Cryptocurrency Project: Assignment Description

CS 168: Cryptocurrencies and Security on the Blockchain
Prof. Thomas H. Austin
San José State University
thomas.austin@sjsu.edu

October 14, 2021

1 Introduction

For the class project, you will implement some significant functionality related to the blockchain. The goal is deliberately open-ended to allow you to work on something you find interesting.

You may work alone or with one other person.

Submit a 2-3 page proposal for your cryptocurrency in PDF format. Include your name and your partners' name(s). Only one partner needs to submit.

Some project ideas:

- Extend SpartanGold with additional features from Bitcoin.
- Port SpartanGold into a different language.
- Review research papers and implement some of the features into SpartanGold (or other codebase).

1.1 Adding Bitcoin Features into SpartanGold

If you would like to get a deeper understanding of how Bitcoin works, you can implement a few of the features *not* implemented in Spartan Gold. Some examples:

- Store transactions into a Merkle tree instead of a map.
- Set the proof-of-work difficulty to adjust to the power of the network over time.
- Use a fixed block size, with miners always selecting the most valuable transactions.
- Create a wallet for SpartanGold, following BIP-32/BIP-39/BIP-44.
- Integrate Bitcoin Script, or something similar.

1.2 Port SpartanGold into a Different Language

JavaScript has many benefits for prototyping different blockchain concepts. However, it does **not** tend to be the most popular language for blockchain development.

The Rust and Go languages are widely used in this space since they have performance similar to C++, but without a lot of the problems of C++. Other languages are possible, but discuss with me first.

Ideally, your client should implement SpartanGold faithfully enough that it can interoperate with my JavaScript implementation.

1.3 Research Project

This project is a particularly good choice if you are interested in extending your work into a thesis project or independent study project.

You will review some research papers, and incorporate their ideas into your implementation.

If you are looking for inspiration, you might skim through Bonneau et al.'s survey paper [1]. The rest of this document also discusses some areas that you might consider.

2 Alternate Consensus Modes

The Bitcoin protocol relies on proof-of-work, which roughly translates to one vote per CPU. However, these computations are essentially waste of the Bitcoin protocol; that is, they offer no value beyond their utility in validating Bitcoin transactions. Furthermore, these computations are more easily performed by specially equipped mining rings, giving rise to powerful mining farms that reduce Bitcoin's decentralization of consensus.

Some work has explored making use of these transactions. Primecoin [9] bases their proofs on searching for prime numbers. Other approaches have tried to use computations that rely less on number crunching, with the goal of making ordinary machines more effective in mining [1].

In this section, we focus on two alternate consensus mechanisms. *Proof-of-stake* translates coin ownership into voting rights, with the theory that the more coins a miner has, the more vested interest they have in protecting the currency's reputation. *Proof-of-space* protocols use storage in place of computation.

2.1 Proof-of-Stake

The most popular alternative to proof-of-work is proof-of-stake; in essence, *virtual mining* involves use of currency in circulation on the mining network to establish consensus. Just as with proof-of-work blockchains, the proof-of-stake miners receive additional currency for their mining efforts.

Peercoin [10] first introduced the concept of proof-of-stake. Their approach uses a mix of proof-of-work and proof-of-stake based on coin age. Essentially,

miners still compete in a proof-of-work protocol, but the difficulty of the target varies depending on the amount of coin age consumed. Therefore, a miner who has a lot of coins that have not been spent over a period of time has a greater advantage in finding the proof first. Once the proof has been found, the coin age value is consumed and the miner faces greater difficulty in finding another block.

One potential concern of their approach is that an attacker could steadily build up mining power, possible accumulating 51% of the coin age in the market. The Peercoin authors discuss checkpointing as a possible defense to limit the danger of this attack.

Tendermint [11] instead uses a coin-deposit approach; coins are locked for some set period of time in exchange for the right to propose a block and the right to vote on valid blocks. The more coins that are locked, the more often a miner gets to propose a block and the more voting power that miner has. To spend the coins, the miner must post a special *unbonding transaction* and then wait for a set period of time for the coins to be released. If the miner has voted for multiple competing blocks, an *evidence transaction* can be posted to destroy that miner's stake.

Ethereum is currently developing the Casper protocol [2], which is a proof-of-stake system designed to work in conjunction with their existing proof-of-stake system.

2.2 Proof-of-Space

Another alternative for establishing consensus relies on storage rather than computation.

Burst's proof-of-capacity (PoC) [5] and SpaceMint's proof-of-storage [17] are both variants of proof-of-space. The data their miners store is not intended to be useful outside of consensus. The authors argue that their approach is both more energy efficient and more equitable in rewarding miners than proof-of-work protocols.

Miller at al. [16] discuss how to replace Bitcoin's Proof-of-Work with a Proof-of-Retrievability solution, which they use in a modified version of Bitcoin called Permacoin. Essentially, Permacoin miners prove that they are storing some portion of archival data. Since the proof itself provides part of the data, miners inherently serve as backup storage.

In Filecoin [4], miners prove that they are storing data through special Proofs-of-Spacetime, which are themselves based on Proofs-of-Replication [19]. Filecoin miners store the data in an encrypted format using a modified form of cipher-block-chaining. This design deliberately introduces latency to their "Storage Market" network, preventing cheating miners from being able to produce the data in time. A second "Retrieval Market" relies on a gossip protocol to provide the needed data off-chain. The Filecoin literature discusses many interesting attacks, such as Sybil attacks, outsourcing attacks, and generation attacks.

Sia [20] and Storj [21] are two other cryptocurrencies that provide decentralized storage in peer-to-peer networks. These systems rely on Merkle trees made of the hashes of file contents, allowing them to (probabilistically) verify that the storage provider is actually storing the data that they claim to store without verifying the entire file. They guarantee the reliability of their systems by using erasure codes to divide data between the data storage providers.

3 Increasing Throughput of Blockchains

Bitcoin's fixed block size and slow block generation has occasionally led to a backlog of transactions. Several protocols have attempted to address this problem with different designs.

Bitcoin-NG [3] modifies Bitcoin to allow a winning miner to continue to produce blocks without a proof-of-work until another miner finds a valid proof. Essentially, the Bitcoin proof acts like a leader election for producing blocks until a new miner takes control.

Several proof-of-stake protocols attempt to reduce consensus on blocks down to minutes or even seconds. Dfinity [8], Thunderella [18], and Algorand [6] are all protocols in this direction. Algorand in particular is worried about denial-of-service attacks; to address this issue, the protocol is designed so that no one knows who the current leader is until after they have done their work. (This design has a parallel to Bitcoin – in Bitcoin, no one knows who will produce a block until they share a valid proof).

4 Smart Contracts

The Bitcoin Script language is purposefully limited to avoid computations that would be too burdensome for the blockchain. It is not Turing complete. It does not support loops, or even multiplication or division. While it is sufficient for validating transactions, the power of the language is extremely limited.

Ethereum [22] expands on Bitcoin's work to provide a Turing-complete[1] language designed to be run on the blockchain. In Ethereum, clients pay a fee in *gas*. At each step of the computation, the gas fee increases; once the gas given by the client is exceeded, the miner rejects the transaction, and the client forfeits their gas fee.

While Ethereum's flexibility and power has made it very popular, it has also lead to some well-publicized faulty contracts. The most well-known of these led to a substantial amount of stolen Ethereum coins (ether) [14]. This theft lead to the majority of the network rolling back the history of the blockchain to invalidate the transaction, while a substantial minority of Ethereum miners

[1] Ethereum's "Yellow Paper" [22] refers to this as *quasi-Turing complete*. Since all computations are bounded by a gas parameter, they are not truly Turing-complete. Said another way, smart contracts in Ethereum would be Turing-complete if they had an infite supply of gas.

decided to maintain the original history; this second group formed what is now know as Ethereum classic.

Zilliqa [23] seeks to offer a balance between the flexibility of Ethereum and the safety of BitScript. Zilliqa's smart contract language is not Turing-complete, but is substantially more powerful than Bitcoin Script. Its model is built around MapReduce, a popular model for running distributed computation. Like Ethereum, Zilliqa uses a notion of gas to prevent a poorly written contract from stalling the blockchain.

5 Other Cryptocurrencies of Interest

EOS [12] aims to be a distributed operating system based on the blockchain. They use a form of proof-of-stake referred to as *delegated proof of stake* [13], which works vaguely like a representative democracy. Block producers do not need to have a large amount of stake, but must gain (and keep) the approval of voters, where voting is determined by stake. While this leads to more centralization, which has been a constant criticism of EOS, it allows much more rapid block generation.

Tezos [7] is a blockchain protocol designed to rewrite itself by community vote. A proposal is shared in the form of an OCaml tarball. If the community approves the changes, the OCaml code becomes the new official codebase.

0Chain uses a *token-locking reward protocol* [15], where users lock tokens for 90 days to generate interest for miners. In this way, 0Chain allows for "free" transactions, in the sense that clients do not permanently sacrifice their tokens. This approach combines some of the benefits that you get from transaction fees and from coinbase transactions in the Bitcoin system. (Full disclosure – I was involved in this project and helped to design this protocol).

References

[1] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *IEEE Symposium on Security and Privacy*, pages 104–121. IEEE Computer Society, 2015.

[2] Vitalik Buterin and Virgil Griffith. Casper the friendly finality gadget. 2019. URL: `https://arxiv.org/pdf/1710.09437.pdf`.

[3] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert van Renesse. Bitcoin-ng: A scalable blockchain protocol. In *Symposium on Networked Systems Design and Implementation (NSDI)*, pages 45–59. USENIX Association, 2016. URL: `https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/eyal`.

[4] Filecoin: A decentralized storage network. Technical report, Protocol Labs, August 2017.

[5] Seán Gauld, Franz von Ancoina, and Robert Stadler. The burst dymaxion: An arbitrary scalable, energy efficient and anonymous transaction network based on colored tangle. `https://dymaxion.burst.cryptoguru.org/The-Burst-Dymaxion-1.00.pdf`, accessed July, 2018, 2017.

[6] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 51–68, 2017.

[7] L.M Goodman. Tezos a self-amending crypto-ledger. Technical report, Tezos Foundation, 2014.

[8] Timo Hanke, Mahnush Movahedi, and Dominic Williams. Dfinity technology overview series, consensus system. *arXiv preprint arXiv:1805.04548*, 2018.

[9] Sunny King. Primecoin: Cryptocurrency with prime number proof-of-work. `http://primecoin.org/static/primecoin-paper.pdf`, 2013.

[10] Sunny King and Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. `http://primecoin.org/static/primecoin-paper.pdf`, 2012.

[11] Jae Kwon. Tendermint: Consensus without mining. `https://tendermint.com/static/docs/tendermint.pdf`, 2014.

[12] Dan Larimer. Eos.io technical white paper. Technical report, block.one, 2017. URL: `\url{https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md}`.

[13] Daniel Larimer. Delegated proof-of-stake (dpos), 2014.

[14] Matt Leising. The ether thief. *Bloomberg*, 2017.

[15] Paul Merrill, Thomas H. Austin, Jenil Thakker, Younghee Park, and Justin Rietz. Lock and load: A model for free blockchain transactions through token locking. In *IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*. IEEE, 2019.

[16] Andrew Miller, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. Permacoin: Repurposing bitcoin work for data preservation. In *IEEE Symposium on Security and Privacy*, pages 475–490. IEEE Computer Society, 2014.

[17] Sunoo Park, Krzysztof Pietrzak, Albert Kwon, Joël Alwen, Georg Fuchsbauer, and Peter Gazi. Spacemint: A cryptocurrency based on proofs of space. *IACR Cryptology ePrint Archive*, 2015:528, 2015.

[18] Rafael Pass and Elaine Shi. Thunderella: Blockchains with optimistic instant confirmation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 3–33. Springer, 2018.

[19] Proof of replication. Technical report, Protocol Labs, July 2017.

[20] David Vorick and Luck Champine. Sia: Simple decentralized storage. Technical report, Nebulous Inc., November 2014.

[21] Shawn Wilkinson, Tome Boshevski, Josh Brandoff, James Prestwich, Gordon Hall, Patrick Gerbes, Philip Hutchins, and Chris Pollard. Storj: A peer-to-peer cloud storage network. Technical report, Storj Labs Inc., December 2016.

[22] Gavin Wood. Ethereum: a secure decentralised generalised transaction ledger. `https://gavwood.com/paper.pdf`, 2014.

[23] The zilliqa technical whitepaper. `https://docs.zilliqa.com/whitepaper.pdf`, 2017.