

ITR - (ASE) - (int) - Timelock encryption for use in decentralized systems

Wesley Smith

1 Project Summary

1.1 Intellectual Merit

Conventional encryption paradigms follow a simple outline; Alice encrypts a message with a key and sends the resultant ciphertext to Bob, who decrypts it also with a key. In the case of private-key cryptography, the two use the same key, while in the case of public-key crypto encryption and decryption are done with public-private keypairs. *Timelock* encryption is a proposed paradigm with the potential to revolutionize the way sensitive information is handled. Timelock encryption has a simple high-level description: instead of using a key, *anyone* can decrypt who can prove that a predetermined amount of time has passed. There exist current ways to achieve this kind of functionality, but only with caveats; we are interested in working to achieve timelock encryption **a) without a trusted third party**, and **b) without requiring expensive computation on the part of the decrypter**. Timelock encryption in this form would constitute a significant contribution to National Priority Area *ASE* (advances in science and engineering) as well as Technical Focus Area *int* (integration of computing, networking, human-computer interfaces, and information management, to support reliable, complex, distributed systems).

1.2 Broader Impacts

Decentralized systems is an exploding area of research; Bitcoin, the world's foremost cryptocurrency, showed the world the potential of technologies in the vein of blockchain. The utility of decentralized systems goes far beyond facilitating financial transactions, however: proposed uses of decentralized tech include everything from coercion-resistant e-voting schemes to protocols for on-chain enforcement of nearly every kind of off-chain agreement, record, or transaction. Timelock encryption that doesn't involve trust or expensive computation on the part of the decrypter would slot seamlessly into distributed technology. More importantly, both of those characteristics would be required; the entire purpose of distributed systems is to eliminate the need for trusted third parties, and requiring expensive computation defeats the purpose of an encryption scheme that allows anyone to decrypt. Take electronic voting: an important property of

a voting scheme is *fairness*, or that the system cannot accept information about votes while still accepting votes. Timelock encryption in the form we propose would allow the construction of a decentralized voting scheme with strong fairness guarantees that would be empirically auditable, removing trust from the voting process. Voters could publish their candidate choice to a decentralized platform such that anyone auditing the vote could easily decrypt, but only after all votes were received. This constitutes a clear contribution to science and engineering, as well as more specifically information management and computing to support complex, distributed systems. This is of course not the only example; timelock encryption of this form would subsume many time-sensitive trusted-third-party operations such as will or trust management.

2 Project Description

The concept of timelock encryption was introduced in Rivest, Shamir, and Wagner's 1996 paper *Time-lock Puzzles and Timed-Release Crypto* [3]. Their paper discusses two main approaches: using a trusted third party to release information at a set time, and decryption through solving a set-difficulty computational puzzles. The first of their proposals is in essence how time-sensitive information is handled today, and the second requires prohibitive cost for decryption. A more modern solution was proposed in Liu et. al's 2018 paper *How To Build Time-Lock Encryption* [1]. The insight in this paper is simple and brilliant. They combine two nascent technologies to propose their scheme: the Bitcoin blockchain and witness encryption.

Witness encryption is an interesting twist on standard key-based decryption: in the words of Garg et. al. who introduced the concept, "What if we don't really care if [the decrypter] knows a secret key, but we do care if he knows a solution to a crossword puzzle that we saw in the Times? Or if he knows a short proof for the Goldbach conjecture? Or, in general, the solution to some NP search problem?" [4]. More technically, witness encryption requires that the decrypter provide a *witness* to an *instance* of an NP-problem.

Liu et. al. [1] combine witness encryption with blockchain technology in an exceedingly clever way for their proposed timelock encryption scheme. They suggest building an NP-relation from the Bitcoin blockchain in the form of a SAT problem: a SAT instance where the literals represent possible input and output bits of SHA-256 applied twice (the Bitcoin hash relation) where the statement evaluates to true if and only if the output literals represent the output of passing the input literals to two rounds of SHA-256. Encryption is then done with a SAT instance encompassing suitably many blocks, and a witness is a valid blockchain of the proper length (the starting block is specified to be the current block at the time of encryption). Decryption, then, requires waiting until the specified number of Bitcoin blocks have been generated after encryption (BTC block generation is very predictable). Since the ledger is public, anyone can do this decryption. More specifically, for a message encrypted for **a** minutes,

decryption requires providing a valid Bitcoin blockchain of length $(\mathbf{a}/10)$, where the first block of that chain was the current block at the time of encryption. This form of encryption is not fundamentally different than the set-difficulty computational puzzles proposed before; the cleverness lies in leveraging the Bitcoin blockchain as a large, public computation. The Bitcoin blockchain doubles as a security guarantee, as an attacker wanting to decrypt early would have to outpace the Bitcoin mining pool.

Last year we worked with a group at the University of Edinburgh in an attempt to implement Liu's proposal by combining witness encryption with the Bitcoin blockchain. We found their scheme to be unimplementable in the form they propose [2]; we were successful in building the relevant SAT instances but were not able to use existing witness encryption tools with our created instances. Specifically, we found existing cryptographic multilinear map (a tool in performing witness encryption) resources vastly inadequate: current implementations can create schemes with multilinearity level < 100 , while we require multilinearity level $> 1,000,000$ due to the complexity of working with SHA-256's logical circuit. Our proposal is an extension of this research. We are interested in revisiting the Liu scheme for building timelock encryption, with a focus on the under-the-hood witness encryption. Research will then consist of some or all of the following:

1. Investigation of witness encryption schemes compatible with very large SAT or Subset-Sum problem instances not built from multilinear maps
2. Development of a multilinear map scheme capable of generating parameters with high levels of multilinearity
3. Analysis of existing multilinear map codebases to find potential changes or optimizations
4. Development of a new, ad hoc witness encryption scheme

The ideal result of this project, a functional timelock encryption scheme, is what has potential for broad impacts on society; between the theoretical contributions of Liu et. al. and the practical headway we have made, this result is in sight. With that said, the remaining pieces (witness encryption and/or multilinear maps) are cutting-edge cryptographic technologies and will require a significant amount of time, computational resources, and effort to realize.

References

- [1] Jia Liu, Tibor Jager, Saqib A. Kakvi, and Bogdan Warinschi. How to build time-lock encryption. *Des. Codes Cryptography*, 86(11):2549–2586, 2018.
- [2] Wesley Smith, Myrto Arapinis. *Towards Timelock Encryption*. University of Edinburgh, 2019.
- [3] Ronald L Rivest, Adi Shamir, and David A Wagner. *Time-lock puzzles and timed- release crypto*. 1996.
- [4] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 467–476. ACM, 2013.