DonorCoin: A Semi-Centralized Cryptocurrency for Donations

Srihari Radhakrishnan∗
Sudarshan Balaji†
Deniz Polat‡

Reference Format:

1 INTRODUCTION

In the past few decades alone, humanitarian aid in the form of food, money and medical aid worth hundreds of billions of dollars have been invested in nations suffering from war, poverty or natural disasters. However, not all the donations reach their intended recipients [2], for many reasons such as corruption, fraudulent charities, and siphoning the aid through unnecessary "administrative tasks". Part of the reason why fraud is difficult to detect and control in this setting, is that aid is difficult to track at the "ground level". Once disbursed, the aid money is easy for middlemen (corrupt officials, merchants, etc.) to misuse, since fiat or electronic currency is difficult to track once out of the donor’s "jurisdiction". In the more insidious cases, charities can end up doing more harm than good, with humanitarian donations being one of the largest sources of funding for terrorist organizations [7].

Modern cryptocurrencies have revolutionized the way we conduct transactions by offering a means to publicly inspect the legitimacy of transactions while protecting the identities of users. We observe that cryptocurrencies can be the building block for a transactional system that can help track donations and humanitarian aid transparently and anonymously with cleverly design incentives for all principals in the system, thereby minimizing fraud.

Existing cryptocurrencies are designed to function in a fully decentralized setting, with no trust assumptions among the principals of the system. Decentralized cryptocurrencies are built upon blockchains, a powerful primitive to ensure accountability in a trust-less system. Due to this inherent lack of trust, decentralized cryptocurrencies trade performance for correctness, and use complex checks and balances (Proof-of-Work [11], Proof-of-Stake [1]) to ensure that all principals in the system act fairly. These checks, while completely necessary in the trust-less setting, introduce a high degree of latency in the system, and bottleneck its throughput as well. In this work, we restrict our interest to a class of cryptocurrency applications which operate in a more trusted setting than Bitcoin and existing cryptocurrencies, and consequently do not require such stringent checks and balances. We propose DonorCoin, a semi-centralized crypto-currency system to work in a distributed setting with concrete trust assumptions. Additionally, we motivate and explore the design for such a system, and use the Ethereum and Solidity framework to illustrate the expressibility of smart contracts in building a limited-scope prototype of DonorCoin.

2 MOTIVATION

The design of DonorCoin is motivated by the following:

Evidence of good use: Over the years, hundreds of billions of dollars, both in terms of money and food/medical aid have been poured into African governments towards war relief, poverty and other humanitarian causes. However, a significant portion of this aid is siphoned away by local regimes[2] for personal and political gain, leaving little aid for its intended use. This is a key issue DonorCoin intends to tackle - the system seeks to provide evidence of good use of monetary funds; for example, we must be able to show evidence of war relief funds going towards rehabilitation of victims, instead of being misappropriated.

Convenience: Recently, mobile person-to-person payment systems such as M-Pesa [4], introduced by Vodafone in Kenya and Tanzania, have emerged as an effective and easy method of payment. Over $10 billion in transactions between July and September 2016 took place via mobile payment systems such as M-Pesa [3]. As of 2017, there are over 31 million
subscribers in various forms of mobile money services in Kenya, of which 20.7 million (66%) are M-Pesa subscribers [3]. DonorCoin seeks to provide a convenient and secure payment system that integrates well into existing methods of mobile payment.

**Statistical analysis**: The system needs to provide means for donors to obtain statistics on how donations were spent in order to better reallocate funds, while maintaining privacy guarantees for the principals involved (i.e., merchants, customers, etc.). This is doable if a public ledger (blockchain) model similar to Bitcoin is used.

### 3 BITCOIN PRIMER

In designing a new cryptocurrency system, it is worthwhile to look at existing systems deployed in the real-world. The most common cryptocurrency in use is Bitcoin—a digital currency running over an electronic payment system, which can be used to buy goods and services.

Bitcoin is a fully decentralized peer-to-peer network of machines running a family of distributed consensus protocols that assure security and anonymity of transactions. These properties make Bitcoin (and cryptocurrency in general) attractive as a means of carrying out transactions. In traditional payment systems such as credit cards and privatized banking transactions, the identities of the parties involved (buyer, seller, and financial institutions) are linked to each individual transaction, and typically involve additional fees per transaction. In Bitcoin, however, new accounts (called addresses, a hash of the user’s public key) are minted for each transaction, and though transactions are available in the public ledger (blockchain) for all parties to view, the claim offered [11] is that the ledger does not reveal user identities of principals involved in any transaction.

#### 3.1 Transactions

All state in Bitcoin is represented as a series of transactions between principals. A transaction is intended to transfer currency from one principal to another, possibly in exchange for goods and/or services. It is important to note that there is no hidden or meta state in Bitcoin that reveals user identities or links transactions to principals (these do not ensure total anonymity or privacy, as we shall see in section 6.2).

A transaction is represented as a string of bits, which is a hash of an array of inputs and outputs. This string of bits is unique in the system, and serves as the transaction’s unique ID in the system. Transactions flow from one address to the next, where each address is defined as a hash of a public key created by a principal that owns the address. This address serves as a pseudonym for a user to conduct transactions without revealing their identity. Users are allowed to create as many addresses as they want, and are encouraged to created a new one per transaction.

In order to ensure the consistency of the number of “coins” in the system, each transaction’s input is a previous transaction signed by its owner using their private key(a hash of the coins the owner received) combined with the public key of the next owner. In this way, each transaction references a previous one, forming a chain which can be verified using the public keys embedded in the chain.

#### 3.2 Consensus

The main contribution of Bitcoin is a solution to the Byzantine consensus problem using clever incentives built into the system. Transactions are added to a global, public ledger known as the blockchain. Any party in the system can add to the blockchain by forming a block of a set of valid pending transactions, and solving a computationally hard proof-of-work puzzle to determine if their block gets added to the blockchain.

The process of adding blocks to blockchains is important for two reasons: first, it determines if transactions are valid, and can be part of the blockchain and second, this process is how money is “pumped” into the Bitcoin system.

The fundamental principle is simple - principals solving the proof-of-work puzzle (known as miners) race to finish and publish the new block with the solution to the network. The first block to contain the solution is considered correct. The transactions in the block are verified by the other participants, and work on the next “valid” block begins. When a new valid block has been formed, referencing the old one, the latter can be considered part of the blockchain. In the case of blocks formed concurrently, there will be a fork in the blockchain, and the longest chain (eventually guaranteed to form due to the random nature of block generation) is considered the “valid” chain. The other chains/forks are rolled-back, and the transactions contained within them nullified.

When a proof-of-work puzzle is solved, a miner includes a special transaction, a coin generation transaction, which is the miner’s reward for creating the valid block. This is also included in the block generated by the miner. This reward incentivizes miners to work on valid blocks, since invalid ones are rejected by its peers (and therefore will not be part of the longest chain).

#### 3.3 Networking

Bitcoin is a decentralized peer-to-peer broadcast network which uses gossip protocols to announce new transactions. New blocks and pending transactions are flooded to all the neighbouring nodes, which flood their neighbours, and so on.
The underlying network has important implications for the design of the consensus protocol: high latency networks can increase the possibility of new blocks being discovered late (or not at all), thereby causing more forks in the network. For this reason, the block creation time has been set to 10 minutes in the original design, with each block carrying a few hundred transactions (at the time of writing).

Therefore, we see that while Bitcoin provides decentralized service, there are trade-offs made to maintain this decentralization of power - block creation time vs block-size is one such trade-off. While increasing block size can increase the throughput of transactions, it slows down block creation. On the other hand, lowering block creation time can increase the number of invalid blocks in the network, causing instability and more forking in the blockchain.

### 3.4 Smart Contracts

We use Ethereum instead of Bitcoin in our prototype implementation of DonorCoin (Sec. 5) because Ethereum offers a powerful declarative primitive – smart contracts. A smart contract is a piece of code expressing an agreement between principals in the distributed currency system. The code and the agreements contained therein exist across the decentralized blockchain, and are executed on it. Smart contracts permit trusted transactions and agreements to be carried out among the principals without the need for a central authority, legal system, or external enforcement mechanism, similar to the way cryptocurrency flows through Bitcoin.

In a smart contract approach, an asset or currency is expressed as code, and the code validates a condition and it automatically determines whether the transaction should proceed or abort. In the meantime, the decentralized public blockchain (ledger) stores and replicates the code, which gives it a certain security and immutability.

### 3.5 Control

For a system such as DonorCoin, we require not just privacy and anonymity guarantees for principals, but also a certain degree of control over fraudulent activity, as described in section 2. Even in a completely decentralized system such as Bitcoin, there have been a few proposals to keep fraud in check - for example, green addresses were introduced as a method to use trusted third parties (whose IPs and identities are made publicly known) and trusted key-pairs to facilitate transactions. The Bitcoin community, however, considers this proposal not advisable for implementation, since it draws away from the decentralized model, leading to trust anchors in the system which can act as a single point of attack or failure. In DonorCoin, the donor is the central point of trust (trust anchor) in the system, and is expected to act honestly. Though it is a single point of failure, we do not consider malicious donors in this treatment, and leave it for future work.

### 4 DONORCOIN PROPERTIES

We propose a crypto-currency system, DonorCoin, in an ecosystem defined by the following characteristics and assumptions:

**Semi-centralized**: Unlike existing crypto-currency systems which are completely decentralized and generate currency from within, DonorCoin pumps currency into the system from large principals such as foundations or NGOs, and disseminates it to other principals such as governments or humanitarian organizations, or other middlemen offering goods and services (hereby referred to as merchants). The end users of the currency exchange it for goods or services from these merchants. Transactions are executed by users, but the currency is disseminated from a central entity (a single node or cluster of trusted nodes), referred to as the donor.

**Semi-anonymous**: Existing crypto-currencies boast complete anonymity. DonorCoin assumes partial anonymity, whereby the identities of the principals carrying out transactions are kept anonymous, but sufficient information is available to make inferences about how money is spent.

**Trust**: It is assumed that there is poor adherence to legal contracts or good-faith agreements between the donors and merchants; donors can register users after verifying them (externally to the system). Donors honor transactions initiated by registered users, and reject untrusted user transactions. It is assumed that the donor is not malicious, and therefore will not compromise the user’s account.

**Incentives**: This property follows from the trust assumptions above - merchants are incentivized to act honestly; each transaction conducted by a merchant carries with it a reward, which a merchant can collect once the transaction is verified as being legitimate by the donor. Verification happens in two steps – first, the transaction is posted on DonorCoin’s public ledger. Then the end-user initiates a confirmation transaction once they receive the goods/services from the merchant, along with proof of receipt, which is also posted on the chain. In addition, end-users can only use the currency towards goods/services “registered” with the system. This is an essential element of control in the system, and ensures that the currency cannot be (easily) exchanged for other forms of payment such as fiat currency, and that such exchanges are explicitly trackable. This is vital to preventing misappropriation and abuse of funds. For instance, merchants cannot issue “fake” receipts to items that are not registered.
Poor infrastructure: Existing crypto-currency systems such as Bitcoin require an unusually high amount of computational power, both for miners and clients. The targeted ecosystem is assumed to have wimpy client nodes, both in terms of networking and computational bandwidth. Therefore, a peer-to-peer system (such as Bitcoin) where all principals (except miners) perform roughly the same amount of computational work may not be feasible. Since the donor executes contracts, verifies transactions and mines most of the currency (see sec. 5), the user and merchant nodes do not perform heavy mining tasks, like the Proof-of-Work or Proof-of-Stake required in Bitcoin or Ethereum.

5 PROTOTYPE

Setup: There are three principals in DonorCoin – the donor, merchant and user. Each entity operates on a locally running private blockchain with different visibility to the contents of the blockchain. The blockchain can be accessed by peers locally using interprocess communication (IPC) or remote procedure calls (RPC). Note that this blockchain is separate from the public Ethereum blockchain. We make use of GoLang Ethereum (geth) console to initialize our blockchain by creating a genesis block with the required parameters. These parameters, for instance, dictate the mining difficulty or the maximum amount of gas that can be spent on running a transaction (often decided by the miners in the network). Once the blockchain is initialized with the necessary parameters, three accounts are created for the different principals. Smart contracts (written in Solidity) manage the account balances (state) for each principal and provide a set of functions to manipulate the state of variables on the blockchain through transactions. In order to execute transactions and contracts on the blockchain, principals require gas. Gas is the execution fee for every operation made on Ethereum. Its price is expressed in ether and it is decided by the miners, which can refuse to process transaction with less than a certain gas price. We set the gas price, mining difficulty and other parameters when we setup our blockchain initially. Principals can view and/or edit the state of certain variables (assuming they have visibility and permissions to do so), using gas. It is also possible to retrieve the value of certain variables through their local copy of the chain, without having to spend any gas.

Transaction: The contract is then deployed on the chain by the donor and this returns the address at which the contract is made publicly available. Contracts can only be executed by the donor – this is ensured using a simple check in the contract logic. The miner network is active in the background, which is required to deploy contracts or accumulate gas (ether) to initiate transactions. While deploying the contract, the donor also adds an initial sum of currency to its account through a constructor of the contract. The donor then transfers some currency to the user by initiating a transaction. Once this transaction is validated by the donor using callbacks (or optionally events), the user proceeds to issue another transaction to the merchant to make a purchase. At this point, the user has currency but no Ether. Without ether, the user cannot use any gas. So, the user mines for some ether prior to launching the transaction. The contract moves the requested merchandise into the account (cart) of the respective user and fires a completion event. The merchant can optionally embed a receipt for the goods, including shipping information and other data. In order to make it so that the user account data and transaction is not visible to other merchants and users, the block can be encrypted with the donor’s public key. This way, the donor can verify the transaction, but no one else can read its details. This ensures privacy of users and merchants involved in each transaction. The completion event fired by the merchant indicates to the donor that the transaction has finished.

In order to identify the participants of the transaction and verify the amounts involved, the donor retrieves the context of the trade from the event fired from the contract. Since callbacks are asynchronous, it is possible to take note of the contract address and check the balances of every principal. The donor, upon verifying the legitimacy of the trade, further initiates two transactions – one to deduct the purchase amount from the user’s account and the other to transfer the purchase amount plus some additional reward to the merchant account. Additionally, in order to ensure that the user has sufficient balance for a purchase, the donor can run a check in the contract that aborts the transaction if funds are unavailable. This way the merchant never has access to an individual user’s account balance details.

The task of deploying the contract is placed with the donor (the trusted third party). The peer which deploys the contract becomes the owner of the contract, and this ownership is stored in a variable on the contract. The owner can then create a whitelist of peers who can initiate certain transactions. This prevents non-trusted members from initiating any transaction, and deploying contracts on the blockchain.

6 FUTURE WORK

While our project is a good first step in determining if smart contracts are expressible enough to describe the interactions between principals in our system, there are few challenges that emerge:

6.1 Collusion/Coercion

A fundamental assumption in our design is that end-users are honest. This is not an unreasonable assumption to make, given that the end-users of DonorCoin will likely be victims
of poverty, war, crime or natural disasters, or in some cases soldiers participating in an active war requiring arms – it is therefore in their own interest to act honestly. However, it is possible for merchants to collude with users to generate fake transactions, and siphon the currency or goods. Siphoning the currency is partly mitigated by restricting the use of the crypto-currency towards pre-registered goods/services, and making the currency non-redeemable outside the system. Additionally, by maintaining a record of trusted users, the donor can decline a transaction initiated by a non-authorized user. However, merchants could still profit from siphoning the goods themselves, especially if the goods are arms or rations. Even if users stay honest, merchants could apply coercion to make users act in their interest. Our approach to this issue is optimistic, assuming that the reward system and public ledger are incentive enough for the merchants to stay honest. However, depending on the volume of goods a merchant or group of merchants could siphon, the reward might not be sufficient motivation for honest behaviour. We leave the investigation of cleverer merchant incentives as future work.

6.2 Privacy & Anonymity Concerns

Though user and merchant identities are kept private and anonymous (to everyone except the donor), we will see that it is possible to reverse-engineer some aspects of the system to deanonymize users. As discussed in Section 3.1, cryptocurrencies assure privacy through cryptographically generated and hashed keys in the form of pseudonymous addresses. Addresses are pseudonymous, not anonymous, since they are publicly available, but are not tied to the principal (user or IP address) that generated them. However, recent work [10, 12] has shown that bitcoin privacy guarantees are not absolute, and can be compromised with both heuristic and rigorous analyses of the public ledger and associated transaction metadata. What follows is a brief discussion of privacy and deanonymization threats that DonorCoin will need to solve, in order to keep principals’ activity and identities safe from untrusted entities (everyone except the donor).

6.2.1 Threats. Though Bitcoin does not maintain an internal mapping of user (or IP address) to transactions or any other identifying transaction metadata [8], it is possible for attackers to link transactions with offline metadata such as shipping addresses, emails and obtain specific information about the user’s identity. Since the complete history of a particular Bitcoin (or satoshi, the atomic unit of Bitcoin currency) is publicly available, a reasonably performing blockchain-analysis algorithm could, in principle, help de-anonymize Bitcoin users. Recent work has explored methods to de-anonymize Bitcoin users and user-groups, some of which are described below.

**Idioms of Use Heuristics**: Recent work [10] has shown that Bitcoin users can be deanonymized to a certain extent based on certain *idioms of use* which are implementation dependent. For example, a customer might need to pay a merchant using multiple accounts that he owns (this is standard practice in Bitcoin), and so multiple accounts can be linked back to a single user, given the usage pattern that multiple users rarely are involved in a joint or linked transaction.

Another idiom of use exploited by Meiklejohn et al. [10] is the use of *change addresses*. Change addresses are addresses created due to a peculiarity in the Bitcoin system - a principal is not allowed to spend part of the bitcoin contained in an address. Therefore, payers typically split the money from an address (or group of addresses) between the merchant and themselves, storing the change from a transaction in a so-called change address. Currently, a change address can be identified by the following properties: it has only one input, and has no output (i.e., is never re-used by the client that issued it).

Once accounts have been clustered as belonging to a user, the next step (linking accounts to real users) was also managed by Meiklejohn et al. [10] by interacting with them directly. For instance, entities such as merchants, online wallets and conversion services sometimes post their IP addresses or leak it in transactions. Using this with cluster information from heuristic analysis methods above, it might be possible to discover entire clusters of accounts belonging to a principal. The authors suggest that identifying other regular users of the system might be easy for authorities that can subpoena a central service like an online wallet, which requires to map addresses to customer identities.

Heuristics such as the ones used in [10] are implementation dependent, and could be rendered irrelevant if the usage patterns change, but there is an important lesson here – the privacy of users in a system such as Bitcoin or DonorCoin is far from guaranteed, and there can be several ways to make inferences regarding user identity in the system. These need to be accounted for when designing the system.

**Network Anonymity**: There are privacy concerns in the network layer as well – nodes can leak IP addresses in transactions, and recent work by Biryukov et al. [5] has shown that even Bitcoin layered over Tor can be compromised to launch a DoS attack. Therefore, it remains to be seen whether Bitcoin’s networking layer will evolve better privacy and anonymity.
6.2.2 Proposed solutions. **Coin Mixing**: In this approach, a collection of users create a joint set of transactions, randomizing ownership of coins, and therefore making the members of the group impervious to privacy attacks that exploit certain idioms of use. For example, users of a group could form a single transaction, where inputs and outputs are controlled by a single user each, resulting in external anonymity for the members of the group [9]. However, this implies a certain degree of trust within the group which is important to understand - this can be managed with certain incentives and the use of trusted third parties [6] to discourage adversarial behaviour within the group.

7 CONCLUSION

In this project, we propose DonorCoin, a new crypto-currency system to operate in a semi-centralized, distributed environment with concrete trust assumptions. We use Ethereum and Solidity to build a limited-scope prototype of DonorCoin, and demonstrate the feasibility of such a system with smart contracts as a building block. Additionally, we identify design constraints and challenges of building such a system on a large scale for practical use.

REFERENCES


