Distributed multi-ledger and asset exchange model for the financial industry

Pavel Kravchenko, Vladimir Dubinin
pavel@distributedlab.com, dubinin@distributedlab.com

Abstract

This paper points out problems that financial institutions have when applying blockchain ideas to their needs, such as limited **scalability**, **trust** needed to (semi-)anonymous validators, low **speed** of transaction settlement, limited **privacy** of trading positions, absence of **AML** and **KYC** mechanisms. All these issues are consequences of the trustless nature of existing blockchain implementations such as Bitcoin, Ethereum and (partially) Ripple that work under the assumption that there should not be a central party that controls the system. This implies the presence of a single blockchain/ledger and global consensus over it. While we fully agree on the need for decentralized financial infrastructure, we believe that the techniques that are used in Bitcoin etc are not directly applicable to it. The basic mechanism described in this paper is the following - 1) each entity has its own ledger(s) that contains its transactions and balances 2) there is no global shared ledger at all 3) there is no global consensus 4) there are no designated validators - financial institutions (gateways and other trusted parties) are validators. To simplify things you can imagine the existing state of the things when each business and individual has its own financial ledger and they are updated individually and then apply the blockchain data structure, cryptographic techniques and a standard set of communication protocols to it. Basically we are describing the financial internet.

1 Introduction

The financial industry needs a unified and standard set of open-source protocols to trade and store information about accounts (for easy and provable audit) and settlement. These protocols should be able to support processing of hundreds of thousands transactions per second, have built-in AML/KYC mechanisms and identification techniques, protection of information about trading positions, provable finality of a trade, minimal trust to third parties.

2 Issues of existing consensus models applied to needs of the industry

1. Bitcoin's blockchain cannot be considered suitable for the financial industry because of 1) the anonymity of validators, 2) the longest chain rule that allows anonymous actors to change the past (51% attack), 3) low transaction speed and 4) the requirement for all details of all transactions to be replicated to all participants indiscriminately and without encryption
2. The Ripple consensus mechanism does not provide information for the user whether the rest of the network agrees with user's UNL state.
3. Both above-mentioned approaches imply the presence of a single blockchain/ledger that is impossible to scale to the need of the whole world.
4. Similarly, achieving global consensus about each transaction, as well as changes of the protocol, are hardly imaginable.

In addition to that:

1. final users don't have a way to analyze/prove situations that caused problems (forks).
2. financial institutions face privacy problems when working on the same blockchain - all their trading positions are public.
3. situation with KYC vs privacy is not clear - on the one hand there is no identity information attached to a public key, on the other hand the blockchain is open so everybody can analyze transaction graphs.

3 A new model and protocol of asset emission, registration and trading

3.1 Risk model and assumptions

1. There are no designated validators. In our model, validators are gateways, banks and regulators. The number of validators is obviously limited.
2. The validators are responsible solely for promising never to sign a transaction that would result in a double-spend and for validating that the business rules associated with the transaction were implemented correctly. Validators that operate in some regulated environment are legally punishable for their actions. Validators that represent non-regulated entities (i.e. merchants that issue loyalty points for their customers) risk only the trust of their customers.
3. All validators can establish trusted relationships with other validators by exchanging/certifying public keys with each other. This will allow their users to pay/exchange currencies.

4. The process of establishing trust (between validators, validators and regulators, validators and users) is beyond scope of the system.
5. There are no anonymous parties in the system. Each public key can be traced down to its owner (but it doesn't imply the presence of global address book). Validators' public keys are either self-signed or signed by the central authority in central jurisdiction. Signed transaction could be proven legally binding.
6. There is no global identity provider or identity storage. Users are registered with certain validators and the correspondence between username (such as john@citi.com) and public key is stored in a database that a validator (e.g. Citi) maintains. A validator provides limited access to this database to other validators/users. (see Pic. 1)
7. There is no need for a single central authority. The network is based on trusted p2p relationships between validators (financial institutions). Trust is ability to provide credit to each other. But at the same time, any kind of (hierarchical) relationships can be established using the same set of protocols. (see Pic.
8. DDoS attack is not applicable, since everybody knows who is initiating (or co-signing) the transaction.
9. Validators don't obligatory share the state of their ledgers with each other and don't achieve consensus within the whole set of validators.

3.2 Terms

Consensus - an agreement between some parties about a transaction and corresponding state of the accounts involved. It is explicitly expressed via digital signature under the transaction. It is verifiably provable - because of the cryptographic nature of the signatures and identification of the public keys owners.

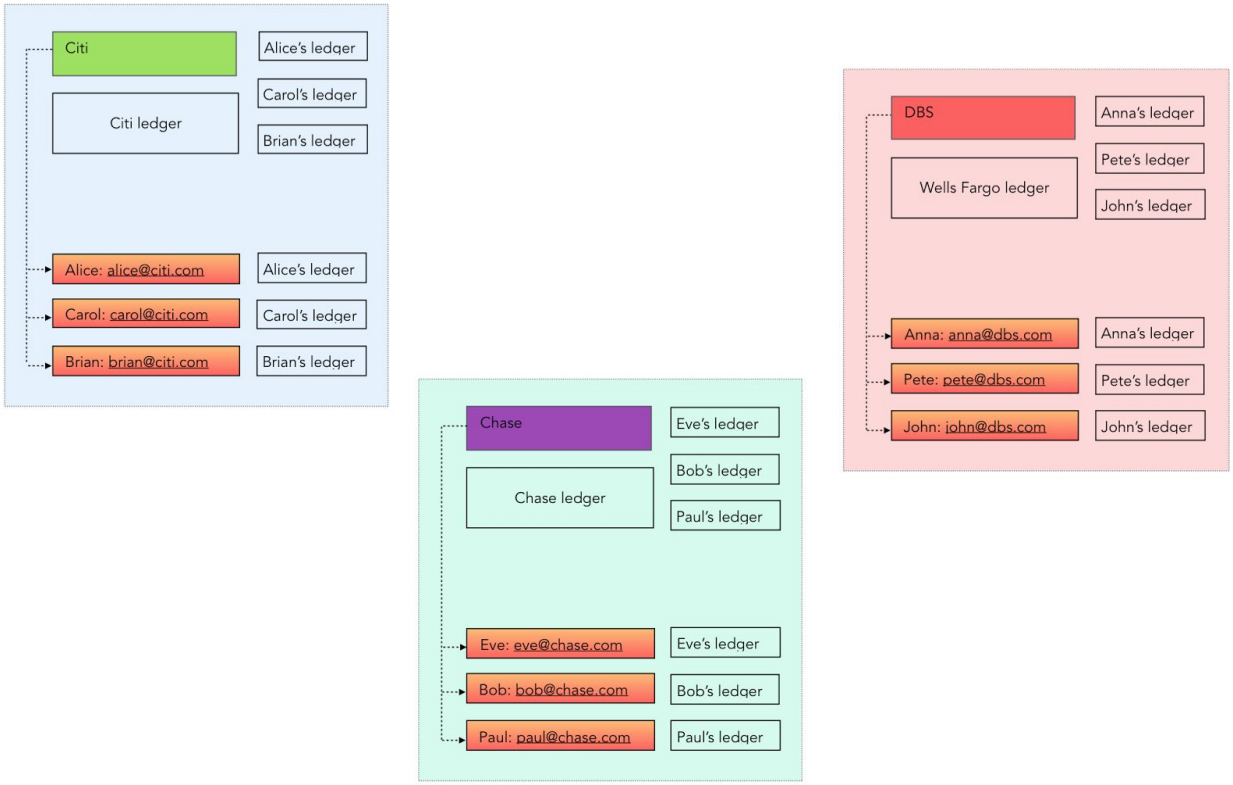
Unified asset identifier (UAI) - unique identifier (may include namespace) that is generated during asset emission and attached to it during the whole lifecycle of the asset. Used to refer to a certain asset.

Unified blockchain identifier (UBI) - needed to search for and refer to a certain blockchain (ledger).

Asset life cycle - process of asset creation, exchange, distribution over multiple blockchains, locking, deletion.

TChain - transaction chain, when transactions are validated individually (without blocks).

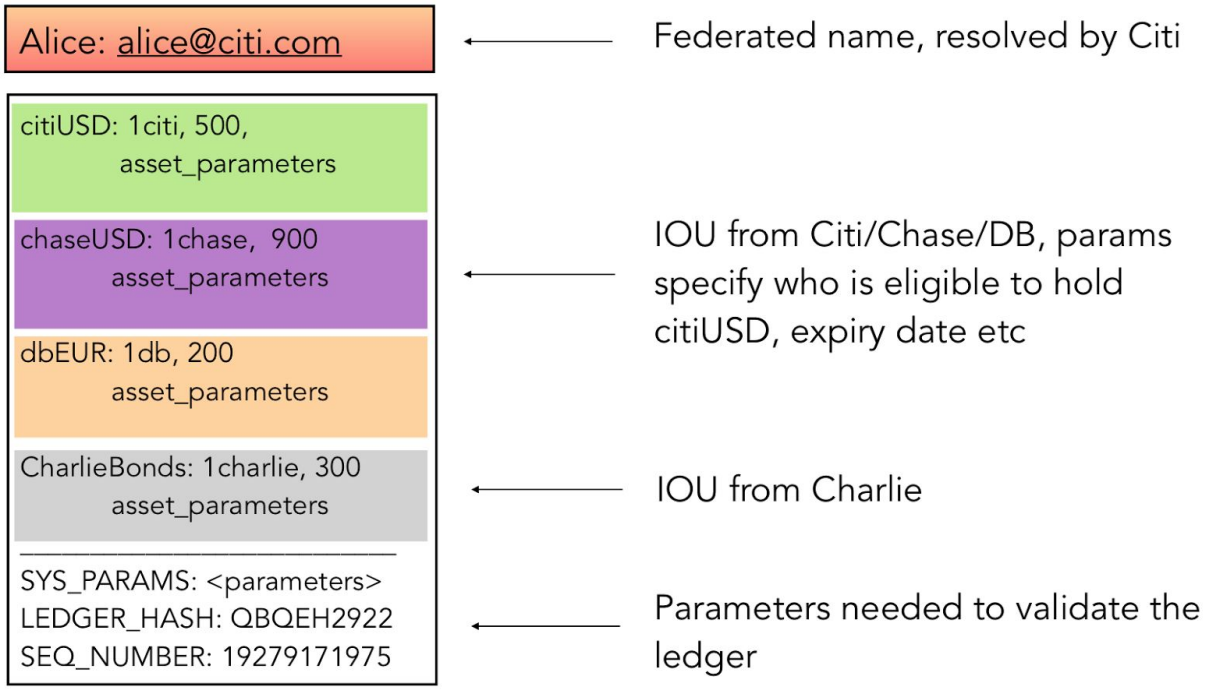
3.3 Architecture



Pic. 1 - Each entity has its own ledger

1. Each business or individual maintains a separate ledger and chooses its own validator (for example the bank where their account is held).
2. The validator keeps a copy of the ledger* for each its customer.
3. It is possible to maintain the entire ledger, or some part of it, privately.
4. Assets are ledger-independent and are referred to by their UAI.
5. Banks perform KYC and store identity data on the federation servers.
6. To access public blockchain data or identity data stored by the validator, one needs to know the public key of the validator and the domain name. The federation API is used to access that.

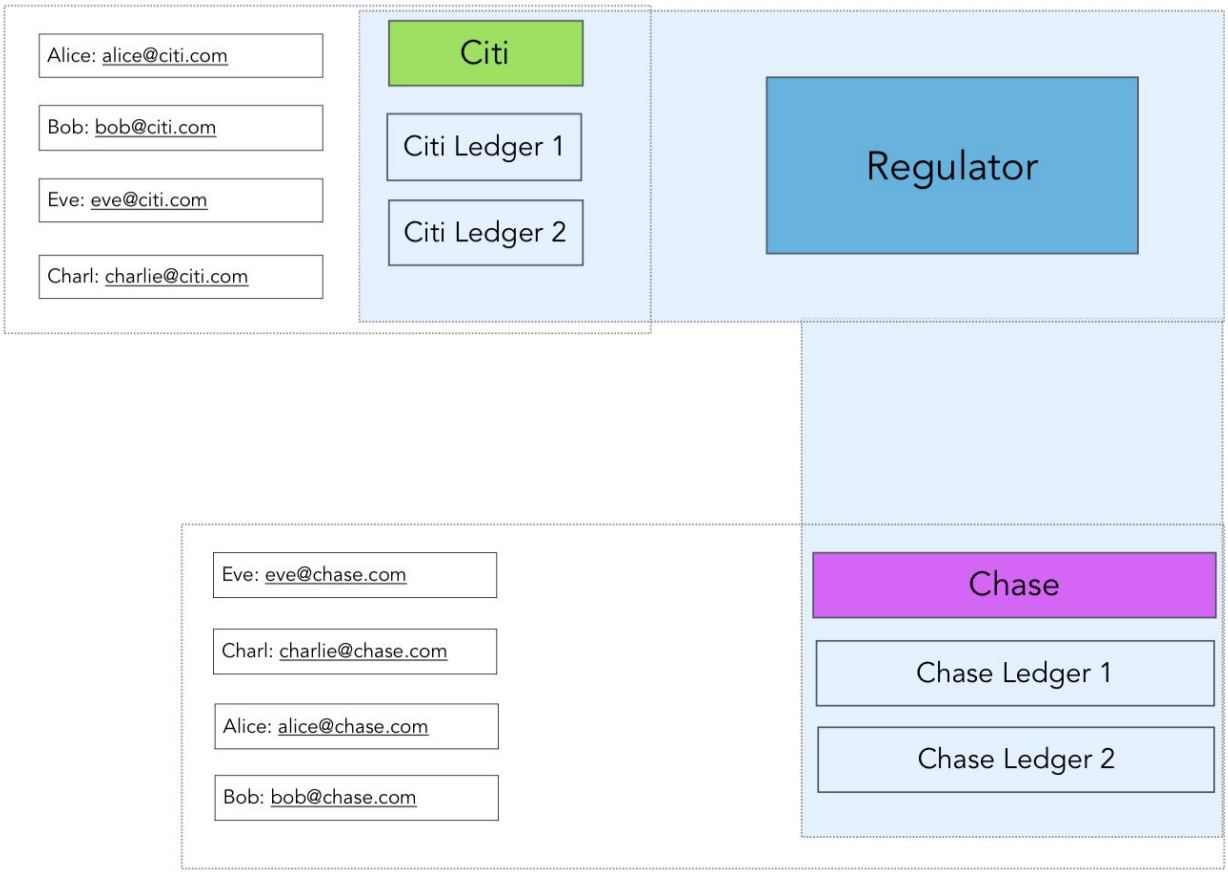
*Trust in the validity of a certain ledger is based on trust in a particular validator and (in some cases) on non-contradiction of this ledger with all other ledgers.



Pic. 2 - Ledger structure

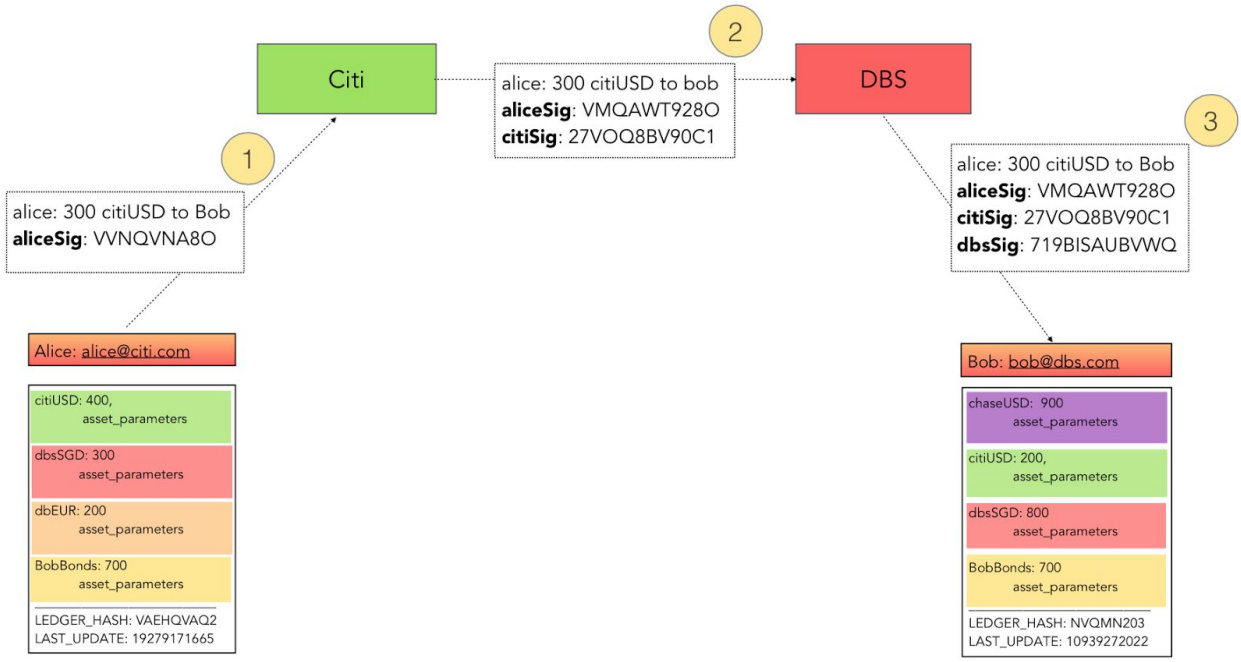
3.4 Process of ledger lifecycle

1. Each user selects validator(s) and creates a ledger.
2. Each ledger contains system parameters such as: system cryptographic parameters, rules of asset emission, <list of other possible validators, back-up rules, rules of payment (frequency, required KYC for the recipients etc)>.
3. The user creates/deposits assets at the bank (the same idea as IOU in Ripple, bank = gateway). The ledger may contain many assets owned by the user.
4. In order to initiate a transaction (emission of an asset, payment or exchange offer) the user has to sign a transaction and submit it to the validator to sign. It is similar to multisignature, but has different meaning in our case - the bank is needed to guarantee compliance and perform 2FA.
5. As soon the validator signs the transaction, it is applied to the ledger (both user's and bank's copies).
6. The user is able to verify and prove that a given transaction is final by referring to the signature of the validator.
7. Other users/validators may able to access certain ledgers and verify the correctness of some balance.

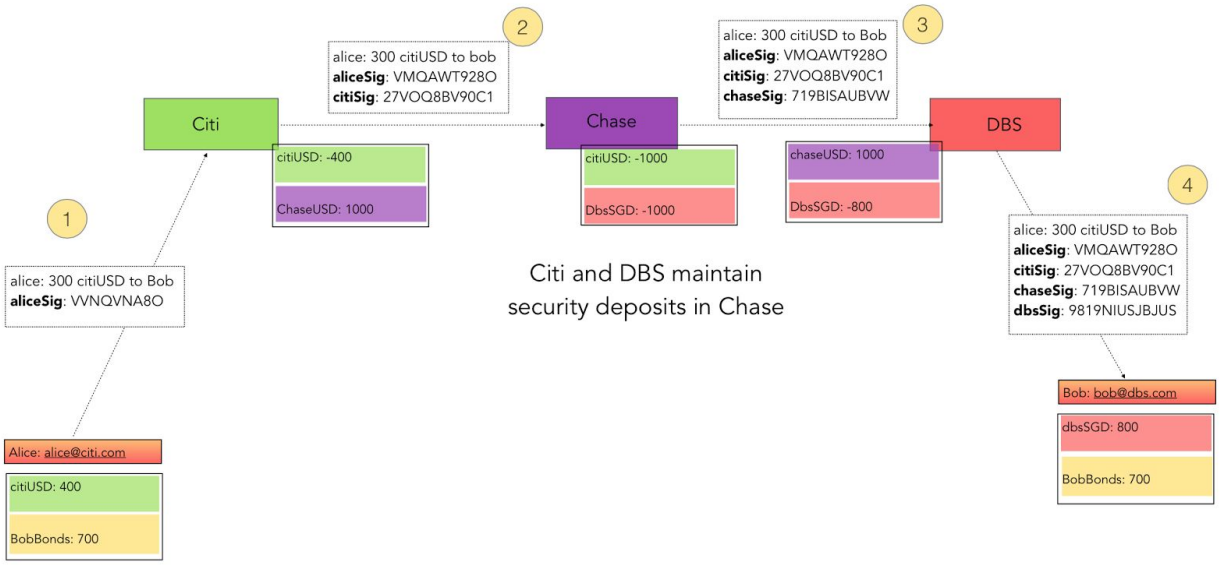


Pic. 3 - Ledger of customers are shared with their banks and ledger(s) of banks are shared with regulator(s)

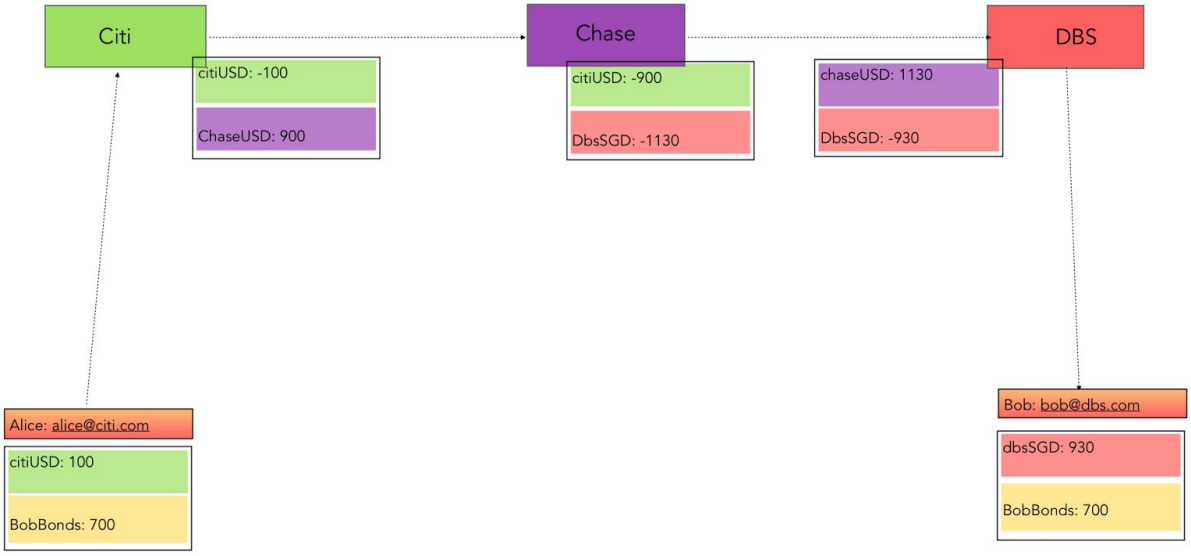
4 Payment flow



Pic. 4 - Payment flow. Signatures of both banks (sender's and recipient's) are needed to validate the transaction



Pic. 5 - Payment and updating ledgers when a correspondent/central bank is used. Initial state of the ledgers.



Pic. 6 - State of the Ledgers after the transaction

Note: There is no way to send/exchange assets in case there is no trust (even transitional) between gateways.

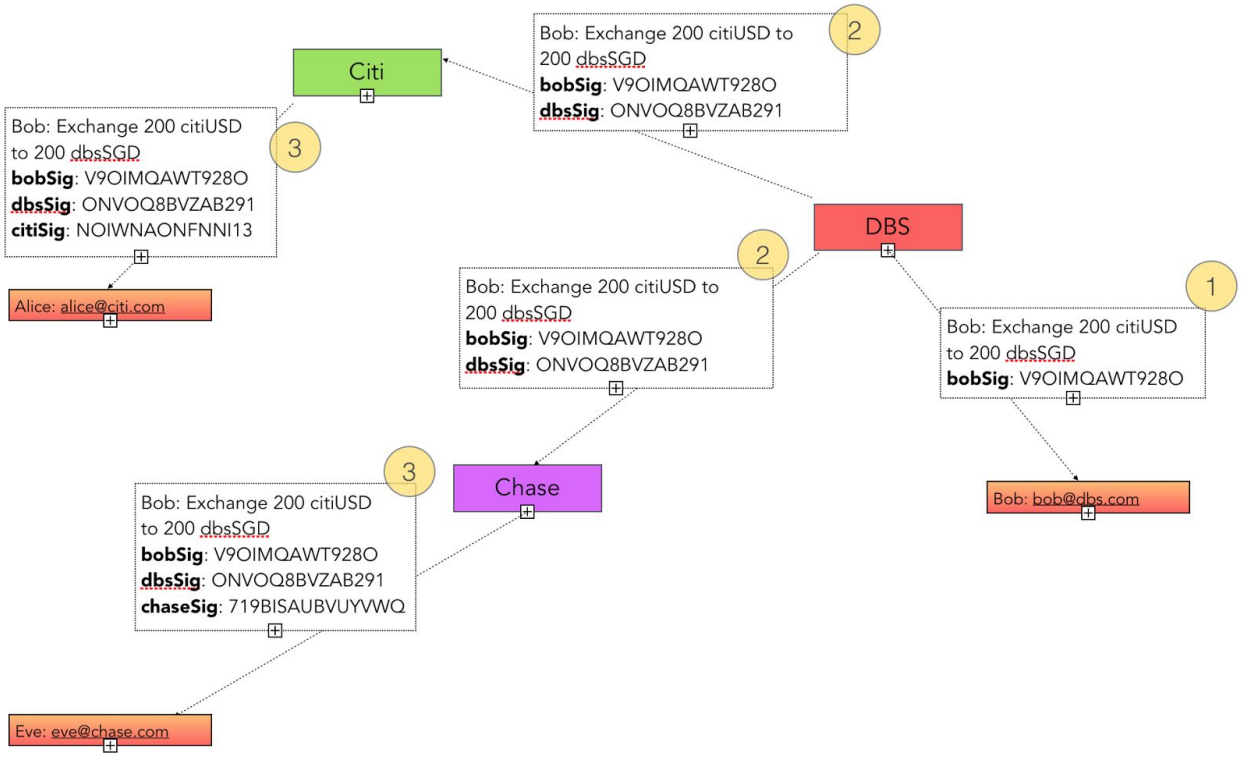
5 Exchange flow

1. Payment flow within the same ledger - straightforward (as soon as all signatures of sender and validator are obtained, the transaction is considered final)
2. Exchange flow within 2 ledgers:

Terms: user #1 controls ledger #1 and asset #1 and is KYC'd by bank #1
user #2 controls ledger #2 and asset #2 and is KYC'd by bank #2

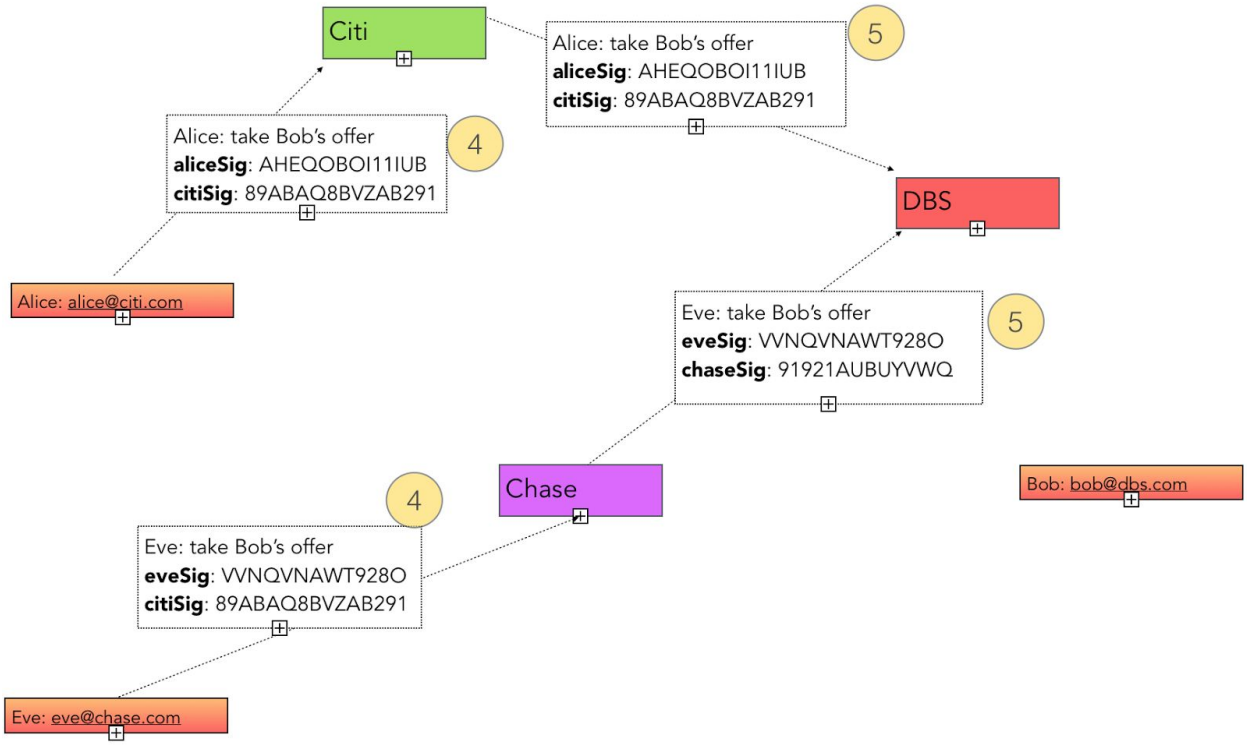
1. User #1 creates an offer to exchange asset #1 against asset #2 and signs it.
2. Bank #1 signs the offer.
3. User #2 sees that asset #2 was mentioned in some offer (search is done by UAI by bank #2 - user #2 has to sign-up to see all relevant offers).
4. User #2 decides to accept this offer and creates a corresponding transaction.
5. Bank #2 validates the transaction of user #2 and passes it on to bank #1.
6. Bank #1 signs the transaction that represent offer acceptance, notifies user #1 and updates ledger #1, then notifies bank #2.
7. Bank #2 sees the notification and updates ledger #2.

8. If bank #2 doesn't update ledger #2 (should bank #1 check that?), then bank #1 can prove that there was fraud.
9. In case users (#3, #4 …) want to make an offer to user #1, it is up to bank#1 to select the best one.

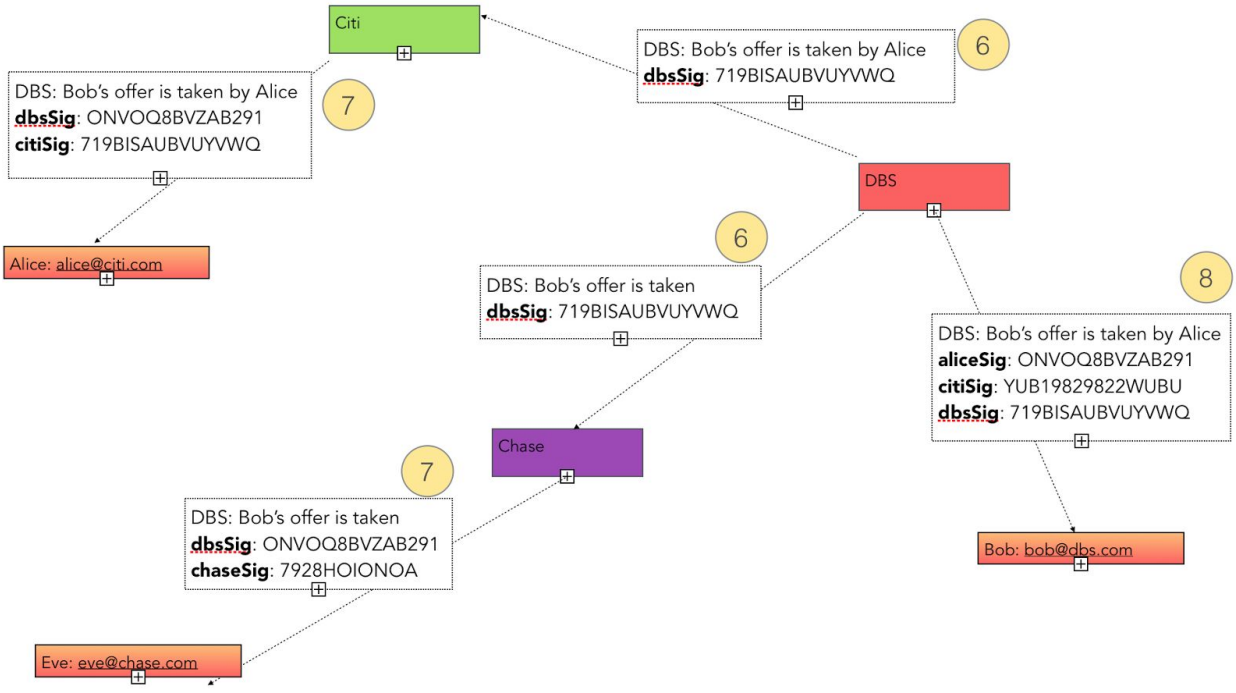


Pic.7 - Offer creation flow.

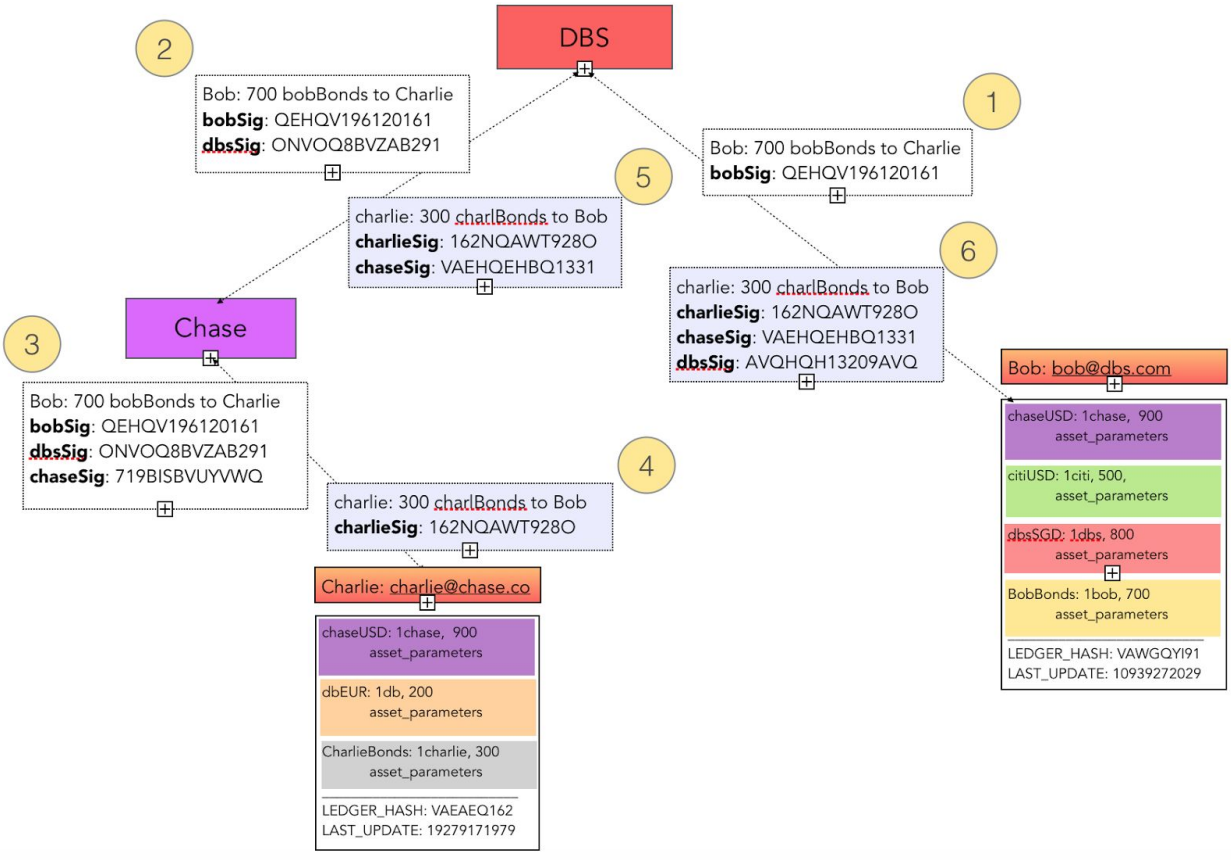
Security considerations: User #2 doesn't have to wait until user #1 updates his ledger, because he/she has a proof (which is included in the transaction) that contains signatures of bank #1.



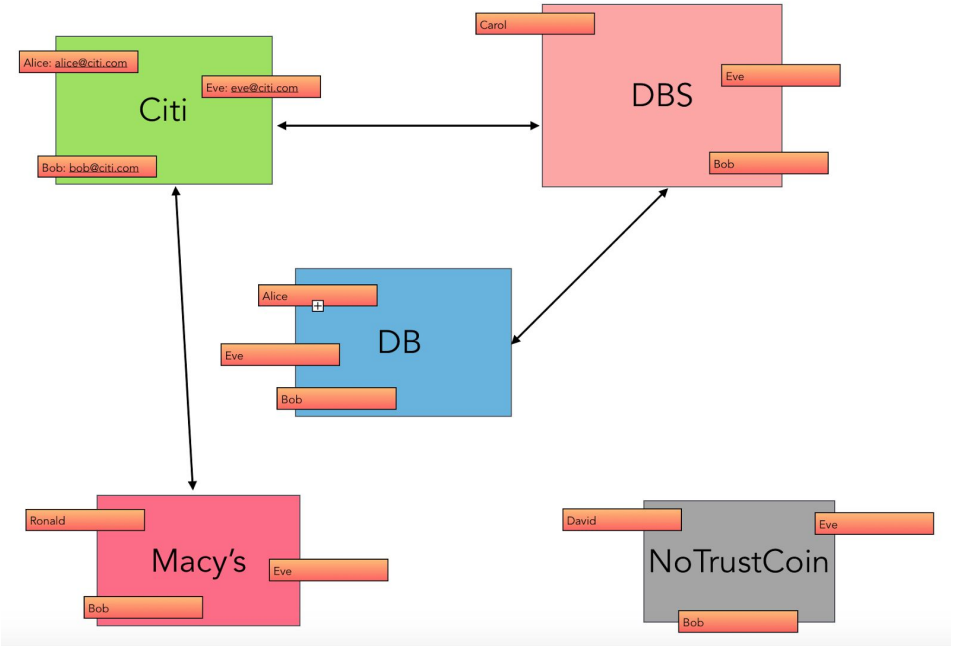
Pic.8 - How a trading offer is spread



Pic. 9 - Bob's bank approves only one transaction and updates Bob's ledger



Pic. 10 - Trade usecase



Pic. 11 - There is trusted relationships between all gateways except NoTrustCoin

6 Frequently asked questions

Question: The system has to search through the offers ledgers to find a payment path / execute a payment. What is system in this case?
Answer: The system is a set of protocols and executable code that is able to negotiate the conditions of the trade and to handle edge situations. It includes decentralized search.

Double spending prevention

Question: How can you provide proof that you've destroyed (or frozen) something in another place and you cannot sell your assets twice?
Answer: Since you don't exchange asset in one ledger, but it can be stored in many ledgers that support the same UAI and UBI namespace/format, you don't need to do anything specific to move asset to another ledger (which will be controlled by other validators).

Question: Can blockchain users be sure that you haven't issued electronic assets twice?
Answer: Your validator is responsible for verifying that an asset was issued only once and for maintaining physical custody over them (if applicable).

Question: Why is a validator/bank/trustee needed at all?
Answer:

1. It verifies the correctness of the consensus (double spending etc).
2. It verifies KYC of the counterparty.
3. It authenticates users if needed (daily limits etc).
4. It verifies that the user physically owns assets that it claims to own.

Architecture questions

Question: Does a validator maintain single database for all its customers, or many?
Answer: Not decided yet, but there will probably be different databases, so that each of them (duplicated on the customer's side) can have a unique blockchain ID.

Question: What is the blockchain/database?
Answer: It is TChain - a transaction chain, with each transaction is linked to its predecessor and validated individually. Each party has a different blockchain, including only relevant transactions.

Question: Will there be native currency in the system?
Answer: No.

Question: How many blockchains can I have with single validator?
Answer: As many as you want.

Question: Do you need trustlines (like in Ripple) in the system?
Answer: No. At least not explicitly.

Question: Does a validator sign consensus (basically transactions) with each customer individually, or with all of them simultaneously?
Answer: Individually, especially if we will select an individual blockchain architecture for the validator.

Question: Can a validator stop signing my transactions? How can I prove that it happened?
Answer: Yes, it can. We expect that there will be a specific response that notifies users about this situation.

User-Validator relationships

Question: How will a validator prove that you are you?
Answer: You will sign your personal data during the registration with your own private key.

Question: How will I know that certain signer is a validator?
Answer: We will use principles of PKI, web-of-trust, PGP. Validators will know about each other and their corresponding public keys.

Question: Do we need central validators (central banks)?
Answer: Central validators may play role of CSDs that track ownership of each asset for a particular country and provide quick access about it for the foreign actors.

Question: Who is going to store all the transactions of a user who died/closed the company etc?
Answer: Their validator. The data should be stored forever.

Question: How to organize decentralized trading in our case?
Answer: Offers have to be spread througout the network, so all interested parties can get this information. Since validators store a list of all trusted validators (and potentially untrusted too) it is possible to broadcast offers (and their originator, so the interested party is able to contact offer initiator directly) through a p2p network. Other option is to create set of sub-Ripples - network

that maintains single ledger and clear transaction really fast. It means that the whole system will consists of many cores that are formed ad-hoc, based on trusted relationships.

Question: How can I check the solvency of the user that makes the offer?
Answer: The user's validator verifies balances, but as in Ripple we are working with IOUs, so the presence of physical assets is guaranteed by the issuer.

Question: How will a user start working with the validator?
Answer: The validator just creates a ledger with the user and starts signing particular transactions. The actual registration process is beyond the scope of our system, because of its trivial nature and the analogy to current registration mechanisms.

Question: How will identities be managed in the system?
Answer: We will use the federation server technique to manage identities. Each gateway/bank maintains a set of identities related to them. Identities verified by a trusted validator can be reused by external parties (with maybe a lower trust level).

Acknowledgment

The authors would like to thank Richard Brown, Alex Kampa, Jeremy Drane, David Lee, Mikkel Larsen, Ryan Singer, Robert Sams, Robert David, Vlad Zamfir, Patrick Salami, Tim Swanson, Stanislav Cherviakov and Alex Oberhauser for passionate discussions and productive feedback.