Decentralized Transaction Clearing Beyond Blockchains

Fabio Massacci[a], Chan Nam Ngo[a,*], Julian Williams[b]

[a]*Department of Information Engineering and Computer Science, University of Trento, Trento, Italy.*
[b]*Durham University Business School, Durham, UK.*

Abstract

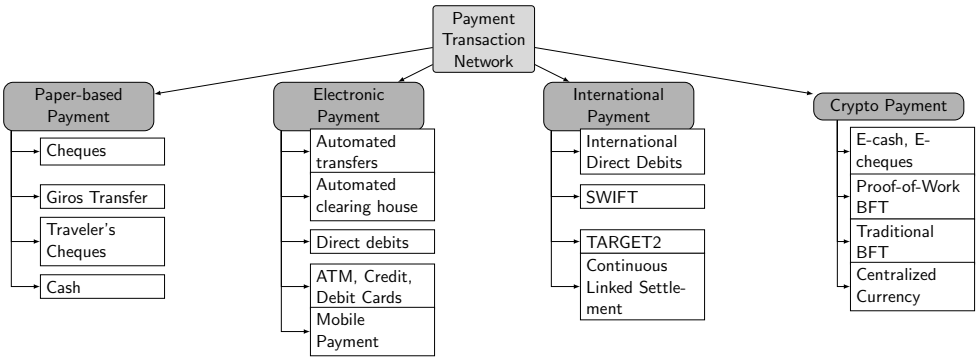
Blockchains and Byzantine Fault Tolerance form the basis of decentralized currencies and ledgers such as Bitcoin, Ripple, Hyperledger, and RSCoin. A large slate of literature has focused on the currency aspects (e.g. anonymity, independence from central banks, etc.). We argue that, as-far-as Distributed Payment Transactions Networks (PTNs) are concerned, there are other, possibly more interesting, properties. This paper provides a systematic review of both traditional PTNs and their analogues in decentralized ledgers and associates different technological features to the corresponding business and financial requirements. We provide a conceptual classification of the key properties (value creation, payment promise, transaction realization, and value storage). We map existing (distributed) PTNs into the classification showing different alternatives are possible. Furthermore, the ideas behind distributed ledgers can be extended beyond payments and contracts. We illustrate the idea of *derivatives-contracts-as-programs* that are marked to market (or an account that is margined) automatically by computations run on, and whose ownership transitions are recorded, in a distributed payment network.

Keywords: Blockchain, Byzantine Fault Tolerance, Distributed Payment Transaction Network, Smart Contract, Derivative Contracts As Programs
CCS Concepts: Applied computing → Digital cash, Secure online transactions, Computer systems organization → Dependable and fault-tolerant systems and networks, Distributed architectures, Peer-to-peer architectures, Security and privacy → Distributed systems security, Pseudonymity, anonymity and untraceability
JEL Classification: E42 Monetary Systems and Payment Systems, G23 Non-bank Financial Institutions, Financial Instruments, and Institutional Investors, G13 Contingent Pricing and Futures Pricing, L86 Information and Internet Services and Computer Software, O33 Technological Change: Choices and Consequences, Diffusion Processes

1. Introduction

The current discussion on distributed systems providing transactions mediated by electronic communications networks has focused primarily on two areas: decentralized currencies such as

*Corresponding Author
Email addresses: fabio.massacci@unitn.it (Fabio Massacci), channam.ngo@unitn.it (Chan Nam Ngo), julian.williams@durham.ac.uk (Julian Williams).
Submitted for publication. Comments welcomed. Cite as F. Massacci, C.N. Ngo and J. Williams, Decentralized Transaction Clearing Beyond Blockchains. Available at SSRN: http://ssrn.com/abstract=2794913



Payment Transaction Networks can be classified into four diagonal sub-groups. Domestic payment includes "Paper-based" and "Electronic-based Payment". International Payment is another sub-group while the Crypto-based Payment is borderless. BFT stands for "Byzantine Fault Tolerance".

Figure 1: Payment Transaction Network Classification

cryptocurrencies on one side and the traditional provision of trading platforms on the other side, such as the US National Market System (where multiple exchanges are networked together) and the European Target 2 Securities platform (a centralized platform).

Figure 1 illustrates the classification of PTNs. In Table 1, we divide the Crypto PTNs into four sub-categories. E-cash/cheques is simply a token exchange from fiat currency. PoW (Proof-of-Work) BFT (for Byzantine Fault Tolerance) are the digital currency systems that solve agreement in a decentralized settings with PoW. Ripple and Hyperledger solve agreement with traditional BFT algorithms. Finally, different from most cryptocurrency, RSCoin is a centralized currency that delegate the currency creation and blockchain management to a central bank.

So far, most attention has been devoted to the "currency" and cryptographic aspects of decentralized PTNs (see for example survey [6]) with either enthusiasts (claiming democratization), or vibrant detractors (claiming collusion with money laundering). This article seeks to provide a different viewpoint. To this extent we start by a systematic review of deployed PTNs, whether centralized or distributed, traditional or digital which are dissected into their engineering components. By linking each technological component to a possible business objectives of PTNs it emerges clearly that some choices are not mandatory and several interesting new combinations are possible to produce new systems for the broad variety of financial services. One of the key aspects of our analysis will be an understanding of how a Distributed PTN can be (or in many cases has already been) re-tooled for different tasks.

For example, one of the critical aspects of building a Distributed PTN is its implementation of consensus building within a distributed system. A Distributed PTN requires agreement and consistency across the nodes on how transactions are initiated and cleared. If it can be tricked into presenting different transaction records to different parties then (a) parties within the system will be vulnerable to fraud and (b) persistent failures will lead to the discontinuity of the system as the PTN collapses. Hence, the ability to demonstrate consensus is arguably far more important than the demonstration of crypto effort to show the validity of one's currency, the anonymity of payers and payees, or the exchangeability of digital currencies with existing fiat currencies or commodities such as gold.

Table 1: Crypto Payment Transaction Networks

Among all listed PTNs, E-cash, Bitcoin, ZeroCoin, Ripple, RSCoin will be selected to demonstrate different features in §(4).

Categories	**Example**
E-cash/cheques	E-cash, E-cheques
PoW BFT	B-money, RPOW, BitGold, Bitcoin, ZeroCoin, Dogecoin, Dash, Primecoin, Ethereum
Traditional BFT	Ripple, Hyperledger
Centralized	RSCoin

These are classical Byzantine fault tolerant (BFT) problems, well studied in distributed system. The transition from impossibility results in general BFT approaches (see [24, 18]), to a practical BFT approach relies heavily on the assumptions of the type and structure of the adversary seeking to subvert the consensus (and hence in our case defraud the ledger) and the effectiveness of cryptographic techniques in facilitating the conveyance of messages between nodes in the distributed system. The applicability of those assumptions to a traditional PTN is what makes possible the transformation of a traditional PTN with a handful of trusted critical third parties to a distributed PTN with less trusted and/or less critical third parties. Many of the current implementations of distributed ledgers (such as Hyperledger) already implicitly or explicitly utilize the essentials of the practical BFT algorithms in their operation.

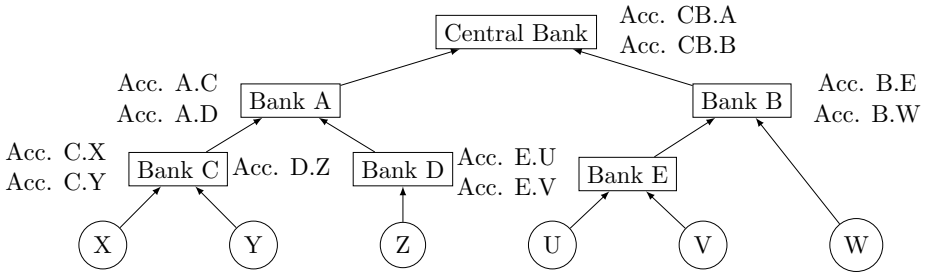
To illustrate the decomposition of both the engineering structures and requirements of transaction systems, we provide a summary of how various transaction systems operate, categorize primitive elements, and finally map them into the business requirements of PTNs. Such comparative analysis has an immediate advantage: even a cursory look at the Tables in Section 4 clearly shows that the very same business requirement can be met in different ways and by involving different actors; slightly different combinations among rows or columns can yield different systems.

Furthermore, we illustrate the idea of *derivatives-contracts-as-programs* that are marked to market (or an account that is margined) automatically by computations run on, and whose ownership transitions are recorded, in a distributed payment network.

The remainder of this paper is organized as follows: §(2) provides a review of classic non-distributed payment systems. Additionally, we will introduce the electronic communications network as payment systems and then provide a short overview of the development of these systems. §(3) provides a non-exhaustive overview of the current cryptopayment universe, whilst §(4) presents our decomposition of the features and reviews the critical engineering issues associated with them. §(5) and §(6) provide a summary of the threat environment and how nodes may be used by malicious attackers to subvert, destroy or commit fraud on the network. Finally, in §(7) and §(8) we postulate some ideas for the future. Critically, we outline how a distributed ledger might be used in an active and automated processing mechanism to replicate more complex markets, we use a futures market as an example, we also utilize this section to summarize our arguments and conclude the paper. §(9) then provides a brief summary.

2. A primer on Traditional Payment Transaction Networks

In most PTNs without physical cash transfers, the payer gives the payee a cheque (we use the Anglo- French spelling to avoid confusion with our use of the work "check" in the sense of



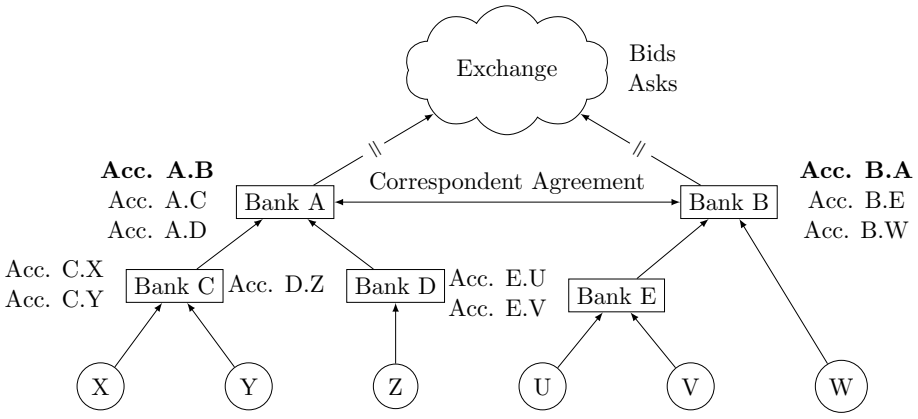
The transactions between X and Y only need Bank C. In the case of the transactions between X and Z, Bank A needs to settle the account A.C and A.D to transfer money from Bank C to Bank D and Bank C will deduce the account C.X while Bank D credit the account D.Z. More sophisticated transaction between Y and V will involve more banks, including the Central Bank, and thus accrue more transaction fees.

Figure 2: The tiered-banking model

verification), with which the payee can withdraw cash from the payer's bank or lodge the cheque at the payee's own bank so that the funds eventually are transferred to the payee's account from the payer's. When a payee lodges the cheque, the funds will usually be credited after some time and the cheque will be sent to a clearing house where the payer's bank will validate the cheque. If the cheque is correct, e.g. funds available and signature matched, the payer's bank will debit the amount written on the cheque to the payer's account once the two banks've settled the amount by crediting and debiting the respective accounts at their central bank. An invalid cheque will be returned and the payee will not be able to acquire the funds. In cheque payment, the payee suffers from such reversible payment and usually is on the disadvantage side. Electronic transfer essentially eliminates the physical transfer of cash and their risk of both delayed authentication, e.g. signature matched, and delay authorization, e.g. funds available.

Figure 2 illustrates the tiered-banking model. There exists a central bank on the top of the hierarchy whereas below are multiple levels of dependent banks and the clients at the lowest level. The central bank may also control the monetary supply of the currency and its dependent banks have to hold an account in its ledger. The clients are at the leaves of the hierarchy. In the correspondent-banking model of Figure 3, the banks A and B form direct relationship with each other, possibly through an exchange, so that they can settle the transactions without a third party. In the case of exchange, a single bank may have accounts in both currencies and will convert the currencies and the clients suffer from some fees due to currency conversion. Alternatively, both of the payer and payee's banks may have a correspondence account with a *multinational bank* and the exchange is then conducted as an internal transfer by the *multinational bank*[1]. In real world scenarios, the transaction may go through multiple levels and two central banks that have correspondent agreements, thus combining Figure 2 and Figure 3. The more third parties the money flows through, the more transaction fees final clients have to pay.

[1] A correspondence account held by a banking institution to receive deposits from, make payments on behalf of, or handle other financial transactions for another financial institution, usually overseas. Commonly, these are referred to as "nostro" and "vostro" accounts in international finance. A "nostro" account is our account of our money, held by the other bank. A "vostro" is our account of the other bank money, held by us.



Bank A and Bank B hold an account of each other in their ledger, A.B and B.A respectively. The transactions between X and V will need the cooperation of both Bank A and B to settle the amount. Obviously, the settlement also needs to be propagated down to Bank C and Band E. The transactions could also involve currency exchange. In this case, Bank A and Bank B will go through the Bid/Ask orders to sort out the exchange.

Figure 3: The correspondent-banking model

2.1. The Development of Traditional Digital PTNs

Traditional electronic payment systems typically either replace the physical form of direct transfer from payer to payee, such as a cash payment, i.e. E-cash[9] and electronic cheque [1], or indirect form of transfer, i.e cheque payment (Debit Cards and Credit Cards). The previous form is usually classified as pay-now system while the latter is described as pay-later. PayPal offers the capability to process payment via the Internet by using the client's cards detail. Recently, innovations in payment technologies provides wrappers to existing payment infrastructures, e.g Google Wallet and Apply Pay which allows user store their cards information inside a phone to use for payments.

Payments across intermediaries are not necessarily instantaneous. In the *Differed Net-Settlement System*, the bank intermediaries may settle the outstanding differences at the end of the day. A Real-Time Gross-Settlement System (RTGS) provides the function of both "real-time" and "gross-settlement", which means low-volume, high-value and immediate payments. In presences of securely authenticated parties, RTGS is a low-risk transaction environment as payments are final and irreversible at the execution point. An example of RTGS is CHAPS in the UK where its counterpart is BACS that handle Net-Settlement. Other examples include FedWire and CHIPS in the US. The Eurosystem is in the process of implementing the Target2 system which provides RTGS for banks across the European Union.

For international payments, financial institutions around the globe make use of SWIFT (Society for Worldwide Interbank Financial Telecommunication) to send the payment orders in a secure and standard environment. Then the payments will be settled using their correspondent accounts. SWIFT identifies the financial institutions via a Business Identifier Code (the SWIFT code).

In parallel to this mostly "monolithic" architecture, the new architectures aim to transfer cash quantities without centralized clearing and we will now give a short technical summary before providing a new set of insights.

3. A Brief History of Crypto PTNs

The concept of a completely digital currency dates back to 1982 when David Chaum introduced the E-cash scheme [9]. E-cash took a "central bank" approach with a completely flat hierarchy where there is no intermediary institution. The scheme also provided such strong anonymity that even cooperation between the E-cash coin issuing bank and the merchant would still fail to identify the spender of the coin. The scheme allowed both online and offline verification of the coin. In the case of offline verification, the payer's identity could be revealed if the coin is spent twice [9].

Subsequent attempts at a purely digital currency were B-Money, RPOW , and BitGold which utilize "Proof-of-Work" [19], a hard cryptographic computational puzzle, as a mean of determining inherent value for the medium of exchange. However, they still require a central authority. These conceptual ideas later matured into the Bitcoin-esque approach which is free of a trusted central bank.

The Bitcoin protocol introduces a decentralized transaction database, namely blockchain [27]. The blockchain is shared by all nodes participating in the Bitcoin system. Every executed transaction is included in the blockchain so that any node can track the balance of each address at any point of the system history. The transfer of value in the Bitcoin network is carried out via the transfer of amount of BTC from an address to another one. The receiver will have to provide the private key to unlock the funds that they received in an earlier transaction.

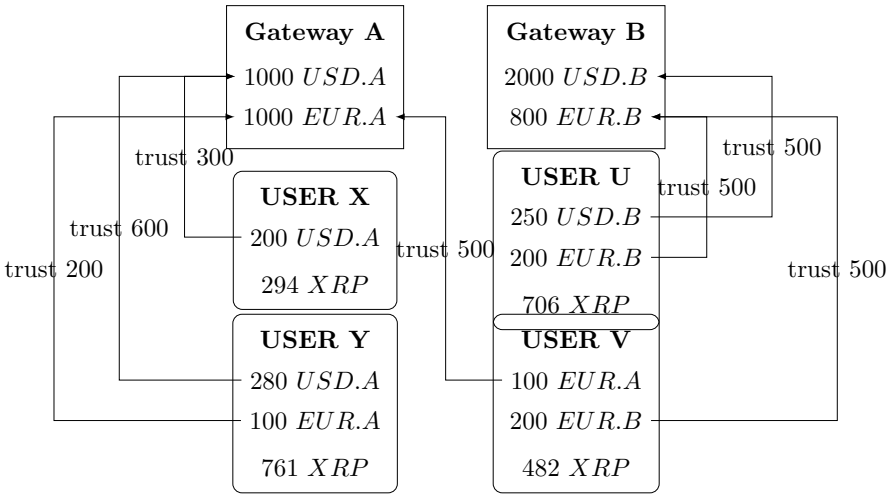
Since the launch of Bitcoin in 2009, many cryptocurrencies have been developed. Some use different hash functions than Bitcoin. Dogecoin, Litecoin, and PotCoin utilize the hash function of Scrypt [28] which is more memory-costly to deter the use of hardware-based mining devices. Dash utilizes X11 which consists of eleven different hashing algorithms in a chain while Primecoin requires finding a prime chain composed of Cunningham chain and Bi-twin chain. BlackCoin and Nxt apply Proof-of-Stake mechanism where the miners need to prove the ownership of a certain amount of cryptocurrency.

Many of the new cryptocurrencies try to offer extended functionality. At an extreme this is represented by Etherum which claims a "programming languages for financial contracts" [14]. The proposal of this idea is far from new. It has been first proposed by J. M. Eber in [20] and is currently marketed by the company LexiFi for the management of traditional financial contracts.

In contrast to the preceding types, ZeroCoin [26] is a combination of E-cash scheme and Bitcoin to improve Bitcoin's anonymity. Subsequently, the ZeroCoin apporach was implemented as Moneta. Another case is Monero which offers sender and receiver privacy by hiding connection between the sender and the receiver's addresses and applies Cryptonote [32] which utilizes traceable link signature [17] to confirm the validity of the transactions.

Among the cryptocurrencies, Ripple (another variant is Hyperledger) takes a relatively different approach from Bitcoin . Ripple [31] provides the functions of an RTGS, and at the same time, currency exchange of a traditional PTN. Figure 4 shows the components of the network where the gateways A and B, usually financial institutions such as banks, introduces liquidity into the systems. They create issuances that represent the external tokens to transfer in the network. The gateway A issues USD.A and EUR.A while the gateway B issues USD.B and EUR.B. Out-of-band the other nodes of the network deposit some "real world" money at the gateways so that they can declare a trust-line with a gateway to accept their digital currency and uses the currency for transactions with other parties.

An interesting direction stems from a recent Bank of England report [3] which covers several salient questions for the financial industry given the developments in technology. One of the primary



Gateways A and B corresponds to financial institutions and provide liquidity, i.e. create issuances that represent the external tokens to transfer in the network. The client X, Y, U, and V declares their trust-line to the gateways A and B. The actual deposit of external values at the various gateways is external to the system as in the e-cash protocol.

Figure 4: The Ripple Network Components and Gateways with the External World

concerns that central banks may have is loosing control of money-supply as an instrument of policy. This specific question is tackled directly by RSCoin [12]. This network delegates the monetary supply to a central authority, such as a bank, but utilizes a distributed network of other parties to perform transaction validation. The RSCoin network claims to provide strong transparency and audit capacity combining a central monetary provider with the attractive features present in a distributed transaction system (chiefly decentralized ownership). However, it suffers from the fact that it relies on the Central Bank to merge the high level blocks and that would create a central point of failure let alone a computational bottleneck [2]. RSCoin can be basically considered as a traditional digital PTN with cryptography extension borrowing from Bitcoin. The Central Bank is still on the top of the hierarchy and maintains the ultimate ledger.

4. High level features of Payment Transactions Networks

Simple inspection of the developments shows that new proposals tends to be capability driven aiming at improving some features an existing frameworks without challenging the overall design or purpose.

To challenge the "feeling of being necessary" behind each architecture, we decompose the foundational requirements of a transaction system and then categorize the current engineering approaches against those requirements. In this section, we present a comparative analysis of the steps for E-cash [9], Bitcoin [27] and its alternative, ZeroCoin [26], Ripple [31], and the centralized currency RSCoin [12]. Their features are summarized in a dedicated Table for each step of a payment system.

[2] In Europe only German and Italy's central banks have the computational horsepower to do it.

4.1. The 4 steps of payment: Creation, Promise, Realization, and Storage

Four critical steps of a digital PTN in common with traditional ones are: creation, promise, realization, and storage.

The *Creation* and preservation of value This step involves putting value into the payment system for circulation. The two steps required are (1) "out-of-band" deposit and (2) payment-token creation. The payment-token is usually referred as currency.

The *Promise* of a payment Payment promise step refers to the action of announcing that a payer is willing to transfer value to a payee. In terms of cheque payment, that is the action when the payer presenting the payee a cheque whereas in a traditional digital payment that is a payment request the payer sends to the central authority. This promise step is however not always necessary.

The *Realization* of transactions For "physical" currencies, the exchange is an instaneous realization of the promise. However, in digital PTN, whenever a "promise" is made, it needs to be realized. Firstly, the transaction's validity is ensured through validation rules and time-stamping. The validation rules cover the procedure to validate a transaction. It requires a check of the payer's balance, and sometimes, the authenticity of the currency. The time-stamping of the transaction ensures a correct order of the transaction ledger.

The *Storage* of value Currency storage usually involves storing two kinds of data. The first one is the balance of a PTN client's account. The second kind of data is the transaction log which allows the audit of the transaction history. The transaction log provides the transparency in the currency network and prove that some transactions are fraudulent or not, e.g. the Bitcoin blockchain is also the transaction log storage. The commodity and token currencies can be stored in physical form such as gold, silver, bank notes, etc. For these physical form, the client keeps them on their own. Another form is an account balance recorded at a third-party's ledger. For this kind of account, the bank is needed to validate the client's available funds. Without a central bank, Bitcoin and other digital currencies keep their client's balance in multiple copies of the transaction ledger.

Four actor types are involved in the four steps above:

Central Authorities CAs have special power in a PTN. They are known and recognized by other actors. They are also the ultimate actors to authenticate other actors in the PTN.

Brokers They validate transactions in a PTN. They consolidate transactions into the PTN and, upon realization, may receive some fees.

Payer A payer owns the value in the PTNs and initiate a transaction by executing promise steps in a PTN.

Payee This is the entity that will receive the value from the transaction.

For the avoidance of doubt, in this scenario we do not consider the "physical" identity of the entity behind the nominal identity in the PTN. Taken on their own, a Bitcoin address and a traditional bank account number have no less (and no more) anonymity. It is only the information held by the bank in its enterprise information system, usually *outside* the payment network, that allows to

Table 2: Creation of Value

Creation of value involves putting value into the payment system for circulation. The two steps required are (1) out-of-band deposit and (2) payment-token creation. The payment-token is usually referred as currency. As we can see, both E-cash, Ripple, and RSCoin requires the ability of all nodes in the network to recognize the signatures of CAs and, possibly more importantly, acknowledge the authority of CAs to introduce value into the PTN. The only difference between those systems is how many CAs can actually participate.

System	Central Authorities	Brokers	Payer	Payee
E-cash	Sign the Serial#. Balance account. Send coin to *Payer*.	N/A	Out-of-band deposit value into *CA*. Blind and send the Serial#. Unblind the CA-signed coin.	-
Bitcoin	N/A	Verify PoW, add the block. Broadcast the block.	Solve the PoW and then broadcast a coin-base transaction.	-
ZeroCoin	N/A	Verify PoW and ZeroCoin. Add and broadcast the valid block.	Solve the PoW and then broadcast a coin-base transaction. Mint Bitcoin to Zero-Coin.	-
Ripple	Sign transactions for creation of issuances. Broadcast a transaction to send value to the *Payer*.	-	Declare a trust-line upon the *CA*' tokens. Out-of-band deposit value into *CA*.	-
RSCoin	Sign transactions for creation of issuances. Broadcast a transaction to send value to the *Payer*.	-	Out-of-band deposit value into *CA*.	-

associate a 27 alphanumeric international bank account number to a physical or legal entity. For each of the systems we will discuss the corresponding properties. For example besides presentation of value we consider two important properties: *confidentiality* and *anonymity*. Confidential means that the owner is known but the value may not be known while anonymous means that the owner is is unknown while the value is known.

4.2. The Creation of Value

Table 2 compares the key steps of the 5 protocols subject of our analysis.

In E-cash, the client withdraws coins from the bank, out of the client's account, by sending a blinded serial number to let the bank signs the corresponding blind signature to the information.

In the Bitcoin network, upon solving a Proof-of-Work, the solver is rewarded with some bitcoin (BTC). The BTC is bound to the owner by an address, i.e a public key whose private key is only known to the owner. The Proof-of-Work puzzle must be hard to solve but easy to check. It must also be a progress-free puzzle, which means the chance of solving doesn't increase with effort. This process is referred to as "mining" and the individual nodes are commonly referred to as "miners".

ZeroCoin [26] is based on Bitcoin. The protocol provides additional anonymity for the network users by allowing the user to "mint" the basecoin (BTC) into a ZeroCoin. To achieve this, the user generates a random serial number s and a random number r, then s is encrypted into the

Table 3: Promise of Payment

The *Promise* of a payment is the action by a payer to announcing to the PTN a transfer of value to a payee. In terms of cheque payment, that is the action when the payer presenting the payee a cheque whereas in a traditional digital payment that is a payment request the payer sends to the clearing house. It is a promise as eventual fulfillment is not initially warranted. The payee plays a relatively minor role and the requirement of some protocols to use a private key to pocket the transaction could be eliminated altogether, when the transactions are public. There would be no need to "unlock" the fund, as the whole PTN could simply acknowledge that the payee is the recipient. Only when the payee needs to use the received funds he would need a private key. Yet, when this would happen s/he will be a payer...

System	**Central Authorities**	**Brokers**	**Payer**	**Payee**
E-cash	-	N/A	Encrypt the Serial#, send the data to *Payee*.	-
Bitcoin	N/A	Receive the transaction.	Create a transaction with a value and *Payee*'s address, then broadcast the transaction. *Payer*'s private key is needed to unlock the available funds.	-
ZeroCoin	N/A	Same as Bitcoin but requires the *Payer* to apply spending function on the stored Serial# to unlock the available funds.	-	
Ripple	-	Receive the transaction from *Payer*.	Sign a transaction and broadcast. The transaction includes address of *Payee*, the amount, and issuer of the tokens.	-
RSCoin	-	Same as Bitcoin	Same as Bitcoin	-

ZeroCoin z with r. The minting operation requires a Bitcoin transaction that spends d BTCs, which is also the value of the ZeroCoin z, plus a small amount of transaction fee. The authors argue that ZeroCoin provides real anonymity compare to the pseudonyms in Bitcoin due to the fact that the pseudonyms' transactions are traceable in the blockchain.

In Ripple, the gateways create issuances to transfer in the network but the payer must first deposit value into the gateways. To ensure the integrity of the issuances, the gateway will sign them with their private keys. The clients declare a trust-line with a gateway to accept its digital currency and uses the currency for transaction.

RSCoin [12] simply delegates the value creation to a Central Bank. The Central Bank is a Central Authority and assumed to be honest. To generate value, the Central Bank signs a transaction to allocate the value to an address.

4.3. The Promise of Payment

Table 3 illustrates how payment is initiated.

The E-cash scheme's payment is initiated as the payer send the coins to the payee. The coin's data is encrypted with the bank's public key so that the payee cannot peek the coin's serial number.

In the Bitcoin network, a payment is started with the payer creating a transaction. The payer specifies the unspent transaction outputs to use as inputs in a new transaction. The outputs are the address of payees with the respective amount of coin. The transaction will be broadcast into the whole Bitcoin network to notify all nodes.

ZeroCoin requires an additional step to initiate the payment. For a payer to spend the ZeroCoin, the payer must first construct a transaction to claim an unspent mint transaction output as input. The output of the transaction is the same as Bitcoin's. The unspent mint transaction output is claimed by providing the serial number s and the random number r as a zero-knowledge proof.

Ripple's users also create a promise of payment via transactions. A Ripple transaction must be signed by a user's private key. The transaction must include the address of a payee and the specific amount of tokens along with their issuer ID. Then the transaction is broadcast into the whole Ripple network for validation.

RSCoin's payment promise is similar to Bitcoin's. In the RSCoin network, there exists a large number of nodes, called mintettes, authorized by the central bank via PKI-infrastructure. The payer must create a transasction and send it to the corresponding mintettes to prove that the payment input is unspent and thus for them to include the transaction into a block.

4.4. The Realization of Transaction

Table 4 shows how payments are validated and the PTN converges to a new state where the value of the transactions is finally accrued to the payee and subtracted to the paye's account in the PTN.

The E-cash scheme requires the original minting bank to perform the verification of the coins. When a payee receive the encrypted coin data from the payer, the payee forwards the data to the minting bank. The bank will decrypt the coin data with its private key. The serial number will be checked within the spent serial number database to prevent double-spending. If everything is correct, the bank will accept the coin data and credit the payee's account with the corresponding amount. The deposited coins' serial number will be added into the spent serial number database.

For the Bitcoin network, the miners receive the transactions from the payer and validate the correctness of the transactions before adding them into blocks. After solving the PoW and creating a block that includes the transactions, the miners will broadcast the block into the whole network. Other nodes, upon receiving it, will perform the validation for the PoW, the transactions' correctness, and the integrity of the block before adding it into their blockchain and broadcast their acceptance.

In the case of ZeroCoin, a transaction is realized after the nodes apply the verifying function to the ZeroCoin z and check that the serial number s is unspent. After the transaction is accepted, the payee will need to initiate a new minting transaction to create a new ZeroCoin from the transaction's output.

In a different approach, Ripple's consensus protocol requires more than 80% of nodes' agreement to consider a transaction valid and add it into the consensus ledger. On the Ripple network the nodes have to check for over-spending only. The payment is conducted by moving a certain amount of issuances from one account to another.

RSCoin requires the collaboration of mintettes and the central bank for the realization of a transaction. Mintettes are authorized by the central bank to collect transactions and put them into the low-level blocks. The low-level blocks are periodically sent to the central bank to merge into a high-level blocks and add to the main blockchain. A transaction initiated by the payer, once sent to the corresponding mintettes will be validated before being added into low-level blocks. Since the mintettes are semi-trusted, PoW is no longer required to create a low-level block.

4.5. The Storage of Value

Table 5 summarizes the approaches to the storage of value in each selected PTN.

Table 4: Realization of Transaction

During the realization step, the promise to pay is finally accrued to the payee and subtracted to the payer's account in the system. It requires checking the payer's balance and possibly the authenticity of the currency. A time-stamping of the transaction may be needed to ensure a correct order of the transaction ledger. A critical issue in distributed PTNs is that every broker must be able to check the balance of the payer and broadcast the result to achieve a consensus on the ledger. Suitable incentives must then be in place to guarantee the timely cooperation of a sufficient number of brokers.

System	Central Authorities	Brokers	Payer	Payee
E-cash	Check the spent Serial# database for double-spending. Credit the *Payee*. Mark the spent Serial#.	N/A	-	-
Bitcoin	N/A	Validate the transaction by checking the public and private key of the available funds. Verify that transaction inputs are previous unspent outputs. Solve the PoW, add the transaction into a block, then broadcast the block.	-	-
ZeroCoin	N/A	Validate the transaction by applying the verifying function Check that the Serial# is not in spent database. Solve the PoW, add the transaction into a block, then broadcast the block.	-	Mint Bitcoin to ZeroCoin.
Ripple	-	Validate the transaction. Broadcast the validation result and add the transaction into ledger upon agreement .	-	-
RSCoin	Aggregate the result from the *Brokers*' low-level blocks to form high-level blocks and add into blockchain.	Validate the transaction by checking the public and private key of the available funds. Then send the block to *CA*.	-	-

For E-cash, the balance of each payer must be stored in the minting bank's database to prevent over-drafting. To counter double-spending, a serial number database must be maintained to check the serial number of each deposit request. The payer must store the serial number carefully to use in a payment. The storage is only partially private as the minting bank knows the balance of each payer. Only anonymity of the payer in a transaction is guaranteed because the bank cannot trace back the original owner of the coin from the serial number.

Bitcoin miners store the whole blockchain in their hard-disk. The blockchain is everything in the Bitcoin network as every executed transaction is included in it. It is the same for the ZeroCoin network, except that ZeroCoin also needs a global accumulator which is a state-keeper for the binding between value and the secret serial number. The Bitcoin network users store private keys to unlock funds while the ZeroCoin users only store the serial numbers as they get rid of the private keys after finishing minting a ZeroCoin. The RSCoin network maintains low-level blocks at the mintettes and high-level blocks which form the final blockchain at the central bank while the users need to store the private keys to unlock funds from previous transactions. These three networks

Table 5: Storage of Value

Notes: The *Storage* of value usually involves storing two kinds of data. The first one is the balance of a PTN client's account. Whilst commodity and token currencies can be stored in physical form, a digital PTN must store value as numeraire in some ledger whose authenticity is recognized by all parties in the PTN. The second kind of data is the transaction log which allows the audit of the transaction history. Several systems are more vulnerable to theft or failures as they require to store data to unlock the funds as payees (as opposed to use them as payers). Further, distributed PTNs require all brokers to have access and to store the transaction log of the *entire* system.

System	**Central Authorities**	**Brokers**	**Payer**	**Payee**
E-cash	Store available funds of a *Payer* and Spent-Serial#.	N/A	Store the Serial#.	-
Bitcoin	N/A	Store the blockchain.	Store the private keys to unlock the unspent transaction outputs.	-
ZeroCoin	N/A	Store the blockchain. The global accumulator stores the binding of value and Serial#.	Store the Serial#.	-
Ripple	Store the out-of-band deposit of clients.	Store the consensus ledger Store the available funds of the corresponding *Payer*.	Store the private key to sign the transaction.	-
RSCoin	Store the "ultimate" blockchain.	Store the blockchain.	Store the private keys to unlock the unspent transaction outputs .	-

all provides anonymous but not private storage as every transaction is recorded in the blockchain. The anonymity of the storage is only provided if the payer use different pairs of public/private key for each transaction.

The Ripple network nodes store the consensus ledger for the balance of each payer and also the transaction history. Thus the payer only needs to store a signing key for the transaction as the Ripple network provides neither private nor anonymous storage.

5. Threats to Payment Transaction Networks

We consider two types of high level threats. The first type involves victims losing money in the PTN while the second type concerns the disclosure of information about its users.

For the first type, the criteria for threats classification are whether the others benefit from the victim's loss and whether the victim is actively involved in one's loss. We identify three high level threats: loss (systemic and individual), fraud, and theft. See Table 6.

The threat of losses due to individual sloppiness or misfeasance are also present in traditional PTNs and can be broadly characterized in the following classes:

Over-Drafting When the payer promises a transaction whose value exceeds one's available funds.

Double-Spending When the payer can use a token twice.

Table 6: Loss of value

Notes: The first threat to a distributed PTN is the possibility of a party to lose money. A key distinction is whether such losses can be due to systemic failures or to individual actions either by a victim's own lack of care or by malicious activities of other parties. From the table it is clear that distributed PTNs make individual thefts harder as it requires changing the value of the global ledger, whilst systems which include a role of the payee in the transaction make losses due to lack of care easier. However, technological failures to achieve consensus (eg. Bitcoin forks) introduce systemic risk akin to the failure of a central bank.

Threats	**Others Benefit**	**Victim Active Involved**	**Examples**
Systemic Loss	-	-	*CA* fails and all values are lost; Blockchain forks.
Individual Loss	-	yes	Lose the Serial# or private keys.
Fraud	yes	yes	Over-drafting; Over-spending.
Theft	yes	-	Unauthorized-spending.

Unauthorized-Spending A payer spends another one's money or changes the value in another's account.

Individual Loss This can happen when the E-cash or ZeroCoin owner loses the serial numbers. Anther possibility is that the Bitcoin or RSCoin network users lose their private keys.

Systemic Loss A systemic loss in a PTN is not caused by an individual but rather by some intrinsic fragility of some components of the system itself.

In the presence of centralized CAs, the obvious sources of fragility is the trustworthiness of those very CAs. An example is the failure of Central Banks. By design, CAs can print money into the system, and therefore protection against their failure must be dealt with outside the protocol.

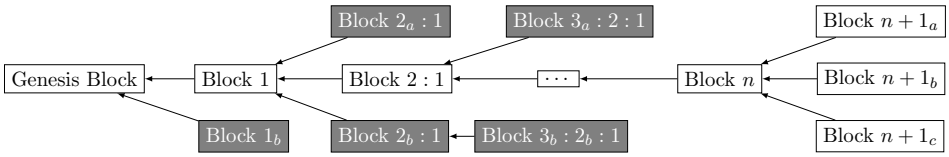
Distributed PTN are amenable to a different type of systemic risk, namely the *failure to reach consensus*: the blockchain forks and some legitimate transactions are not included in the main chain. We illustrate this scenario in Figure 5 for the particular case of Bitcoin. At some point in time several nodes are generating new blocks or proposing transactions. The member of the PTN closer to the promising payers will create different blockchains. In that case, the blockchain is forked and every nodes will maintain the fork until conflict resolution mechanisms kicks in.

In Bitcoin, the consensus mechanism is based on the presence of a "longer" chain. The longer chain becomes the main chain and the other tentative chains can be considered invalid. The blocks in the invalid chain are "orphaned" blocks. From the perspective of the distributed PTN, they are all potentially valid but rejected payment promises. This happens irrespectively of the actual validity of the promises from the view point of authenticity of the payers requests and the availability of funds at the payers' accounts. This threat is *not* present in traditional PTN.

Every transactions in the orphaned blocks needs then to be put back to the transaction queue to be added in the new block. As in Figure 5, the blocks 1_a, 2_a, 2_b, 3_a, and 3_b are orphaned transactions as a result of previous fork resolutions. At block $n-1$ the chain suffers from three forks (blocks n_a, n_b, and n_c) and has to wait for the next fork resolution. Another threat of systemic incoherence is when the outcome of a transaction is different from individual to individual in the network [4].

The second type of information disclosure involves answering three questions.

Instantaneous Networth At time t, can an attacker identify the total value v of a nominal identity I?



Notes: After $n-1$-blocks, three concurrent transactions (n_a, n_b, n_c)have been proposed concurrently. The global chain forked and has to wait until some payer decide to append a transaction onto one of the three alternatives. Some previous forks in the past at block 2 resulted in orphaned transactions (in grey). Those are discarded from the system, no matter whether they were actually valid from the perspective of a "normal" payment system. The situation is particularly dire for those along the forked chain $3_b:2_b:1$: they have added transactions along what they thought was an eventually legal chain. The payers need to re-play the entire subchain again (both 2_b and 3_b) if they want the transaction to be inserted in the final ledger.

Figure 5: The Systemic Risk of Blockchains: Forked and Orphaned Transactions

Table 7: Loss of value countermeasures

System	Over-Drafting	Double-Spending	Individual Loss	Theft
E-cash	*CA* check *Payer*'s balance before signing blinded Serial#	*CA* stores spent Serial# and check upon deposit.	*CA* stores the last n batches of blinded Serial#, send back to *Payer*. *Payer* unblinds the data.	N/A
Bitcoin	*Brokers* check that input $\geq$ output	*Brokers* check the inputs are unspent & blockchain is hard to rewrite.	N/A	*Brokers* check the private key.
ZeroCoin	Same as Bitcoin.	*Brokers* check the serial numbers are unspent & Blockchain is hard to rewrite.	N/A	*Brokers* check the Serial#.
Ripple	*Brokers* verify balance $\geq$ output	N/A	N/A	*Brokers* check the signature.
RSCoin	Same as Bitcoin.	Same as Bitcoin.	N/A	Same as Bitcoin.

Transient Value At time t, can an attacker know about a transaction of total value v between two nominal identities I_1 and I_2?

Persistent Identity Can an attacker link two nominal identities I_1 at time t and I_2 at time $t+1$?

The countermeasures against loss of either confidentiality or anonymity are summarized in Table 8.

Beside the threats which we have just mentioned, there are still potential threats related to the technology and some of them have been actually realized .

Concurrent irreconcilied transactions This threat allows a malicious actor to combine two valid transactions to create a fraudulent one, e.g. "0-confirmation attack" [21].

Selfish-mining Selfish-mining is an attack on the block mining mechanism that allows unfair block distribution [15].

Table 8: Information disclosure countermeasures

System	Instantaneous Networth	Transient Value	Persistent Identity
E-cash	Only *CA* knows the balance.	Encrypt coin data with *CA*'s key.	*CA* keeps the balance DB private.
Bitcoin	NO	NO	*Payee* uses different key pair per transaction.
ZeroCoin	*Payee* applies coin minting.	NO	*Payee* uses different key pair per transaction.
Ripple	NO	NO	NO
RSCoin	NO	NO	Same as Bitcoin.

Table 9: Threats

System	Realized Threats	Potential Threats
E-cash	Lose both Serial# + blinding factor.	N/A
Bitcoin	Loss of coins (invalid PubKey or losing PriKey). 0-confirmation attack. Deanonymization. Transaction malleability. Exchanger fraud.	Majority attack.
ZeroCoin	Same as Bitcoin.	Same as Bitcoin. Loss of Serial#
Ripple	Disagreement	Majority attack. Gateway fraud.
RSCoin	Loss of coins (invalid PubKey or losing PriKey). Deanonymization. CA & Exchanger fraud.	Same as Bitcoin. The final blockchain (only one copy at the Central Bank) is rewritten.

Deanonymization Although the distributed PTNs often offer anonymity in coins ownership and payment, the attacker has a chance to group and link the coins back to the owner by different methods.

Malleability This refers to an implementation flaw that allows the modifications in the transaction data without changing the hash value of the block header.

CA or Exchanger Failure Exchangers are third-party organizations that provide exchange for the digital currency and another currency such as commodity, fiat or another digital currency [5]. When the CA or Exchanger fails, the deposited funds are simply lost.

We summarize those threats in Table 9. Other threats also exist. The instability of a currency's value, inflation, or the realization of another digital currency can drive the participants away from a currency network. Some national governments may try to control or even ban the digital currency from usage in the country because they have no control over it, and the anonymity of the digital currency allows illegal activities (e.g. money laundering, illegal financing, and evading taxes).

6. Technological Components

We now briefly summarize the technological components of digital PTNs.

Digital Signature In public-key cryptography, a digital signature is obtained by encrypting a plain text (human-readable message) with the private key. The others can verify the signature by decrypting the cipher text (encrypted message) with the public key.

Blockchain The basic idea is that every block references to its previous block header's hash and thus forming a chain. This mechanism firstly ensures a chronological order of every blocks. Secondly, it is unfeasible to modify a block after it is linked to by the next blocks as that would require to modify all the following blocks.

Decentralized Agreement This refers to the property that all participants produce the same result upon querying a single certain data in a decentralized setting.

6.1. Proving Authenticity and Integrity

Digital signature [22, p. 439] is used to provide some form of authenticity to payers and payees. For example, a Bitcoin, ZeroCoin, and RSCoin user receives the coins via an address in the form of the hash of a public key. Later, to unlock the funds, a user has to provide a corresponding private key to prove the ownership of the coins. Ripple, however, use digital signature only to authenticate the user as s/he has to sign the transactions with the private key before sending the transactions to a validator. This also applies to the gateways in their transactions for value creation. Vanilla digital signatures are also used by RSCoin to authenticate messages from "bank-endorsed" participants (i.e. mintettes) in the consensus protocol.

In the E-cash scheme, the blind signature [8] allows the CA sign the serial number without knowing its content. This is done by multiplying a blinding factor. After the central authority has signed the blinded serial number with its private key, the content is unblinded by dividing the signed message with the blinding factor. However, the E-cash model only offers the ability to create fixed-value coins. To allow the different value coins, the central authority can introduce various coin keys, each of which, upon signing, stating that the coin has a specific pre-defined value. A decentralized e-cash scheme [26] is used by ZeroCoin to maintain anonymity yet still allow a user to prove ownership of a minted coin.

Hash functions [22, p. 153] are typically used to guarantee the integrity of the ledger. A blockchain is simply a sequence of applications of a hash function to a sequence of transactions. Every block contains the information of the current transactions and a reference to its previous block header's hash. Figure 6 outlines the process of conducting transactions in the Bitcoin network. There are 2 types of transaction: a coinbase and a regular transaction. While the latter one indicates some BTCs are being transfered from an address to another, the former is for the introduction of new coin into the system for circulation (value creation). As such, it is a coinbase transaction that distributes a miner's reward. Each block in the blockchain is included with a set of transactions. Each transaction consists of a vector of inputs and a vector of outputs. Each input of a transaction T spends a certain amount of BTCs from a previous output of a concluded transaction $T0$. To spend the previous output, the input references a tuple of $(hash, n)$. A *double-SHA256* is performed on the T_0's raw data for the $hash$, while the n is the index of the output in T_0. Each output of a transaction T specifies an amount that will be transfered to a Bitcoin address. A block also hashes all of its transactions into a Merkle Tree [25] to maintain integrity.

Cross-hashing [30] is used by RSCoin which also uses a signed hash from the central bank to mark epochs in the chain. Due to lack of space, we refer to [22] for some further discussion of crypto systems.

The transaction that creates the value is the coinbase transaction T_0 which send 10 BTC to the public keys address A_{pub}, 8 BTC to B_{pub} and 7 BTC to C_{pub}. In order to spend those coins, in transaction T_N, the owners have to prove they process the private keys A_{pri}, B_{pri} for the output index 0 and 1 of the transaction $T0$ respectively. This also applies to transaction T_M with D_{pub}, D_{pri} and E_{pub}, E_{pri}.

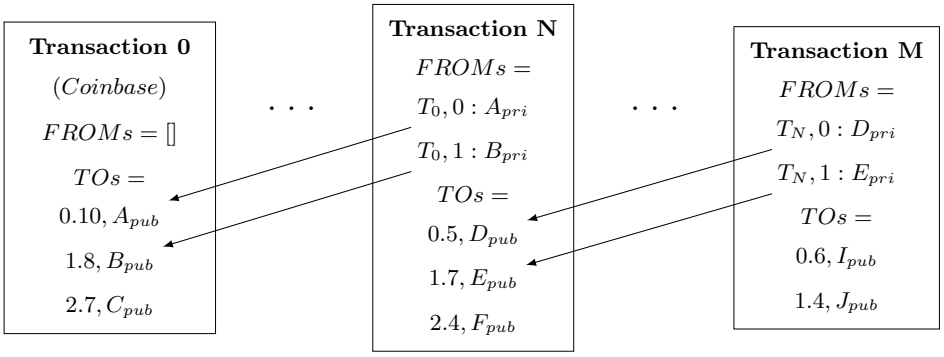


Figure 6: The Bitcoin Transactions

6.2. Decentralized Agreement

The classic distributed consensus problem can be described as a set of n participants, out of which t are possibly faulty, seeking to agree on a single value v drawn from a set of value V proposed by the participants. The most trivial solution would be majority voting. The classic paper on Byzantine agreement on synchronous system [23] defines three properties that an agreement protocol must hold.

Validity The single value v must be initially proposed by non-faulty participants.

Agreement Every non-faulty participants must agree on the same value v.

Termination The agreement must be reached in finite time.

The robustness of a Byzantine protocol is measured in terms of the number of faulty participants that it can tolerate. It is proven that Byzantine agreement is only possible with $n \geq (3t + 1)$ [23].

A further FLP impossibility result, applicable to deterministic and asynchronous system [16] has proven that even with a single faulty-participant, the distributed consensus cannot be achieved in asynchronous settings because the nodes fail to differentiate a failure from a delay. To circumvent the impossibility results, researchers introduced techniques such as randomization[29], failure detectors [7], or assume partial synchrony [13]. In such systems, safety is always prioritized over liveness.

The Bitcoin network introduced a solution to a variance of the Byzantine agreement problem. The mechanism of adding blocks allows the happening of "forks" where disagreement arises within the network. The Bitcoin network's consensus problem is different from classic ones as the number of nodes in the network is exceptionally large and dynamic. The consequence is that nodes cannot decide on a value of majority. The nodes will vote for the "chain with most work" according to the blockchain by mining on the next block referencing to that chain's last block. Additionally, the block generation rate is technically maintained at 10 minutes to allow the propagation of block across the whole network so that every node shares the same blockchain. This solution practically allows the tolerance of faulty-nodes up to 50%. However, a consequence is that the transaction

clearing speed is unacceptably slow for any financial implementation (e.g. the Chicago Mercantile Exchange marks to market millions of trades in less than a minutes).

Ripple's agreement is achieved through the *Ripple Protocol Consensus Algorithm* [33] (RPCA). In each round, each server will collect the transactions, validate them, and then broadcast the valid transactions as candidates. The candidates will be voted for veracity by the other servers in the server's unique-node-list. If a transaction achieves at least 80% of agreement, it will be put in the ledger. The ledger will then close for current round and become the last-closed ledger for next round. RPCA can only tolerate up to $n \geq 5t + 1$. However, the RPCA implementation (by Stellar) has been invalidated by an actual fork [11]. Ripple is an example of traditional BFT PTN comparing to the PoW approach of Bitcoin. For a comparative analysis between PoW and traditional BFT, see [35].

Ripple, and similar systems, have not yet accounted for all possible cases that must be faced by real payment systems. One of the problem is churn when servers leave the network for maintenance or other reasons. In this case, the number of faulty nodes may exceed the threshold and compromise the ledger. If this is perceived as realistic threat by the network stakeholders then the incorporation of a stronger consensus algorithm [2] may be needed. In the cited work, the authors managed to tolerate up to j churning servers where $n \geq 5t + 3j$. Furthermore, some crypto PTNs are fully asynchronous (e.g. Bitcoin), whilst real systems actually have a pre-defined synchronization point, e.g. the Chicago Mercantile Exchange Globex Futures synchronize all trades between 13:59:00 and 14:00:00 Central Time for each trading day (see §(7)). For such systems, a synchronous protocol could be more suitable such as [36].

7. Smart contracts and Futures

The notion of "smart" contracts has been posited by Bitcoin and some other blockchain-based distributed ledgers e.g. Ethereum, Hyperledger, and Hedgy. However, the term "smart" is a misnomer as the actual engineering requirements for the envisioned applications are quite primitive. Indeed, the contract is simply stored as data in the blockchain. The contract must then be executed by the provider of the PTN infrastructure itself (e.g Hedgy) or another third party that execute and monitors the contracts (e.g LexiFi). Hence, those smart contracts are essentially indistinguishable from the Eber's original proposal of smart contract [20].

The current prevailing smart contract implementation replicates a bond or dividend payment. In this case the contract is pre-programmed with a regular payment date and recipient until the maturity of the debt. Hence, the only information the contract needs after its inception is the date and time. A more complex approach that covers a wider variety of financial instruments requires some measurement of ex-post state of a contingent variable, such as a stock index, commodities price or reference interest rate. In this section we will discuss the natural extension to the standard distributed ledger which contains passive information: that is active executables within the distributed framework. Hence, the ledger evolves to a distributed database with active execution.

The presence of both passive and active components is not new in current centrally cleared ledgers, such as the Chicago Mercantile Exchange (CME) Globex Futures trading platform [10] and the European Central Bank coordinated initiative Target 2 securities.

Amongst a large number of trades it facilitates three very important futures markets: Eurodollars (the forward delivery of dollar cash at the three month London Interbank Offered Rate, LIBOR), S&P 500 E-mini futures (forward looking equivalent cash delivery of the S&P 500 index) and NYMEX traded West Texas Intermediate (WTI) Light Crude futures (one of the two main

benchmarks for crude oil, indeed, the alternative contract for the delivery of the Brent Blend is also partially traded on NYMEX).

Similar features are present in the Target 2 (Trans-European Automated Real-time Gross Settlement Express Transfer System). Whilst the standard Target 2 interbank payment system is a standard settlement system owned by the Eurosystem members, there is also a Target 2 securities, a separate entity, that provides centralized delivery versus payment systems within a centralized system. One of its interesting feature is the ability of the system to automatically determine cash settlement of contingent claims depending on the positions of the counter-parties. To execute settlements, the Target 2 Securities engine needs to process executions determined by the contractual structure of the instrument being traded. For a review of the technical specifications of Target 2 Securities see [34].

For illustrative purposes, in the sequel we focus on the CME Globex platform.

Forward delivery contracts, where parties agree to buy and sell an asset at a particular forward price (often referred to as the "fix") on a particular date (the maturity, or delivery date) are often used by firms to hedge risks in underlying assets, the classic being forward rate agreements and forward currency contracts.

Futures contracts operate in the same manner, but with two significant distinguishing features: the size of the contracts are standardized and the contract is marked to market each day in a predefined fixed interval[3]. The purpose of marking to market is to prevent individual bilateral positions from accumulating substantial losses relative to the reference (or spot) price.

Consider a forward contract that agrees to deliver (sell) one million barrels of oil at \$90 per barrel in three years, i.e. the point estimate jointly determined by the counter parties of expected value in three years from now is that the price of oil in the spot market will be \$90 per barrel. However, oil spot delivery prices fluctuate over time. Suppose that after two years the spot/nearest delivery oil price has dropped to \$60 dollars. The party on the purchase (buy) side of the contract would then stand to loose \$30 per barrel in the contract or \$30 Million versus buying on the spot market and the seller is set to gain the opposite versus the spot. This aggregation of losses and gains may place a substantial strain on the credit worthiness of the counter-parties agreement, if the notional size of the agreement is large enough. Eventually, a counter-party may have accrued enough losses to want to break the contract (despite the forward repercussions that might entail). Default, would then focus the losses on the other counter-party, in the worst case scenario a whole chain of back-to-back forward fixes may fail as counter-parties default.

Futures contracts eliminate this effect by settling adjustments in prices to cash each day. Hence, any price fluctuations in the forward delivery price stemming from trading in the spot and/or futures market are noted and the net position of each trader relative to their originating positions in the contracts are then credited and/or debited from the traders account with the exchange. If we reconsider the example above, the equivalent futures position would have already settled out the loss on the buyer side gradually as the price dropped.

To trade futures with the central counter-party several lots of collateral in the form of margins are needed. The initial margin is a deposit needed to trade, in addition a account or trading margin must be maintained (hence it is referred to sometimes as a maintenance margin) above a certain level to continue or the exchange settles out the traders position and deducts any losses above the level of the account margin from the initial margin (a "margin call").

[3]Daily settlements in Eurodollars occur in the one minute between 13:59:00 and 14:00:00 Central Time for each trading day.

At maturity, the exchange executes the final settlement of the contract and then rolls the reference code over to the next maturity date.[4] Final settlement of a cash settled contract operates in effectively the same manner as a daily settlement, only the reference is now the agreed spot reference (the 3 month LIBOR rate, the spot delivery price for crude, etc) and not a traded futures price. The final net positions of traders is calculated relative to the final spot reference and positions are settled.

By trading futures on the CME Globex platform a trader never actually takes possession of the long or short position in contract or execute it. The exchange computes the net position of traders and then credits or debits their accounts accordingly each day at the daily settlement time. Hence, away from processing order-flow, the CME Globex platform is both a passive ledger recording trades and an active computation platform determining where daily cash (and some terminal) settlements should be directed, subject to the stated reference (the futures price between 13:59:00 and 14:00:00 for daily settlements in crude for instance).

8. Distributed implementation of Derivatives-contracts-as-programs

This scenario can be implemented in a distributed PTN if we associate a future contract to an executable program. The current literature already associates a simple program to a derivative contract and use the blockchain technology to guarantee the integrity of the program in the PTN. As we have already noted, this is not sufficient to unleash the full potentiality of the system.

For the association of *derivatives-contracts-as-programs* to work properly and reflect accurately the functions of the CME, we need to define two additional properties of a contract beside the code itself: input and output bindings.

Input binding to spot/reference prices The spot or reference prices of the contract corresponds to the input of the program and the distributed PTN must bind it to the market or source where such values needs to be extracted. The blockchain technology provides a solution in this extent.

Output binding to traders' positions The output of the program must then be bound to traders accounts in the PTN. From an operational perspective, this is simply a list of pairs (trader's nominal identity in the PTN and a corresponding output element of the program). Once again the blockchain technology can be used to guarantee the integrity of the binding.

Sell and buy positions corresponds to the value of the corresponding output variables as illustrated in Figure 7

With this set up we can now transfer the CME functionalities to a distributed PTN.

Contract Settlements Each broker of the PTN executes the code of the contract as embedded in the blockchain, by reading the input values from the bound reference and computing the corresponding output values. As the contract includes the binding of the outputs to the nominal identities of the traders in the PTN, each broker then uses the consensus algorithm of the PTN to initiate the corresponding transactions (either crediting or debiting the results). There is no need of the promise phase from Table 3 and the brokers can directly proceed to

[4] For instance, CLH0 stands for first quarter (March) delivery of crude, in the nearest year ending in 0, for instance 2000, 2010 and 2020.

A sample contract

```
ContractID CLH1-345698-09678
in spotPrice;
out outputSell=0;
out outputBuy=0;

const value=...;

if  value > spotPrice
then
outputSell = spotPrice-value;
outputBuy = value-spotPrice;
else
outputSell = value-spotPrice;
outputBuy = spotPrice-value;
```

Its bindings

```
ContractID CLH1-345698-09678
spotPrice = .... crude oild in March 2021 ....

ContractID CLH1-345698-09678
outputSell = jw@192.152.234

ContractID CLH1-345698-09678
outputBuy = fm@myfund.somewhere
```

Notes: A simple derivatives-contract-as-program. The act of selling and buying the contract itself on the exchange can be captured by changing the binding of the appropriate outputs from the owner A to the acquirer B's own account in the PTN. The initial step is essentially identical to the promise one for PTN (Table3) and can use the same types of validation checks used by brokers to validate a transaction from Payer A to Payee B (Table 4). Additional checks might be required by the distributed exchange (e.g. mark to margin). Brokers executing the contract to mark it to market must also ensure that the sum of the two outputs at the time of daily settlement amounts to zero. This latter check is implicit in the controls in Table 4, but must be made explicit here.

Figure 7: A contract-as-program and its bindings

the realization phase in Table 4. After the transactions are finally realized the accounts of the traders are aligned to the market value as they would be in a centralized exchange.

Contract trading The trading of a derivative contract in the CME simply corresponds to change in the bindings between outputs of the contracts and the corresponding nominal identities in the PTN. This simply requires the owner of a (sell or buy) position in the contract to initiate a transaction as a Payer of the blockchain and the corresponding Payee is the the new owner of the position (Table 3). Each broker would register the "transaction", i.e. the change in ownership of the derivatives contract's positions, after having verified its validity (e.g. by marking to margin). A part of the steps are already detailed in Table 4 but additional steps specifically aimed at guarantee the balance of the ledger must be implemented.

Reference swap It is also possible to change the reference point of a contract by changing the binding between the input of the contract and the corresponding source of information. This would requires to set a party in the PTN that is actually authorized to do so.

An obvious issue for checking the validity of a trade is whether the Payer is actually the entity mentioned in the binding. These checks are already discussed in Table 4.4. Additional check may generate distributed exchanges of different nature. They could mimic the checks performed by the centralized exchange mechanism. For examples, if traders are not allowed to bid for contracts whose current payout exceeds their margin, each broker in the PTN would have to execute the contract and make sure that the Payee has enough capacity to stand the losses with the current value (as opposed to the value at time of settlement, or the previous year average, etc.). A whole variety of different arrangements is possible provided they only required to execute the contract

A Multi-party contract

```
ContractID CLH0-345698-32456
in spotPrice;
out outputSellA=0;
out outputSellB=0;
out outputBuy=0;

const value=...;

if (some condition)
then
outputSellA = 60% spotPrice-value;
outputSellB = 40% spotPrice-value;
outputBuy = value-spotPrice;
else
outputSellA = 30% spotPrice-value;
outputSellB = 70% spotPrice-value;
outputBuy = spotPrice-value;
```

Its bindings

```
ContractID CLH0-345698-32456
spotPrice = .... crude oild in March 2020 ....

ContractID CLH0-345698-32456
outputSellA = jw@192.152.234

ContractID CLH0-345698-32456
outputSellB = cnn@superbank.com

ContractID CLH0-345698-32456
outputBuy = fm@myfund.somewhere
```

Notes: A multi-party contract with an asymmetric share of risk from three counter parties. A party in a PTN can trade only a fraction of the risk by acquiring only one of the sell bindings (e.g. `cnn@superbank.com`) rather both sell positions. Brokers executing the contract to mark it to market must ensure that the *aggregated* outputs amount to zero.

Figure 8: A Multi-party contract-as-program and its bindings

from the bound inputs and possibly check the resulting outputs against the current account of the new owner.

In this set-up we can easily devise and deploy contracts in which risk is shared by several traders and where traders may only trade a share of the contract (see Figure 8).

New threats also become possible: *malicious (or buggy) contracts* that makes it possible to create a Ponzi scheme or even a cluster of schemes that mutually feed each other. Consider as an example the contract in Figure 7 where the instruction `outputSell = spotPrice-value;` is deleted. Cleary the execution of that contract would generate a money pump for the holder of the Buy position as he gets credited for each time the spot prices is larger than the value without charging the holder of the Sell position of the corresponding amount. To avoid such cases, an additional requirements must be checked: brokers executing the contract to mark it to market must ensure that the *aggregated* outputs amount to zero. This check is essentially implicit in the current implementation of distributed PTNs, but it must be included as a explicit protocol step in the contract-as-program PTN. We will examine this facet of active contracts in follow up work.

9. Conclusions

In this paper we have provided a systematic review of both traditional payment systems and their analogues in decentralized ledgers. We have dissected the different technological features and associated them to the corresponding business and financial requirements. In this way we have highlighted the essential characteristics of a distributed Payment Transactions Networks and, above all, identified possible novel combination of these features.

In particular, a key property of a distributed Payment Transactions Network is to ensure that the ledger can guarantee global consistency against nodes that either by malice or by sloppiness

undermine the correct functioning of the ledger (for instance to double-spend). Failure to address such issue at the forefront of any distributed payment system is a road to a systemic failure akin to the failure of a central bank in a traditional system.

An interesting follow-up of this observation is the idea of distributed computations, whereby the transactions in the blockchain need not be simply stored but actually be executed. We discussed the idea of a *derivatives-contract-as-program* that is marked to market (or an account that is margined) automatically by computations run on, and whose ownership transitions are recorded, in the distributed ledger. This proposal opens up new possibilities and new challenges.

Acknowledgement

We would like to thank the participants of the TechUK seminar on Blockchain technologies (Blockchain and Beyond) and in particular Richard Adams (Univ. of Surrey), Philip Bond (Univ. of Oxford), Martin Glasspool (Government Office for Science), Leda Glyptis (Sapient Ltd), Phil Godsiff (Univ. of Surrey), Craig Hogan (Cisco Systems Ltd), Jonathan Liebenau (London School of Economics and Political Science), Alistair Milne (Loughborough University), Glenn Parry (Univ. of West of England), and Peter Randall (SETL) for useful discussions.

We would also want to express our gratitude towards Marko Vukolic (IBM Research - Zurich) for the valuable feedbacks.

References

[1] Anderson, M. (1998). The electronic check architecture. Technical report, Financial Services Technology Consortium.

[2] Baldoni, R., S. Bonomi, and A. S. Nezhad (2011). *An algorithm for implementing BFT registers in distributed systems with bounded churn*, Volume 6976 LNCS of *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 32–46.

[3] Bank of England (2015). One bank research agenda. `http://www.bankofengland.co.uk/research/Documents/onebank/discussion.pdf`. Accessed: 2015-12-30.

[4] Bitcoin.org (2015). 11/12 march 2013 chain fork information. `https://bitcoin.org/en/alert/2013-03-11-chain-fork`. Accessed: 2015-12-30.

[5] Bitcoin Wiki (2015). Bitcoin mt. gox. `https://en.bitcoin.it/wiki/Mt._Gox`. Accessed: 2015-12-30.

[6] Bonneau, J., A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten (2015, May). Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pp. 104–121.

[7] Chandra, T. D. and S. Toueg (1996). Unreliable failure detectors for reliable distributed systems. *Journal of the ACM (JACM) 43*(2), 225–267.

[8] Chaum, D. (1982). Blind signatures for untraceable payments. In *Advances in cryptology*, pp. 199–203. Springer.

[9] Chaum, D., A. Fiat, and M. Naor (1990). Untraceable electronic cash. In *Proceedings on Advances in cryptology*, pp. 319–327. Springer-Verlag New York, Inc.

[10] CME Group (2015). Eurodollar. `http://www.cmegroup.com/confluence/display/EPICSANDBOX/Eurodollar`. Accessed: 2015-12-30.

[11] Coindesk (2014). Stellar network fork prompts concerns over ripple consensus protocol. `http://www.coindesk.com/stability-questions-dog-ripple-protocol-stellar-fork/`. Accessed: 2015-12-30.

[12] Danezis, G. and S. Meiklejohn (2015). Centrally banked cryptocurrencies. Technical report, University College London.

[13] Dwork, C., N. A. Lynch, and L. Stockmeyer (1988). Consensus in the presence of partial synchrony. *Journal of the ACM (JACM) 35*(2), 288–323.

[14] Ethereum (2015). A next-generation smart contract and decentralized application platform. `https://github.com/ethereum/wiki/wiki/White-Paper`. Accessed: 2015-12-30.

[15] Eyal, I. and E. G. Sirer (2014). Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security*, pp. 436–454. Springer.

[16] Fischer, M. J., N. A. Lynch, and M. S. Paterson (1985). Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM) 32*(2), 374–382.

[17] Fujisaki, E. and K. Suzuki (2007). Traceable ring signature. In *Public Key Cryptography–PKC 2007*, pp. 181–200. Springer.

[18] Gilbert, S. and N. A. Lynch (2012). Perspectives on the cap theorem. *Computer 45*, 30–36.

[19] Jakobsson, M. and A. Juels (1999). Proofs of work and bread pudding protocols (extended abstract). In *Secure Information Networks*, Volume 23 of *IFIP — The International Federation for Information Processing*, pp. 258–272. Springer US.

[20] Jones, S. P., J.-M. Eber, and J. Seward (2000). Composing contracts: an adventure in financial engineering. In *ACM SIGPLAN Notices*, Volume 35–9, pp. 280–292. ACM.

[21] Karame, G. O., E. Androulaki, and S. Capkun (2012). Double-spending fast payments in bitcoin. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pp. 906–917. ACM.

[22] Katz, J. and Y. Lindell (2014). *Introduction to modern cryptography*. CRC Press.

[23] Lamport, L., R. Shostak, and M. Pease (1982). The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS) 4*(3), 382–401.

[24] Lynch, N. A. (1989). A hundred impossibility proofs for distributed computing. In *Proceedings of the eighth annual ACM Symposium on Principles of Distributed Computing*, pp. 1–28. ACM.

[25] Massias, H., X. Serret-Avila, and J.-J. Quisquater (1999). Design of a secure timestamping service with minimal trust requirement. In *the 20th Symposium on Information Theory in the Benelux*. Citeseer.

[26] Miers, I., C. Garman, M. Green, and A. D. Rubin (2013). Zerocoin: Anonymous distributed e-cash from bitcoin. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pp. 397–411. IEEE.

[27] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Technical report, Unknown.

[28] Percival, C. (2009). Stronger key derivation via sequential memory-hard functions. Self-published.

[29] Rabin, M. O. (1983). Randomized byzantine generals. In *Foundations of Computer Science, 1983., 24th Annual Symposium on*, pp. 403–409. IEEE.

[30] Raynal, M. and M. Singhal (1996). Logical time: Capturing causality in distributed systems. *Computer 29*(2), 49–56.

[31] Ripple Labs (2015). Executive summary for financial institutions. `https://ripple.com/solutions/executive-summary-for-financial-institutions/`. Accessed: 2015-12-30.

[32] Saberhagen, N. v. (2012). Cryptonote v 1.0. `https://cryptonote.org/whitepaper_v1.pdf`. Accessed: 2015-12-30.

[33] Schwartz, D., N. Youngs, and A. Britto (2014). The ripple protocol consensus algorithm. Technical report, Ripple Labs Inc White Paper.

[34] Target 2 (2015). T2s graphical user interface - business functionality. `https://www.ecb.europa.eu/paym/t2s/pdf/2015-02-18_bfd_v2_0.pdf`. Accessed: 2015-12-30.

[35] Vukolić, M. (2016). *The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication*, pp. 112–125. Cham: Springer International Publishing.

[36] Wang, X., Y. M. Teo, and J. Cao (2008). Message and time efficient consensus protocols for synchronous distributed systems. *Journal of Parallel and Distributed Computing 68*(5), 641 – 654.