IACR Policy for Cryptology Schools

August 2015*

1 Related Works

Nodes on [2]:

- This model starts with describing how to model execution of *synchronous* protocols that can access a global setup clock.
- In a previous treatment, the clock in UC was local to each party and it would have to receive update messages from the other parties (everyone is doing this operation). Hence, with GUC the environment can control the clock speed and define when clock updates happen (as other protocol sessions might also be accessing it).

There are several works from the past few years that try to model a blockchain within the Universal Composability framework—some attempting to model it in its extendion, (G)UC [**?**, **?**].

Kiayias et al. [2] models a Bitcoin-like blockchain for fair and robust multi-party computation. It is motivated by the impossibility result for fairness in secure MPC [1] and circumeventing it by imposing monetary penalties on participants. The model consists of two global functionalities, $\overline{\mathcal{G}}_{\mathsf{clock}}$ and $\overline{\mathcal{G}}_{\mathsf{blockchain}}$. The blockchain functionality enables the expected functionality like submitting tranasctions, validating them, batching them into blocks, and allowing an adversary to reorder transcations. Because of the GUC framework, the state of the blockchain is available to all parties including the environment and any other protocol sessions (or dummy parties). This work however, fails to prove that their model of the blockchain is GUC-realized in any currently existing blockchain system. Such a security proof is essential as it provide credibility to the possibility of implementing protocols in the $\overline{\mathcal{G}}_{\mathsf{blockchain}}$-hybrid world. Furthermore, the assumptions that are made for the blockchain and what the adversary can do severly limit the scope of adversaries in the rearl-world. The first failure of this model is to consider an adversary which can change the view some parties have of the blockchain state. For example, if the adversary mines a new block and keeps it a secret, or if some nodes have not received new blocks because of communication delays. Another failure is that all transactions in the buffer between blocks are always included in the next block. This, again, prevents a miner-like adversary which can censor transactions and delay their entry into the chain. Finally, the state of the blockchain

*The most recent version of this document can be obtained from `http://www.iacr.org/docs/`.
Editors of this document: M. Abdalla, A. Boldyreva, C. Cachin, A. Kiayias, B. Warinschi (2014).

[1] Fairness in MPC is defined as: either all parties learn the output or none of them do.

is updated at fixed time intervals which does not accurately convey the consensus model of Bitcoin or Ethereum.

Badertscher et al. [1] attempt to solve these problems by allowing a more unrestricted in the GUC framework. The shared functionality in this case is a global clock functionality, $\overline{\mathcal{G}}_{\mathsf{clock}}$, which enables modelling a synchronous system in the UC framework by proceeding in rounds. Because it is a shared functionality, the clock allows any other protocol session in the environment to be synchronized with the challenge protocol. The blockchain functionality is a local functionality (only available to the parties within the protocol session) that allows the adversary to have more power in what it can do. The adversary can inject transactions and modify the state of the chain that all parties that query it can see. This is accomplished by allowing a maximum distance, d, that the adversary can specify and return a prefix of the chain which is at most a distance d from the head of the chain. Furthermore, the adversary can choose exactly which transactions are allowed to be in the next block. The blockchain functionality is modularized by allowing the definition of subroutines that capture extending the blockchain state (specifically for Bitcoin in this paper). The authors of this work admit that the paper's only intent is to model the Bitcoin blockchain hence the choice to use the ledger as only a local functionality. This prevents other protocol sessions from using the same blockchain (definitely a limitation of modelling the reality of a blockchain environment). Furthermore, this paper makes the argument that it is dangerous to have a global ledger functionality as such replacement does not "in general, preserve a realization proof of some ideal functionality that is conducted in a ledger-hybrid world, because the simulator in that proof might rely on specific capabilities that are not available any more after the replacement (as the global setup is also replaced in the real world)". It claims that [**?**] provides a sufficient condition for such a replacement, but that the condition is too strong to be satisfied by any ledger implementation.

- Assumes that all transactions get into the next block
- blocks arrive at fixed time intervals (adversary can prevent this with delays)
- every one sees the same state (not realisitic if some parties fall behind or shallow fork)
- adversary can only modify existing transactions and can not insert his own transactions (frontrunning, for example)

References

[1] Christian Badertscher, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Bitcoin as a transaction ledger: A composable treatment. In *Annual International Cryptology Conference*, pages 324–356. Springer, 2017.

[2] Aggelos Kiayias, Hong-Sheng Zhou, and Vassilis Zikas. Fair and robust multi-party computation using a global transaction ledger. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 705–734. Springer, 2016.