P2P Marketplace Project

A blockchain framework for building competitive marketplaces

`

EX FIDA BONA

Version 0.1

Note: This paper assumes the reader is familiar with and has read both the Ethereum whitepaper[1] and the Quorum whitepaper[2]. Active research is under way, and new versions of this paper may change. Project progress can be tracked at https://github.com/exfidabona/p2p-marketplace. For comments and suggestions, please email us at research@coen.io.

[1] https://github.com/ethereum/wiki/wiki/White-Paper
[2] https://github.com/jpmorganchase/quorum-docs/blob/master/Quorum%20Whitepaper%20v0.1.pdf

1 Introduction

A common problem for new social networks and digital marketplaces is that they struggle to gain user adoption. These systems require a minimum amount of users to fill multiple roles in order for the network to be functional. This issue is often referred to as the “chicken and egg problem[3]”. It can require millions of dollars in capital to incentivize usage to fill roles until the network reaches a critical mass of participants.

One of the key advantages of networks built on top of blockchain technology is that a secure value token can easily exist natively in the platform. Since these tokens can be exchangeable for other tokens and currencies, they have real world asset value as a cryptocurrency. Starting with a built-in incentive system is a huge advantage over internet based centralized applications, since value can be captured immediately in a token and passed on to users instead of having to pre-pay for value with fiat money[4].

However, existing cryptocurrency blockchain platforms such as Ethereum[5] and Bitcoin[6] lack the governance and permission features that a marketplace may require.

JPMorgan Chase & Co. has developed a permissioned blockchain based on the Ethereum[7] protocol named Quorum[8]. Quorum added both data privacy and governance for its usage within the financial services industry, but also removed the native cryptocurrency features.

The goal of this project is to extend quorum to build a new framework for marketplaces to define their own currency, privacy settings and business logic. This paper focuses on the modifications to Quorum and by extension the Ethereum protocol.

2 The P2P Marketplace Project

The P2P Marketplace Project(P2PMP) is a framework written in Go that helps define a cryptocurrency driven marketplace on top of a custom permissioned blockchain. P2PMP is

---

[3] http://cdixon.org/2009/08/25/six-strategies-for-overcoming-chicken-and-egg-problems/
[4] https://en.wikipedia.org/wiki/Fiat_money
[5] https://www.ethereum.org/
[6] https://bitcoin.org
[7] https://github.com/ethereum/go-ethereum
[8] https://github.com/jpmorganchase/quorum

building upon the work done by Quorum and Ethereum which have built out a production hardened product. P2PMP will make a few core changes to the design, but keep core functionality in place in order to minimize code changes required to synchronize with future Ethereum and Quorum releases. The functionality goals for the project are as follows:

1. Design a new governance structure that optimizes open competition for all participants
2. Introduce a new byzantine fault tolerance (BFT) consensus algorithm called proof-of-merit (PoM)
3. Add customization for a native marketplace cryptocurrency and marketplace business logic

3 Transaction

Marketplace transactions usually involve the exchange of an asset for a store of value such as a currency. In our framework we want to ensure the authenticity of the assets being exchanged as well as securing the payment processing.

In order to do so we require a new proof field be added to the existing ethereum transaction structure. This added field, aptly labeled “proofOfExistence” is an array of executable steps needed to repeat the verification of the asset. For example, if the asset is a piece of information, it will need to be stored independently using a separate proof-of-existence blockchain service such as Factom. Proof-of-existence creates a unique hash of a digital asset and stores it in a blockchain for reference. The proofOfExistence array will hold the steps to reference the proof-of-existence.

The existing “gasprice”, “startgas”, and “value” fields will still exist but will take units of the newly defined currency instead of ether.

*Note: Verifying the proofOfExistence steps will be a manual process in the first version

```
var proof = [
     {Proof-of-existence verification Step 1},
      {Proof-of-existence verification Step 2},
      …
      {Proof-of-existence verification Step n},

];

var tx = {
      to: '0x0000000000000000000000000000000000000080', // address of
marketplace bidding contract
      data: '0x3b3d3de7a3271f0f6bea0b6b99e6f26a64dd3617', // Pointer to
asset
      proofOfExistence: proof,
      from: '0xd416324aa49f3a6db3e999ec823a7c38'

      value: 21000000 // This must match the original bid price from the
asset bid

};


web3.eth.sendTransaction(tx).then(function(receipt){
    ...
});
```

Figure 1: Example proofOfExistence Transaction

4 A Network Steward and the Genesis Block

The creator of each blockchain network will be called the Network Steward. This Steward will be creator of the initial block in the network, called a genesis block, and will be charged with creating and configuring the network using smart contracts defined by the block. These specific contracts will define the network currency, the consensus implementation, and define specific business rules to define the utility of the marketplace. Each contract will have a default contract that gets installed from the information in the genesis.json file, but it can be overwritten with customized smart contracts. The steward will be defined by 3 accounts which will all get an equal distribution of the initial supply created in the genesis block. These 3 account public keys

will be required to make all changes and create new smart contracts. All other smart contract creation attempts will get rejected.


```
{
  "alloc": {
    "0x0000000000000000000000000000000000000020": {
      "code": "[Code String Redacted]",
      "storage": {

"0x0000000000000000000000000000000000000000000000000000000000000001":
"0x02",

"0x0000000000000000000000000000000000000000000000000000000000000002":
"0x04"
      }
    },
"0x0000000000000000000000000000000000000040": {
      "code": "[Code String Redacted]",
      "storage": {

"0x0000000000000000000000000000000000000000000000000000000000000003":
"0x02",

"0x0000000000000000000000000000000000000000000000000000000000000004":
"0x04"
      }
    }
"0x0000000000000000000000000000000000000060": {
      "code": "[Code String Redacted]",
      "storage": {

"0x0000000000000000000000000000000000000000000000000000000000000005":
"0x02",

"0x0000000000000000000000000000000000000000000000000000000000000006":
"0x04"
      }
    }
"0x0000000000000000000000000000000000000080": {
      "code": "[Code String Redacted]",
      "storage": {

"0x0000000000000000000000000000000000000000000000000000000000000007":
"0x02",
```

```
"0x0000000000000000000000000000000000000000000000000000000000000008":
"0x04"
      }
    }
    "0xed9d02e382b34818e88b88a309c7fe71e65f419d": {
      "balance": "10000000"
    },
    "0xca843569e3427144cead5e4d5999a3d0ccf92b8e": {
      "balance": "10000000"
    },
    "0x0fbdc686b912d7722dc86510934589e0aaf3b55a": {
      "balance": "10000000"
    }
  },
  "coinbase": "0x3333333333333333333333333333333333333333",
  "config": {
    "homesteadBlock": 0,
    "isP2PMP":true,
    "currencyName":"Verit",

"steward":["0xed9d02e382b34818e88b88a309c7fe71e65f419d","0xca843569e3427144
cead5e4d5999a3d0ccf92b8e","0x0fbdc686b912d7722dc86510934589e0aaf3b55a"]
  },
  "difficulty": "0",
  "gasLimit": "2100000",
  "mixhash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "nonce": "0",
  "parentHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp": "0x00"
}
```

Figure 2: Example genesis.json configuration

4.1 The Network Currency Smart Contract

Direct ether transactions are turned off in Quorum and therefore will not be available in our framework. Therefore the majority of the transactions on the system will be transacting with the smart contracts defined in the genesis block.

The currency smart contract defines the name of the currency, its initial supply, the gas cost formula and the mining reward. This contract has built-in functions that allow the Steward to

adjust gas cost and mining reward equations. These changes can only be completed with a request sent signed by at least 2 of the 3 steward accounts.

This smart contract will also manage the account balances for the associated cryptocurrency.

4.2 The Validation Rules Smart Contract

Asset The custom validation rules for asset verification will be defined here. Otherwise the default fact based information proof will be used. This contract will also have the array of current validator keys, that can be adjusted by a multisig update by the steward accounts. The rules and logic for automatically obtaining and removing validators can also be set inside this smart contract. By default new validators can only be added by the Steward, and automatic removal will happen when accuracy score drops below 75%.

4.3 The Voting Consensus Smart Contract

This smart contract will define the nodes that currently have the permissions to vote for next block consensus, called Validators. The consensus algorithm will be defined and accessed through this smart contract. Additional functions will be built into the smart contract for the steward to manage the list of Validators.

4.4 The Marketplace Business Rules Smart Contract

This smart contract defines the rules for asset transactions and manage the buy and sell bidding system for the marketplace. It can also handle other business logic rules that me need to be applied to transaction data.

5 Merit Consensus

In distributed peer-to-peer systems, with multiple copies of the data controlled by separate parties, a consensus method needs to be established to ensure the integrity and consistency of the network data. When data from two peers differs, the other peers need an algorithm to guide them in determining which dataset to choose as the accurate one. Specifically in a blockchain, peers need to determine block order in a chain.

A practical solution to the consensus problem is to allow the different blockchain peers to vote for the block order they think is accurate, and then require some percentage threshold to be reached amongst the population for consensus to be reached. A pure democratic voting system is slow to scale and is very susceptible to corruption if the personal incentive to act negatively outways acting in the best interest of the system. These risks become greater when users can vote anonymously, as is the case with most blockchain networks. For this reason consensus algorithms need to be more nuanced.

Quorum take the authoritative approach with its QuorumChain consensus algorithm. It works by assigning voting rights to a few non-anonymous authoritative nodes that it trusts, and only these nodes can vote on the next block to be added to the chain.

A new smart economy blockchain called Neo, takes an approach that more closely resembles a representative democracy. Their consensus algorithm called delegated byzantine fault tolerance (dBFT). The anonymous voting peers in the dBFT algorithm elect delegates to represent their vote. These delegates are special nodes in the ecosystem. The network participants can change their delegate for every new block that needs to be added. Every time a new block needs to be written to the chain, one block is randomly selected to share their version of the new chain. If 66% of the other delegates agree with the validity of this block than that block gains consensus, otherwise a new random delegate is selected and the process is repeated. This model ensures that as long as at least ⅓ of the delegates are acting in good faith, then the integrity network should remain safe.

Popular public blockchain networks Bitcoin and Ethereum, use a different algorithmic approach to consensus called proof-of-work. A proof-of-work system allows anyone to participate as a peer anonymously and add new data blocks to their own version of the chain. However, doing so requires solving a hard guess and check math problem that requires a lot of computer processing power to be expended in the process. This process is called mining and it introduces the idea of upfront work to the consensus process. In order to reward the block creator for doing the work, the network creates a new amount of cryptocurrency tokens and then grants these tokens to the block creator. This is analogous to mining for gold, hence the term of phrase.

Peers are then told to vote for the longest chain, which should theoretically have taken the most work. The positive incentive along with work influenced voting rules creates a system in which it is easier to personally profit from positive work on the network rather than negative work that could compromise the system. This model has proven to be reliable and secure, but scales poorly because of slow transaction times.

A functional marketplace requires fast transaction times, trust in the value exchange system and also the promise of fair competition to buy and sell value. In order to achieve these goals we are combining an authoritative voting system with the idea of work incentives and merit based reputation in a new consensus algorithm called proof-of-merit(PoM).

Proof-of-merit has special nodes charged with voting for consensus. These special nodes are called Validators. Validators are tasked with verifying assets and creating the repeatable proof steps in a transaction. All other nodes are considered Maker nodes. Maker nodes can initiate new asset bids as well as respond to buy requests with a matching asset for exchange. The maker node that responds to a bid with an asset transaction can provide proof steps for the Validator in order to speed up the verification process. The initial proof array can also be left blank for the Validator to fill in, but this will severely slow down the processing time for that transaction.

Once one or more transaction is packaged in a block, the special node submits that block for vote to be written as the next block on the chain. The other Validators then validate the structure of all of the block transactions and submit a yes or no vote. If a majority of validators consensus is reached, the maker that initiated the transaction can sign the block as the maker node and the other nodes in the network will adopt their version of the blockchain as the truth. The new block will create a reward in the cryptocurrency that is native to that block. In the event of a tie, Validators will be told to vote for the block with the largest sum of transaction values within the block.

This process is equivalent to mining in the PoW algorithm, and the Validator node that originally provided the proof and created the block as well as the maker that creates the block will share the mined incentive.

In order to protect marketplace competition for all participants, there will be two more safeguards built into the algorithm. First, Validators proof accuracy and voting history and voting participation over time and will be algorithmically combined into a reputation score. This can be done programmatically within a smart contract or manually off the chain by the Steward. Either way, this reputation of Validators will dictate their merit for remaining in the voting position of power. The steward will control the marketplaces standards and rules for removing Validators, but the reputation score will be the driving factor.

Secondly, the Steward cannot participate as a Validator, or profit on the network from anything other than transaction fees. This will help prevent Stewards from garnering too much short term profit incentive that could negatively affect the competitive balance of the ecosystem.

6 Miner Adjustments

Quorum has removed mining capabilities so for our project we will need to add them back in. However, we do not want to add ether back in as the mining reward. We want to make sure that the miner looks for the currency defined in the genesis block currency smart contract, as well as

creating the mining reward based on settings in the currency smart contract defined in the genesis block. If that smart contract is not defined, we will make sure to throw an error so that no mining can be done on that version of the network.

7 Applications

The first obvious application for this framework is to build an information marketplace to replace the large centralized information bureaus.

The pilot product released with the framework by Ex Fida Bona will be the Compliance of Everything Network. This will be an application of an information marketplace that will add extra business logic in order to incorporate a data structure to help simplify the complexity of complicated compliance problems.

This framework could be used by any potential marketplace that can define its assets or utility in way such that the information can be verified and then continually re-validated.

Optionally because the network will be based on Quorum, a marketplace can have private transactions as well as public. In the case of private transactions, a consensus algorithm is not used, and the private state of transactions is stored separately from the consensus public state. Also, because consensus is not used for private transactions, rewards are not earned, and any third party asset verification required will have to paid purely through higher transaction fees.

8 Conclusion

Blockchain is the perfect medium for decentralized marketplaces, and Quorums modifications to the go-ethereum project get us most of the way to where we can build a model that can be widely used by marketplace entrepreneurs that want more control over their own economy. In order to make our product flexible enough to be configured to the marketplace creators needs, while doing our best to create a standard set of rules for open marketplace competition with a cryptocurrency.