WIP: Formalizing the Concordium consensus protocol in Coq

Thomas Dinsdale-Young
Concordium ApS, Aarhus

Bas Spitters
Computer Science, Aarhus University

Søren Eller Thomsen
Computer Science, Aarhus University
DIGIT

Daniel Tschudi
Computer Science, Aarhus University

ACM Reference Format:
Thomas Dinsdale-Young, Bas Spitters, Søren Eller Thomsen, and Daniel Tschudi. 2019. WIP: Formalizing the Concordium consensus protocol in Coq. In *Proceedings of Workshop on Coq for Programming Languages (CoqPL19).* ACM, New York, NY, USA, 2 pages.

We report an industrial application of Coq: our work towards formalizing the Concordium blockchain consensus protocol. This work is currently under NDA, but we expect to be able to release more information by the time of CoqPL. Eventually, all the code will be published under a permissive open source license. This work is a collaboration between Concordium and Aarhus University.

The Concordium consensus protocol is a provably secure Proof-of-Stake protocol (PoS). The protocol has a prototype implementation in Haskell. Our aim is to connect the cryptographic security proof with the Haskell implementation. Ideally, we would either use Coq's extraction mechanism or hs-to-coq [SBRW17] to connect the formalization with the Haskell implementation. Our work is loosely based on the toychain formalization in Coq [PS18]. Toychain is a formalization of a modified bitcoin-like Proof-of-Work (PoW) protocol. Like toychain, we build on the math-components library.

Formalizing a PoS protocol is inherently different from formalizing a PoW protocol. In PoS a block-correctness-proof not fixed to a specific block, but rather to a specific slot. Based on [BMTZ17] we do however aim to find a good modularization of the protocol. This will allow for a common generalization of both PoS and PoW protocols.

In our Coq formalization, we abstract away from the P2P-layer and formalize an ideal semi-synchronous network functionality. In toychain every peer relays a message containing hashes of its current stash of blocks to all peers, each time the peer receives a block. We verify a protocol without this message redundancy. At the workshop we will present the Coq implementation of the protocol and its *functional correctness*.

To prove *security*, we need to add probabilistic computation. This allows us to add cryptographic primitives such as hash functions. To do so, we will use a probabilistic library in Coq: either the one from information theory [AG15], following a successor of toychain [GS18], or the classical analysis library in math-components which has been used to formalize the logic underlying easycrypt.

Next we will need to reason about adversaries. This will most likely combine ideas from the formalization of multiparty computation in easycrypt [HKO$^+$18, ABB$^+$18] and insights from abstract cryptographic frameworks [BDLKK18, Mau11] and the rational analysis of Bitcoin [BGM$^+$18].

Finally, to support that using extracted code can be efficient, we mention the impressive work for the Raft (permissioned) consensus protocol which is formalized in Coq using the Verdi-framework [WWP$^+$15, WWA$^+$16]. It proves that the raft protocol is linearizable, and extracts an efficient implementation of the protocol with performance comparable to a state of the art implementation. A small trusted shim connects the extracted functional code to the running distributed code.

Concordium aims to collaborate with other science heavy companies in setting up a collaborative effort to develop clear quality standards for blockchains. At the end of this lecture, we invite discussion on how Coq is best used in this process.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CoqPL19, January 19, 2019, New York, NY, USA
© 2019 Association for Computing Machinery.

Acknowledgments

We would like to thank Elaine Li who was involved in the beginning of this work.
Ilya Sergey has been very helpful in discussing their upcoming work on formalizing the bitcoin backbone protocol.
This work has been supported by the Concordium Blockchain Research Center, Aarhus University, Denmark.

References

[ABB+18] José Bacelar Almeida, Manuel Barbosa, Gilles Barthe, Hugo Pacheco, Vitor Pereira, and Bernardo Portela. Enforcing ideal-world leakage bounds in real-world secret sharing MPC frameworks. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018* [DBL18], pages 132–146.

[AG15] Reynald Affeldt and Jacques Garrigue. Formalization of error-correcting codes: from hamming to modern coding theory. In *International Conference on Interactive Theorem Proving*, pages 17–33. Springer, 2015.

[BDLKK18] Chris Brzuska, Antoine Delignat-Lavaud, Konrad Kohbrok, and Markulf Kohlweiss. State-separating proofs: A reduction methodology for real-world protocols. Cryptology ePrint Archive, Report 2018/306, 2018. https://eprint.iacr.org/2018/306.

[BGM+18] Christian Badertscher, Juan Garay, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. But why does it work? a rational protocol design treatment of bitcoin. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 34–65. Springer, 2018.

[BMTZ17] 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.

[DBL18] *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018*. IEEE Computer Society, 2018.

[GS18] Kiran Gopinathan and Ilya Sergey. Towards mechanising probabilistic properties of a blockchain. In *CoqPL 2019 The Fifth International Workshop on Coq for Programming Languages*, 2018.

[HKO+18] Helene Haagh, Aleksandr Karbyshev, Sabine Oechsner, Bas Spitters, and Pierre-Yves Strub. Computer-aided proofs for multiparty computation with active security. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018* [DBL18], pages 119–131.

[Mau11] Ueli Maurer. Constructive cryptography–a new paradigm for security definitions and proofs. In *Theory of Security and Applications*, pages 33–56. Springer, 2011.

[PS18] George Pîrlea and Ilya Sergey. Mechanising blockchain consensus. In *Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pages 78–90. ACM, 2018.

[SBRW17] Antal Spector-Zabusky, Joachim Breitner, Christine Rizkallah, and Stephanie Weirich. Total haskell is reasonable coq. *CoRR*, abs/1711.09286, 2017.

[WWA+16] Doug Woos, James R. Wilcox, Steve Anton, Zachary Tatlock, Michael D. Ernst, and Thomas Anderson. Planning for change in a formal verification of the raft consensus protocol. In *Proceedings of the 5th ACM SIGPLAN Conference on Certified Programs and Proofs*, CPP 2016, pages 154–165, New York, NY, USA, 2016. ACM.

[WWP+15] James R. Wilcox, Doug Woos, Pavel Panchekha, Zachary Tatlock, Xi Wang, Michael D. Ernst, and Thomas Anderson. Verdi: A framework for implementing and formally verifying distributed systems. In *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '15, pages 357–368, New York, NY, USA, 2015. ACM.