The University of Sydney

School of Information Technology

INFO5993 Research Methods - Assignment II

Unforkable Blockchain Cryptocurrencies with Efficient Zero Knowledge Contingent Proof on Mobile Devices

Author:
Lin Han

Supervisor:
Dr. Vincent Gramoli

October 13, 2017


SIDERE MENS EADEM MUTATO

Contents

1 Introduction

From the time when Block 0 of the Bitcoin blockchain, the Genesis Block, is created at 18:15:05 GMT on January 3rd, 2009, the words "cryptocurrencies" and "Blockchain" become one of the most popular fields in information technology. The "decentralized" and "anonymous" nature of cryptocurrencies overcomes the weakness of traditional *trust-based* electronic payments who relies heavily on trusted third-party financial institutions. The *cryptographic proof-of-work* of bitcoin enables reliable transaction between two parties directly. Through almost 10 years development, cryptocurrencies turns out to be a large family and can be deployed onto multiple devices.

Though bitcoin give a great solution on *double spending* problem, it doesn't mean that it is secured in all aspects. One possible issue is attacks targeting blockchain's forkable feature. Other cryptocurrencies allowing forkable chains all suffer from the very same issue. In this sense, unfokable blockchain is proven to solve this problem. On the other hand, another possible solution to secure transaction is to adopt mechanism like *zero knowledge contingent payment* which are released if and only if some knowledge is disclosed by the payee and to do this in a trustless manner where neither the payer or payee can cheat. Whilst there are several theoretical discussion and practice in a variety of contexts, this paper will concentrate on their application on cryptocurrencies blockchain, especially on mobile devices.

2 Bitcoin

2.1 Blockchain

Blockchain is the way how Bitcoin keeps its public ledger. To some extent, blockchain is simply a peer-to-peer distributed timestamp server. The ultimate goal of this design is to solve *double-spending* problems and prevent modification of transaction records.

Each full node in the Bitcoin network keeps a full copy of the blockchain, in which all blocks validated by this particular is stored. When several nodes within the network independently arrive at identical blockchains, they are considered to be in *consensus*. As its name suggests, a blockchain is a digital chain of blocks, where a timestamp, a nonce, and a Merkle Tree is stored. The blocks are chained cryptographically using hash. In detail, each

block contains the hash of its previous block ,finally leading to the Genesis Block. Any modification on blocks in the chain would violates all subsequent hashes, which is vital for consistency of the ledger. Figure 1 shows part of a blockchain.

However, computing a hash is expensive. This truth enables the adoption of *proof-of-work* in bitcoin network.

2.2 Proof of Work

According to blockchains' feature, a huge amount of computation is required in the generation of each block. Meanwhile, there is a *proof-of-work* mechanism to make the distributed timestamp server work and determine representation in majority decision making. Especially, when there are multiple chains (forked chains), consensus rules will pick up the longest chain, which contains the most proof of work during its generation.

In this way, any malicious changes on previous blocks would violate its following blocks. That is to say, hacker with huge computing power can hijack the blockchain if he can generate the longest chain from the block he hacks, in turn, he has to own more than half of the computing power within the whole blockchain network.

2.3 Contracts

There are distributed contracts in Bitcoin transactions for agreement enforcements, which provides another way to formalize and guarantee agreements rather than traditional court system. Examples include Escrow, Micropayment channels and CoinJoin.

Some of the contracts can be implemented in Bitcoin Script, especially the zero knowledge contingent payments in Bitcoin is achieved using it. However the Red Belly Blockchain doesn't have a robust script language like Bitcoin does.

3 Ethereum

3.1 Previous Work

Bitcoin provides a protocol allowing weak implementation of *smart contracts*. However, several limitations exists in Bitcoin's scripting language:

1. **Not Turing-Completeness** - Bitcoin scripts lacks loops.
2. **Lack of States** - UTXOs scripts is only for one-off contracts.
3. **Blindness of Blockchain** - Bitcoin scripts cannot access blockchain data.
4. **Blindness of Value** - Bitcoin either consumes the entire UTXO or none of it

3.2 Rationale

Ethereum implements a blockchain with Turing-complete scripts, states, value awareness and blockchain awareness, which enables development of smart contracts, and even new protocols.

4 Balance Attack

As the previous review mentioning, to attack a blockchain, or specifically to rewrite the content of a block, the hacker should have more than half of the computing power of the whole blockchain network which is almost unfeasible in real world. In particular, by delaying the propagation of blocks in Bitcoin system, the hacker can in result delay the growth of the longest branch of the system. In other word, he can then hijack the blockchain even without a large amount of computing power. Ethereums' "Blockchain 2.0" somehow fixes this problem, but there is still other possible method against forked blockchain. One practice is the **Balance Attack** against *proof-of-work* blockchain systems.

To achieve a balance attack within the blockchain network, the attacker should divide the network into subgroups of similar mining power by cutting

off their communications. During this down time, the attacker issues transaction in the *transaction group*, and mine blocks in the *block group* simultaneously. This action only ends when it comes to the point where the tree of the block subgroup outweighs the tree of the transaction group, which is with high possibility. The balance, in result, can leverage the *GHOST* protocol that accounts for sibling or uncle blocks to determine on a chain of blocks. This strategy allows the attacker to mine a branch regardless of the rest of the network so that he can influence the branch determination process while merging. The process is as shown in Figure 2.

5 Unforkable Blockchain

5.1 Byzantine Consensus Problem

The *Byzantine Consensus Problem* refers to the *Byzantine General's Problem* proposed by Leslie Lamport, Robert Shostak and Marshall Pease in 1982. The problem is complicated by the presence of traitorous generals who may not only cast a vote for a suboptimal strategy, they may do so selectively. All the votes and results are simplified to attack or retreat. The problem is complicated further by the generals being physically separated and having to send their votes via messengers who may fail to deliver votes or may forge false votes.

In computer science area, typically computers or participants in network are mapped to generals and links between them are mapped to messengers.

5.1.1 Traditional Blockchain Byzantine Consensus Problem

The *proof-of-work* of Bitcoin blockchain is the primary solution to *Byzantine Consensus Problem*.

In detail, a model of Bitcoin network can be built upon the classic Byzantine Consensus Problem. The distributed system is the alliance of generals, in which the upper bounds on the delay of communicating and decision-making is unknown.

However, regarding to our previous discussion on *Balance Attack*, attackers can still disrupt the consensus system to beat the Bitcoin Byzantine Consensus. Because there is no guarantee that the decided value is proposed by a valid process.

5.1.2 Democratic Byzantine Consensus

Democratic Byzantine Fault Tolerance is a system specially tailored for consortium blockchains based on *Binary Byzantine Consensus*. In *Binary Byzantine Consensus*, each trusted participant issue proposal with either 0 or 1 and decides that the final agreement such that:

1. No pair of trusted participants have different decision.
2. Every trusted participant decides
3. If all correct participant propose the the same value, then no other value can be deiced

In safe Democratic Byzantine Fault Tolerance, a mechanism called binary broadcast for binary Byzantine Consensus system is adopted.

5.2 The Red Belly Blockchain

6 Zero Knowledge Contingent Proof

6.1 Bitcoin

6.2 Efficient Implementation of Zero Knowledge Proof in Cryptocurrencies

7 Conclusion

References

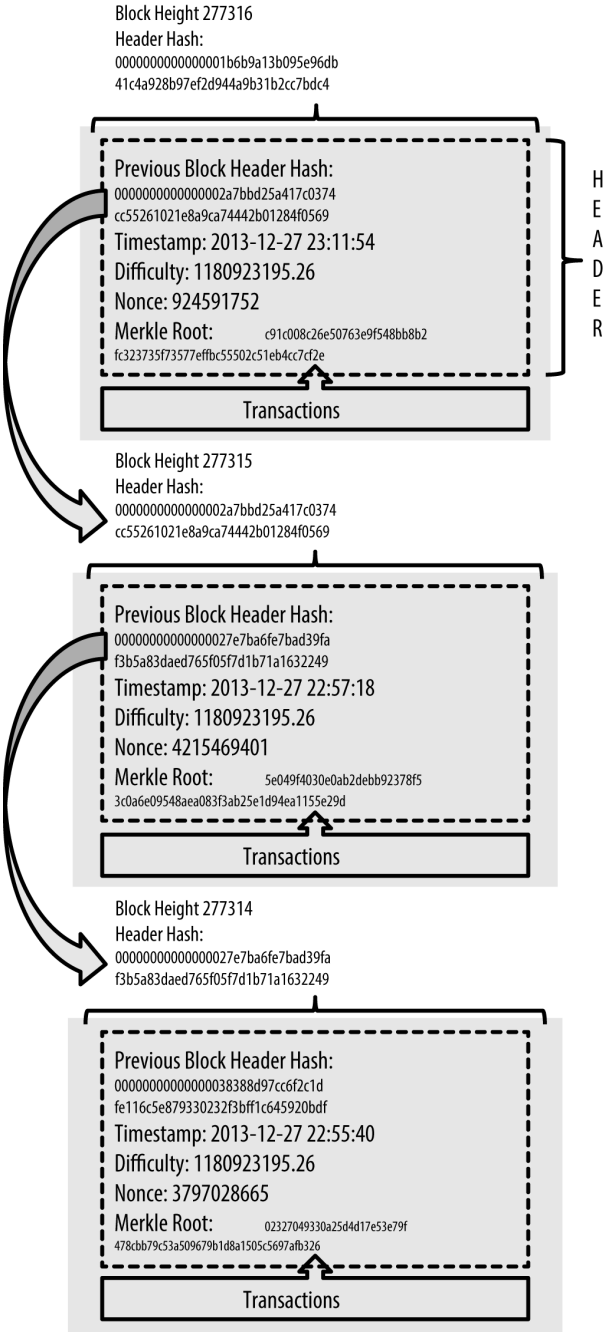

Figure 1: Blockchain

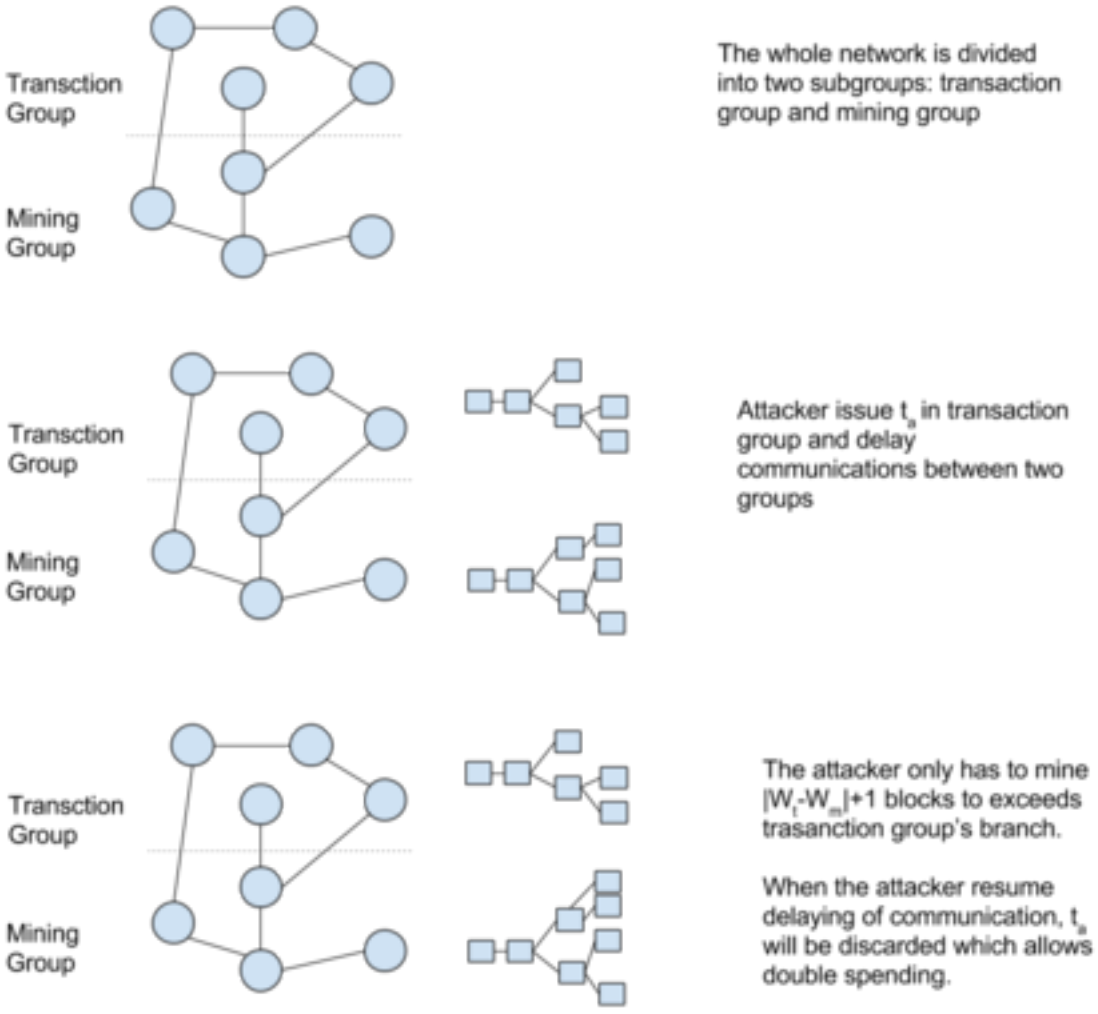

Figure 2: Balance Attack