THEMISv2 on Solana: Protocol Analysis

Solana Labs

1 Introduction

As we explain in our previous submission, Solana is well-suited to implement the L1 pieces of THEMISv2. Solana's speed, capacity, and transaction costs are among the lowest in the industry. As the exact details of the THEMISv2 protocol is not yet finalized, we do not include any prototype code in our submission. Once the cryptography research is further along and the protocol is fully specified, we can build a prototype and provide performance measures in very little time.

We did examine the protocol in much greater depth since our previous submission to the RFC. In this report, we highlight some of our internal discussions regarding THEMISv2 at the protocol level. This report complements our original submission to the RFC and it consists of two main components:

- In Section 2, we provide our analysis of the THEMISv2 and give our general thoughts on the protocol. According to our analysis, the use of black box accumulators (BBAs) of Bobolz et al. [BEK+20] as specified in the RFC is well-suited to THEMISv2. For the accompanying proof system, we believe that Bulletproofs [BBB+18] most naturally complements the BBAs as the accompanying inner product and range proof system.

  With minor modifications to the protocol as specified in our protocol analysis, we expect that a single THEMISv2 payout requires approximately 156 exponentiation operations on a pairing-free elliptic curve. The cost of these exponentiations can be reduced by a factor of 6 using a suitable multi-scalar exponentiation algorithm such as that of Pippenger [Pip80].

- In Section 3, we discuss BBAs and Bulletproofs in the specific context of Solana. In short, a Solana validator with a modern GPU can process up to 7 million elliptic curve exponentiation operations per second. This means that Solana can theoretically perform 270,000 verifications per second.[1] Given that 10M verifications per month equates to approximately 4 verifications per second, this implies that with Solana:

  - There are no availability concerns due to other applications (i.e. DeFi users)
  - There is no problem even if users requested all verifications on the last day of the month
  - There is no need to offload any computation to L2 solutions
  - Payout fees are very cheap

[1] This is under specific hardware assumptions on the Solana mainnet validators. We refer to Section 3 for additional details.

2 Protocol Analysis

In this section, we present our findings from a systematic study of the THEMISv2 protocol. The analysis in this section is purely from a cryptographic protocol point-of-view and is largely independent of Solana. We discuss the protocol in the context of Solana in Section 3.

2.1 Black Box Accumulators

Applications of cryptographic accumulators have gained a considerable amount of study in the last few years with the rise of blockchain technologies. However, many of these applications are either too theoretical to be used in a practical setting or require heavy use of general SNARKs, which we believe is excessive for a simple application like THEMISv2. We believe that the use of simple black box accumulators (BBAs) [JR16] that uses standard Pedersen commitments as specified in the RFC report [rfc] is quite natural in the context of THEMISv2.

All existing BBA constructions follow a general blueprint:

- *Token Structure*: A token $T = (C, \sigma)$ consists of a commitment C to the user's credit value and a signature σ on C that is issued by an authority.
- *Token Update*: The user proves that it owns a valid token $T = (C, \sigma)$ to the authority. The authority issues a new token $T' = (C', \sigma')$ with the user's new credit value.
- *Token Payout*: The user prove that it owns a token $T = (C, \sigma)$ such that:
  - σ is a valid signature on C,
  - the user's committed credit value in C satisfies certain constraints that is required by the system.

The main design decisions in BBA constructions relate to the details of *token update* and *token payout*. We defer the discussion on token payout to Section 2.2 and focus on token update in this section.

In the cryptography literature, there are two main ways to do token updates such that the users' tokens are unlinkable:

- *Structure preserving signatures*: The digital signature σ is a structure preserving signature (SPS) [PS16] as in the construction of Bobolz et al. [BEK+20]. This allows users to randomize their tokens before submitting it to be verified by the authority.
  - *Pros*: Since the signatures can be re-randomized while correctness is preserved, users can achieve token unlinkability without the need for any sophisticated zero-knowledge proofs.
  - *Cons*: All existing structure preserving signatures today require the use of pairings on elliptic curves. For the construction of Hanser and Slamanig [HS14] that is described in the RFC report, signature verification requires 6 pairings. This means that even though the re-randomization computation of the tokens (to be submitted to the authority) is efficient, the time to verify whether the authority issued the new token correctly is costly on the users' machines.
- *Zero-knowledge proofs*: Each user commits to its token T and proves in zero-knowledge that the token is indeed valid as in the constructions of [HHNR17, HKRR20].

  - *Pros*: Zero-knowledge proofs can be constructed without the need for pairings. Zero-knowledge proof systems on standard Pedersen commitments are generally efficient. When considering both proof generation *and* verification, we expect that proofs on Pedersen to be more efficient than SPS.
  - *Cons*: Zero-knowledge proofs on Pedersen commitments are generally efficient; however, in the context of Themis, Pedersen commitments are used as *vector* commitments. In this case, there does not appear to be a simple zero-knowledge proof system that justifies its use as an alternative to structure preserving signatures.

We took some time to think about a zero-knowledge proof system that performs better than structure preserving signatures in the context of vector commitments. However, it appears that any such system would fundamentally require Bulletproofs or SNARKs, and therefore, does not justify its use over SPS.[2]

The main downside of structure preserving signatures is the use of pairings. However, in the context of `THEMISv2`, this can be mitigated to a certain extent:

- As already specified in the RFC report, at the end of each epoch, users can submit its token (after re-randomization) to the Brave server. The server can verify the signature and replace it with a different signature (i.e. ECDSA) on the commitment. This removes the need for pairing computation on chain.

- Similarly, at the end of each epoch, users can optionally submit its original token $T = (C, \sigma)$ to the Brave server along with a new commitment C' and proof π:

  - C' is a Pedersen vector commitment on the same as vector as is committed C. Whereas C is a commitment over a pairing friendly curve, C' is a commitment over a pairing free curve.
  - π proves that C and C' are commitments to the same vector (but over different curves). The proof can be constructed using Bulletproofs or SNARK techniques.

  Now, the token verification on chain can be done entirely over pairing-free curves. According to our experiments, operations on pairing-free curves are at least $5\times$ cheaper than operations on pairing-friendly curves.

- Another downside of the use of SPS is that it is expensive for users to verify the validity of their tokens in their local machines. However, assuming that a user interacts with at most ≈ 10 ads per day, this is not completely unreasonable. Brave can possibly provide users (especially users on mobile devices) with an option to forgo signature verification.

Overall, we believe that the use of Pedersen commitments and structure preserving signatures as specified in the RFC report to be quite natural and well-suited for `THEMISv2`. If the Brave servers replace the structure preserving signatures with regular signatures at the end of each epoch (as specified above), then the pairing operation never runs on chain. Therefore, the bulk of the on-chain computation will consist of token payout proof verification, which we discuss next.

[2] This is for *token updates* only. For *token payout*, the need for Bulletproofs or SNARKs is inevitable.

2.2 Zero-Knowledge Proof Systems

Practical zero-knowledge proof systems today can be classified into three main categories:

- *Proofs over traditional EC groups*: Most notably Bulletproofs [BCC+16, BBB+18] and its variants.
  - *Pros*: These proof systems rely on traditional elliptic curve groups and are quite efficient for specific languages such as inner product relations or range relations.
  - *Cons*: The verification time scales linearly with the size of the prover's witness. The proof system is also designed specifically for inner product proofs and range proofs. Therefore, the concrete efficiency quickly degrades for more general class of languages.
- *Pairing-based*: Most notably SONIC [MBKM19], PLONK [GWC19], and their variants.
  - *Pros*: Proof size and verification times are the smallest compared to Bulletproofs or hash-based SNARKs.
  - *Cons*: The proof system requires a common reference string that must be trusted to have been generated properly. The reliance on pairings also make pairing-based SNARKs excessive for small languages.
- *Hash-based*: Most notably Ligero [AHIV17], STARKs [BSBHR18], Aurora [BSCR+19], and their variants.
  - *Pros*: The amount of computational work that is required by both the prover and the verifier is small as there are no expensive elliptic curve (public-key) operations. The system is also secure against quantum computers.
  - *Cons*: The proof sizes are fundamentally large (kilobytes) due to the reliance on Merkle trees.

Since Solana is a layer 1 blockchain, we are indifferent to any specific proof system, and any one of the systems above can be implemented by the Solana blockchain. However, from a protocol perspective, we found that an adaptation of Bulletproofs is most naturally suited for `THEMISv2` as we explain below.

Bulletproofs. Generally, there are two main drawbacks of the Bulletproofs system:

1. Although the *size* of a user's proof scales logarithmically the user's witness size (in the case of `THEMISv2`, the vector size), the amount of *computational work* that is required by the verifier scales linearly with the dimension of the vector.
2. The proof system is not well-suited as a general SNARK system. Its concrete efficiency is tailored for inner product proofs and range proofs.

For the `THEMISv2` protocol, users are required to prove simple inner product relations and range relations and hence, the second drawback is not applicable. Furthermore, given that each user maintains a vector of dimension 128 with each entry at most an 8 or 9-bit number, the witness sizes for the inner product relations and range relations that the user must generate a proof for are also quite small. The verification algorithms for inner product relations and range relations would require

≈ 136 and ≈ 20 elliptic curve exponentiation operations respectively. Our experiments show that the cost of these exponentiation operations can be reduced by a factor of roughly $6\times$ and $3\times$ respectively by using a preprocessed multi-exponentiation algorithm such as that of Pippenger [Pip80].

Comparison to pairing-based systems. A pairing operation on pairing-friendly elliptic curves is generally $\approx 10\times$ more costly than an exponentiation operation on pairing-free elliptic curves. Existing pairing-based SNARK systems such as Aurora or PLONK requires two pairing operations at the minimum. Therefore, we estimate that the concrete cost of proof verification for Bulletproofs, when multi-scalar exponentiation is used, will be comparable (if not better) than those of pairing-based SNARKs in the context of `THEMISv2`. Bulletproofs also do not require a trusted setup. Given that Bulletproofs and other pairing-based SNARKs are comparable in cost/efficiency, it makes sense to use Bulletproofs that does not require a trusted setup.

Comparison to hash-based systems. The main benefit of hash-based SNARKs is that the computation is lightweight for both proof generation and verification. However, for instance sizes that `THEMISv2` is concerned with, we do not expect significant improvement in computational cost. Hash-based proof systems have much bigger proof sizes that are generally in the order of kilobytes. Unless, post-quantum security is an immediate concern for `THEMISv2`, we believe that there is no significant reason to adopt hash-based SNARKs over Bulletproofs.

3 THEMISv2 and Solana

Using the Bulletproofs system and assuming that the pairing operation is removed from the BBA verification procedure, a single `THEMISv2` payout verification would require approximately ≈ 156 elliptic curve exponentiation operations. With a preprocessed multi-exponentiation algorithm such as that of Pippenger [Pip80], we expect that the total cost of verifying a single proof equates to that of ≈ 26 independent exponentiation operations.

Solana validators with modern GPUs[3] can process up to 7 million elliptic curve exponentiation operations per second. This means that such a validator can theoretically process $\approx 270,000$ `THEMISv2` payout verifications per second. Currently, not all mainnet validators meet the GPU requirements to process this amount of exponentiation operations per second; however, we expect that an appropriate upgrade of mainnet hardware to meet this requirement would not be expensive.

Ten million reward verifications per month translate into ≈ 4 verifications per second. With these numbers, we do not see a need for any L2 solutions. We expect that even if other applications such as DeFi take part of the blockchain availability and Brave sees exponential growth in the number of active users, Solana can comfortably support Brave well into the future.

References

[AHIV17] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 2017 acm sigsac conference on computer and communications security*, pages 2087–2104, 2017.

[3] Nvidia GeForce RTX 3080 or above.

[BBB$^+$18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 315–334. IEEE, 2018.

[BCC$^+$16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 327–357. Springer, 2016.

[BEK$^+$20] Jan Bobolz, Fabian Eidens, Stephan Krenn, Daniel Slamanig, and Christoph Striecks. Privacy-preserving incentive systems with highly efficient point-collection. In *CCS*, 2020.

[BSBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. *IACR Cryptol. ePrint Arch.*, 2018:46, 2018.

[BSCR$^+$19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P Ward. Aurora: Transparent succinct arguments for r1cs. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 103–128. Springer, 2019.

[GWC19] Ariel Gabizon, Zachary J Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *IACR Cryptol. ePrint Arch.*, 2019:953, 2019.

[HHNR17] Gunnar Hartung, Max Hoffmann, Matthias Nagel, and Andy Rupp. Bba+ improving the security and applicability of privacy-preserving point collection. In *CCS*, 2017.

[HKRR20] Max Hoffmann, Michael Klooß, Markus Raiber, and Andy Rupp. Black-box wallets: Fast anonymous two-way payments for constrained devices. *Proc. Priv. Enhancing Technol.*, 2020(1):165–194, 2020.

[HS14] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In *ASIACRYPT*, 2014.

[JR16] Tibor Jager and Andy Rupp. Black-box accumulation: Collecting incentives in a privacy-preserving way. *Proceedings on Privacy Enhancing Technologies*, 2016(3):62–82, 2016.

[MBKM19] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2111–2128, 2019.

[Pip80] Nicholas Pippenger. On the evaluation of powers and monomials. *SIAM Journal on Computing*, 9(2):230–250, 1980.

[PS16] David Pointcheval and Olivier Sanders. Short randomizable signatures. In *Cryptographers' Track at the RSA Conference*, pages 111–126. Springer, 2016.

[rfc] Black box accumulators in the context of themis. Available at https://github.com/brave-intl/themis-rfcc/blob/main/rfcc-themis-bbas-v1.0.pdf.