

Block structure. Different from hash-based mining, in VRF-based mining a block should additionally attach h and π, but does not need the signature of a block. Other miners without knowing sk cannot produce h, but can use π to verify h is generated by someone knowing sk. Through proving the authorship of h, π also proves the authorship of the block. Thus, miners only need to sign h, but do not need to sign blocks.

Feasibility and practicality

We experimentally show that VRF-based mining is easy to implement and introduces small overhead. First, we compare the performance of VRFs with three other hash functions used for cryptocurrency mining, namely SHA256D used in Bitcoin, Scrypt used in Litecoin and CryptoNight used in Monero. The comparison shows that VRF can be much faster than Ethereum and CryptoNight. Second, we measure the runtime of each step ($H_1(\cdot)$, point multiplication and $H_2(\cdot)$) of VRFHash. The result shows that the elliptic curve scalar multiplication dominates VRFHash's runtime, which can be optimised in the future.

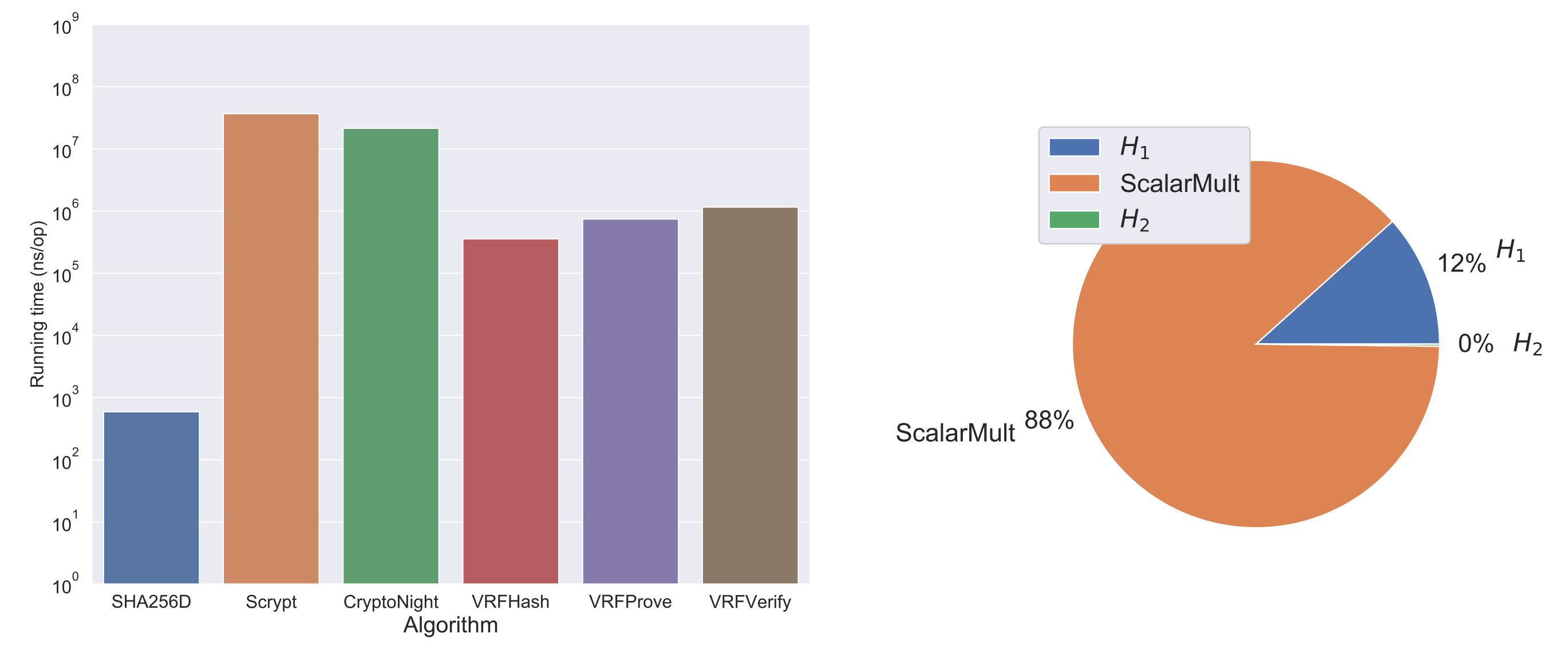

Figure 1: Runtime and profiling of VRFs.

Comparison with other solutions

To the best of our knowledge, VRF-based mining is the first construction that makes pooled mining *impossible*. In this section, we briefly review related research on preventing mining pools, and compare them with VRF-based mining. We classify related research to two types, namely mining protocols trying to address pooled mining, and decentralised mining pools.

Table 1: Comparison between mining protocols.

	VRF-based mining	Puzzle-1 [5]	Puzzle-2 [5]	2P-PoW [2]
Punishment	New reward	New reward	New reward	New reward
Stealing	Unlinkable	Linkable	Unlinkable	Unlinkable
No partial outsourcing	✓	✓	✓	✗
Support randomised signatures	✓	✓	✓	✗
No complex cryptography	✓	✓	✗	✓

Table 2: Comparison with decentralised mining pools.

	VRF-based mining	P2Pool [6]	SmartPool [3]	BetterHash [1]
Complexity	-	Blockchain	Smart contract	-
Decentralisation	Mining	Mining	Mining	Select txs

Conclusion and future work

In this paper, we propose VRF-based mining, that can make pooled mining in Proof-of-work-based consensus impossible. VRF-based mining is simple and intuitive: Miners produce hashes of blocks using VRFs rather than hash functions, so a pool operator should reveal his private key to outsource the mining process to other miners.

As aforementioned, ruling out pooled mining can achieve better decentralisation but may harm the incentive of mining. How to achieve decentralisation while preserving the incentive of mining is still an open challenge.

References

[1] Matt Corallo. Betterhash mining protocol(s), 2019. last accessed on 08/12/19.

[2] Eyal Ittay and Emin Sirer, Gün. How to disincentivize large bitcoin mining pools, 2014.

[3] Loi Luu, Yaron Velner, Jason Teutsch, and Prateek Saxena. Smartpool: Practical decentralized pooled mining. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pages 1409–1426, 2017.

[4] Silvio Micali, Michael Rabin, and Salil Vadhan. Verifiable random functions. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 120–130. IEEE, 1999.

[5] Andrew Miller, Ahmed Kosba, Jonathan Katz, and Elaine Shi. Nonoutsourceable scratch-off puzzles to discourage bitcoin mining coalitions. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 680–691. ACM, 2015.

[6] Forrest Voight. p2pool: Decentralized, dos-resistant, hop-proof pool, 2011.

Acknowledgements

We thank Cheng Wang, Omer Shlomovits and John Tromp for their valuable feedback.