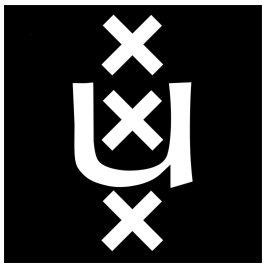


University of Amsterdam

Master Thesis

Software Engineering

Facilitating Peer-to-Peer Decentralized Transport Contracts

Student Name:	Dylan Bartels
Student Number:	10607072
Supervisor:	Jaap van Ginkel
Host Organization:	CargoLedger

July 5, 2018

Abstract

Amid growing centralization in the economy and progressing technological advancement realising decentralized alternativesThe logistical domain operates on certain trusting ways, transportation of physical object means a custodian has to be trusted. This paper suggest a trustless insurance mechanism which could minimalize intermediation and conflict resolution due to a trustless incentive structure.

Amid growing centralization in the economy and progressing technological advancement realising decentralized alternativesThe logistical domain operates on certain trusting ways, transportation of physical object means a custodian has to be trusted. This paper suggest a trustless insurance mechanism which could minimalize intermediation and conflict resolution due to a trustless incentive structure. amid growing boundry of entry and rising costs to participate in the logistics domain

this paper introduces a trustless decentralized incentive structure to sucomd reputation systems and centralized intermediation. We created a working proof of concept operating on the Bitcoin network to lower the participation boundary and create the possibility for everyone to participate if they reach agreement on the constructed Ricardian Contract.

Contents

1 Introduction **3**
- 1.1 Initial Study . . . 3
- 1.2 Problem Statement . . . 3
- 1.3 Related Work . . . 4

2 Background **7**
- 2.1 Bitcoin . . . 7
- 2.2 Ricardian Contracts . . . 8
- 2.3 Decentralized Marketplace . . . 8

3 Mechanism Setup **9**

4 OpenLogistics **12**
- 4.1 Transparency . . . 12
- 4.2 Advantages & Disadvantages . . . 13
- 4.3 Claim . . . 13

5 Evaluation **14**
- 5.1 Research Questions & Answers . . . 14
- 5.2 Evidence . . . 15

6 Conclusion **17**
- 6.1 The Challange of Data Integrity . . . 17
- 6.2 Relevance . . . 17
- 6.3 Future Challanges . . . 17

A List of Symbols **18**

B Ricardian Contract **19**

C Bitcoin Mempool Anomaly **20**

D Transaction Script Byte-map **21**

1 Introduction

The gig economy is in full effect, individual actors get paid for the execution of short term contracts and centralized companies intermediate in the supply and demand of this labour. With the intermediation of these parties the companies profit of margins and deny individuals full ownership of the value of the produced labour. Recent advancements in peer-to-peer technologies and decentralized possibilities are of interrest for innovative logistical service providers and the academic domain looking for alternative options to shift towards decentralized alternatives.

1.1 Initial Study

Recent successful technical innovations have been due to a shift from centralization to peer-to-peer services, examples of these are Uber, Airbnb and Kickstarter. In the domain of supply chain logistics this innovation has been lacking. O. Gallay et al. [3] have proposed a peer-to-peer framework supporting interoperability between different actors in the logistics chains. This research lacks insight related to trust, network and technical implementation but establishes interest in the domain regarding implementation.

According to N. Hackius et al. [5] surveys in the logistics domain show that there is a clear demand on what blockchain technology can realistically do for the domain.

With the recent progression in the domain of trustless value transference, research towards the applicability of this in the supply chain offers relevancy. In [7] M. Klems et al. have formulated possible implementation of trustless intermediation in blockchain based decentralized service marketplaces. The research topic arises if this or other intermediation solutions can also be applied to peer-to-peer logistics marketplaces and to which degree will there be a custodian in the process due to it including transferring a physical asset.

1.2 Problem Statement

The problem which will be explored in this study is how to trust actors to transport a physical object without trusting centralized intermediation and reputation systems being a necessity. Currently the transport domain operates around centralized reputation systems, whereby the companies with aggregated reputation and trust offer the service and carry responsibility for conflict resolution.

However reputation loss might not be the only incentive available to achieve transport. In chapter 4 an alternative incentive construction will be demonstrated which aims to achieve decentralization, reduction on trusting reputation systems and every actor being able to fulfil every role in transport. The setup uses trustless escrow to lock the transport actor into not behaving hostile due to possible punishment. Deviation of rational behaviour would result in loss of value to counteract the unavailability of punishing reputation.

1.2.1 Research Questions

We examine the proposed mechanism from the perspective of the following research question:

- How can you provide a mechanism to facilitate peer-to-peer decentralized trustless transport contracts?

To contribute to a clear view of what the mechanism provides we state the following subquestions:

- Can trustless intermediation exists on this marketplace without centralized dispute resolution?
- How decentralized is the mechanism?
- To what degree do oracles play a role in the verification of the information?
- What level of anonymity is possible?
- Which attack vectors are possible to undermine this mechanism?

1.2.2 Solution Outline

Our solution uses a digital representation of the transport contract which includes the begin address, end address and the end address public key. This contracts will be tied to the owner of the physical object which is intended to be transported with the process. When the transport actor is custodian of transporting the asset an equivalent value or more will be put in escrow which can be unlocked by the endpoint actor. The setup incentivizes for the transport actor to move the asset to the endpoint else losing the escrow equivalent value.

We chose to implement the escrow on the Bitcoin network due to it being well tested and offering a baseline environment in the decentralized domain.

1.2.3 Research Method

The study will apply the action research methodology research method. Action research can be defined as an approach in which the action researcher and a client collaborate in the diagnosis of the problem and in the development of a solution based on the diagnosis. With this method a prototype of the marketplace and transport intermediation solution will be build in collaboration with Cargoledger. The methodology has the downside that biases might occur towards the chosen solution due to also being responsible for the development.

1.3 Related Work

1.3.1 Building Trust in Decentralized Peer-to-Peer Electronic Communities [12]

Summary: Many players in electronic markets have to cope with much higher amount of uncertainty as to quality and reliability of the products they buy and the information they obtain from other peers in the respective online business communities. One way to address this uncertainty problem is to use information such as feedbacks about past experiences to help making recommendation and judgment on product quality and information reliability. This paper presents PeerTrust, a simple yet effective reputation-based

trust mechanism for quantifying and comparing the trustworthiness of peers in a decentralized peer-to-peer electronic marketplace. There are three main contributions in this paper. First, we argue that the trust models based solely on feedbacks from other peers in the community is inaccurate and ineffective. We introduce three basic trust parameters in computing trust within an electronic community. In addition to feedbacks in terms of amount of satisfaction, we incorporate the feedback context such as the total number of transactions and the credibility of the feedback sources into the PeerTrust model for evaluating the trustworthiness of peers. Second, we develop a basic trust metric that combines the three critical parameters to compare and quantify the trustworthiness of peers. Third, we present a concrete method to validate the proposed trust model and report the set of initial experiments, showing the feasibility, costs, and benefits of our approach.
Differ from my approach: Xiong & Liu, 2003 focus on the wide domain of trust in decentralized peer-to-peer communities. With my specific case regarding a marketplace with transport of goods trust between the peers is an important aspect but is more contained in a specific domain than Xiong & Liu.
Obtained result: Xiong & Liu, 2003 present PeerTrust a trust mechanism for building trust in peer-to-peer electronic communities. They identified three important trust parameters, these are: amount of satisfaction, number of interactions and balance factor of trust. They put the results into experiments which demonstrated the effectiveness, costs, and benefits of the approach.
Remaining open questions: Looking at ways to make the approach more robust against malicious behaviours, such as collusions among peers. Combining trust management with intrusion detection to address concerns of sudden and malicious attacks. How to uniquely identify peers over time and associate their histories with them.

1.3.2 The challenge of decentralized marketplaces [11]

Summary: Online trust systems are playing an important role in todays world and face various challenges in building them. Billions of dollars of products and services are traded through electronic commerce, files are shared among large peer-to-peer networks and smart contracts can potentially replace paper contracts with digital contracts. These systems rely on trust mechanisms in peer-to-peer networks like reputation systems or a trustless public ledger. In most cases, reputation systems are build to determine the trustworthiness of users and to provide incentives for users to make a fair contribution to the peer-to-peer network. The main challenges are how to set up a good trust system, how to deal with security issues and how to deal with strategic users trying to cheat on the system. The Sybil attack, the most important attack on reputation systems is discussed. At last match making in two sided markets and the strategy proofness of these markets are discussed.
Differ from my approach: Very similair by giving a rundown of all the research done towards trust enforcements in peer-to-peer file sharing, decentralized markets and sybil attacks.
Obtained result: B van Ijzendoorn gives a summary of academical research on decentralization, Sybil attacks, trust and peer-to-peer in relation to marketplaces.
Remaining open questions: Not applied.

1.3.3 A Peer-To-Peer Platform for Decentralized Logistics [3]

Summary: We introduce a novel platform for decentralized logistics, the aim of which is to magnify and accelerate the impact covered by the integration of the most recent advances in Information and Communication Technologies (ICTs) to multi-modal freight operations. The essence of our peer-to-peer (P2P) framework distributes the management of the logistics operations to the multiple actors according to their available computational resources. As a result, this new approach prevents the dominant players from capturing the market, ensures equal opportunities for di erent size actors, and avoids vendor lock-in. The latest ICTs such as Industrial Data Space (IDS), Blockchain, and Internet-of-Things (IoT) are used as basic building blocks which, together, enable the creation of a trusted and integrated platform to manage logistics operations in a fully decentralized way. While IDS technology allows for secured data exchange between the di erent parties in the logistics chain, Blockchain technology handles transaction history and agreements between parties in a decentralized way. IoT enables the gathering of real-time data over the logistics network, which can be securely exchanged between the di erent parties and used for managing the decision-making related to the control of the freight transportation activities. The practicability and the potential of the proposed platform is demonstrated with two use cases, involving various actors in the logistics chains.
Differ from my approach: It is an academic research which originated from the logistics domain and aimed at solving the contradiction between interoperability and data sovereignty.
Obtained result: High level decentralized logistics system architecture with data flows. Two initial use cases.
Remaining open questions: Development of business models in parallel.

2 Background

2.1 Bitcoin

Introduced by the anonymous Satoshi Nakomoto in 2008, Bitcoin (BTC) is a decentralized peer-to-peer currency system. Bitcoin allows digital payments without going through a financial institution and solves the double spend problem by hashing timestamped transactions into an continues chain of hash-based Proof-of-Work (PoW) [8]. Payment are possible by creating transactions, signing them and sending them to Bitcoin addresses. The user has a public/private keypair whereby addresses are a mapping function of the public key and the private key can sign transactions. Creating of transactions is only possible if they have been send to one of the user owned addresses and are then called unspent transactions (UTXO).

The broadcasted transactions are send to the memory pool and included in blocks once the network has formed consensus on the correct order of transactions. To generate a block the miners have to find a nonce value, peers than include it in a block which allows anybody to verify the PoW. Miners get rewarded upon generating a block thus incentivising to support the network with computational power. The generated computational power of the network, also expressed in hash rate or hash power, protects the integrity of the PoW chain. For a user the definition of owning a bitcoin is the right to sign a UTXO with your keypair.

The PoW mechanism solves the double spend problem by guaranteeing that the transaction is not spend twice when it is included in a block. For a malicious actor to double spend a BTC without detection they would have to recompute all previous blocks, so as long as the honest peers in the network exceed the malicious the integrity of the work is guaranteed [8].

The average block time of the bitcoin network is 10 minutes, to guarantee that the double spend would not occur on average the receiver of the transaction would have to wait 10 minutes minus the time of last found block. Fast transactions (i.e. in the order of seconds) are not reliable because low cost attacks can be mounted effectively to spoof a transaction [6].

2.1.1 Trustless

Markus Klems et al. define trustless and trustless intermediation as follows:

> *A system property which guarantees rules of interaction that are known to and agree upon by all participants of the system, and which cannot be unilaterally changed. These guarantees are enforced through, what we call trustless intermediation, a set of mechanisms for decentralizing the enforcement of rules in a system, thereby removing the need for and existence of trusted intermediaries. [7]*

Bitcoin can be defined as a trustless system because once an UTXO is signed and broadcasted to the memory pool with a high enough transaction fee it is guaranteed that it will be put inside a block. No intermediation takes place for this mechanism to occur. The mechanism cannot be changed *easily*, only if a hard forking is proposed including a code change. However this chain split would not count as bitcoin unless it is backed by the majority of hash rate thus being classified as longest chain [8].

Extrapolating the definition to the logistics domain means that trustless logistics can be defined as an logistic contract which has predefined unchangeable rules without the need for trusted intermediaries.

2.1.2 Script

Bitcoin uses a stack-based scripting system for transactions which is intentionally not Turing-complete. A typical Bitcoin address is known as a Pay-to-PubKeyHash (P2PKH) address and can be identified by the 1 prefix, an example of such an address alongside other address types can be found in Chapter 4.1. Table 1

Another important address is the Pay-to-ScriptHash (P2SH) address identified by the 3 prefix. The Pay-to-ScriptHash (P2SH) address type allows any of *N* out of *M* signatures to spend the UTXO available in the P2SH address. To send a P2SH transaction *M* amount of public keys are needed and the required amount of signatures *N* to allow to redeem from the address need to be given. *M-of-N* multisignature P2SH addresses owned by $\{X_{pk}, Y_{pk}\}$ will be denoted as $XY^{m/2}_{adr}$ from now on.

The Bitcoin scripting language provides operators which can be applied to created transaction scripts. The OP_CHECKMULTISIG operators for multisig verification makes it possible to verify a generated transaction script for the content before it is broadcasted.

2.2 Ricardian Contracts

A Ricardian contract is designed to register a legally binding digital document to a specific object [4]. The contract puts all information in a format which is parsable by software and humans. It represents a legal agreement between individuals and a protocol for integrating the agreement securely within a infrastructure.

The Ricardian contract can be used to form an agreement by forming the liability when trading with another party. It can represent a unit of goods or service and uses signed agreement between the parties which cannot be falsified once signed.

Similar to smart contracts [1] Ricardian Contracts can achieve taking out the middle man in contract construction, execution and enforcement. A Ricardian contract is acceptable within the existing legislation frameworks.

2.3 Decentralized Marketplace

Markus Klems et al. define centralized marketplaces[7] as providing mechanisms to facilitate efficient spot trades between numbers of sellers and buyers by providing matchmaking and payment transaction processes that are accompanied by trust-building mechanisms, most importantly, reputation and dispute resolution systems. Some examples of centralized marketplaces for logistics are Postmates for deliveries, Uber for the transport of people and Uship for shipping of goods. These peer-to-peer marketplaces facilitate the intermediation process of transport contracts.

Decentralized marketplaces facilitate the same spot trades without a central provider and intermediaries. The upside of such a mechanism is the lowering of the entry barrier [2], no intermediation fees [13], increasing sensorship resistance [9], and improving privacy [10].

3 Mechanism Setup

As a new idea, this study introduces a framework for trustless insurance of transport contracts thus replacing the need for centralized intermediation. We propose a mechanism which punishes hostile actors automatically resulting in no conflict resolution required from central entities. The meaning of used abbreviations and function structures in this chapter can be found in the list of symbols appendix A.

In our scenario seen at figure 3.1, we assume that the service consumer A wants to send an physical object to the endpoint actor B. The third and last actor is the service provider C who will execute the transport contract. Let A, B and C all have an Bitcoin public-private key and Bitcoin address which is created with a mapping function from the public key.

$$CreateKeypair(A) = \{A_{pk}, A_{sk}\}$$

$$CreateAddress\colon A_{pk} \rightarrow A_{adr}$$

The scenario starts of with A creating a Ricardian contract representing a request for transport, this contract contains A location (A_{loc}), B location (B_{loc}), B public key (B_{pk}), physical object equivalent value (Ev) or more and the transport reward (Tr).

$$\{A_{loc}, B_{loc}, B_{pk}, Ev, Tr\} \subseteq Rc$$

This contract is accepted by C, he will then create and sign an P2SH transaction, denoted by tx_1. This transaction contains Ev to be send from C_{adr} to the 2-of-2 multisig P2SH address of B and C. Signing a Bitcoin transaction with your private key will include a signature (C_{sig}) into the transaction which cannot be decoded back to the private key.

$$CreateAddress\colon \{2, B_{pk}, C_{pk}\} \rightarrow BC^{2/2}_{adr}$$

$$SignTransaction\colon \{Ev, C_{adr}, BC^{2/2}_{adr}, C_{sk}\} \rightarrow tx_1$$

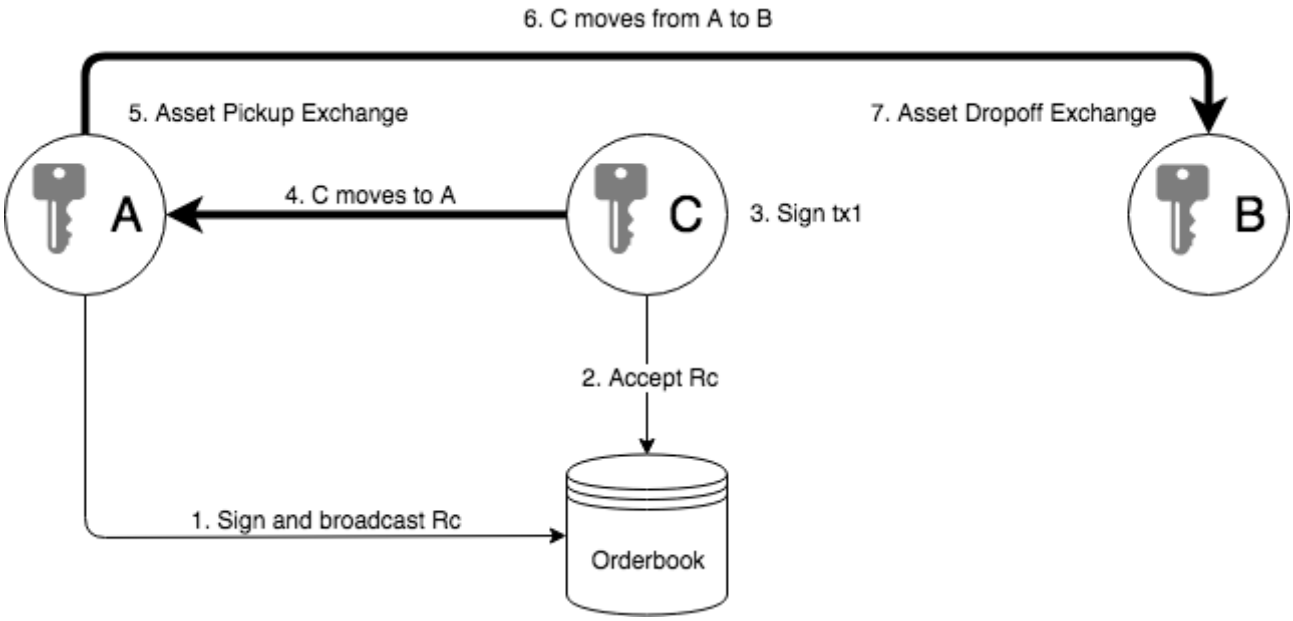


Figure 3.1: Overview test scenario

Upon accepting C moves to the physical location of A bringing the signed transaction script tx_1. As illustrated with figure 3.2, upon C arriving at A the asset pickup exchange can take place.

1. C innitiates the exchange by giving A the transaction script tx_1
2. A creates and signs a transaction from A_{adr} to the multisig address containing the reward for transporting the physical object.

$$SignTransaction\colon \{Er, A_{adr}, BC^{2/2}_{adr}, A_{sk}\} \rightarrow tx_2$$

3. A broadcasts the transport incentive into escrow $\{tx_1, tx_2\}$
4. A signs and broadcasts the ownership change of the Rc from $A \rightarrow C$
5. Wait block confirmation containing $\{tx_1, tx_2\}$
6. Exchange the physical object from $A \rightarrow C$

Before $\{tx_1, tx_2\}$ get broadcasted A or C can individually verify if the signed transactions actually contain what they should.

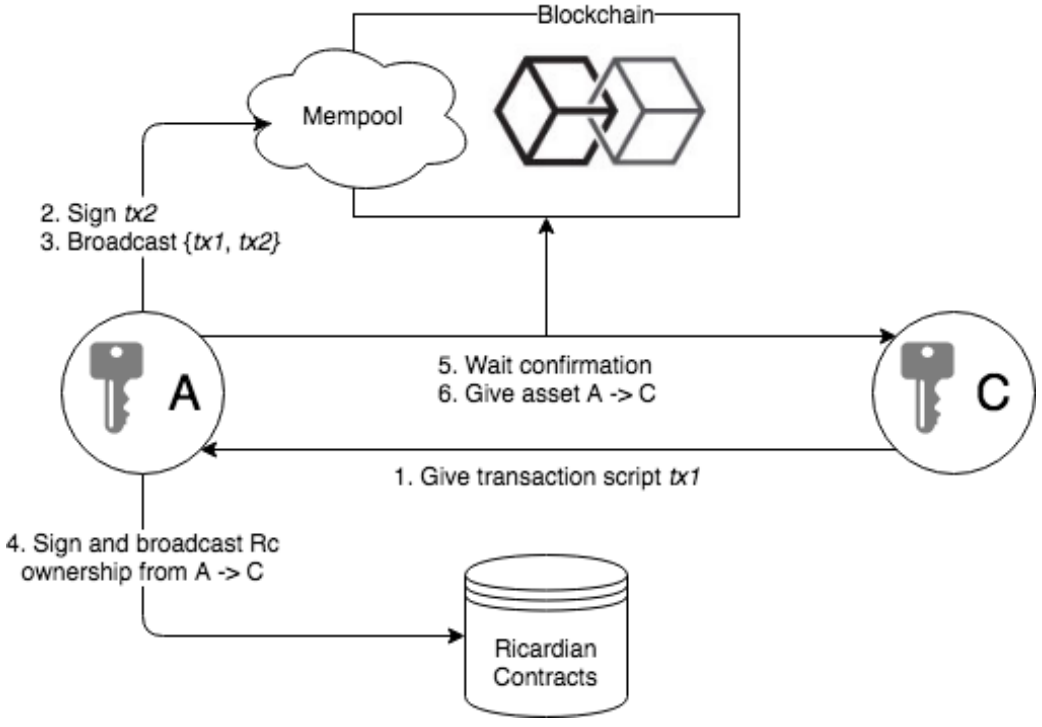


Figure 3.2: Asset Pickup Exchange

When *C* receives the physical object he is the custodian and will move to the endpoint B_{loc} to claim the reward locked in the escrow. As illustrated with figure 3.3, upon *C* arriving at B_{loc} the physical object drop-off exchange takes place which consist out of the following steps:

1. B signs tx_3 containing the equivalent value and transporting reward from $BC^{2/2}_{adr}$ to C_{adr}

$$SignTransaction\colon \{\{tx_1, tx_2\}^{0/2}, BC^{2/2}_{adr}, C_{adr}, B_{sk}\} \rightarrow tx_3$$

2. B gives the transaction script tx_3 to C
3. C signs and broadcasts the ownership change of the Rc from $C \rightarrow B$
4. Exchange physical object from $C \rightarrow B$

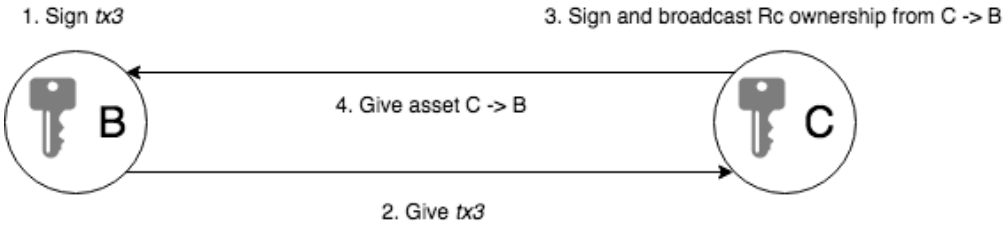


Figure 3.3: Asset Dro-poff Exchange

At the end of the second exchange C now owns tx_3 which has been signed by B. He can now sign and broadcast the transaction with his own keypair whenever he wants to redeem the funds.

$$SignTransaction\colon \{\{tx_1, tx_2\}^{1/2}, BC_{adr}^{2/2}, C_{adr}, C_{sk}\} \rightarrow tx_3$$

4 OpenLogistics

In this chapter we describe OpenLogistics[1], the created decentralized marketplace proof of concept for facilitating trustless transport contracts. OpenLogistics uses the Bitcoin main network for the multisig trustless intermediation mechanism, the distributed data storage bigchaindb to store legally binding liability Ricardian contracts and the software client aggregates the contracts and facilitates the interaction.

To analyze the obtained result of OpenLogistics three roles which can interact with the mechanism will be defined. Every role owns a Bitcoin and BigchainDB keypair once interacting with the software client. The roles which interact with the mechanism are the following:

- Service consumer
- Endpoint actor
- Service provider

The **service consumer** uses OpenLogistics because he wants to transport a physical object. He can create and sign a Ricardian contract which will than be stored on BigchainDB. This represents a request for transport, the ownership of the physical object and includes the information regarding the transport, an example of the data model can be found in appendix B.

The **service provider** can acces the OpenLogistics marketplace order-book and accept the contract in his local client. Upon accepting he creates a transaction script and travels to the location of the **service consumer**. Upon the **service provider** arriving at the **service consumer** the **service consumer** accesses the client and selects his own Ricardian Contract and the pickup exchange takes place illustrated in figure 3.2. The **service provider** now moves to the **endpoint actor** and the **endpoint actor** accesses the client and the drop-off exchange takes place illustrated in Figure 3.3. Eventually the **service provider** endup with tx_3 and can claim the reward in his own client for his provided labor.

4.1 Transparency

Due to the inherit transparancy of the Blockchain it is possible to see the interaction between OpenLogistics and the network. A example testrun of the mechanism can be seen locally, if a Blockchain copy is present, or through commonly used block explorers[2] by searching on the addresses found in Table 1.

The **service provider** actually has no transaction being broadcasted from his address, this is due only using his keypair to sign the transactions from the multisig address. The function **service provider** actually fulfils is being the oracle for conflict resolution upon drop-off exchange by signing or not.

The transferring of Ricardian Contract ownership can be seen by exploring bigchaindb addresses. The aggregation of the ownership change between addresses can be used for a proof of delivery. This proof could be the foundation for a reputation system with further research.

[1] https://www.github.com/DylanBartels/master_thesis/src
[2] https://www.blocktrail.com/BTC

Role	Address
Service Consumer	17F4ZhEJp83qqEG1S6z8YcPbWW7AdqbkZ3
Endpoint Actor	19exDB5Fb2gQAv7k2dH93WbLga1ZUNz9mh
Service Provider	1EY38FGwuSg3uRzetBwYqYh9jjbX55fHsL
Multisig (Endpoint Actor, Service Provider)	3MiFyavsRpMZBzfxFk94WdeZUnbQP1hdDy

Table 1: Bitcoin addresses used to execute a testun of the mechanism in OpenLogistics

4.2 Advantages & Disadvantages

The advantage of the facilitated mechanism of OpenLogistics is that if all actors agree upon the set rules in the Ricardian Contract *anybody* can participate. We chose to implement the payment and escrow mechanism on the Bitcoin network, which is in the blockchain domain perceived as simpler and less vulnerable in terms of incentive structure than other networks.

The downside of the Bitcoin network is that the average blocktime is 10 minutes, which is not consistant which can create long waiting periods when the pickup exchange is taking place. An anomly example of a 1 hour blocktime can be found in appendix C. Other blockchain alternatives can be chosen depending as long as they support multisignature. Another disadvantage of the structure is that you have to spend three transaction fees to make the mechanism work, price fluctuations of the Bitcoin price can disrubt the mechanism.

We chose to implement the OpenLogistics client with Javascript libraries which can easily be developed towards mobile. We assumed with the proof of concept that the actor keypair is bounded to a physical location. The signing of the transactions takes place in the actors local client resulting in the actor his private key never leaving the software client bounded physical location.

4.3 Claim

We claim that OpenLogistics facilitates peer-to-peer decentralized transport contracts while aiming to create trustless intermediation. Through the evaluation of the research questions in Chapter 5 we will provided evidence of our claim.

5 Evaluation

5.1 Research Questions & Answers

In Chapter 1 we stated our research questions and so far we answered them considering the motivating prototype. Generally, we answered the research questions as follows:

How can you provide a mechanism to facilitate peer-to-peer decentralized trustless transport contracts? From our research we conclude that to facilitate this mechanism a Bitcoin multisignature escrow incentive strucure can incentivize the transportation without relying on trusted intermediaries.

Can trustless intermediation exists on this marketplace without centralized dispute resolution? Trustless intermediation cannot take place with the transport of physical goods. During the process of transport a person will *always* be custodian of the physical good. Due to this restriction you can never garantee the expected outcome, centralized conflict resolution will remain to play a roll in transport.

How decentralized is the mechanism? We claim that the mechanism supporting network is as decentralized as the two networks it is build upon.

To what degree do oracles play a role in the verification of the information? The verification of information by an oracle takes place twice during the mechanism. The first time by the service provider when he verifies that the physical asset is simalair to the stated Ricardian Contract when it is being picked up. The second time by the endpoint actor when he verifies that the physical asset being delivered is correct.

What level of anonymity is possible? We claim that the starting point and endingpoint of the transport identity will always be known. However the service provider can maintain privacy in this mechanism, the only aspect of privacy he will have to turn in is the sight of his physical apperance to the service consumer and endpoint actor.

Which attack vectors are possible to undermine this mechanism? The mechanism uses the PoW solution twice to counteract the double spending possibilities of the escrow and the Ricardian contract proof of ownership. We claim that if the actors behave rational the incentive structure holds.

5.2 Evidence

To outline the proof of our incentive structure we will analyze the possible malicious actions the actors can take during the two exchange stages of the mechanism.

5.2.1 Malicious actions pickup exchange

The asset pickup exchange has the following possible malicious actions:

1. Service Provider (C)
   (a) Signing tx_1
      i. Wrong amount equivalent value (Ev)
      ii. No signature from start address (C_{sig})
      iii. Wrong endpoint address (B_{adr})

$$\{Ev, C_{sig}, BC^{2/2}_{adr}\} \notin tx_1$$

2. Service Consumer (A)
   (a) Signing tx_2
      i. Wrong amount transport reward (Tr)
      ii. No signature from start address (A_{sig})
      iii. Wrong endpoint address ($BC^{2/2}_{adr}$)

$$\{Tr, A_{sig}, BC^{2/2}_{adr}\} \notin tx_2$$

   (b) Ricardian Contract (Rc)
      i. Giving wrong pickup exchange location (A_{loc}) in Rc
      ii. Giving wrong dropoff exchange location (B_{loc}) in Rc
      iii. Giving wrong dropoff exchange public key (B_{pk}) in Rc
      iv. Change his mind on request
      v. Wrong change of ownership address

$$\{A_{loc}, B_{loc}, B_{pk}, Ev, Tr\} \not\subseteq Rc$$

   (c) Physical Asset
      i. Not giving the physical asset to C
      ii. Incorrect content

Result malicious actions

1.a. A can verify the content of the tx_1 before broadcasting, an example of a transaction script byte-map can be found in Appendix D. C_{sk} cannot be extracted from tx_1 after it has been signed and encoded into a transaction script. A also has to wait till the transaction is confirmed into the ledger before proceeding with the process.

$$VerifyTransaction(Rc, tx_1)\colon \{Ev, B_{adr}\} \in Rc \wedge \{Ev, C_{sig}, BC^{2/2}_{adr}\} \in tx_1$$

2.a. C can verify the content of the tx_2

$$VerifyTransaction(Rc, tx_2)\colon \{Tr, B_{adr}\} \in Rc \wedge \{Tr, A_{sig}, BC_{adr}^{2/2}\} \in tx_2$$

2.b.i. C would lose the laborcost of moving to A_{loc}
2.b.ii. C would lose the laborcost of moving to A_{loc}
2.b.iii. C can verify the change of ownership of the Rc before it is being broadcasted
2.c. C can organize contract enforecement of Rc through legal institutions because Rc can be legally binding and A_{loc} is known.

5.2.2 Malicious actions dropoff exchange

Possible malicious behaviour at asset dropoff exchange:

1. Service provider (C)
    (a) Ricardian Contract
        i. Wrong change of ownership address
    (b) Physical asset
        i. Opening the package and keeping it
        ii. Opening package and continue contract
2. Endpoint actor (B)
    (a) Signing tx_3
        i. Wrong amount ($\{Ev, Tr\}$)
        ii. No signature from start address ($BC_{adr}^{2/2}$)
        iii. Wrong endpoint address (C_{adr})

$$\{\{Ev, Tr\}, BC_{adr}^{2/2}, C_{adr}\} \notin tx_3$$

    (b) Ricardian Contract
        i. Giving wrong dropoff exchange information in Rc
        ii. State incorrect content

Result malicious actions

1.a. B can verify the change of ownership of the Rc before it is being broadcasted
1.b.i. C will lose $\{Ev, Tr\}$
1.b.ii. Depending on content it might be prevented by chosen packaging, but will not be prevented through incentive mechanism.
2.a. C can verify the content of the tx_3

$$VerifyTransaction(Rc, tx_3)\colon \{Ev, Tr, B_{adr}\} \in Rc \wedge \{Tr, A_{adr}, BC_{adr}^{2/2}\} \in tx_2$$

2.b.i. C can keep package or return to A.
2.b.ii. If B does not agree with content he will not sign to release the escrow. C can keep package or return to A.

6 Conclusion

The proposed trustless escrow mechanism in combination with the digital ownership representation of a physical good can leverage a incentive towards the correct outcome of transportation without central intermediation. However you can never garantee the correct outcome of the physical transportation even if it is encapsuled by a decentralized incentive mechanism. The verification of the ricardian contract correctness into the physical domain has to be done by oracles, and is currently not solved by the same solution as the double spend solution.

Creating the correct legally binding digital contract is very challanging and when edge cases take place not covered by the contract central intermediation is very practical.

The enforcement of punishment of certain malicious actions still need to be resolved by centralized legal means resulting in not a fully trustless mechanism.

6.1 The Challange of Data Integrity

Putting work behind the mechanism components solves the double spend attack but does not solve the data integrity which need to be executed by oracles verification of the Rc.

6.2 Relevance

6.3 Future Challanges

Oracle Problem.
Decentralized Storage.

A List of Symbols

A	Actor A
$\{PubK_a, PrivK_a\}$	EDSCA keypair containing private and public key owned by actor A
Adr_a	Bitcoin address owned by A
$MSig_{ab}$	Multisignature address, a tx_i originating from this address can be signed by private key of $A \vee B$
Loc_a	Physical location of A
tx_1	Transaction script number 1
$A \rightarrow B$	Exchange physical asset from A to B
Rc	Ricardian Contract
EqC	Equivalent cost of the physical asset
Tr	Reward for transporting the physical asset
$GenerateAddress(PubK_n)\colon Adr_n$	Generating a bitcoin address with script language mapping function
$SignTransaction(PrivK_y)\colon \{TRANSACTION_INPUT, y, z\} \mapsto tx_i$	Signing a transaction requires a unspent transaction input, from bitcoin address y to bitcoin address z and resulting in transaction script tx_i

B Ricardian Contract


```
{
  "type": "OpenLogisticsBeta",
  "created_on": "2018-06-19T10:10:13.627Z",
  "value_category": "50",
  "pickup": {
    "name": "John Doe",
    "address": "Kinkerstraat 236",
    "postal": "7634YX",
    "city": "Amsterdam",
    "country": "Netherlands",
    "date_day": "2018-06-21",
    "date_time": "10:09:23",
    "public_key": "03e1b9f8f7114ffb05ee6a006479230fa2d7635e32f8655728cbc29a77ccdbbfe0"
  },
  "dropoff": {
    "name": "Harry de Vries",
    "address": "Amstelstraat 12",
    "postal": "9283JS",
    "city": "Amsterdam",
    "country": "Netherlands",
    "date_day": "2018-06-22",
    "date_time": "11:09:23",
    "public_key": "02d7cef6f0c3109cef11c2bb3304dcb0d453551233ad5873abe9e292a0f9eff226"
  }
}
```

C Bitcoin Mempool Anomaly

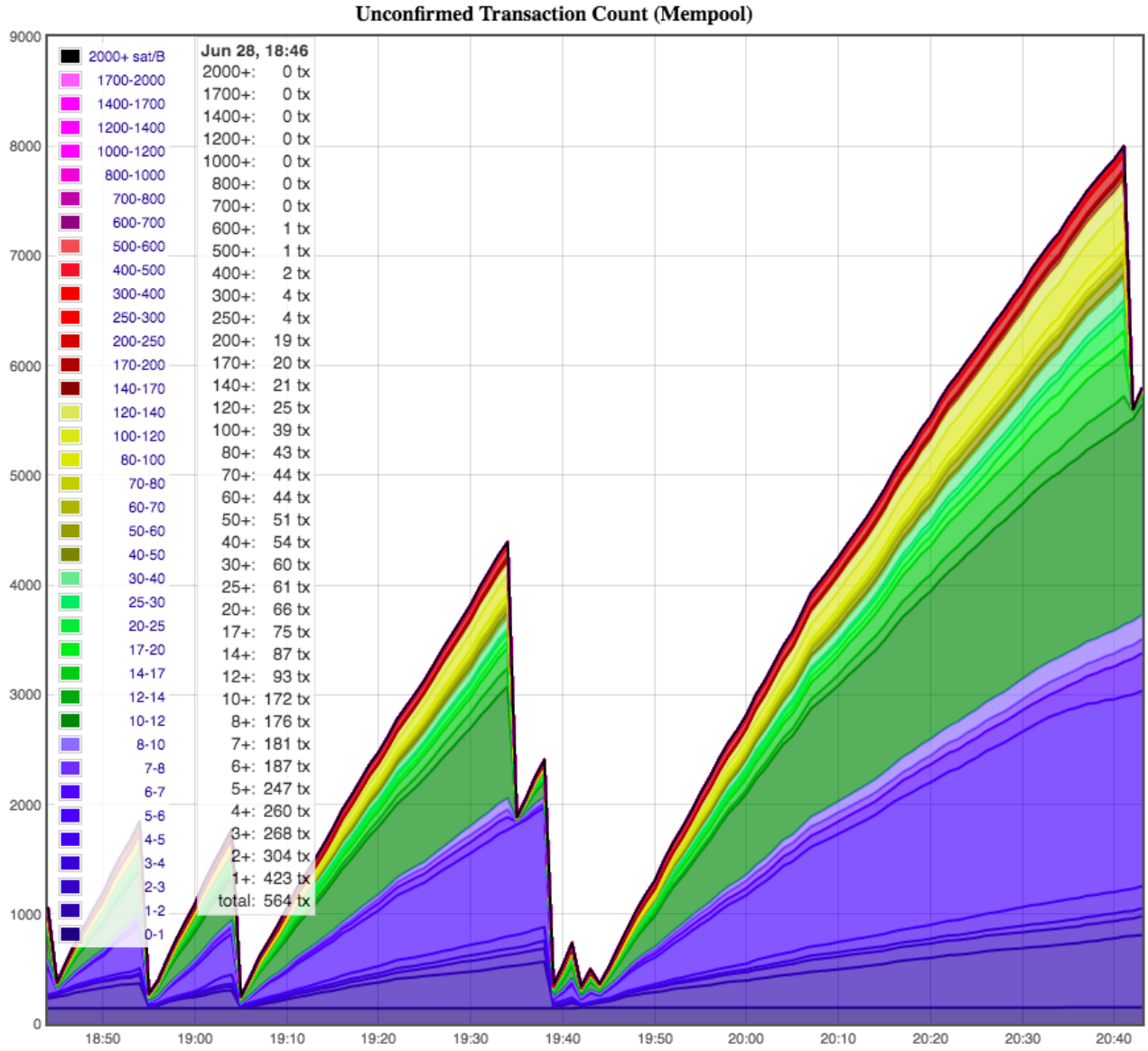
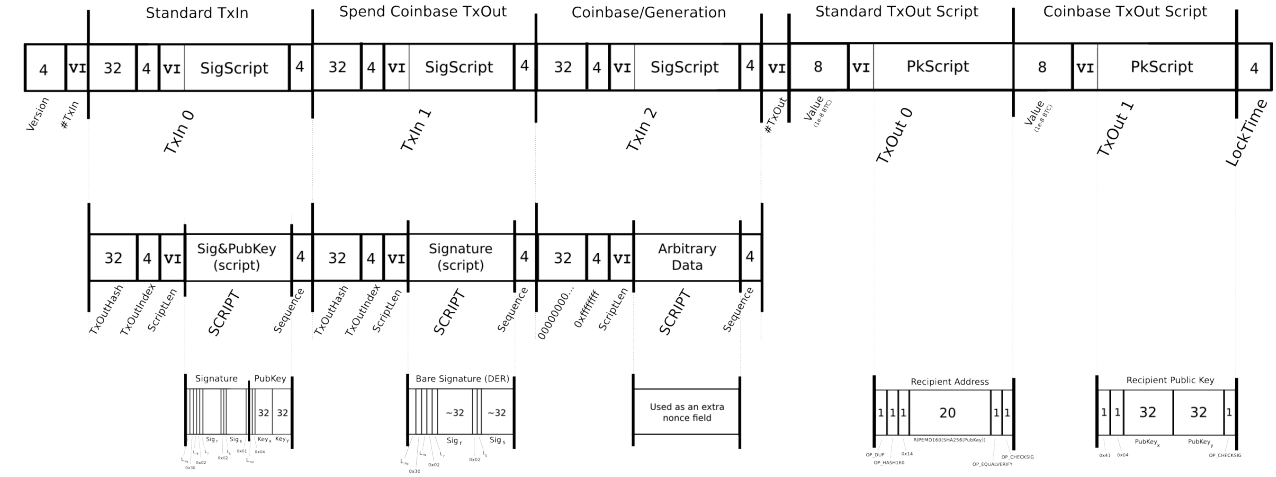


Figure C.1: Block confirmation far exceeding average of 10-minutes

D Transaction Script Byte-map

Transaction



Scripts and DER encoding both use big-endian values, all other serializations use little-endian

etotheipi@gmail.com / 1Gffm7LKXcNFPrtxy6yF4JBoe5rVka4sn1

Figure D.1: Bitcoin transaction byte-map

References

[1] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 2014.

[2] Liran Einav, Chiara Farronato, and Jonathan Levin. Peer-to-peer markets. *Annual Review of Economics*, 8:615–635, 2016.

[3] Olivier Gallay, Kari Korpela, Niemi Tapio, and Jukka K. Nurminen. A peer-to-peer platform for decentralized logistics. In *Digitalization in Supply Chain Management and Logistics*, oct 2017. http://tubdok.tub.tuhh.de/handle/11420/1476; Proceedings of the Hamburg International Conference of Logistics (HICL).

[4] Ian Grigg. The ricardian contract. In *Electronic Contracting, 2004. Proceedings. First IEEE International Workshop on*, pages 25–31. IEEE, 2004.

[5] Niels Hackius and Moritz Petersen. Blockchain in logistics and supply chain: trick or treat? In *Proceedings of the Hamburg International Conference of Logistics (HICL)*, pages 3–18. epubli, 2017.

[6] Ghassan Karame, Elli Androulaki, and Srdjan Capkun. Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. *IACR Cryptology ePrint Archive*, 2012(248), 2012.

[7] Markus Klems, Jacob Eberhardt, Stefan Tai, Steffen Härtlein, Simon Buchholz, and Ahmed Tidjani. *Trustless Intermediation in Blockchain-Based Decentralized Service Marketplaces*, pages 731–739. Springer International Publishing, Cham, 2017.

[8] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[9] J. Winter M.J.G. Olsthoorn. Decentral market: self-regulating electronic market. *DelftUniversity of Technology*, 2016.

[10] Kyle Soska, Albert Kwon, Nicolas Christin, and Srinivas Devadas. Beaver: A decentralized anonymous marketplace with secure reputation. *IACR Cryptology ePrint Archive*, 2016:464, 2016.

[11] Bas van IJzendoorn. The challenge of decentralized marketplaces. *CoRR*, abs/1703.05713, 2017.

[12] Li Xiong and Ling Liu. Building trust in decentralized peer-to-peer electronic communities. In *In The 5th International Conference on Electronic Commerce Research. (ICECR*, 2002.

[13] Dionysis S. Zindros. Trust in decentralized anonymous marketplaces, 2015.