Fair Exchange E-Commerce Protocol for Multi-Chained Complex Transactions

Cătălin V. Birjoveanu and Mirela Birjoveanu

1Department of Computer Science, “Al.I.Cuza” University of Iași, Iași, Romania
2Continental Automotive, Iași, Romania

Keywords: B2C/B2B, Multi-Party Electronic Commerce, Chained Transactions, Complex Transactions, Fair Exchange.

Abstract: In this paper, we introduce the concept of chained transaction as a transaction in which the customer obtains a physical product from a provider using one or more active intermediaries (brokers). Even if there are many multi-party fair exchange protocols with applications in buying digital or physical goods, digital signature of contracts and certified e-mail, no one can be used to solve our problem: complex transactions in which a customer wants to buy several physical products, where each product is acquired in a chained transaction, providing fair exchange. In this paper, we propose the first solution to the problem mentioned above. Our protocol is optimistic and provides fairness, timeliness, non-repudiation and confidentiality.

1 INTRODUCTION

In e-commerce was taken into consideration a significant type of transaction namely complex transaction. Complex transactions are very important as they allow the customer a great flexibility regarding the products he wants to buy. A complex transaction is the combination in any form of aggregate and optional transactions. An aggregate/atomic transaction is a transaction in which the customer wants to buy several products and he wants to obtain either all of them, or none of them. An optional transaction is a transaction in which the customer wants to buy exactly one product from many options he expresses.

In the transaction model that considers complex transactions, we introduce the concept of chained transaction. A chained transaction is a transaction in which the customer obtains a physical product from a provider using one or more active intermediaries (brokers). A broker is a party involved in the chained transaction that receives from another broker (or customer) a request to provide a certain physical product, and to fulfill the request, he buys the product from another broker (or a provider) and sells it to the requesting party. This type of transaction is relevant to be considered being often used in practice, for example when a broker makes available the products from many others brokers/providers as a single interface to the customer. A key business requirement is that after execution of a chained transaction, each party knows only the identities of the parties with whom he/she communicates in the exchanges in which is involved. The importance of this requirement derives from the fact that if after the chained transaction’s execution a broker from chain knows a provider, then in the transactions that follow he could skip certain brokers in the chain and contact the provider directly.

We define the Multi-Chained Complex Transactions Protocol (MCCTP) as the protocol that considers complex transactions in which the customer acquires each physical product in a chained transaction. A standard security requirement in e-commerce protocols is fair exchange. Generally speaking, an e-commerce protocol ensures fair exchange if either the customer/each broker receives the successful payment evidence and the corresponding broker/provider receives the payment for product, or none of them obtains nothing.

There are protocols for buying physical products that provide fair exchange and consider only one customer and one merchant (Djuric and Gasevic, 2015), (Alaraj, 2012). Many multi-party fair exchange protocols were proposed with applications in e-commerce transactions for buying physical goods (Birjoveanu and Birjoveanu, 2019a), buying digital goods (Birjoveanu and Birjoveanu, 2019b), (Liu, 2009), exchange of digital goods (Zhang et al., 2004), digital signature of contracts (Draper-Gil et al., 2013a) and certified e-mail (Zhou et al., 2005).
There are also multi-party fair exchange protocols that considers intermediaries in different scenarios: digital signature of contracts (Draper-Gil et al., 2013b), non-repudiation (Onieva et al., 2004), and exchange of electronic items (Khill et al., 2001). From the multi-party fair exchange protocols considering intermediaries proposed until now, there is no one to address our problem: complex transactions in which a customer wants to buy several physical products, where each product is acquired in a chained transaction using one or more brokers, providing fair exchange. (Khill et al., 2001) considers a ring architecture that can not be applied to our problem because in our model the customer and the provider are not directly communicating. The proposal from (Onieva et al., 2004) considers a customer, many providers, but only one intermediary without considering complex transactions. Thus, this solution is not suitable for our problem because in our model we consider a customer that buys physical products in a complex transaction where each product is obtained using one or more intermediaries (brokers). A solution for digital contracts signing between a customer and many providers using intermediaries is proposed in (Draper-Gil et al., 2013b). This solution considers a contracts signing scenario that is different from our scenario, so it can not be applied to our problem. Moreover, the proposal from (Draper-Gil et al., 2013b) assures weak fair exchange between digital signed contracts, while our protocol assures strong fair exchange between successful payment evidences and payments for physical products. Weak fairness requires that all parties receive the expected items, or all honest parties will have enough evidence to prove that they have behaved correctly in front of an arbiter.

As a result, from all known solutions for multi-party fair exchange considering intermediaries, no one can be applied to solve our problem.

**Our Contribution.** In this paper, we propose the first e-commerce protocol for complex transactions in that the customer wants to buy several different physical products, where each product is acquired in a chained transaction using one or more brokers, providing strong fair exchange. Our protocol provides also effectiveness, timeliness, non-repudiation and confidentiality.

The paper is structured as follows: section 2 presents use cases of our protocol, section 3 defines security requirements, section 4 describes our protocol. Section 5 presents the security analysis of our protocol and section 6 contains the conclusion.

### 2 USE CASES IN B2C/B2B

Our protocol has use cases in Business to Consumer (B2C) and Business to Business (B2B) scenarios. For a B2C scenario, the customer is browsing through the online catalog where there are a great variety of products for home decorations. In this catalog are products from HomeDept, BestHome, RusticHome, and so on, that are home decorations hypermarkets which collaborate with many other intermediaries (brokers) to get the merchandise they sell. The customer wants to redecorate a room. He would like primary a rustic style, but if this is not possible (due to lack of stock or delay in delivery time), also a modern style would fit. Anyhow, in both options, he would like a white carpet and white curtains. So, he specifies his options as a multi-chained complex transaction: ((rustic couch and rustic table and rustic painting) or (2 modern armchairs and steel table and abstract painting)) and white carpet and white curtains. The multi-chained complex transaction is an aggregate transaction in which first component is an optional transaction, the second and third components are individual products. In the optional transaction, both preferences are aggregate transactions. Each individual product is acquired through a chained transaction using one or more brokers. The rustic couch (RC) and the rustic painting (RP) are sold by the broker HomeDept, while the rustic table (RT) is sold by the broker RusticHome.

In a chained transaction in which the customer acquires RC, the customer initiates an exchange by contacting the first broker HomeDept that is the receiver in this exchange. HomeDept initiates the second exchange from chain by contacting the second broker AllCouches that is the receiver in this exchange. AllCouches goes in his turn to the next broker BestSleepProducts that further goes to the provider that produces the couches. Similarly, in the chained transaction in that RT is acquired, RusticHome goes to the second broker WoodenStuff that goes to the carpenter that produces the table. Also, in the chained transaction in that RP is acquired, HomeDept collaborates directly with a painter. In a similar manner, the white table and curtains are acquired in chained transactions using one or more brokers.

For example, a pack of products that solves the customer’s options is: RC and RT and RP and white carpet and white curtains. This means that each product from the pack was successfully acquired in a chained transaction: each party that initiates an exchange (the customer or a broker) receives a successful payment evidence from the corresponding receiver (a broker or the provider) and the corresponding receiver receives the payment for product from the cor-
responding initiator.
A similar scenario can be used for B2B. In this case, the customer is an interior design company that redecorates a hotel.

3 SECURITY REQUIREMENTS

In what follows, we will discuss the security requirements we want to achieve: effectiveness, fairness, timeliness, non-repudiation, and confidentiality. These requirements are also stated and analyzed in (Birjoveanu and Birjoveanu, 2019a).

Effective-ness requires that if every party involved in MCCTP behaves honestly, does not want to prematurely terminate the protocol, and no communication error occurs, then the customer receives his successful payment evidences (digital receipts) from brokers and each corresponding broker receives the payment for product, each broker that initiates an exchange receives his successful payment evidence from the corresponding broker/provider and each receiver broker/provider receives the payment for product, without any intervention of Trusted Third Party (TTP). This is a requirement for optimistic protocols, where TTP intervenes only in case of unexpected situations, such as a network communication errors or dishonest behavior of one party.

Fairness in electronic payment protocols for physical products from (Djuric and Gasevic, 2015) is defined only for one customer and one merchant. So, we must adapt the fairness requirement to our scenario: for chained transactions and also for multi-chained complex transactions.

Fairness for a chained transaction requires:

- either the chained transaction is successful: each party that initiates an exchange in the chained transaction (the customer or a broker) receives a successful payment evidence from the corresponding receiver (a broker or the provider) and the receiver receives the payment for the product from the corresponding initiator, or
- the chained transaction is aborted: each party that initiates an exchange (the customer or a broker) in the chained transaction receives an aborted payment evidence from the corresponding receiver (a broker or the provider) and the receiver doesn’t receive the payment for the product.

Fairness in MCCTP (for a multi-chained complex transaction) requires:

- fairness for any chained transaction from the complex transaction, and
- for any optional transaction from the complex transaction, either the chained transactions corresponding to exactly one option are successful, or all chained transactions are aborted, and
- for any aggregate transaction from the complex transaction, either all chained transactions belonging to the aggregate transaction are successful, or all are aborted.

This requirement corresponds to the strong fairness. Timeliness requires that any party involved in MCCTP can be sure that the protocol’s execution will be finished at a certain finite point of time, and after the protocol’s finish point, the fairness level achieved cannot be degraded.

Non-repudiation requires that neither the customer, any of brokers, nor any of providers can deny their involvement in MCCTP.

Confidentiality in MCCTP requires that the content of messages sent between participating parties is accessible only to authorized parties. Also, after any chained transaction’s execution, each party knows only the identities of the parties with whom he/she communicates in the exchanges in which participates.

The last requirement is important in chained transactions: if after the chained transaction’s execution a broker from chain knows a provider, then in the transactions that follow he could skip certain brokers in the chain and contact the provider directly.

4 MULTI-CHAINED COMPLEX TRANSACTIONS PROTOCOL

In MCCTP, we consider that one customer can buy products in complex transactions from many providers using brokers.

4.1 Participants, Roles, Communication Channels

MCCTP has the following participants: the customer, the providers, the brokers, the payment gateway and the bank. Table 1 presents the notations used in the description of MCCTP.

In MCCTP the following roles are identified: initiator, receiver and payment processing. An initiator is an agent that initiates an exchange with a receiver by sending a request to the receiver to buy a product. A receiver is an agent that responds to the initiator’s request by sending to the initiator the corresponding evidence of product’s buying. The payment processing is an agent that performs payments between initiators and receivers. These roles can be played by the
participants as follows. C initiates a complex transaction and can play only the initiator role. A provider is an agent that provides a product to the agent that initiates an exchange with him. So, any provider can play only the receiver role. A broker is an intermediary agent in a chained transaction, communicating with both initiators and receivers. In an exchange in that the broker provides a product to an initiator, the broker plays the receiver role. In an exchange in that the broker buys a product from a receiver, the broker plays the initiator role. The \textit{payment processing} role is played by \textit{PG} that is a trusted party.

Hybrid encryption \( \{ m \}_{P_{A}} \) of the message \( m \) with the public key \( P_{A} \) means \( \{ m \}.K \), \( \{ K \}_{P_{A}} \): the message \( m \) is encrypted with an AES session symmetric key \( K \), which is in turn encrypted using \( P_{A} \). If two parties use the session symmetric key \( K \) in a hybrid encryption, then they will use \( K \) to hybrid encrypt all the messages that will be transmitted between them for the remainder of session. We consider unreliable communication channels (messages can be lost) between initiators and receivers, and resilient communication channels (messages can be delayed but not lost) between \textit{PG} and the other participants.

\section{Setup}

In what follows, we describe the setup phase which is needed before \textit{MCCTP} execution. The customer is browsing through the online catalog where the products from brokers are posted. After \( C \) decides the products the buyer wants to buy and the options/alternatives for each product from the pack, he clicks a “submit” button on the online catalog. We consider that each initiator (customer or broker) has a payment Web segment downloaded from \textit{PG}. The payment Web segment is a software digitally signed by \textit{PG} that runs on the initiator’s computer.

For each subtransaction from the complex transaction (corresponding to the product the initiator wishes to buy), the corresponding payment Web segment requires from the initiator the credit card information and a challenge code that will be used to authenticate and authorize the initiator for using the credit card. Also, the payment Web segment generates a RSA session public/private key pair for the initiator. We consider that the payment Web segment has the digital certificates for the public keys of \textit{PG} and of each receiver he communicates with. Also, each receiver/\textit{PG} has the digital certificate for \textit{PG} each receiver’s public key.

\textit{MCCTP} uses the \textit{Chained Transaction Protocol (CTP)}. Next, we will describe \textit{CTP} in which the customer buys a certain physical product using one or more brokers and a provider.

\subsection{Chained Transaction Protocol}

A \textit{subtransaction} is an exchange in which an initiator (customer or a broker) buys a physical product from a receiver (a broker or a provider). A \textit{chained transaction} in which \( C \) buys a certain physical product using the brokers \( B_{1}, \ldots, B_{n} \) and the provider \( P \) is a sequence of subtransactions \( s_{0} \), \( s_{1} \), \ldots, \( s_{n} \), where \( C \) is the initiator in \( s_{0} \), \( B_{1} \) is the receiver in \( s_{0} \), and \( B_{1} \) is the initiator in \( s_{1} \), with \( 1 \leq i \leq n \), and \( P \) is the receiver in \( s_{n} \). In such a chained transaction, \( C \) knows only the identity of the broker he communicates with in the subtransaction \( s_{0} \), because \( C \) participates only in \( s_{0} \). Also, \( B_{i} \) knows only the identities of the agents from the subtransactions \( s_{i-1} \) and \( s_{i} \) in which he participates, and \( P \) knows only the identity of the broker \( B_{n} \).

\( C \) initiates \textit{CTP} by sending to \( B_{1} \) a purchase order for buying a certain physical product. To fulfill \( C \)'s request, \( B_{1} \) initiates a new subtransaction by sending a purchase order to \( B_{2} \). In the same manner, \( B_{2} \) initiates a new subtransaction with \( B_{3} \), and so on until \( B_{n} \) initiates a new subtransaction with \( P \). In this step, \( n+1 \) subtransactions \( s_{0}, s_{1}, \ldots, s_{n} \) are started belonging to the chained transaction \( s_{0}, s_{1}, \ldots, s_{n} \). The successful finish of the entire chained transaction starts successfully finishing the subtransaction \( s_{n}, s_{n-1} \), until successfully finishing \( s_{0} \). Successful finish of \( s_{n} \) means that \( B_{n} \) as initiator received from \( P \) the successful payment evidence for product, and \( P \) received the payment from \( B_{n} \). Only after successful finish of \( s_{n} \), \( B_{n} \) sends as receiver in \( s_{n-1} \) the payment request to \( PG \). Successful finish of \( s_{n-1} \) means that \( B_{n} \) received the payment from \( B_{n-1} \), and \( B_{n-1} \) received from \( B_{n} \) two successful evidences: the current payment evidence of \( B_{n-1} \) and \( B_{n} \) in \( s_{n-1} \), and the past payment evidence of \( B_{n} \) in \( s_{n} \). In this manner, any subtransaction \( s_{i} \) from chain is successfully finished only after the successful finish of \( s_{i+1}, \ldots, s_{n} \) subtransactions.

\textit{CTP} for a chained transaction \( s_{0}, s_{1}, \ldots, s_{n} \) consists of the \textit{Exchange} sub-protocol for each chained subtransaction \( s_{i} \), and two \textit{Resolution} sub-protocols.
Next, we will describe the CTP’s sub-protocols.

4.3.1 Exchange Sub-protocol

In this section, we will describe the Exchange sub-protocol for an arbitrary subtransaction $s_i$, where $0 \leq i \leq n$, that is presented in Table 3. The notations used in CTP are provided in Table 2. In the subtransaction $s_i$, the payment Web segment of the initiator $B_i$ sends to the receiver $B_{i+1}$ the purchase order $PO_{i+1}$ to buy the product from the chained transaction. $PO_{i+1}$ contains a payment message $PM_i$ and the order information $OI_i$ both encrypted with $PkB_{i+1}$. $PM_i$ is build by $B_i$’s payment Web segment by encrypting with $PG$’s public key the payment information $PI_i$ of $B_i$ and the signature of $B_i$ on $PI_i$.

$PI_i$ contains the data provided by user as initiator: card number $CardN_i$ and a challenge code $CCode_i$ issued by bank. The challenge code is provided to user by bank via SMS and it has a minimum length of four characters. In each subtransaction $s_i$ from chained transaction, the payment Web segment of $B_i$ generates a fresh random number $N_i$. The identifier $Id_i$ of $s_i$ is built by concatenating the identifier $Id_{i-1}$ of $s_{i-1}$ with $N_i$. So, the identifier $Id_i$ of $s_i$ is the sequence of numbers $N_0,N_1,...N_i$ generated in all subtransactions of $s_0,s_1,...,s_n$. This way of assigning identifiers to subtransactions allows tracing the subtransactions from the chained transaction. Thus, in case of a latter dispute, the identification of the subtransactions from chain is done in a simple manner. Also, $PI_i$ includes the identifier $Id_i$, the amount $Am_i$, and $PkB_i$.

$OI_i$ contains the identity of the initiator $B_i$, of the receiver $B_{i+1}$, the product identifier $PId_i$, the transaction identifier $Id_i$, the amount $Am_i$, and $PkB_i$.

In each subtransaction $s_i$, the receiver decrypts $PO_{i+1}$ and checks the signature of the initiator on $OI_i$. If the receiver $B_{i+1}$ is not the provider $P (i < n)$, then he stores $PO_{i+1}$ and sends $PO_{i+1+2}$ to $B_{i+2}$ for buying the product requested by $B_i$.

At the line 2, if the receiver $B_{i+1}$ is the provider $P (i = n)$, then $s_i$ is the last subtransaction from chain. So, $B_{i+1}$ stores $PO_{i+1}$ and sends the payment request $PR_{i+1}$ to $PG$ to get payment from $B_i$. $PR_{i+1}$ is built from the payment message $PM_i$ and $B_{i+1}$’s signature on the subtransaction identifier $Id_i$, the identity of the initiator $B_i$, of the receiver $B_{i+1}$, $PkB_i$, and $Am_i$. Upon receiving $PR_{i+1}$, $PG$ decrypts it, checks $B_i$’s signature on $PI_i$ and checks if $B_i$ is authorized to use the card by checking if the combination of $CardN_i$ and $CCode_i$ is valid. If these checks are successfully passed, then also $B_i$ proves as being the owner of the public key $PkB_i$. $PG$ checks $B_{i+1}$’s signature, and if checking is successful, then it has the confirmation that both $B_i$ and $B_{i+1}$ agreed on $Id_i$, $PkB_i$ and $Am_i$. If some check fails, then $PG$ sends to $B_{i+1}$ an aborted current payment evidence $CE_{E_{i+1}}$ (with $Resp = ABORT$). If all checks are successful, $PG$ sends the payment message to the bank. The bank checks $B_i$’s account balance, and if it is enough, then the bank makes the transfer in $B_{i+1}$’s account providing a successful current payment evidence $CE_{E_{i+1}}$ (with $Resp = YES$) to $PG$ that forwards it to $B_{i+1}$ at the line 3. Otherwise, if checking $B_i$’s account balance fails, then also the transfer fails and the bank provides an aborted current payment evidence $CE_{E_{i+1}}$ (with $Resp = ABORT$) to $PG$ that forwards it to $B_{i+1}$ as a proof of $s_i$’s abortion. We remark that the current evidence $CE_{E_{i+1}}$ in $s_i$ includes the evidence $E_i$ that will be latter send by $B_i$ to $B_{i+1}$ to inform $B_{i+1}$ if $s_i$ was successfully finished or aborted. Also, $PG$ stores $PR_{i+1}$ and $CE_{E_{i+1}}$ in its database. Upon receiving $\{CE_{E_{i+1}}\}_{PkB_{i+1}}, B_{i+1}$ decrypts it and sends to $B_i$ (line 4) the current evidence $CE_{E_{i+1}}$ and the provider certificate $Cert_p$ both encrypted with $PkB_i$. $B_i$ decrypts the message received from $B_{i+1}$, and checks the authenticity of $Cert_p$ and $CE_{E_{i+1}}$. If $PG$’s signature from $CE_{E_{i+1}}$ is successfully checked, then $B_i$ has the guarantee of $CE_{E_{i+1}}$’s authenticity and it corresponds to the current subtransaction $s_i$, because of the presence of the identifier $Id_i$.

In each subtransaction $s_i$ that is not the last from chain $(i < n)$, in order to answer the request received from the initiator $B_i$, the receiver $B_{i+1}$ must ensure that either all the subtransactions that follows $s_i$ in chain have been successfully completed, or all the subtransactions that follows $s_i$ in chain have been aborted. An arbitrary subtransaction $s_j$ is successful if the receiver $B_{j+1}$ received the payment and $B_i$ received the successful current payment evidence $CE_{E_{j+1}}$ and payment evidence $E_{j+1}$. $s_j$ is aborted if the receiver $B_{j+1}$ did not receive the payment and $B_j$ received an aborted $CE_{E_{j+1}}$.

So, at line 6, $B_{i+1}$ as initiator in $s_{i+1}$ waits to receive from $B_{i+2}$ the current evidence $CE_{E_{i+2}}$ in $s_{i+2}$ and the evidence $E_{i+2}$ in $s_{i+2}$. If both evidences $CE_{E_{i+1}}$ and $E_{i+2}$ received by $B_{i+1}$ are successful (with $Resp = YES$) then this means that $s_{i+1}, s_{i+2},...,s_{n}$ are successfully finished. In this case, $B_{i+1}$ sends $PR_{i+1}$ to $PG$ in $s_i$ at line 8.

If $B_{i+1}$ receives from $PG$ (line 9) a successful $CE_{E_{i+1}}$, then he sends $CE_{E_{i+1}}$ and $E_{i+1}$ to $B_i$ (line 11). $B_i$ verifies evidence’s authenticity by checking $PG$’s signatures, checks the corresponding identifiers from both evidences to ensure the freshness of evidences and that these belong to successive subtransactions. If both evidences are successful, then $CE_{E_{i+1}}$ ensures $B_i$ that $s_i$ was successful and $E_{i+1}$ ensures $B_i$ that $s_i$ was successful. $B_i$ will use $E_i$ in $s_{i-1}$ in the same man-
Table 2: Notations used in CTP.

| POi,i+1 | = | \{PMi, Oi, SigBi(Oi)\}PBi,i+1 |
| PMi | = | \{Pi, SigBi(Pi)\}PBi |
| CTP | = | \{Idi, CardNi, CCCodei, Idi, Ami, PKBi\} |

Table 3: Exchange sub-protocol for the subtransaction \( s_i \).

1. \( B_i \rightarrow B_{i+1} : PO_{i,i+1} \)
2. if \( i = n \), \( B_{i,i+1} \rightarrow PG : PR_{i,i+1} \)
3. \( PG \rightarrow B_{i,i+1} : \{\text{CE}_{i,i+1} \} \) \( P_{Bi,i+1} \)
4. \( B_{i,i+1} \rightarrow B_{i,i+1} : \{\text{CE}_{i,i+1}, CertP\}P_{Bi,i+1} \)
5. else
6. \( B_{i,i+1} \rightarrow B_{i,i+1} : \{\text{CE}_{i,i+1}, Resp, E\}P_{Bi,i+1} \) in \( s_{i+1} \),
7. \( \text{with} \ \text{CE}_{i,i+1+2} \text{Resp}=YES \) and \( \text{E}_{i,i+1} \text{Resp}=YES \)
8. \( B_{i,i+1} \rightarrow PG : PR_{i,i+1} \)
9. \( PG \rightarrow B_{i,i+1} : \{\text{CE}_{i,i+1} \}P_{Bi,i+1} \)
10. if \( \text{CE}_{i,i+1} \text{Resp}=YES \)
11. \( B_{i,i+1} \rightarrow B_{i,i+1} : \{\text{CE}_{i,i+1}, E_{i,i+1} \}P_{Bi,i+1} \)
12. else Resolution 1
13. end if
14. else if \( \text{CE}_{i,i+2} \text{Resp}=\text{ABORT} \)
15. \( \text{Resolution 2} \)
16. else Resolution 2
17. end if
18. end if
19. end if

If \( i = 0 \), then \( B_i \) denotes the customer \( C \). If \( i = n \), then \( B_{i,i+1} \) denotes \( P \), and \( E_{i,i+1} \) is replaced with \( CertP \).

4.3.2 Resolution 1 Sub-protocol

Let be \( s_i \), where \( 0 \leq i \leq n \), the first subtransaction (in reverse order: from \( s_n \) to \( s_0 \)) from the chained transaction \( s_0s_1 \ldots s_n \) in which \( B_{i+1} \) receives from \( PG \) an aborted \( CE_{i,i+1} \) and forwards it to \( B_i \). If \( s_i \) is not the last subtransaction from chain \( (i < n) \), then fairness in CTP is not ensured because the subtransactions \( s_{i+1} \), \ldots, \( s_n \) are successful, and \( s_i \) is aborted due to aborted \( CE_{i,i+1} \). If \( s_i \) is the last subtransaction from chain \( (i = n) \), then \( s_n \) is aborted, and also the subtransactions \( s_{n-1} \), \ldots, \( s_0 \) must be aborted. So, to obtain fairness in CTP: Resolution 1 sub-protocol described in Table 4 is applied to abort all subtransactions \( s_i \) from chain.

If \( s_i \) is not the last subtransaction from chain, then \( B_{i+1} \) initiates Resolution 1 by sending to \( PG \) a request to abort \( s_{i+1} \). The request contains the aborted \( CE_{i,i+1} \) in \( s_i \) and the successful \( CE_{i+1,i+2} \) in \( s_{i+1} \). \( PG \) verifies evidence’s authenticity by checking his signatures and checks the corresponding identifiers from both evidences to ensure that these belong to successive subtransactions. If all checks are passed, \( PG \) aborts the successful \( CE_{i+1,i+2} \). We denote by \( CE_{i+1,i+2} \), the evidence generated by \( PG \) that aborts the successful \( CE_{i+1,i+2} \). \( PG \) sends \( CE_{i+1,i+2} \) to \( B_{i+1} \) and \( B_{i+2} \) as a proof of aborting \( s_{i+1} \). The first for loop aborts in a similar manner \( s_{i+2}, \ldots, s_n \), each iteration aborting a new subtransaction from chain until aborting \( s_n \).

\( PG \) obtains \( CE_{i+1,i+2} \) from the bank by sending to it the request of \( B_{i+1} \). The bank aborts the successful \( CE_{i+1,i+2} \) by canceling the transfer from \( B_{i+1} \)’s account into \( B_{i+2} \)’s account, and builds \( CE_{i+1,i+2} \) as follows:

1. \( E_{i+1} = \text{SigPG}(\text{ABORT, Id}_{i+1}, E_{i+1}) \)
2. \( CE_{i+1,i+2} = \text{ABORT, B}_{i+1}, B_{i+2}, I_{i+1}, \text{SigPG}(\text{ABORT, B}_{i+1}, B_{i+2}, I_{i+1}, Am_{i+1}, CE_{i+1,i+2}, E_{i+1}) \)

The second for loop aborts \( s_{i+1}, \ldots, s_0 \) in this order. In the first iteration, \( B_i \) sends to \( PG \) a request to abort \( s_{i+1} \). The request contains the aborted \( CE_{i+1} \) in \( s_i \) and the content of \( PO_{i+1} \), received by \( B_i \) in \( s_{i+1} \). \( PG \) verifies evidence’s authenticity by checking his signature and \( PO_{i+1} \)’s authenticity by checking
Table 4: Resolution 1 sub-protocol.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i &lt; n$</td>
<td>$B_{i+1} \rightarrow PG: {CE_{i,i+1}, CE_{i+1,i+2}}_{PG}$</td>
</tr>
<tr>
<td>$PG \rightarrow B_{i+1}: {CE_{i+1,i+2}}<em>{PK</em>{B_{i+1}}}$</td>
<td></td>
</tr>
<tr>
<td>$PG \rightarrow B_{i+2}: \ldots$</td>
<td></td>
</tr>
</tbody>
</table>

end if

for $(j = i + 1; j < n - 1; j = j + 1)$

$B_{j+1} \rightarrow PG: \{CE_{j,j+1}, CE_{j+1,j+2}\}_{PG}$

$PG \rightarrow B_{j+1}: \{CE_{j+1,j+2}\}_{PK_{B_{j+1}}}$

$PG \rightarrow B_{j+2}: \ldots$ |

end for


If $(i < n)$ $B_{i+1} \rightarrow PG: \{CE_{i,i+1}, CE_{i+1,i+2}\}_{PK_{PG}}$

$PG \rightarrow B_{i+1}: \{CE_{i+1,i+2}\}_{PK_{B_{i+1}}}$

$PG \rightarrow B_{i+2}: \ldots$


end if

for $(j = i + 1; j < n - 1; j = j + 1)$

$B_{j+1} \rightarrow PG: \{CE_{j,j+1}, CE_{j+1,j+2}\}_{PK_{PG}}$

$PG \rightarrow B_{j+1}: \{CE_{j+1,j+2}\}_{PK_{B_{j+1}}}$

$PG \rightarrow B_{j+2}: \ldots$ |

end for


the $B_{i-1}$’s signatures. Also, $PG$ verifies the corresponding identifiers from $CE_{j,j+1}$ and $PO_{i-j,i}$ to ensure that these belong to subsequent subtransactions. If all checks are passed, $PG$ generates an aborted evidence $CE_{i-1}$ and sends it to $B_i$ and $B_{i-1}$ as a proof of aborting $s_i$. Each iteration continues in a similar manner aborting a new subtransaction from chain until aborting $s_0$.

### 4.3.3 Resolution 2 Sub-protocol

Let be $s_i$, where $0 \leq i \leq n$, the first subtransaction (in reverse order: from $s_0$ to $s_n$) from the chained transaction $s_0s_1 \ldots s_n$ in which $B_i$ sends $PO_{i-j,i}$ to $B_{i+1}$, but $B_i$ does not receive a payment evidence or receives an invalid payment evidence from $B_{i+1}$. It is obvious that in this scenario, fairness in $CTP$ is not ensured.

In this case, in any subtransaction $s_i$ from chain, a timeout interval $t$ (e.g., in the order of seconds or minutes) is defined, in which $B_i$ waits a payment evidence from $B_{i+1}$. If $t$ expires and $B_i$ does not receive a payment evidence or receives an invalid payment evidence from $B_{i+1}$, then $B_i$ initiates Resolution 2 with $PG$ to receive a payment evidence w.r.t. $s_i$ and to obtain fairness in $CTP$. $B_i$ sends to $PG$ the content of purchase order $PO_{i,j+1}$ and the invalid evidence $R$ received from $B_{i+1}$, if such an evidence is received.

$B_i \rightarrow PG: \{PM_i, OL_i, R, Sig_{B_i}(OL_i, R)\}_{PK_{PG}}$

$PG$ decrypts the message, checks $B_i$’s signature and checks the purchase order’s content and $R$.

If $R$ does not contain the evidence $CE_{i,i+1}$, then $PG$ checks if $CE_{i,i+1}$ has been generated for the entry $Id_i, AM_i$ and $PK_{B_i}$, as follows:

1. If $PG$ finds in its database the successful $CE_{i,i+1}$ for the entry above, then $PG$ requires $E_{i+1}$ from $B_{i+1}$. If $B_{i+1}$ responds to $PG$ with a successful $E_{i+1}$, then $PG$ sends it to $B_i$ and $CTP$ continues with $s_{i+1}$. Otherwise, if $B_{i+1}$ responds with an aborted $E_{i+1}$ or does not respond, then $PG$ generates the aborted $CE_{i,i+1}$ and sends it to $B_i$ and $B_{i+1}$ as a proof of aborting $s_i$. Now, Resolution 1 is applied to abort $s_{i+1}, \ldots, s_n$ if these are successful, and also to abort $s_{i-1}, \ldots, s_0$.

2. If $PG$ finds in its database the aborted $CE_{i,i+1}$ for the entry above, then $PG$ sends $CE_{i,i+1}$ to $B_i$ and $B_{i+1}$ as a proof of aborting $s_i$. Resolution 1 is applied as in item 1.

3. If $PG$ does not find an evidence for the entry above, then $PG$ generates the aborted evidence $CE_{i,i+1}$ and sends it to $B_i$ and $B_{i+1}$ as a proof of aborting $s_i$. Resolution 1 is applied as in item 1.

If $R$ contains a successful $CE_{i,i+1}$ but a successful $E_{i+1}$ is missing, then $PG$ requires $E_{i+1}$ and $CTP$ continues as in item 1.

A chained transaction $s_0s_1 \ldots s_n$ successfully finished $CTP$ if all subtransactions $s_i$, where $0 \leq i \leq n$, are successfully finished ($B_i$ receives the successful $CE_{i,i+1}$ and $E_i$ and $B_{i+1}$ receives the payment for the corresponding product). A chained transaction $s_0s_1 \ldots s_n$ is aborted if all subtransactions $s_i$, where $0 \leq i \leq n$, are aborted ($B_i$ receives an aborted $CE_{i,i+1}$ and $B_{i+1}$ does not receive the payment for product). As we can see, after running $CTP$, either the chained transaction successfully finished $CTP$, or is aborted.

**Example.** In Figure 1, we describe an instance of $CTP$ considering a customer $C = B_0$, two brokers $B_1$, $B_2$ and a provider $B_3 = P$. In this example, $C$ knows only $B_1$, $B_1$ knows only $C$ and $B_2$, $B_2$ knows only $B_1$ and $P$ and $P$ knows only $B_2$. The chained transaction is the sequence of the subtransactions $s_0s_1s_2$. In $s_0$, $C$ initiates $CTP$ by sending $P_0$ to $B_1$ for buying a certain physical product. To acquire the product requested by $C$ in $s_0$, the broker $B_1$ initiates a new subtransaction $s_1$ by sending $P_{O1,2}$ to $B_2$. Also, to acquire the product requested by $B_1$, $B_2$ initiates $s_2$ by sending $PO_{2,2}$ to $P$. $s_2$ is the last subtransaction from the chain because the receiver in $s_2$ is $P$. So, $P$ sends $PR_{2,3}$ to $PG$ to get the payment for his product from $B_2$. After $PG$ successfully verifies $PR_{2,3}$, he sends to $P$ the successful $CE_{2,3}$. To complete $s_2$, $P$ sends to $B_2$ the successful $CE_{2,3}$ and $Cert_P$. After receiving message 6, $B_2$ is ensured that $s_2$ is successfully finished. In this step, $B_2$ continues $s_1$ (in which he is the receiver) by sending in message 7, $FR_{1,2}$ to $PG$ that responds to him with a successful $CE_{1,2}$. To complete $s_1$, $B_2$ sends to $B_1$ the successful $CE_{1,2}$ and $E_2$. The successful $CE_{1,2}$ ensures $B_1$ that $s_1$ successfully finished and the successful $E_2$ ensures $B_1$ that $s_2$ successfully finished. In this step, $B_1$ continues $s_1$ by sending the message 10 to $PG$. After $B_1$ obtains
successful $CE_{0,1}$, he sends $CE_{0,1}$ and $E_1$ to $C$. If $C$
obreakspace \noindent obtains the successful $CE_{0,1}$ and $E_1$, then the chained
transaction is successful. So, in each subtransaction $s_i$, the initiator $B_i$
retrieves the successful current evidence and the receiver $B_{i+1}$ obtains the payment.

On the other side, for example, if $s_2$ was successfully finished and in $s_1$ the broker $B_2$
receives in message 8 an aborted $CE_{1,2}$, then an unfair case appears
w.r.t. the entire chained transaction. In this case, $B_2$
initiates Resolution 1 sub-protocol with $PG$ to abort $s_2$ and $s_0$. So, the chained transaction is aborted.

### 4.4 MCCTP Description

**MCCTP** is based on the same idea as the proposed protocol in (Birjoveanu and Birjoveanu, 2019a).

For an aggregate transaction, the aggregation operator, denoted by $\land_k$, is defined as follows: $P_i \land \ldots \land P_k$, meaning that $C$ wishes to buy exactly $k$ products with the product's identifiers $P_i, \ldots, P_k$. For an optional transaction, the option operator, denoted by $\lor$, is defined as follows: $P_i \lor \ldots \lor P_k$, meaning that $C$ wishes to buy a product that is exactly one of the products with the product's identifiers $P_i, \ldots, P_k$, where the apparition order of the product's identifiers is the priority given by $C$. This means that $C$ wishes first of all to buy the product $P_i$, but if this is not possible, his second option is $P_{i+1}$, and so on.

From the choices of $C$ describing the sequence of products he wishes to buy, we build a tree over the product identifiers selected by $C$ using $\land$ and $\lor$ operators. To represent the tree, we use the left-child, right-sibling representation in that each internal node corresponds to one of the above operators or to an identifier, while each leaf node corresponds to an identifier. Each node of the tree is represented by a structure with the following fields: info for storing the useful information (identifier or one of the operators), left for pointing to the leftmost child of node, and right for pointing to the sibling of the node immediately to the right. The access to tree is realized through the root. An example of tree derived from the complex transaction from section 2, is shown in Figure 2.

$Pid_1$ corresponds to RC, $Pid_2$ to RT, $Pid_3$ to RP, $Pid_4$ to modern armchairs, $Pid_5$ to steel table, $Pid_6$ to abstract painting, $Pid_7$ to white carpet and $Pid_8$ to white curtains. The root node has $\land$ operator as info. The root does not have any right sibling and its children are the node having $\lor$ operator as info and two nodes with the info $Pid_7$ and $Pid_8$.

In the multi-chained complex transactions protocol we proposed, a chained transaction $ct$, denoted by $CTP(C, B_i, Pid_i)$, is an instance of $CTP$ in which $C$ buys the physical product with $Pid_i$ identifier from the broker $B_i$, using many other brokers and a provider. We define $St(ct)$ the state of the chained transaction $ct$ as being the current evidence $CE_{0,i}$ that $C$ will receive in $ct$.

We define $Ns(p)$ - the state of the node $p$ as a sequence of chained transaction states $St(ct_1) \ldots St(ct_n)$ corresponding to the product defined by $p$. For a node $p$, $Ns(p)$ is calculated similarly as in (Birjoveanu and Birjoveanu, 2019a), depending on $p \rightarrow info$ as follows:

- if $p \rightarrow info = Pid_i$, where $1 \leq i \leq n$, then $Ns(p) = St(CTP(C, B_i, Pid_i))$. For simplicity, we consider that $B_i$ sells the product with $Pid_i$ identifier.
- if $p \rightarrow info = \lor$, then

$$Ns(p) = \begin{cases} Ns(l), & \text{if } \exists l, \text{the leftmost child of } p \text{ such that } St(ct).Resp = YES, \text{ for all } St(ct) \in Ns(l) \\ Ns(r), & \text{otherwise} \end{cases}$$

where $r$ is the rightmost child of $p$. 

![Figure 1: CTP message flow.](image1)

![Figure 2: Tree describing the customer's choices.](image2)
• if $p \rightarrow \text{info} = \land$, and $c_1, \ldots, c_k$ are all children of $p$, then we have two cases:

1. if $S_t(ct).\text{Resp} = \text{YES}$, for any $1 \leq j \leq k$, then $N_s(p) = N_s(c_1) \ldots N_s(c_k)$.

2. otherwise, let be $c_j$, where $1 \leq j \leq k$, the leftmost child of $p$ with $N_s(c_j) = S_t(ct_j) \ldots S_t(ct_{j+m})$ such that $S_t(ct_j).\text{Resp} = \text{ABORT}$, for all $1 \leq l \leq m$. In this case, $N_s(p) = N_s(c_1) \ldots N_s(c_j)$.

Thus, $N_s(p)$ contains a sequence of chained transaction states in which either all chained transactions successfully finished CTP or all chained transactions are aborted.

MCCTP, described in Table 5, recursively calculates $N_s(t)$ ($t$ is the root of the tree derived from the customer’s choices), traversing the tree in a similar manner with depth-first search. For any node $p$ of the tree, we use a child array to store the node states of all children of $p$.

At the lines 2-3, the protocol computes $N_s(p)$ for a node $p$, depending on the node state of the left most child of $p$. For a node $p$ with at least two children, the while loop (the 5-12 lines) computes the node state of any child of $p$ except the left most one. If an aborted chained transaction/sequence of chained transactions leads to aborting the entire aggregate transaction, but some chained transactions from the aggregate transaction successfully completed CTP, then the ones that are successfully must also be stored in the node state corresponding to $\land$ operator (line 9) and afterwards aborted by applying AggregateChainsAbort sub-protocol (line 10). We will describe AggregateChainsAbort sub-protocol in section 4.4.1.

At line 13, the protocol computes $N_s(p)$ for a node $p$ with a product identifier as info.

### 4.4.1 AggregateChainsAbort Sub-protocol

In MCCTP, an unfair case for $C$ may occur: the entire aggregate transaction is not successful, but $C$ has paid for certain products purchased in the successful chained transactions. For example, for a node $p$ that corresponds to $\land$ operator, MCCTP computes the node state $N_s(p) = S_t(ct_1) \ldots S_t(ct_k)S_t(ct_{k+1}) \ldots S_t(ct_m)$, where $S_t(ct_i).\text{Resp} = \text{YES}$ for any $1 \leq i \leq k$ and $S_t(ct_j).\text{Resp} = \text{ABORT}$ for any $k+1 \leq j \leq m$.

To solve this unfair case, the customer’s payment Web segment initiates AggregateChainsAbort($N_s(p)$) sub-protocol by sending to PG a customer’s request to abort any chained transaction from $N_s(p)$ that successfully finished CTP.

$C \rightarrow PG : \{N_s(p), \text{SigC}(N_s(p))\}_{PKPG}$

The customer request is built from $N_s(p)$ and $C$’s signature on $N_s(p)$, both encrypted with $PKPG$. PG decrypts the customer’s request, obtains the sequence of chained transaction states $S_t(ct_1) \ldots S_t(ct_n)$ in $N_s(p)$, and checks $C$’s signature on $N_s(p)$. For each $S_t(ct_i) \in N_s(p)$, PG searches into its database the appropriate entry, and if it finds out then also checks his signature.

If all checks are successfully passed, then PG sends to the bank the customer’s request. Each chained transaction $ct_i = \text{CTP}(C, B_i, \text{Pid}_i) = s_0s_1 \ldots s_n$ such that $S_t(ct_i) \in N_s(p)$ and $S_t(ct_i).\text{Resp} = \text{YES}$ will be aborted as follows:

1. The bank cancels the transfer corresponding to $s_0$, from $C$’s account into $B_i$’s account, and generates $S_t(ct_i)$ as in Resolution 1 sub-protocol.

2. After receiving $S_t(ct_i)$ from the bank, PG sends it to $C$ and $B_i$ as a proof of $s_0$’s abortion.

3. $s_1, \ldots, s_n$, in this order, will be aborted by applying the same technique as in Resolution 1 sub-protocol ($B_i$ requires and obtains from PG abortion of $s_1$, and so on, the last broker from chain requires and obtains from PG abortion of $s_n$).

### 5 SECURITY ANALYSIS

In this section, we will analyze the security requirements stated in section 3 for our MCCTP.

#### 5.1 Effectiveness

If every party involved in MCCTP behaves according to the protocol’s steps, does not want to prematurely terminate the protocol and there are no network communication delays/errors, then each chained transaction executed in MCCTP is successfully finished, without TTP involvement. Therefore, our protocol meets the effectiveness requirement.

#### 5.2 Fairness

To show that MCCTP ensures fairness, three requirements must be satisfied. First requirement to obtain fairness in MCCTP is to obtain fairness in CTP. So, we will show that CTP ensure fairness by analyzing all possible behaviors of each participant involved in CTP: the customer, the brokers and the provider.

**Customer.** In CTP, each initiator (customer or broker) has a payment Web segment that performs initiator’s side protocol actions and is a soft digitally signed
Table 5: Multi-Chained complex transactions protocol.

\[
MCCTP(t)
\begin{align*}
1. & \text{if } (t \rightarrow \text{left} \neq \text{NULL}) \text{ child}[0] = MCCTP(t \rightarrow \text{left}); \\
2. & \text{if } ((t \rightarrow \text{info} = \lor \text{ and } St(ct) \text{ Resp} = \text{YES}, \text{for all } St(ct) \text{ from child}[0]) \text{ or } \\
3. & (t \rightarrow \text{info} = \land \text{ and } St(ct) \text{ Resp} = \text{ABORT}, \text{for all } St(ct) \text{ from child}[0])) \text{ Ns}(t) = \text{child}[0]; \text{ return Ns}(t); \\
4. & j = 1; \text{ k} = t \rightarrow \text{left} \rightarrow \text{right}; \\
5. & \text{while } (k \neq \text{NULL}) \\
6. & \text{child}[j] = MCCTP(k); \\
7. & \text{if } (t \rightarrow \text{info} = \lor \text{ and } St(ct) \text{ Resp} = \text{YES}, \text{for all } St(ct) \text{ from child}[j]) \text{ Ns}(t) = \text{child}[j]; \text{ return Ns}(t); \\
8. & \text{if } (t \rightarrow \text{info} = \land \text{ and } St(ct) \text{ Resp} = \text{ABORT}, \text{for all } St(ct) \text{ from child}[j]) \\
&& \text{for } (c = 0; c \leq j; c = c + 1) \text{ Ns}(t) = \text{Ns}(t)\text{child}[c]; \text{ end for} \\
9. & \text{AggregateChains} \text{Abort}(Ns(t)); \text{ return Ns}(t); \\
10. & k = k \rightarrow \text{right}; \text{ j} = j + 1; \\
11. & \text{end while} \\
12. & \text{if } (t \rightarrow \text{info} = \text{Pid}_i) \text{ Ns}(t) = \text{St}(\text{CTP}(C, B_i, \text{Pid}_i)); \text{ return Ns}(t); \\
13. & \text{else if } (t \rightarrow \text{info} = \land) \text{ k} = t \rightarrow \text{left}; \\
14. & \text{while } (k \rightarrow \text{right} \neq \text{NULL}) \text{ k} = k \rightarrow \text{right}; \text{ end while} \\
15. & \text{Ns}(t) = \text{Ns}(k); \text{ return Ns}(t); \\
16. & \text{else for } (c = 0; c \leq j - 1; c = c + 1) \text{ Ns}(t) = \text{Ns}(t)\text{child}[c]; \text{ end for} \\
17. & \text{return Ns}(t); \\
18. & \text{end if} \\
19. & \text{end if} \\
20. & \end{align*}
\]

by \(PG\) that is a trusted party. So, any corruption of the payment Web segment by user is impossible. If \(C\) initiates \(CTP\) with a broker \(B_i\), but a network communication error appears and \(B_i\) does not receive the message from \(C\), then \(C\) will not receive any payment evidence from \(B_i\). If we consider only the current chained transaction, then this is not an unfair case. However, the state of this chained transaction is undefined, and this scenario must be solved if this chained transaction belongs to a multi-chained complex transaction. To ensure fairness in \(MCCTP\), each chained transaction must have a defined state (successful or aborted). So, in this case, \(C\) waits for an evidence from \(B_i\) until the timeout interval \(t\) expires in which case \(C\) initiates \(Resolution\ 2\) sub-protocol to abort the current chained transaction.

The only scenarios in which \(C\) can behave dishonest in \(CTP\) are when he wants to buy a product and his account balance is insufficient, or \(C\) provides incorrect data (card number, challenge code). In both scenarios, the only case in \(CTP\) in which the fairness is not insured is when in a chained transaction \(ct = CTP(C, B_i, \text{Pid}_i)\) is the first aborted subtransaction (in the reverse order: from \(s_n\) to \(s_0\)) in which \(B_i\) receives from \(PG\) an aborted \(CE_{i+1}\). In this case, \(s_1, \ldots, s_n\) are successful, and to obtain fairness in \(CTP\), \(B_i\) initiates \(Resolution\ 1\) sub-protocol. After \(Resolution\ 1\), all subtransactions from \(ct\) are aborted, so fairness is ensured.

**Broker.** If \(s_i\) is the first subtransaction from a chained transaction \(s_0s_1 \ldots s_n\), in which \(B_{i+1}\) does not receive \(PO_{i+1}\) from \(B_i\) because of a network communication error, then \(B_i\) will not receive any payment evidence from \(B_{i+1}\). By the same reasoning as in \(C\)'s case, \(B_i\) waits for an evidence from \(B_{i+1}\) until the timeout interval \(t\) expires in which case \(B_i\) initiates \(Resolution\ 2\) sub-protocol to abort the current chained transaction.

We detail below all scenarios in which a broker \(B_i\) can behave dishonest in \(CTP\):

- In \(s_i\), \(B_i\) wants to buy from \(B_{i+1}\) a product and his account balance is insufficient, or \(B_i\) provides incorrect data (card number, challenge code). In these scenarios, the only case in \(CTP\) in that fairness is not insured is when \(s_i\) is the first subtransaction (in the reverse order) from the chained transaction \(s_0s_1 \ldots s_n\), in which \(B_{i+1}\) receives from \(PG\) an aborted \(CE_{i+1}\). To obtain fairness in \(CTP\), \(B_{i+1}\) initiates \(Resolution\ 1\) sub-protocol to abort all subtransactions from the chained transaction.
- \(s_i\) is the first subtransaction from the chained transaction in which \(B_i\) does not send \(PO_{i+1}\) to \(B_{i+1}\) and also any evidence to \(B_{i-1}\). In this case, \(B_{i-1}\) waits for an evidence from \(B_i\) until the timeout interval \(t\) expires in which case \(B_{i-1}\) initiates \(Resolution\ 2\) sub-protocol (case 3) to abort all subtransactions from the chained transaction.
- \(s_i\) is the first subtransaction from the chained transaction in which \(B_i\) as initiator does not send \(PO_{i+1}\) to \(B_{i+1}\), but continues as receiver in \(s_{i-1}\) sending \(PR_{i-1,i}\) to \(PG\). In this scenario, the only case in \(CTP\) in that fairness is not insured is when \(B_i\) receives in \(s_{i-1}\) a successful \(CE_{i-1,i}\) from \(PG\).
So, $B_i$ obtains the payment in $s_{i-1}$ from $B_{i-1}$, but $B_i$ did not bought in $s_i$ the product requested by $B_{i-1}$. In this case, $B_i$ cannot respond in $s_{i-1}$ to $B_{i-1}$ with a valid message ($B_i$ has a successful $CE_{i-1}$, but cannot obtain a successful $E_i$). Then, $B_{i-1}$ initiates Resolution 2 sub-protocol (case 1) to abort all subtransactions from the chained transaction.

- $s_i$ is the first subtransaction from the chained transaction in which $B_i$ sends $PO_{i,i+1}$ to $B_{i+1}$, but $B_{i+1}$ does not send the successful payment evidences if and only if each of initiators/receivers tries to deny its involving in MCCTP, then PG has evidences to prove the contrary.

- $s_{i-1}$ is the first subtransaction from the chained transaction in which $B_i$ has received the evidence $CE_{i-1,i}$, and also he has the evidences $CE_{i,i+1}, E_{i+1}$ from $s_{i+1}$. However, $B_i$ does not send the evidences or sends invalid evidences to $B_{i-1}$ in $s_{i-1}$, or a network communication error appears and $B_{i-1}$ does not receive the evidences from $B_i$. In this case, $B_{i-1}$ waits for an evidence from $B_i$ until the timeout interval $t$ expires in which case $B_{i-1}$ initiates Resolution 2 sub-protocol (case 1 or case 2) to ensure fairness in CTP.

**Provider.** In CTP, $P$ can behave dishonest in the following two scenarios. First, $P$ receives $PO_{i,n+1}$ from $B_n$, but $P$ does not send $PR_{i,n+1}$ to PG. Second, $P$ receives the evidence from PG in $s_n$, but $P$ does not send the evidence to $B_n$ or a network communication error appears and $B_n$ does not receive the evidence from $P$. In both cases, $B_n$ waits for an evidence from $P$ until the timeout interval $t$ expires in which case $B_n$ initiates Resolution 2 sub-protocol to ensure fairness in CTP.

As we can see, after analysis above considering all possible scenarios in which the customer, the brokers or the provider behave dishonest, fairness in CTP is ensured.

The second requirement to obtain fairness in MCCTP is that for any optional transaction from the complex transaction, the chained transactions corresponding to exactly one option successfully finished CTP, or all chained transactions are aborted. This requirement is satisfied in MCCTP from the way in which the node state corresponding to $\lor$ operator is computed (Table 5).

The third requirement to obtain fairness in MCCTP is that for any aggregate transaction from the complex transaction, either all chained transactions belonging to the aggregate transaction successfully finished CTP, or all are aborted. When we consider aggregate transactions in the complex transaction, only a scenario in which fairness is not ensured can occur: the entire aggregate transaction is not successful, but $C$ has paid for certain products purchased in the chained transactions that successfully finished CTP. To obtain fairness in this case, the customer’s payment Web segment initiates AggregateChainsAbort sub-protocol from section 4.4.1 to abort any chained transaction that successfully finished CTP and that belongs to the unsuccessful aggregate transaction. Thus, any chained transaction which belongs to the aggregate transaction is aborted, solving this unfair case.

As a result, fairness exchange of payment for successful payment evidence in multi-chained complex transactions is preserved.

### 5.3 Timeliness

MCCTP uses CTP. Resolution 1 sub-protocol from CTP is executed between initiators/receivers and PG and the communication channels between initiators/receivers and PG are resilient. Also, in CTP, we have introduced a timeout interval $t$ in which each initiator waits the payment evidences from the corresponding receiver. If $t$ expires and an initiator does not receive the corresponding payment evidence, then he initiates Resolution 2 sub-protocol with PG. Further, in all cases, Resolution 2 will apply Resolution 1. So, CTP execution will be finished at a certain finite point of time. Moreover, if in MCCTP, the AggregateChainsAbort sub-protocol is executed, then the communication channels used are resilient. As a result, the execution of MCCTP will be finished at a certain finite point of time. After the MCCTP finish point, $C$ received the successful payment evidences if and only if each of involved chained transaction successfully finished CTP. Also, after the finish point, $C$ received the aborted payment evidences if and only if each of involved chained transaction is aborted. So, after the MCCTP finish point, the level of fairness achieved cannot be degraded.

### 5.4 Non-repudiation

In each chained transaction $s_0s_1\ldots s_n$ executed in MCCTP, in each subtransaction $s_i$, PG stores in its database the payment request $PR_{i,i+1}$. In the component $P_{Mi}$ from $PR_{i,i+1}$ there is included the signature of $B_i$ on $PL_i$, which proves the involvement of $B_i$ as initiator in $s_i$. Also, $PR_{i,i+1}$ includes the signature of $B_{i+1}$ on $Id_i, B_i, B_{i+1}, PkB_i$; and $Am_i$, which proves the involvement of $B_{i+1}$ as receiver in $s_i$. So, if any of initiators/receivers tries to deny its involving in MCCTP, then PG has evidences to prove the contrary.
5.5 Confidentiality

In MCCTP, every message transmitted is hybrid encrypted with the public key of the receiver. So, only the authorized receiver of a message can read the message’s content. After execution of any chained transaction $s_0 \ldots s_n$, each party $B_i$ from chain knows information from $s_{i-1}$ where $B_i$ participates as receiver and from $s_i$ where $B_i$ participates as initiator. In addition to the known information from $s_{i-1}$ and $s_i$, $B_i$ receives only $E_{i+1}$ that ensures him that $s_{i+1}$ was successful. We remark that from $E_{i+1}$, $B_i$ obtains the response $Resp$ and the fresh random number generated in $s_{i+1}$, but these can not lead $B_i$ to determine the identity of the party that follows $B_{i+1}$ in chain. As a result, MCCTP ensures confidentiality.

6 CONCLUSIONS

In this paper, we proposed an optimistic Fair Exchange E-Commerce Protocol for Multi-Chained Complex Transactions.

Some of the future work on this proposal are discussed in what follows. First direction is to extend the proposed protocol with a penalty mechanism applied to the agents that behaves dishonest. This mechanism will encourage the agents to behave honestly, otherwise, their dishonest behavior will not bring benefits above the impose penalties. As a result, the number of aborted transactions caused by dishonest behavior will be reduced. Second direction is to allow one to many subtransactions between brokers. This will lead to a more complex business model in which each broker might perform aggregate, optional or chained transaction. Third direction is to extend the proposed protocol to provide anonymity of the customer. This implies an elaborated infrastructure because an online TTP is required.

REFERENCES


