POSEIDON: A New Hash Function for Zero-Knowledge Proof Systems

Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, **Markus Schofnegger**

USENIX Security ’21

TU Graz
SCIENCE
PASSION
TECHNOLOGY

Motivation – Hash Functions in Zero-Knowledge Protocols

- Private cryptocurrency spending
  - Sign transaction $h = H(k, m)$, where k is secret
  - Add h to Merkle tree T of coins
  - Later prove that

    (1) $h \in T$

    (2) $h = H(k, m)$
- h is used in a zero-knowledge context (SNARK, STARK, Bulletproofs, …)
- Proving that $h \in T$ is in general expensive

Traditional Hash Functions

- So why not just use e.g. SHA-256?
  - Too expensive (almost one minute for proofs in early Zcash)
- Proof procedure
  - Express proof verification algorithm as circuit over some field
  - Proof generation time depends on circuit size, width, degree
- Traditional hash functions not well-suited
  - Mainly optimized for e.g. performance on a certain architecture

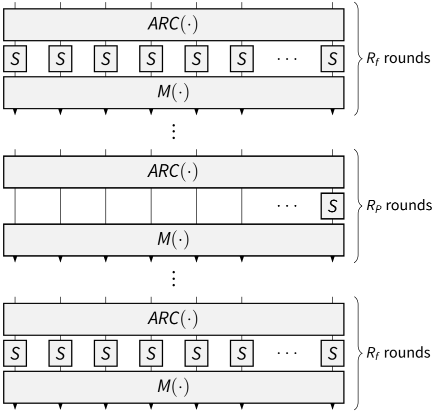
→ Design something new

Which properties are we looking for?

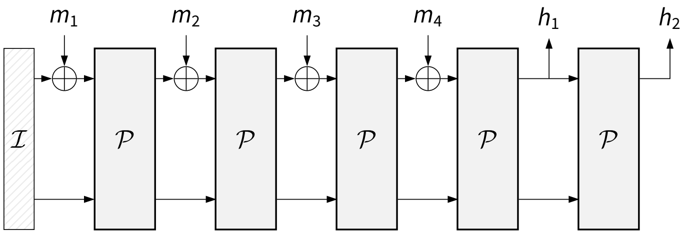
- Operate in big finite field
  - E.g., field of $\approx$ 256 bits
- Consider new metrics
  - Degrees
  - Size of circuit
- Cryptographic security

The POSEIDON$^\pi$ Permutation

- Based on HADES strategy [GLR+20]
- Mixture of full nonlinear and partial nonlinear rounds
- Fixed MDS matrix as linear layer
- “Efficient” S-box
  - Low-degree polynomial
  - E.g., x^3 or x^5
- Flexible design
  - Different field sizes, number of S-boxes, …



The POSEIDON Hash Function



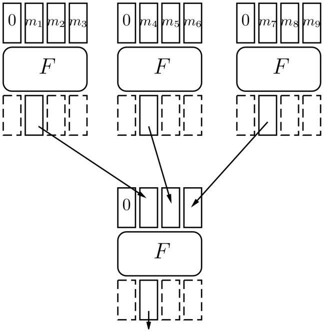
- Sponge hash construction
  - r message elements are added per call
  - c elements remain untouched
- $\mathcal{P}$ is the POSEIDON$^\pi$ permutation
  - Width of $r + c$ elements
- Adjust c according to security level and field size

Cryptanalysis

- Hash-function-specific cryptanalysis
  - Keyless setting, CICO, preimage, ...
- Evaluated many strategies from the last couple of decades
- Algebraic attacks most promising
  - Interpolation, Gröbner bases, ...
- Statistical attacks prevented by external rounds

Sponges in Merkle Trees

- For arity t, use permutation of size $t + 1$
- Fix one element, take out one element



- Arities e.g. $2, 4, 8$

R1CS for POSEIDON$^\pi$

- Single S-boxes in most rounds
- Optimized constraint representation
  - Include linear layer and round constants in fewer constraints
- For R_F full and R_P partial rounds:

$$3tR_F + 3R_P \quad \text{constraints for POSEIDON}^\pi \text{ with } x^5$$

- Merkle tree with 2^m elements:

$$\frac{m}{\log_2(\text{arity} - 1)} \quad \text{levels}$$

Benchmarks

- Focus on security level of 128 bits
- Comparison in two directions
  - R1CS constraints, hashing performance
- Prove leaf knowledge in Merkle tree of 2^{30} elements
- Result:
  - Very low number of R1CS constraints
  - Proof verification in < 1 second with Ristretto
  - Up to 15x hashing performance of comparable competitors

Benchmarks – R1CS

Poseidon-128		
Merkle tree arity	Width	Total constraints
2:1	3	7290
4:1	5	4500
8:1	9	4050
Rescue-x^5		
2:1	3	8640
4:1	5	4500
8:1	9	5400
SHA-256		
510	171	826020

Benchmarks – Runtime

Poseidon-128		
Merkle tree arity	Width	Plain time / call
2:1	3	0.033 ms
4:1	5	0.08 ms
8:1	9	0.259 ms
Rescue-x^5		
2:1	3	0.525 ms
4:1	5	0.555 ms
8:1	9	1.03 ms

Implementations

- Available in various languages
  - Rust, Go, Sage, C++
- Circuit implementations in Bulletproofs and Circom
- Reference implementations available online
  - https://extgit.iaik.tugraz.at/krypto/hadeshash
  - Use version 1.1

POSEIDON in the Wild

- Already used in various protocols
  - Filecoin[1]: Merkle tree proofs
  - Dusk Network[2]: securities trading
  - Sovrin [Lod19]: Merkle-tree-based revocation
  - Loopring[3]: private trading on Ethereum
  - Semaphore[4], Tornado Cash[5], Hermez[6], ...

---

[1] https://github.com/filecoin-project/neptune
[2] https://github.com/dusk-network/Poseidon252
[3] https://tinyurl.com/y7tl537o
[4] https://semaphore.appliedzkp.org/
[5] https://tornado.cash/
[6] https://hermez.io/

Thank you!

More info: poseidon-hash.info

Contact: team@poseidon-hash.info

References

[GLR+20] Lorenzo Grassi, Reinhard Lüftenegger, Christian Rechberger, Dragos Rotaru, and Markus Schofnegger. **On a Generalization of Substitution-Permutation Networks: The HADES Design Strategy.** EUROCRYPT (2). Vol. 12106. Lecture Notes in Computer Science. Springer, 2020, pp. 674–704.

[Lod19] Mike Lodder. Mike Lodder, Sovrin's principal cryptographer `www.sovrin.org`, private communication. 2019.