Advanced FHE Protocols for the Blockchain
(Abstract)

Morten Dahl[1], Clément Danjou[1], Daniel Demmler[1], Tore Frederiksen[1], Petar Ivanov[1], Marc Joye[1], Benoît Libert[1], Dragos Rotaru[1], Nigel P. Smart[1,2], and Louis Tremblay Thibault[1]

[1] Zama, Paris, France
[2] imec-COSIC, KU Leuven, Leuven, Belgium

Speaker: Nigel P. Smart nigel.smart@kuleuven.be

This talk will outline some of the cryptographic protocols and primitives (over and above that of basic FHE encryption and evaluation operations) which are needed to implement private smart contracts. Our motivation is the Zama **fhEVM** protocol (see [3] for details), but the cryptographic primitives we will outline will be of general interest and apply to many FHE-enabled applications.

We outline below the main structure of the talk, pointing to the main papers on ePrint which elaborate on the protocols we will discuss. We also outline how the protocol components fit together into an application.

FHE Protocol: Any FHE application has to sit within a protocol; just as simple Diffie–Hellman key agreement is not used on its own, it is used in a more complex suite such as TLS or Signal. Thus, the first task is to understand what cryptographic protocol or service is being provided, and model this correctly. FHE provides a form of Computing on Encrypted Data, thus it is natural to model FHE applications as MPC protocols. Even a simple symmetric key FHE application is a two-party computation (one person encrypts data, one person performs an outsourced computation, the encrypted result being returned to the first person for decryption). Hence, we first explain how we model our FHE application as an MPC protocol. This follows the protocol outline of [12], which provides a robust actively secure MPC protocol.

We will explain how this MPC model maps onto the application **fhEVM** protocol. Indeed the use of multiple "validators" in the blockchain scenario provides a relatively simple form of Verifiable Computation (needed to obtain the robust MPC protocol), as well as a means to provide Distributed Decryption capabilities (see below).

Zero-Knowledge Proofs: A key component to obtain active security is that we need to protect against entities encrypting data invalidly. This is important in FHE as someone encrypting data can produce an invalid ciphertext, for which it is impossible to determine the invalidity from inspection. This attack vector can enable relatively simple selective failure attacks. To prevent this, every input ciphertext needs to be accompanied by a Zero-Knowledge Proof-of-Correctness.

There are many ways of performing such proofs, however most techniques in the literature introduce forms of "soundness slack". This is perfectly acceptable in many applications, but for FHE applications it may require an increase in parameter values, which results in a very pronounced performance degradation. To avoid this issue we view the encryption process via the lense of the encryptor creating a subset-sum problem. The zero-knowledge proof then becomes a proof of knowledge of a solution to a public subset-sum problem.

We will show how such statements admit efficient solutions, with no soundness slack, in the pre-quantum setting (using vector commitments and the method of [10], which is secure in the Algebraic

Group Model), or in the post-quantum setting (using MPC-in-the-Head techniques and the method of [6] and [7], which are specializations of the KKW [9] approach to this specific problem).

Distributed Decryption: To enable translation of encrypted data into clear data, via control of the MPC program/smart contract, a form of distributed decryption is provided, in which a set of n parties can tolerate up to t corruptions. One can see this operation as equivalent to the *Open* operation in standard LSSS-based MPC programs; thus one can also see it as a means to provide performance optimizations and break the "circuit paradigm" to enable more efficient program representations.

Whilst threshold FHE has been considered in the literature before it is often not considered in the context of a robust decryption operation, or in the context of FHE schemes which enable small parameters (e.g. the TFHE scheme [2] we utilize in the **fhEVM** protocol). The distributed decryption protocol we use is based on a novel bootstrapping methodology, to enable noise-flooding. With the online protocol being completely asynchronous in its operation, making use of the technique of asynchronous online-error correction from the MPC literature. Our talk will outline the distributed decryption methodology, which is given in full detail in [4].

Distributed Key Generation: Our distributed decryption requires a distributed key generation. For this we utilize a standard robust MPC protocol based on that of Damgård and Nielsen [5], which is a synchronous protocol. The same protocol is used in any offline processing for our distributed decryption protocol (which is needed when the value of $\binom{n}{t}$ is large, which happens for some deployments). An efficient asynchronous version of the above protocol is being developed, based on the ideas underlying [11].

Reducing Bandwidth: A key problem in the space of protocols based on FHE, is the large ciphertext sizes. The ciphertext size for BGV and BFV is known to be very large, yet even the ciphertext size for TFHE is orders of magnitude larger than the underlying data. This explosion in size is unavoidable during computation, but for data transmission and storage this is a cost which is important to reduce. This is especially true if encrypted data is stored on a blockchain, or a user wishes to send much encrypted data into the system. We will explain two techniques to reduce such a bandwidth explosion.

The first is a novel public-key encryption methodology tailored for TFHE, which is outlined in [8]. The technique replaces the normal public-key encryption methodology for TFHE (which is usually done via providing many encryptions of zero within the public key), via a methodology which goes via Ring-LWE and the technique more often associated with BGV. This significantly reduces the size of the associated public keys.

The second technique is to use transciphering, i.e. using the FHE algorithm to evaluate the encryption and/or decryption process of another cipher homomorphically. Combined with the distributed decryption functionality described above, this provides a means to go from (say) a symmetric encryption of a value to a homomorphically encrypted value relatively cheaply. However, this "relatively cheaply" requires that the symmetric cipher can indeed be efficiently evaluated homomorphically. In [1] it was shown that TFHE can efficiently evaluate the Trivum and Kreyvium stream ciphers. Whilst Trivium and Kreyvium only provide IND-CPA encryption capabilities, this was shown in [12] to be sufficient for the robust overall MPC protocol based on FHE.

fhEVM *Applications:* Often FHE is considered to be too slow for many applications; thus it is natural to look first for application areas where throughput is already limited due to other constraints.

The relatively high latency of blockchain applications makes FHE not that slow in comparison to computing in the clear. Hence, we feel widespread adoption of FHE applications is more likely to occur initially with the blockchain environment than in other computing environments. The full **fhEVM** stack is described in the whitepaper [3].

We will end the talk with some example smart contracts, which provide enhanced privacy preserving applications in the blockchain space, and present some execution times showing that such smart contracts are already practical. These will be drawn from the list of examples maintained at https://docs.zama.ai/fhevm/resources/examples; for example ERC-20 tokens, Blind Auctions, a Battleship game, or a Darkpool.

References

1. Balenbois, T., Orfila, J.B., Smart, N.P.: Trivial transciphering with Trivium and TFHE. Cryptology ePrint Archive, Paper 2023/980 (2023), https://eprint.iacr.org/2023/980
2. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast fully homomorphic encryption over the torus. Journal of Cryptology **33**(1), 34–91 (Jan 2020). https://doi.org/10.1007/s00145-019-09319-x
3. Dahl, M., Danjou, C., Demmler, D., Frederiksen, T., Ivanov, P., Joye, M., Rotaru, D., Smart, N., Tremblay Thibault, L.: fhEVM: Confidential EVM smart contracts using fully homomorphic encryption (2023), https://github.com/zama-ai/fhevm/raw/main/fhevm-whitepaper.pdf
4. Dahl, M., Demmler, D., Elkazdadi, S., Meyre, A., Orfila, J.B., Rotaru, D., Smart, N.P., Tap, S., Walter, M.: Noah's ark: Efficient threshold-FHE using noise flooding. Cryptology ePrint Archive, Paper 2023/815 (2023), https://eprint.iacr.org/2023/815
5. Damgård, I., Nielsen, J.B.: Scalable and unconditionally secure multiparty computation. In: Menezes, A. (ed.) Advances in Cryptology – CRYPTO 2007. Lecture Notes in Computer Science, vol. 4622, pp. 572–590. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2007). https://doi.org/10.1007/978-3-540-74143-5_32
6. Feneuil, T., Maire, J., Rivain, M., Vergnaud, D.: Zero-knowledge protocols for the subset sum problem from MPC-in-the-head with rejection. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022, Part II. Lecture Notes in Computer Science, vol. 13792, pp. 371–402. Springer, Heidelberg, Germany, Taipei, Taiwan (Dec 5–9, 2022). https://doi.org/10.1007/978-3-031-22966-4_13
7. Feneuil, T., Rivain, M.: Threshold linear secret sharing to the rescue of MPC-in-the-head. Cryptology ePrint Archive, Report 2022/1407 (2022), https://eprint.iacr.org/2022/1407
8. Joye, M.: TFHE public-key encryption revisited. Cryptology ePrint Archive, Paper 2023/603 (2023), https://eprint.iacr.org/2023/603
9. Katz, J., Kolesnikov, V., Wang, X.: Improved non-interactive zero knowledge with applications to post-quantum signatures. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) ACM CCS 2018: 25th Conference on Computer and Communications Security. pp. 525–537. ACM Press, Toronto, ON, Canada (Oct 15–19, 2018). https://doi.org/10.1145/3243734.3243805
10. Libert, B.: Vector commitments with short proofs of smallness. Cryptology ePrint Archive, Paper 2023/800 (2023), https://eprint.iacr.org/2023/800
11. Shoup, V., Smart, N.P.: Lightweight asynchronous verifiable secret sharing with optimal resilience. Cryptology ePrint Archive, Paper 2023/536 (2023), https://eprint.iacr.org/2023/536
12. Smart, N.P.: Practical and efficient FHE-based MPC. Cryptology ePrint Archive, Paper 2023/981 (2023), https://eprint.iacr.org/2023/981