Fair exchange protocols with anonymity and non-repudiation for payments.

Athanasios Polychronis

Technical Report

RHUL–MA–2013– 2

01 May 2013

Information Security Group
Royal Holloway, University of London
Egham, Surrey TW20 0EX,
United Kingdom

M.Sc. in Information Security

Title: Fair exchange protocols with anonymity and non-repudiation for payments

Supervisor: Konstantinos Markantonakis

Submitted as part of the requirements for the award of the MSc in Information Security at Royal Holloway, University of London.

I declare that this assignment is all my own work and that I have acknowledged all quotations from the published or unpublished works of other people. I declare that I have also read the statements on plagiarism in Section 1 of the Regulations Governing Examination and Assessment Offences and in accordance with it I submit this project report as my own work.

Signature:

Date:

Fair exchange protocols with anonymity and non-repudiation for payments

By

Athanasios Polychronis

Abstract

Our daily routine requires the establishment and control of a number of electronic transactions. In addition, the average global internet usage as well as the web activity per person has increased throughout the last few years and business today is reliant on open networks and systems that are always connected to support constant communication. As a result, IT is a critical component for all organisations and there is, often, no fall-back position in case of a failure.

Network protocols are required to facilitate economic transactions and to usually protect the participants mutually in this process. Often, in this context, they are required to provide fairness, anonymity and non-repudiation.

This thesis aims to illustrate the evolution of these protocols and point out situations where some existing protocol schemes have mismanaged their expected services or tried to provide them in a weaker sense. Additionally, we review the proposed formal methods for protocol analysis in order to identify their results as well as their limitations. The main goals of this dissertation, however, are the optimisation of the Netbill protocol in order to make it more secure and fairer as well as suggestions for its adaptation in the modern social networking and payment infrastructure.

Table of Contents

Table of Figures

Chapter 1: Introduction

The aim of this chapter is to provide an introduction which will illustrate the structure of the dissertation as well as present the motivation, the requirements and contributions to the work.

i. Motivation

The introduction of electronic commerce transactions was a revolutionary step which affected enormously the relationship between merchants and customers and redefined the way business is done in a significant degree. The shift from the standard, point of sale (POS) transaction at a cash register, to the virtual, electronic, card-not-present (CNP) one, brought benefits to both customers and merchants. However, as change is inherently ambiguous, it resulted in expanding the attack surface and thus creating the need for higher security in transactions. In the traditional environment most of the measures taken are physical and are there to deter the parties from misbehaving. The effectiveness of these measures is satisfactory and in most cases, it is easy to detect and control any wrongdoing. However, the virtual environment of an electronic transaction can prove less effective in terms of detection as one of the communicating parties can more easily misbehave during the flow of a protocol and then disappear. Achieving a settlement becomes even harder if we have provided some form of anonymity for the communicating parties, which makes spotting them more difficult in an open network with no former relationships, identities, or common physical locations.

Many things can go wrong in an electronic transaction. A merchant could decide not to send a product after receiving a payment from a customer or vice versa. It can be easily deduced that the backbone of this virtual environment is trust. Trust could be explained as “dependence on something future or contingent, reliance on future payment for property delivered” (Merriam-Webster, 2012). As a result, trust is relative, personal, difficult to communicate and essential for business. (Dorey, 2011) It is inevitable to include trust without assuming a certain amount of risk. The participants in an electronic transaction initiate a protocol without any former personal contact or, necessarily, any trust established between them. As a result, they want to feel confident that their trust is not wrongly placed and that the existing risks are kept to a minimum level. Many of the proposed protocols do not treat the communicating parties equally. For example, on many occasions the merchants are treated as trusted whereas customers are not treated as such and are often required to take more steps in order to complete an electronic payment. However, this can cause the suspiciousness and concern of a customer who fears that, if he/she makes a payment in advance, he may be exposed to a malicious merchant who can take advantage of the situation by not sending the merchandise or by claiming he did not receive the payment. The following example is given by (RAY, INDRAJIT RAY and INDRAKSHI, 2002).

“A customer C contacts an on-line merchant M for a product P. The product is an electronic database. Now the customer is not willing to pay for the product without being sure it is the right database sent by the merchant. A merchant is not willing to give the database unless he is sure that he will receive the proper payment. If the merchant delivers the product prior to receiving the payment, the fraudulent customer may simply disappear after getting the product, causing loss for the merchant. If, on the other hand, the customer pays before receiving the product, the merchant may not deliver it or may deliver some wrong product.’’

ii. Document structure

The aim of this dissertation is to identify transactional problems and, if possible, give solutions by providing specific security services. The motive is to examine if truly fair exchange protocols can be realistic and feasible in an e-Commerce environment and if this can be combined with the protection of the anonymity and privacy of the participating parties.

The rest of this thesis is organised as follows:

The main objective of Chapter 2 is to detect distinct categories of protocols after a comprehensive review on the academic literature. Due to the large volume of proposed protocols, we choose the most important and we briefly describe their flow, the parties involved and their contribution to the evolution of payment systems. Additionally, we point out the security requirements for a payment system and define the most common security threats.

Chapter 3 focuses on formal analysis and generally automated software tools for verification of protocols. We choose the (Zhou, Gollmann, 1996) protocol for our review as there have been many attempts to formally analyse it. After having studied these different methods, we are able to describe, compare, and reach conclusions in terms of the authors' perspective as well as the effectiveness of formal methods.

In chapter 4, we modify the protocol of (B. Cox, D. Tygar, and M. Sirbu, 1995) in order to enhance its fairness and make it "more suitable" for payments. Furthermore, we propose a way to deploy this protocol in the current networking/payment service systems, while protecting the privacy of users by making use of modern identity management standards and common features of these services.

In Chapter 5, we summarise the motivation and the contribution of this thesis and we make suggestions for further research.

iii. Definitions

This chapter will provide explanations in order to clarify and avoid ambiguity for terms used above and others that will be used henceforth.

Definition 1: A *third party* is a participating party in an e-Commerce transaction and is someone other than the entities directly involved (merchant, customer). A third party is often responsible for preventing the parties from misbehaving or for settling any differences in case of any wrongdoing.

Definition 2: A *trusted third party* (TTP) is a third party who is trusted not to misbehave or collude with any one of the parties to the detriment of the other.

Definition 3: Fair exchange (or fairness) implies two parties exchange items of value in such a manner that no party can gain advantage over the other by misbehaving, or by prematurely aborting the protocol.

Definition 4: An *optimistic* fair exchange protocol is a fair exchange protocol that relies on a trusted third party (TTP) but does not require for it to be dedicated and actively involved in every session of the protocol. Optimistic protocols expect the parties not to misbehave. Only if something goes wrong (e.g. dispute, network issues) the TTP is asked to participate in order to resolve the matter.

Definition 5: *Strong fairness* in e-Commerce means that, at the end of a protocol run, either both participants have acquired their expected items or no one does. *Weak fairness*, on the other hand, indicates that either strong fairness is achieved, or a participant, who cannot obtain his expected item from another player, is capable of resolving his/her dispute by presenting evidence gathered during the protocol and proving any misbehaving at a court.

Definition 6: In the e-Commerce context, *anonymity* implies that the true identity of an entity is not revealed to any party while privacy protection is usually related to the transaction details, sensitive information, spending habits. *True* anonymity is an unrealistic requirement in a network protocol, as, although required in a payment scheme, it is difficult to achieve. The communication traffic between the participants (IP address, cookies, etc.) compromises their anonymity in a degree. Moreover, on some occasions, the true identity of the players must be known by, at least, one trusted party.

For all these reasons, *pseudonyms* are issued to the users. Pseudonymity is a lesser form of anonymity and implies that a user reveals a special identifier to the provider instead of his true identity. Identifiers are usually generated on a regular basis (Windley, 2005).

Definition 7: *Non-repudiation* indicates that a participant in a protocol cannot later deny any messages he sent (non-repudiation of origin) or received (non-repudiation or receipt) during that protocol run. As Gollmann mentions in his book "Computer Security", non-repudiation services provide unforgeable evidence that a specific action occurred. (Gollmann, 2011)

Definition 8: *Efficiency* implies that a protocol requires the optimal computational time and resources to realise a transaction. By complicating the process or including many parties, efficiency can be undermined.

Definition 9: The European Community Directive defines *Digital Signatures* as data in electronic form attached to, or logically connected with, other electronic data and which serve as a method of authentication. (European Parliament, 1999) In the context of this thesis, when we use the term digital signatures, we will usually refer to digital signature schemes based on RSA.

Definition 10: *Public-key Certificates* are data that link public keys to data relating to the assurance of purpose of that public key. They can be considered as a trusted directory entry in a sort of distributed database. A certificate contains four essential pieces of information: Name of the owner, Public-key value, Validity time period, Signature of the Certification authority (M.Martin, 2012).

Chapter 2: Fair exchange and anonymous protocols

i. History of fair exchange protocols

We can distinguish many different types of fair exchange schemes.

A really basic classification is based on the number of participants in the transaction. As such, we have two categories; two-party based and third-party based protocols.

Two-party based protocols

The first generation of two-party based protocols that attempted to achieve fairness was introduced by Blum (Blum, 1983). Blum describes a situation where a divorced, distrustful couple tries to find a way of trading a set of secret passwords without anyone else obtaining the secret. Due to the lack of trust between them, they require a protocol that prevents cheating from either side. The secret passwords can be prime factors of their publicly announced composite numbers and work in the same way as private keys in RSA. The two players start exchanging their secret keys bit by bit alternately. They start a gradual parallel exchange of parts of items to ensure that this exchange takes place ‘almost’ simultaneously and that no party can gain significant advantage over the other. Each party has to prove that the bit released is correct and that it is a part of the secret. The protocol assumes though that the two parties have equal computational capabilities and the same knowledge of algorithms. The protocol can also be deployed to certify e-mails and sign contracts with the use of digital signatures.

Gradual Secret Exchange

Simultaneous secret exchange schemes are based on the fact that each one of two entities A and B possesses a secret (a and b are of equal size). One party's secret is valuable to the other party, and therefore, they are both willing to exchange secrets. This gradual exchange is realised with the help of two predefined functions f() and g() with the property that, if A sends f(a) to B and B sends g(b) to A, there is no way for the respective recipient to recover the secret. Then, both parties release their secrets bit-by-bit in every round. As Deng et al. (Robert H. Deng, Li Gong, Aurel A. Lazar and Weiguo Wang, 1996) mention in their paper, in order for such a system to be successful, the correctness of the released bits and fairness must be established. The term correctness reflects the necessity of checking the information received so that a party is not trading his secret for worthless data. Furthermore, fairness can only be achieved if both parties have equal computational power at every stage of the protocol.

The idea that Even et al. (Even Shimmon, Oded Goldreich, and Abraham Lempel, 1985) presented in 1985 includes the notion of the 1-out-of-2 oblivious transfer protocol. "This 1-out-2 oblivious transfer allows a participant to transfer exactly one secret, out of two recognisable secrets, to his counterpart". In other words, the introduction of recognisable secrets provides solutions in an environment, where a receiver wishes to receive a message but the sender is reluctant to send it. Upon receipt, the receiver cannot compute the message but can authenticate that it came from the sender. An example of such a message is a digital signature on a known message. During the protocol one of the parties, the sender S, sends one out of two recognisable secrets to the receiver R with a probability of 0.5. The a-posteriori probability, that R got that message, is still one half if the two parties executed the steps properly. R can detect any misbehaving by S, trying to increase his probability of guessing which message was read by R with a probability at least one half. In contrast to Blum's scheme, this one requires only same computational ability between the parties as well a secure public key signature scheme. If we compare the two schemes it is obvious that they are both based on gradual exchange of information to face issues of mistrust and maintain a simultaneous and fair exchange of information.

By examining these two proposed schemes, we can identify some drawbacks that emanate from the requirement for equal computational abilities. Soon, it was obvious that the need for equal, or even related, computational power made this approach prone to errors and cheating and, of course, the party with the more resources was in a privileged position as it could take advantage over the other. Let us assume that, while transferring a secret bit-by-bit between a bank with infrastructure worth of millions and an individual customer using a common PC, the conversation gets disrupted. If the bank starts an exhaustive key search using thousands of computers for the remaining bits, it has a profound advantage over the individual, for whom an exhaustive key search would be far less effective. Another example that illustrates the shortcomings is the case where both parties have to perform 10^{10} operations and the bank has 10000000 computers capable of performing 1000 operations per second and the customer has only one computer of equal computational power. In that case, the bank would have obtained the secret in 10 seconds, whereas it would take approximately 3 months and 24 days for the customer to perform the same series of calculations.

A breakthrough, in terms of these problems, was made in 1990, as Ben-or et al. (Ben-Or Michael, ODED GOLDREICH, SILVIO MICALI, AND RONALD L. RIVEST, 1990) introduced the notion of a probabilistic approach. Each party has to gradually exchange information in order to reach fairness. While the two parties exchange messages the possibility that a fair exchange is valid increases. Let us examine the case where two parties, A and B, exchange messages over a communication network with the intention of obtaining each other's proof of commitment to a contract C without them being committed in advance. The problem of contract signing cannot be solved with a simultaneous exchange of information; as we saw this is unfeasible in practice. Up until then, fairness was treated as a matter of equal computational effort. Ben-or et al. tried to achieve fairness with the deployment of a high probabilistic correlation scheme. In each round of the protocol A (or B respectively) sends to B a message saying that, with a probability λ, the contract will be valid at an end date D (previously agreed). The probability λ increases after each round until it becomes 1. The two parties are alternately privileged after each round. For instance, after the first round, B can go to a judge with the message he received and the judge can find with a probability $p=\lambda_A$ that the contract is binding. If, however, the deadline is reached, any one of the two parties has the right to invoke the "early stopping procedure". In other words, they can call the judge who decides whether the contract is binding or not. The judge first acquires the last message sent during the protocol flow. Such a message would say something similar to: "With a probability p the contract C will be valid. Signed by A." The judge checks the validity of the signature and, if it is indeed valid, he/she checks for any previous verdicts regarding that contract. If the previous computed value $p_c \leq p$ then the judge decides that the contract C is binding to both parties. Otherwise, he chooses a random value p_c between 0 and 1 and compares it with the current value of p and sends his signed verdict to both parties.

As (RAY, INDRAJIT RAY and INDRAKSHI, 2002) point out, all the above protocols lack simultaneity and, as a result, that causes uneven problems in case one of the parties decides to stop the transaction early. It is also worthy of note that, due to that gradual exchange of information over multiple rounds (sometimes bit-by-bit), we have huge amounts of transmissions and, as a result, a very busy channel which is, arguably, a difficult situation to control.

As appealing as the absence of a dedicated TTP may sound, Even and Yacobi (Even Shimon, Yacobi Yacov, 1980) proved that deterministic protocols cannot guarantee true (100%) fairness in contract signing unless there is some sort of a judge or a trusted third party involved during the signing phase.

In 1996, Sandholm and Lesser (Sandholm Tuomas, Lesser Victor, 1996), in a different attempt that circumvents these problems, adopt an approach suggested in game theory that includes contingency contracts for self-interested parties. The idea behind this approach is that a leveled commitment is being built for the participating parties, as long as they adhere to the protocol, but they also maintain the option of “decommitting” by paying a penalty, in case of any wrongdoing or unexpected behavior in the course of the transaction. The protocol can become fully committing for the parties by setting the penalty rates really high. Bearing in mind that the authors assume that the parties will also behave rationally in such a transaction, we can conclude that this is not a viable solution in the modern e-commerce environment.

Third-party based protocols

Currently, most of the research done in this area involves the use of one or even more trusted third parties in order to tackle the problems we examined above. Again, third-party based protocols can, in turn, be divided according to the role and the nature and, more importantly, the level of involvement of the third party. Thus, we can distinguish in-line, on-line and off-line TTPs.

In-line TTP based protocols

The first use of in-line TTP protocols was for the implementation of certified e-mail schemes. The most important characteristic of this approach, which was also reflected in the case of a certified e-mail framework, is that the TTP is totally dedicated and intervenes in each and every message sent between the participants. The TTP acts as an intermediary who filters and relays the information to one direction or the other.

In 1994, Bahreman and Tygar (Bahreman Alireza, J. D. Tygar, 1994) introduced the first in-line TTP based protocol for certified e-mail. They treated the certified e-mail (CEM) problem as an asymmetric exchange problem, unlike all the previous approaches we examined. The authors present two families of protocols that would enable two equally and mutually suspicious participants to send, and receive respectively, a certified e-mail. The first is called “the believer’s protocol” and it requires the involvement of a trusted third party whereas the second one, called “the skeptics protocol”, is basically a two-party based attempt. As illustrated by figure 1 below, the *Postmaster* participating in the *believer’s protocol* handles every single message transferred between the users and keeps secure records of them. The objective is to easily settle future disputes by establishing non-repudiation of origin and receipt of messages. The authors claimed that this approach provides little to no chance for the participants to cheat or to gain advantage over each other and, in that sense, that it achieves strong fairness.

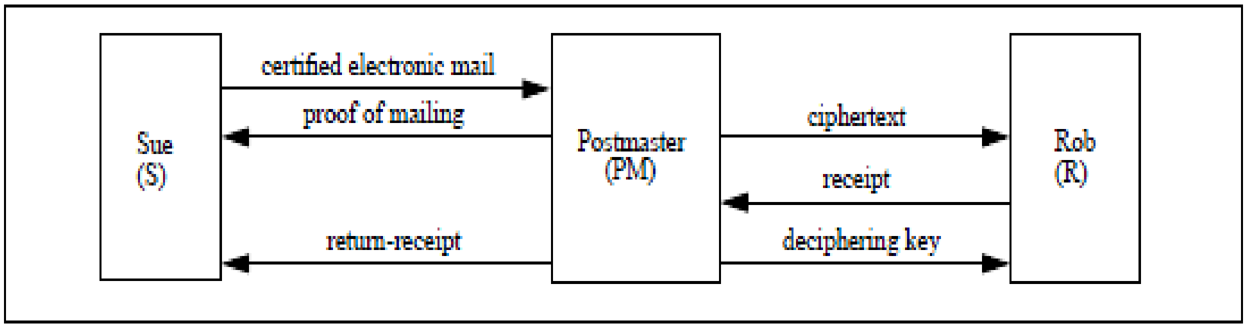


Figure 1: High-level overview of the interactions among the parties in a B-CEM protocol. (Bahreman Alireza, J. D. Tygar, 1994)

Coffey and Saidha (Saidha Puneet, Tom Coffey, 1996) introduced a fair, non-repudiation protocol which uses a "non-repudiation server" (NRS) as a trusted intermediary. Before any exchange of information takes place, the two parties first send their digitally signed evidence to the NRS (proof or origin and receipt). Moreover, the scheme requires that the digital signatures must be supported by "reliable and trustworthy time-stamps" provided by the "time-stamping authority" (TSA). As depicted in figure 2 below, the originator A starts the protocol by generating and sending his partial proof of origin to the TSA where a time-stamp is appended, provided that the originator's signature is valid. Then, the TSA returns the final proof-of-origin message back to A, who checks the validity of the time-stamp. These two messages are encrypted with the respective receiver's public key. Then the originator forwards both messages to the NRS and asks for initialization of non-repudiative communication. The server checks in turn the messages and the signature values sent by the originator and, if they are valid, the NRS sends the partial proof of receipt to the receiver asking for his/her signature. Subsequently, the receiver signs the message and forwards it to the TSA. Similarly, he/she receives the final proof of receipt message after the TSA has confirmed that the signature is legitimate and then submits it to the NRS. Finally, after checking the last message, the NRS stores a copy of the two messages (for any future disputes) and then provides the originator with proof of receipt and the receiver with proof of origin. Again this protocol provides strong fairness, as the trusted third party has kept track of all the messages exchanged between the two parties and can acquire all the information needed before forwarding these messages. The originator and the receiver are at no point communicating directly. As Kremer mentions in his thesis (Kremer, 2003-2004) a small drawback is that the authors do not define a maximum amount of time after which the message is considered lost. This could present problems; for instance, a message could be expected forever and, as a result, timeliness should be addressed and included.

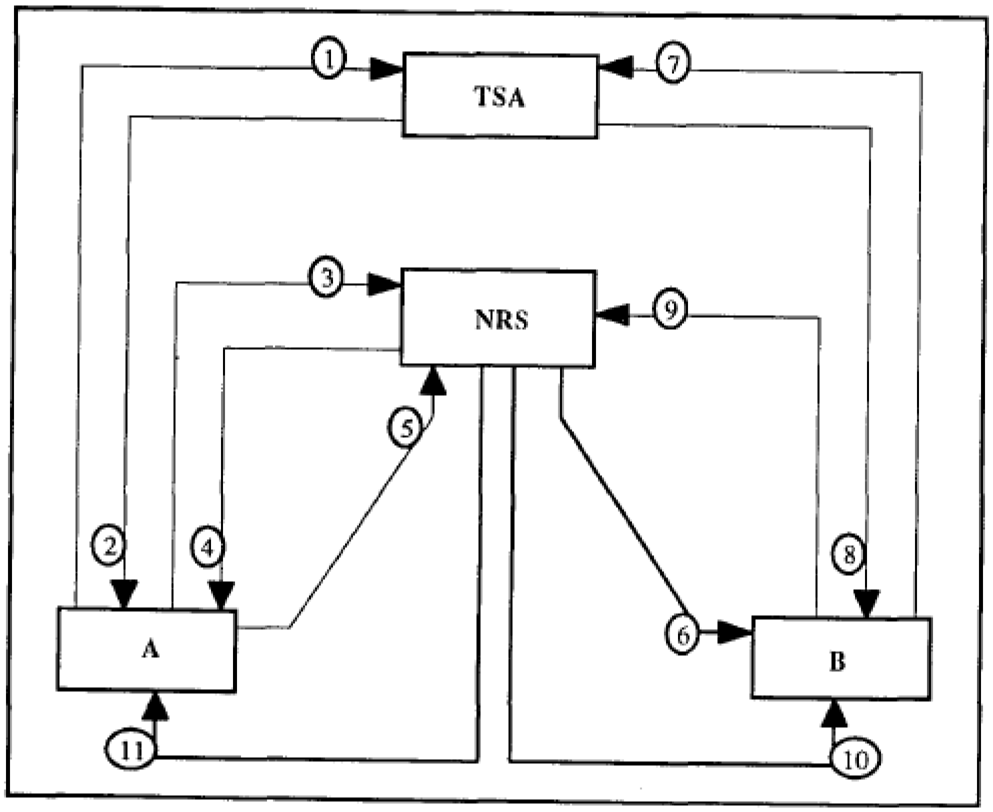


Figure 2 The Non-repudiation protocol (Saidha Puneet, Tom Coffey, 1996)

Summary

To sum up, in-line TTP protocols can provide a satisfactory level of fairness and non-repudiation with the TTP being the cornerstone of the transaction and any dispute process if needed. However, a bottleneck situation can be created due to its constant involvement and total dedication. The trusted third party has to process multiple messages per second and maybe for different transactions. If the resources are not sufficient, then many problems can arise if we also take into account that time is an issue and many schemes do not tolerate delays. In other words, the trusted third party has to be *always* accessible and able to compute and forward messages fast to the respective parties in order for in-line protocols to be efficient and effective. Moreover, such schemes require the TTP to maintain large databases in order to store a plethora of messages that have been or are about to be communicated. The security and storing capacity of these databases with centralized information and resources can prove significant risk factors. The nature of this approach requires these risks to be reduced to a minimum.

On-line TTP based protocols

A comprehensive research on on-line TTP protocols shows that this approach is somewhat more efficient than using an in-line TTP and the main notion is that the TTP should be involved in each run, but not in each transmission. It is no more an intermediary between the communicating parties, but it still generates, validates and stores messages. In case of a dispute, the trusted third party collects and examines the messages transmitted during the protocol in order to 'reach a verdict' and provide solutions.

In 1995, Cox et al. (B. Cox, D. Tygar, and M. Sirbu, 1995) proposed a very important scheme, as it was among the first to realise e-commerce transactions for digital products while using a trusted third party. In particular, Netbill was an online payment-delivery system for low-priced network goods. The authors used a server called Netbill server which handles and maintains accounts for both customers and merchants. A Netbill transaction transfers goods from the merchant to the customer and debits the customer's Netbill account while crediting the merchant's account for the value of the product. The authors tried to include a method of atomic certified delivery, according to which the customer pays for a product if and only he/she receives it intact. Moreover, they tried to incorporate a credential mechanism for the users and enable them to use pseudonyms in order to protect their real identities. The protocol has three phases; the *price negotiation*, the *good delivery*, the *payment* phase and the Netbill server is involved only in the last one. A detailed description of a modified version of this protocol is presented in chapter 4.

In 1996, Zhang and Shi (Zhang N., Shi Q., 1996) tried to tackle the problem of safeguarding EDI (Electronic Data Interchange) systems. To them, it was equivalent to enhancing the non-repudiation service and especially making it hard for a party to falsely deny having received a message during the protocol. They incorporated two simple ideas; a conditional signature process used by a randomised protocol (such as the oblivious 1-out-of-2 transfer) and a public-notice board similar to these we use in our daily lives. The parties participating are the originator, the recipient and a trusted third party called the secure server (SS). The protocol starts with the originator sending the recipient a message encrypted with the recipient's public key. The message comprises the originator's public key certificate and the message M encrypted with a conventional session key of his choosing. In addition, the originator includes his digital signature on a hash of the encrypted message M and a nonce N_A.

M1) A -> B: $\{PC_A \parallel \{M\}_{K_{AB}}\}_{PK_B} \parallel Sign_A(H(\{M\}_{K_{AB}}) \parallel N_A)$

Upon receipt, the recipient B decrypts the encrypted first part using his private key and obtains A's certificate and message M encrypted with the session key. Then B checks

A’s identity and the validity of his/her signature. B also re-computes the hash and checks for any alterations of the sent message M or any replay attempts by verifying that the nonce (session-dependent number) is acceptable.

The next message is sent by the originator and includes his signature on the hash of the encrypted message M along with N_A, a new nonce N_B and t_B which denotes a future time at which - and not later than that - A must publish the session key via the secure server SS. t_B is used as a condition for the validity of B’s signature and prevents A from delaying in order to gain advantage over B.

M2) B -> A: $Sign_B(H(\{M\}_{KAB}) \| N_A \| N_B \| t_B)$

A can now verify the signature and obtain the message. He/she checks the hash and the nonce N_A and, if they do not match the originals, the session is terminated. Moreover, if time t_B is too tight for A to adhere to, A can again terminate the session and restart it later. Otherwise, he/she saves B’s signature and sends the following message to the secure server.

M3) A -> SS: $\{PC_A \| Label \| t_B\}_{PKSS} \| Sign_A(H(\{M\}_{KAB}) \| K_{AB} \| t_B)$

where Label includes all the information needed to be published by the server. Furthermore, t_B is included in both parts so that the server can check its validity. Upon receipt, the server records the time the message was sent, which is practically the time A published the session key. Then the server decrypts the message part using its private key in order to obtain A’s certificate and, subsequently, verifies the signature part using A’s public key. After verifying the validity of t_B, the server publishes the information needed by B to obtain the session key and decrypt message M. It should also be mentioned that the secure server stores the message and the signature part along with the time of arrival of M3 in a database which is available in case any disputes arise concerning the protocol. As Kremer observed, (Kremer, 2003-2004) although the protocol ensures the confidentiality of the messages and keys transferred, a practical problem arises as this requires a lot of messages to be stored indefinitely and thus the database to respectively grow.

In the same year, Zhou and Gollmann (Zhou, Gollmann, 1996) proposed another protocol that would provide fair exchange and non-repudiation services using an on-line TTP. More information about this protocol is provided in the third chapter with an overview of relative formal analyses.

You et al. (You, Zhou, Lam, 1998) tried to modify the previous protocol in order to resolve an issue with compromised keys. We observed that the whole notion of non-repudiation was based on digital signatures. If the signing keys are compromised, the signatures can be forged and, as a result, such keys must be revoked. The question, according to the authors, is how to know for sure that the signature was generated before the keys were compromised. In order to ensure and maintain the validity of non-repudiation evidence during and after a transaction the authors included the idea of evidence chaining. This idea is translated into the insertion of an on-line time-stamping authority and a certificate revocation infrastructure. The time-stamping authority ensures that a message was signed at the time of T_g and, if the certificate is revoked at any time after T_g, the signature will be still considered valid.

A different approach which we can, however, classify as an on-line TTP based scheme was proposed by Franklin and Reiter (Matthew K. Franklin, Michael K. Reiter, 1997). Their solution includes a "semi-trusted" TTP, a TPP that can misbehave on its own but will not collude with any parties. Thus, any member in a network could qualify as a third party and that is very appealing when acting on a large public network. The authors call that fair exchange by "kindness of strangers". The goal of the protocol is to allow two participants A and B to exchange their keys K_A and K_B in a fair way. First, each one of them chooses a random value from a specific group of values and sends it to the other one respectively (x_A to B, x_B to A). The third message is sent by A to the TTP:

$h(K_A)$, $h(K_B)$, $K_A\ x_A^{-1}$, $h(x_B)$,

Where h() is a specific hash function with the property: $h(a) * h(b)= h(ab)$

B respectively sends the TTP the following:

$h(K_B)$, $h(K_A)$, $K_B\ x_B^{-1}$, $h(x_A)$,

and now the TTP is able to check the validity of the communicated messages by checking that $h(K_A)=h(K_A\ x_A^{-1})h(x_A)$ and $h(K_B) = h(K_B\ x_B^{-1})h(x_B)$. If these equations are true the TTP forwards $K_A\ x_A^{-1}$ to B and $K_B\ x_B^{-1}$ to A. Both of them can now retrieve the keys without the TTP obtaining any information about the keys as it is not aware of the random values x_A and x_B. The authors assume that the messages are private and authenticated and that the TTP cannot eavesdrop on the traffic between A and B.

Summary

All the protocols we examined until now require the involvement of a trusted third party at least once during each run and its availability and capacity are crucial for the effectiveness of these schemes. In general, on-line TTP based protocols can also provide strong fairness but the involvement of the TTP, although less, can still cause computational and communication bottleneck, if not addressed properly. Moreover, the TTP has access to almost all messages exchanged between the participants even to the sensitive bits. This, of course, compromises the privacy of the exchanged messages and items, and increases the need for the TTP to be trustworthy.

Off-line TTP based protocols

In this section, we will present the most important of the protocols that utilise an off-line TTP. By off-line we mean that the trusted third party is not involved in a protocol run unless something goes wrong, for example, when a party misbehaves. In case of any wrongdoing, the TTP intervenes in order to make sure that the protocol remains fair and that none of the participants gain advantage over the other. The nature of these protocols is to assume that the parties will most probably behave fairly and that no misbehaving will take place and thus they are called *optimistic* protocols. Of course, these protocols offer significant practical advantages in comparison with the ones we already discussed. The involvement and the workload of the TTP are truly reduced to a minimum and the privacy of the messages transferred is not as broadly compromised. However, they are unable to provide strong fairness, due to their inability to resolve disputes and prevent dishonest players from misbehaving on the fly. Besides, the very nature of the off-line schemes implies that the TTP is always involved after a problem has occurred and not during the normal execution of a protocol.

In 1990, Bürk and Pfitzmann (Bürk, Pfitzmann, 1990) proposed two ways to solve the problems that derive from the lack of simultaneity in exchanges. Both ways included a trusted third party, but especially the first one referred to a TTP that would be involved only in case of detected fraud. The authors basically propose a scheme that would allow the users to use pseudonyms in order to maintain a relative degree of anonymity and check signatures for their validity before they have actually received these signatures. The drawback of this approach is that it is applied on signature exchanges and not on transactions.

A similar attempt was made by Bao et al. (BAO, F., DENG, R. H., AND MAO, W., 1998) who again tried to provide truly fair exchange with the deployment of an off-line TTP, so no loss can be incurred to any party. Their paper describes three different protocols; the first used to exchange signatures on a common file such as signing electronic contracts, the second for exchanging signatures on different files and the third enables the exchange of a confidential file and a signature on the file. The authors use the concept of "Certificate of Encrypted Message Being a Signature" (CEMBS) in order to convince a user that an encrypted message is a certain party's signature on a public file without revealing the signature. Let us briefly examine the first protocol. First, A computes her signature on a message M using her signing key. Subsequently, she generates the ciphertext of that signature using the TTP's public key, generates the CEMBS certificate and sends both to B. Upon receipt, B checks the validity of the certificate by running a verification algorithm and, if it proves to be valid, he can be sure that the ciphertext he received is indeed the ciphertext of A's signature. Then, he computes and sends his own signature on the same message. A in turn, checks B's signature and, if it is valid, she sends her signature to B. Again, this protocol provides a way for the user to check the validity of a signature before he/she has actually received it. The authors claim that this is the first off-line system to achieve true fairness. However, it can only apply to the exchange of signatures and not to transactions.

Asokan et al. (Asokan, Shoup, Waidner, 1998) tried to extend the applicability of these protocols to the exchange of any two digital products, such as digital payments for receipt of products, contract signing, certified email, or even trading one product for another. The proposed protocol, which assumes that the communication channel used is resilient, enables the players to asynchronously and unilaterally complete or abort the protocol run. The protocol is composed of three sub-protocols. The *exchange* protocol is only executed during a normal run, whereas the *abort* and two *resolve* protocols are executed in case of any problems or disputes. The flow of the protocol is somewhat similar to the ones already discussed. A and B send their signed commitments to one another regarding the expected items and, if the protocol is followed properly, they finally exchange these items. Moreover, A (the originator) has an inherent disadvantage as B (a dishonest recipient) can run the recovery protocol after receiving the first message from A. Thus, A is the only one that can initiate the abort protocol whereas the recovery protocols can be run by both parties. The recovery protocols are usually executed when the 3rd or the 4th message (or both) are not received. In that case, the TTP intervenes, receives the first two messages of the exchange protocol by the interested party and provides a successful solution or generates an affidavit which can be used in court.

Asokan et al. (Asokan, N., Schunter, M. and Waidner, 1997) proposed another scheme that, as they claimed, provides fair exchange using an optimistic protocol and that applies to the exchange of any two items. The main contribution of this protocol is that it requires at most five messages to complete its run and, as a result, it is more light-weight than most of the signature based non-repudiation approaches. Again, the two participants A (the originator) and B (the recipient) agree to exchange two items. A initiates the protocol by sending her signed commitment, the description of the products and a specific time frame. After B has received A's commitment he executes the expect() function in order to establish if the descriptions of the two products match. If they do not match, he can abort the protocol. Otherwise, he executes the commit() procedure to produce a commitment to his product and sends it signed to A. At this point, the two participants have agreed on the terms of the exchange and the exchange takes place in messages 3 and 4. The players use the functions open() and fits() to determine respectively whether the open commitment fits the item delivered and whether the description fits the item delivered. The fifth message is sent for recovery purposes. If, for example, B does not send the expected item in message 4 after receiving his item in message 3, A can include the TTP in the process and provides him with the initial agreement in order to reach a solution. The TTP checks the message, provides a reliable channel between them and, if B does not again respond with the correct item, then the TTP issues an affidavit which can be used later for revocation or replacement of the item. It can be easily deduced that, in order for fairness to be achieved, the item sent by the originator has to be revocable or the item sent by the recipient has to be generable. This, of course, may prove to be a major drawback as an actively involved third party might be necessary in order to manage this process. Nevertheless, this method that Asokan et al. used, separating the protocol into one main protocol and the three sub-protocols is widely adopted by many other optimistic fair exchange protocols based on generatability.

ii. Anonymous Payment protocols

In the previous chapter, we focused mostly on fair exchange and non-repudiation services and in what sense these are provided by the proposed protocols. In this chapter, we are going to review how anonymity, which is an element of privacy, is provided in payment protocols. This service is going to be examined separately because anonymity is, in a sense, non-compatible with fair exchange and non-repudiation. The reason is that it is difficult for a user to truly retain his anonymity in a protocol that requires him to authenticate himself and its goal is to prevent him from repudiating his actions. Nevertheless, anonymity is a very important service, maybe the most important in regards to public acceptance. We should always bear in mind that no scheme or protocol, no matter how secure or fair it is, can be successful if it is not user friendly and appealing to the public. Anonymity can be translated into secrecy of payment data and information, but also customer and merchant information. It can be found in three cases in electronic payments. In some cases, one of the two parties wants to remain anonymous. This scenario is often a B2C (business to customer) model, where a customer buys a product from a merchant online. The merchant might want to remain anonymous in order to protect his privacy and personal information (spending habits, sensitive information etc.), whereas the merchant will usually not require the same thing. The second scenario regards transactions where both parties wish to remain anonymous to each other as, for example, in a C2C transaction over the internet. Both customers might want to remain anonymous, the seller while advertising the product and the buyer while paying for it. There is also the scenario which includes participants who already know each other but want to hide their transactional activities from third parties who might be interested in these.

An early approach to realise anonymity in the form of an anonymity channel was proposed by Chaum in 1981 (Chaum D, Untraceable Electronic Mail Return Addresses and Digital Pseudonyms, 1981). The goal of this paper is to find a solution against the "traffic analysis problem", the problem of keeping confidential which parties are communicating and when. The protocol is based on the involvement of a computer between the participants which is called the "mix". A mix is a node in the network that carries a number of messages, usually of equal size, and uses public key cryptography to change them so that they cannot be observed by unauthorised third parties. The sender prepares the message for delivery by encrypting it with the recipient's public key, appends the address of the recipient and then encrypts the whole message with the mix's public key. The mix's objective is to conceal the correspondences between the items in its input and those in its output. Thus the mix strips the incoming messages off the identifying information and forwards the messages to the next mix-node (if multiple mixes are required) according to the routing instructions it obtained after decryption. It should also be mentioned that the items received are not sent out according to the order of arrival but on a lexicographical order. Furthermore, the protocol can provide a backward channel so that the recipient can communicate with the sender anonymously. In particular, the sender can send an anonymous, untraceable address to the recipient so that he can also receive messages anonymously.

However, this approach is not successful in the modern e-commerce schemes as information about the participants can leak from all the messages that are communicated (e.g. personal information included in the message obtained by eavesdropping).

Blind Signatures

A year later, Chaum (Chaum D. L., 1982) proposed one of the fundamental methods of achieving anonymity in e-commerce, the blind signature. Let us examine the use of blind signature in an untraceable transaction example. The customer wants to send his bank a message m in order to be signed but he does not want any information about the message to be revealed. Let us assume that the customer's bank has access to a RSA public and secret key P_{CB}=(e,n) and S_{CB}=(d) respectively. First, the customer generates a random value R so that gcd(x,n)=1, in other words, a number that is relatively prime to n, and sends the message to the bank in order for it to be signed.

Message 1:

C -> CB: $X = (m\ R^e) \bmod n$

The value X is called blind because of the existence of the random value R, which acts as the blinding factor. Upon receipt, the bank signs X and returns it to the customer.

Message 2:

CB -> C: $X^d \pmod n = (m R^e \pmod n)^d = R (m^d \bmod n)$

The customer can obtain the bank's signature on the message m by computing $R^{-1} X^d$.

It is obvious that no information about m has been revealed to the bank.

Zero-knowledge proof

The second important method for providing anonymity is the zero-knowledge proof. It was first proposed by Quisquater et al. (Quisquater Jean-Jacques, Guillou Louis, Annick Marie, Berson Tom, 1990). In general, zero-knowledge mechanisms can bring significant benefits, but also have practical costs. As mentioned by Prof. Keith Martin in his book "Everyday Cryptography" (M.Martin, 2012), their two main advantages are that they can give solution in situations where mutual trust is non-existent between the participants and when we want to prevent any leak of useful information. "The requirement for a zero-knowledge mechanism is that on entity (the prover) must be able to provide assurance of their identity to another entity (the verifier) in such a way that it is impossible for the verifier to later impersonate the prover, even after the verifier has observed and verified many different successful authentication attempts."

Jakobsson et al. (Jakobsson, M., Raihi, D., Tsiounis, Y. and Yung, M., 1999) mention in their paper that privacy is a big concern for customers in electronic payments as their information can be easily mined or sold to advertisers after a transaction without their approval or knowing. The authors detect three different types of privacy: perfect privacy, revocable privacy and limited privacy.

Perfect privacy or, in other words, anonymity can be achieved by enabling the customer to assure a merchant that the payment information provided is correct without, however, revealing any of that information. A way to achieve that is by using blind signatures which, as we mentioned above, were put forth by Chaum (Chaum D. L., 1982) and used by many others in this context.

Again, we could examine and distinguish the various systems proposed in academic literature according to the nature and the involvement of the TTP they use. Although the following schemes may differ on that and other factors, they all share a common structure composed of three phases. A withdrawal phase (customer and bank), a payment phase (customer and merchant) and a deposit phase (merchant and bank) (Camenisch, L. J., Piveteau, J-M. and Stadler, M. A., 1994).

On-line schemes

From our research on on-line schemes, in the first section of this chapter, we established that it is required of the TTP (in payment schemes the bank) to be connected on-line so it can detect and prevent any fraudulent action (e.g. double spending the same coin).

The first on-line schemes supporting anonymity were proposed by Chaum and they required all the participants to be connected and involved at least once during the protocol run. Two of his papers are based on the idea of electronic coin (Chaum D. , Security without Identification: Transaction Systems to Make Big Brother Obsolete, 1985), (Chaum D. , Privacy Protected Payment Unconditional Payer and/or Payee Untraceability, 1989) and one on electronic check (Chaum D. , Online Cash Checks, 1989).

The electronic coin system (Chaum D. , Security without Identification: Transaction Systems to Make Big Brother Obsolete, 1985) he proposed was based on blind signatures and it was the first on-line payment system providing anonymity. Let us assume that a bank has a special signature which is worth a dollar (coin). First, the customer sends the bank the usual carbon-lined envelope with a plain paper inside asking for a dollar from his account. Subsequently, the bank signs the outer envelope without unsealing it and as a result without obtaining any information about the paper inside. Of course, simultaneously, the bank debits the customer's account for the amount of one dollar. The customer now has to extract the carbon image of the bank's valid signature which is worth a dollar. When he finds something of interest, the customer can provide a merchant with the signed slip. The merchant, then, forwards the signature back to the bank which credits the merchant's account for the same amount after it has verified that the coin is not already spent. The bank is not able to identify the payment and trace it back to this particular customer as it issues signatures like that for all its account-holders. In particular, the bank is not able to relate any withdrawal to any deposit.

The paper of Bürk and Pfitzmann, that we discussed above, (Bürk, Pfitzmann, 1990) proposes another way of realising anonymity and preventing the traceability of transactions by deploying digitally transferred currencies called anonymously transferable standard values. A standard value can be described as an anonymous account upon which standard values are deposited. These accounts are truly anonymous and while the bank manages them it cannot relate them to the true identities of their holders but only to their digital pseudonyms. A payment is realised only by transferring ownership of the standard value. The payee has to choose a new digital pseudonym and sign a declaration proving that he accepts this new anonymously transferable standard value which was previously owned by the payer. The drawback is, however, that users have to change their digital pseudonyms and modifications are needed in specific situations (where the payee of a transaction is the payer of another) in order to retain total unlinkability (Camenisch, L. J., Piveteau, J-M. and Stadler, M. A., 1994).

Camenisch et al. (Camenisch, L. J., Piveteau, J-M. and Stadler, M. A., 1994) proposed another more practical implementation of anonymous payments by reducing the needed size of bank databases significantly. The goal of the protocol is to conceal the customer's identity during the withdrawal phase and it tries to achieve that by introducing the concept of anonymous accounts. In particular, the bank will maintain two kinds of accounts: personal and anonymous. The personal account is a normal account that corresponds to an individual with a specific identifier, whereas the anonymous account is linked to someone whose true identity is not known to the bank. A customer of the bank is capable of managing his two accounts and transferring money back and forth by logging in the bank's on-line system. Another significant advantage is that the authors claim that this system can be realised as an add-on feature to the already existing banking system and there is no need for any new mechanisms.

Off-line schemes

Unlike on-line systems, off-line schemes do not require the active involvement of the bank in each transaction and, as a result, the TTP cannot spot and prevent fraud *on-the-fly*, but it must be capable of detecting it and identifying the party responsible. We should always bear in mind though that, if the protocol provides perfect unlinkability – anonymity, this could potentially be a hard task to accomplish. The difficulty of linking withdrawals to deposits gives room for criminal activity, such as blackmailing, money laundering and illegal purchases (B. von Solms and D. Naccache, 1992) and, as a result, the idea of revocable anonymity was adopted by many authors (J. Camenisch, M.Stadler, J.M. Piveteau, 1995), (J. Camenisch, U. Maurer, and M. Stadler, 1996), (M. Jakobsson and M. Yung, 1996), (Y. Frankel, Y. Tsiounis, and M. Young, 1996) (G. Davida, 1997).

Chaum, Fiat and Naor (D. Chaum, A. Fiat, and M. Naor, 1988) proposed an untraceable electronic coin scheme which uses the idea of one-time blind signatures and manages to identify "repeat-spenders" after the fact. By untraceable the authors mean that it should not be easier for someone to trace an electronic transaction back to the payer than it is in a physical transaction with conventional cash. The authors wanted, however, to find a way to trace a user after he forged or cloned a 'blindly signed coin' and used it more than once. In order to achieve that, the information of the accountholder and in particular redundancy based on his/her random values is encoded in the withdrawn information in order to spot any attempt of reusing the same coin. However, the customer's identity is not known by the bank as anonymity and untraceability is guaranteed by the use of blind signatures. A crucial drawback of this attempt lies in the practical costs which will be significant.

Chaum (Chaum D, Unpublished Manuscript, 1988) came up with a solution in order not only to identify but to prevent double-spending. That solution was also adopted by Brands (Brands S, 1993) in his anonymous off-line untraceable payment scheme. The basic concept is the introduction of e-*wallets* with *observers* where the observers are tamper-resistant devices placed into payment devices like smart cards, palmtops or PCs and these two combined form the *wallet*. The observers are necessary in order for a payment to be completed and, thus, they can achieve "prior restraint of double spending". Even if the tamper-resistant device breaks down the protocol in place will still identify the double-spender after the fact.

Another, successful, in terms of security, way of implementing off-line untraceable anonymous payment systems was proposed by Pfitzmann and Waidner (Pfitzmann B., Waidner M., 1992) who tried to optimise a protocol of David Chaum, Ivan Bjerre Damgård and Jeroen van de Graaf (David Chaum, Ivan Bjerre Damgård, Jeroen van de Graaf, 1988). The idea was to implement a two-party computation protocol for the withdrawal phase. In particular, the bank and the customer jointly generate the blind coin. The customer provides his/her private input, a random number R (the coin number), and the bank its secret key SK and they issue a signature on R. The customer then gives the coin number R to the merchant who forwards it to the bank along with his account number. Then the customer gives the bank a perfect zero-knowledge proof that he is in possession of the signature on R and the bank completes the transaction by crediting the merchant's account with the respective amount. Although secure, the system requires long messages and suffers from encryption complexity.

Summary

Our extensive research on off-line and on-line schemes shows that the former are more suitable for low-value transactions, as preventing fraud is difficult to be accomplished in real-time. The latter, although followed by the difficulty in applicability due to the need for large on-line databases, are capable of preventing users ("prior-restraint") from spending with insufficient funds and thus they are more suitable for high-value transactions.

iii. Security Issues

Security is the most important service in a payment-delivery system. There is no point in talking about other services or proposing anonymous or fair exchange protocols if there are security vulnerabilities undermining the rest of the protocols' objectives. For the majority of the protocols we examined in the previous chapters we accepted several assumptions about the infrastructure in place; for example that we are using a resilient, reliable or secure channel. In this chapter, we will discuss what the most famous potential security threats in network protocols are.

Security threats

Usually the security objectives of a secure channel are confidentiality, replay and integrity protection. In terms of payment protocols, these are the respective threats to these objectives.

Information disclosure: Exposure of sensitive information to unauthorised parties. It is closely related to confidentiality which ensures that data can only be viewed by the authorised parties. It is also sometimes referred to as secrecy. (M.Martin, 2012)

Impersonation: An active attacker can use the identity of one of the authorised users in order to participate in a transaction. Tackling this problem would require effective authentication-access control mechanisms.

Data replay: An active attacker can intercept messages on a network and then replay them in order to gain advantage. The protocol needs to have mechanisms to identify sessions and replay detection techniques in place in order to prevent the legitimate entities from rerunning the same protocol. Peterson and Davie mention in their book (Larry L. Peterson, Bruce S. Davie, 2007) that when a protocol detects replays, it provides *originality.*

Man-in-the-middle attacks: An active attacker intercepts messages in transit and alters them in any way.

Tampering: Tampering is the unauthorised modification of data by substitution, insertion, or deletion of messages. A protocol that detects tampering provides data integrity.

Denial-of-service: An active adversary intervenes in the traffic of the transaction in order to cause delay or even deprive the legitimate users from the actual service or the resources needed. "Ensuring a degree of access is called availability." (Larry L.

Peterson, Bruce S. Davie, 2007) According to the authors, a protocol that detects delays achieves *timeliness*.

Repudiation: Repudiation is the ability of users (legitimate or otherwise) to deny that they performed specific actions or transactions. In this case, a participant denies his/her involvement in the transferred messages of a protocol run after the completion of a transaction.

We can now identify the desirable security requirements for a reliable e-commerce system in our scope. The basic security requirements are authentication, confidentially, integrity, non-repudiation and availability. To these we, of course, add other 'business' requirements such as anonymity, fair exchange, simplicity, least involvement of TTPs, speed etc.

There are many implemented protocols that can realise secure e-commerce transactions and the most famous of these are SSL (Secure Socket layer), SET (Secure Electronic Transfer) and PGP (Pretty Good Privacy). The following table illustrates their effectiveness against some of the threats we discussed above.

Secure E-commerce protocols	Confidentiality	Non-repudiation	Integrity	Replay Attack	Man-in-the-middle attacks
SSL	Yes	No	Yes	No	No
SET	Yes	No	Yes	No	No
PGP	Yes	No	Yes	No	No

Figure 3 Security comparison of E-commerce protocols (Khalid Haseeb , Dr. Muhammad Arshad , Shoukat Ali , Dr. Shazia Yasin , Khalid Haseeb , Dr. Muhammad Arshad , Shoukat Ali , Dr. Shazia Yasin, 2011)

Chapter 3: Formal Analysis

In this chapter we aim to examine formal methods, in terms with their established process, their results and also their potential weaknesses.

Network protocols are used everywhere around us nowadays, participating in the majority of electronic activities by accomplishing trivial but also critical tasks. Flaws in payment protocols are, undoubtedly, undesirable and a deterring factor for the user. Formal methods were developed using exhaustive verification and mathematical notation in order to intensively test network protocols and identify potential weaknesses.

Most of these formal methods comprise three stages; Modelling, Specification and Validation. (Kremer, 2003-2004)

Modelling: It refers to the process of converting the protocol into a mathematical model.

Specification: It refers to the process of stating the properties which must hold on the given model.

Validation: We basically prove that the specification holds on the given model. Sometimes this process can be done as an automated process which is called model-checking. This is the approach we will use in our thesis.

i. Modelling and formal analyses of the selected protocol

For our purposes, we have chosen an on-line TTP based protocol which was proposed by Zhou and Gollmann (Zhou, Gollmann, 1996). In every protocol we can identify assumptions, flow, messages and actions. However, in order to define it, we have to complete three important stages. We have to define the protocol's objectives, goals and specify the protocol itself. The objective describes the main problem the protocol intends to solve and the goals are defined by translating the objective in clear requirements. Finally, by using the goals as input in order to identify the needed assumptions, flow, techniques and messages we are in position to specify our protocol. (M.Martin, 2012)

Objective: The Zhou-Gollmann protocol wants to identify participants that misbehave in a distributed communication environment and wants to prevent any users from obtaining an advantage over the others.

Goals: The respective goals which are derived from the objective are non-repudiation (of receipt and origin) and fairness. Their additional goal was to bring the TTP's amount of workload down to the minimum. In order to achieve that, they split the definition of the message into two parts, a commitment C and a symmetric key K chosen by the originator.

Figures 4 and 5, which are presented below, illustrate the flow and the notation of the protocol. The main idea is that the originator A sends the recipient B the message he wants encrypted with a key K, a commitment C, the signed non-repudiation of origin evidence (NRO) as well as a label L indicating the session. If B accepts A's message he/she responds with his signed non-repudiation of receipt evidence (NRR) and A sends the trusted third party a signed copy of the key. The TTP checks the validity of the signature and checks out, the server publishes the information which the two parties obtain with an *ftp get* operation (from a read-only public directory). In case of an incorrect or a not delivered message either one of the parties can end the session at any time during the protocol.

We assume that, even if the channel is not reliable, the parties will eventually receive their respective messages. The protocol can also offer fairness and timeliness and does not require the participants to “play fair”. The TTP works as a delivery agent and handles keys which are smaller than messages. This changes the security requirements of its deployment in comparison with the other third party-based protocols while the involvement of the TTP is still considerable. Finally, it should be noted that the protocol does not define any means of keeping the messages confidential. Even the key is transferred in the clear, free for anyone to observe.

The source of the following protocol, its notation and assumptions is (Zhou, Gollmann, 1996).

1. $A \rightarrow B$: f_{NRO}, B, L, T, C, NRO
2. $B \rightarrow A$: f_{NRR}, A, L, NRR
3. $A \rightarrow TTP$: f_{SUB}, B, L, T, K, sub_K
4. $B \leftrightarrow TTP$: f_{CON}, A, B, L, K, con_K
5. $A \leftrightarrow TTP$: f_{CON}, A, B, L, K, con_K

Figure 4 The Zhou-Gollmann protocol flow (Zhou, Gollmann, 1996)

Notation:

A: The originator of the message

B: The recipient of the message

TTP: Trusted Third Party

C: commitment (ciphertext) for message M (e.g. M encrypted under key K)

K: message key defined by A

L: Label used to identify a specific protocol run

$S_{A,}$ P_A: the public and private key pair of the originator A

sK(m): digital signature of a message m using signature key K

NRO= $sS_A(f_{NRO},$ B, L, C), Non-repudiation of origin for M

NRR= $sS_B(f_{NRR},$ A, L, C), Non-repudiation of receipt of M

sub_K= $sS_A(f_{SUB},$ B, L, K), proof of submission of K

con_K= $sS_T(f_{CON},$ A, B, L, K) confirmation of K issued by TTP

f_{NRO}; f_{NRR}; f_{SUB}; f_{CON}: message flags to indicate the purpose of the respective message.

Figure 5 Notation of the Zhou-Gollmann protocol (Zhou, Gollmann, 1996)

Assumptions:

- All users are equipped with their own signature key and the verification keys.
- B cannot block the message identified by f_{SUB} permanently, thus A, will eventually be able to obtain the evidence of receipt.
- The ftp communication channel is eventually available, thus, also B will eventually be able to obtain K and therefore m and con_K.
- TTP checks that A does not send two different keys with the same label L and the same agents' names. In other words, labels have to be unique, in oder to link commitments and keys effectively. This is necessary because L serves as a unique identifier for con_K. i.e. TTP will overwrite con_K with con_K0 which causes a problem if either A or B have not yet retrieved con_K.
- TTP stores message keys at least until A and B have received con_K.

Let us assume that, after the protocol run, A wants to prove that the message has been received by B. She sends NRR= $sS_B(f_{NRR}$, A, L, C) along with con_K= $sS_T(f_{CON}$, A, B, L, K) to the judge. The first part proves that the commitment was indeed received by B and the second that the key was uploaded and deposited by the TTP and that B has access to it. The label L in both parts proves that these two messages were exchanged during the same protocol run.

If B wants to prove that the message was sent he provides the judge with NRO= $sS_A(f_{NRO}$, B, L, C) and con_K= $sS_T(f_{CON}$, A, B, L, K). The first part proves that A sent him her commitment and the second that A sent the key to the TTP.

Formal Methods

The first attempt to formally verify the protocol was made by the authors who used the SVO logic (Zhou, J., Gollmann, D, 1998). Schneider in his analysis (Schneider, 1998) used the CSP algebra with the FDR model checker followed by Bella et al.. who used the Isabelle theorem prover (Bella, G., Paulson, L.C, 2001). Finally, a third one by Gürgens and Rudolph used the asynchronous product automata (APA) and the simple homomorphism verification tool (SHVT) (Gürgens, S., Rudolph, C., 2002). Their results were not the same and the discrepancies between them are caused by the assumptions each authors take. These assumptions are necessary in order to model the system and translate it into consistent mathematic notation. The modelling of the two main goals we described above is the most difficult and crucial element of each analysis.

- All the authors assume that perfect encryption is in place and that keys in use cannot be guessed be anyone. (Susan Pancho-Festin, Dieter Gollmann, 2005)

Zhou and Gollmann examined the goals from the judge's point of view and especially in regards to the dispute procedure. Only the non-repudiation goals were modelled because of inefficiencies in the SVO belief logic and the output was:

G1 Judge J believes (A said M).

G2 Judge J believes (B received M).

Furthermore, the authors assumed that the judge is in possession of the public keys of A, B and the TTP as well as of all the evidence provided.

Schneider in his analysis adopts a different approach. He examines the fairness requirement form the participants' point of view and the non-repudiation goal from the judge's point of view as correctness of evidence. The judge verifies the validity of origin or receipt claims, without having observed the protocol run. Each participant is modelled in terms of the messages he can receive, send or download from the TTP.

Non-repudiation in the protocol is composed of evidence of receipt and origin (EOR, EOO).

Non-repudiation of Origin:

B possesses NRO = $sS_A(f_{NRO}, B, L, C)$

B possesses con_K = $sS_T(f_{CON}, A, B, L, K)$

It is assumed that if B possesses these messages then A is the one that sent $sS_A(f_{NRO}, B, L, C)$ and $sS_A(f_{SUB}, B, L, K)$ to him.

Non-repudiation of Receipt:

A possesses NRR = $sS_B(f_{NRR}, A, L, C)$

A possesses con_K = $sS_T(f_{CON}, A, B, L, K)$

It is assumed that if A possesses these messages then B is the one that sent $sS_B(f_{NRR}, A, L, C)$ and is also capable of obtaining the key K from the TTP.

A protocol is unfair if one party can acquire the evidence he requires before the other party is able to. According to the author, fairness can be expected by a party only if the protocol is followed properly.

Fairness for the originator A:

If B is able to provide message M and proof of origin from A then A must be able to provide proof of receipt by B. This also means that B cannot obtain the key K until it is provided by the TTP. In other words, no participants know the key but A, and A sends the key to the TTP only once, so that B can only obtain it after TTP has made it available to both of them.

Fairness for the recipient B:

If A has proof of receipt, then B must be capable of presenting proof of origin.

The author modelled the network between the participants and he called it the medium. He made the assumption that the medium is unreliable but the messages in transit are always detected if altered and then disposed of. As a result, no modified messages can actually be delivered by any of the parties. The second restriction is that messages on this medium cannot be misdelivered. Without imposing this restriction, Schneider realised that fairness cannot be guaranteed for A. In particular, if she sends message 3 to B instead of the TTP then B is in an advantageous position. He is in possession of the key and, as a result, of the message itself without the TTP's involvement and without A receiving any evidence. Finally, in his analysis, he assumed that the agents can perform any malicious actions and the only restriction he imposed in their behavior, is that they do not reveal their signing keys.

Bella et al. used a similar way with Schneider, of expressing the respective goals. However, the correctness of evidence and fairness requirements were specified in regards to the guarantees that each party expects from the protocol.

Guarantees for A

Validity of evidence: If A has obtained con_K that means that A has previously lodged key K for a specific label L with the TTP and that B can also obtain it via ftp get. Moreover, if A holds the NRR proof, then B must have accepted the commitment C bound to the same label L and he should be able to obtain the message M.

Fairness: When con_K is known to B, then either A has obtained it also, or it is made available to her.

Guarantees for B

Validity of evidence: If B has obtained con_K then A must have submitted K bound to label L to the TTP. If B has in his possession the NRO evidence, then A must have already sent her commitment C with the same label L.

Fairness: If A holds con_K then B is able of obtaining too.

The authors, again, assumed that although the channel can be unreliable, the messages cannot be altered in transit. However, the only assumption they made in terms of malicious behaviour of the protocol agents is that they can abandon protocol sessions before termination. Gürgens and Rudolph argue that this is a rather unrealistic and limited assumption to make about a non-repudiation protocol and it is also the reason why they did not discover any attacks.

Gürgens and Rudolph translated the protocol's goals into the predicates NRR(B) and NRO(A) that must hold true for the participants.

For A : Predicate NRR(B) states that if B has valid NRO and con_K then A should have valid NRR and access to con_K.

For B: Predicate NRO(A) states that if A has valid NRR and con_K for a message M, then B must have valid NRO and access to con_K.

The most important assumption of the authors is that neither con_K nor the keys of a specific protocol run remain available at the TTP indefinitely. Zhou and Gollmann expressed the need for deletion of keys at the TTP after a specific amount of time and extended their protocol using timestamps (deadline for storage of evidence at the TTP) which are sent by A to B along with the commitment and B can choose whether he accepts the timestamp or not. However, timestamps are not included in the original version of their protocol. This assumption made the following attack possible:

Attack:

Let us assume that after A and B have completed a normal protocol run, A has obtained NRR from B and con_K from the TTP and that the TTP has deleted con_K. We also assume that A has stored K and con_K for a specific L. Then A starts a new protocol run choosing the same key and the same label. This is possible as the original Zhou-Gollmann protocol restricts only the use of two different keys with the same label. A, however, sends now a different commitment $C=\{M_2\}_K$ and receives NRR2 from B. A is now able of presenting this evidence of receipt together with con_K from the first protocol run. B may have kept the same information that A stored from their previous protocol run but this is a rather unrealistic assumption. Even if he has stored this information, it is unlikely that he will start trying old keys on the encrypted message.

ii. Results

None of the first three attempts to verify this protocol identified the aforementioned attack. In Schneider's analysis, it is assumed that the keys and con_K remain at the TTP forever, available for download. If A tries to use the same data used in a previous session, there will be duplicate entries in the TTP's database and the server will detect and prevent it. Bella et al. made the same assumption and as a result, they also did not detect the attack. Zhou and Gollman's SVO logic analysis was focused on determining the beliefs that are derived by A and B and not on finding a possible attack or a flaw in the protocol. However, it could be claimed that Gürgens and Rudolph modified the original protocol by assuming that the evidence is deleted by the TTP *as soon as* it is obtained by A and B. In the original paper of Zhou-Gollmann, it is stated that it is undesirable for the evidence to be held by the TTP forever and thus they further extended the protocol by using timestamps. The authors' 'original' assumption is that the evidence will be held *at least* until A and B have retrieved it.

To sum up, it is clear that in order to formally analyse a protocol or to compare different analyses we first have to thoroughly examine the assumptions and the modelling methods of their respective authors. Differences in the goals of the protocol or the analyses itself as well as differences between the point of view, the expression of the network, the cryptographic primitives or the actions of the participants can cause discrepancies in the results and findings (validation stage).

iii. Limitations of formal methods

We can deduce that formal methods tend to simplify the functions and the events that can take place in a protocol. Most of the tools and languages (like CSP) work only with models that are finite and small. Hence, all models expressed in mathematical notation tend to be only approximations to reality for reasons of efficiency, consistency and simplicity. For example, Gürgens and Rudolph examined the protocol based on the belief that A will misbehave while B and the TTP remain honest throughout the protocol run. Bella et al. assumed that an intruder (spy capable of faking messages) can participate in the protocol, but they did not consider A, B or the TTP as an intruder. One should always take into account that the original protocol does not require A or B to behave honestly and analyse the protocol on that basis. An objective analysis has to rely on the actual security context, the goals and the assumptions of the original protocol, and obviously, these are difficult to incorporate in a formal verification.

Chapter 4: Improvements and Suggestions

i. A protocol for Exchange of Digital Product

We are going to present a modified version of the Netbill protocol (B. Cox, D. Tygar, and M. Sirbu, 1995) for the exchange of generic digital products, such as journals, music, contract signing, etc. Additionally, we develop a more detailed description and conduct a more comprehensive informal analysis than the one presented in the original paper.

Objective: The protocol must enable only authorised users to participate in low-priced transactions after negotiating and agreeing on the terms of the transaction. Both users must receive their expected goods intact, or else they must be able to prove any misbehavior to a judge and solve any disputes efficiently. Furthermore, we want them to be able to conceal their identity and the scheme must provide them proof of their involvement and commitment to the transaction.

Goals: The respective scheme's goals are: authentication, non-repudiation, fair exchange, privacy protection, integrity of messages.

As we discussed earlier, the protocol is composed of three phases, *price negotiation*, *goods delivery* and *payment* phase. Before the start of the negotiation phase, the users can choose not to reveal their true identities to each other and use a pseudonym-unique identifier. In particular, upon registration, the users receive a unique User ID (identifier) associated with an RSA public key pair which is linked to this identifier. This key pair is certified by the Netbill Server.

However, the Netbill server also uses a modified version of Kerberos which uses public key cryptography. Kerberos is a TTP based authentication protocol based on Needham-Schroeder. It provides single sign-on services meaning that a user can authenticate once to a server, who in turn, gives him limited access to other service providers, in a transparent way (silently handled). Thus, the user's credentials are not exposed to several authentication events and this approach is more user-friendly. The basic flow of the protocol is as follows: The client (C) authenticates himself to an authentication server (AS) and obtains a Ticket Granting Ticket (TGT). The client forwards this ticket to the Ticket Granting Server (TGS) and, if the client is authenticated, the TGS issues a service granting ticket (SGT) which is used by the client to access the respective service. All communications are encrypted with a symmetric cryptography already established between the entities and each ticket issued has a validity period. There are, however, both practical and security issues about Kerberos. The practical issues can be identified easily.

To begin with, there is no ticket revocation mechanism in place; the tickets have a fixed time. We should bear in mind that Netbill works also as a Certification authority who issues keys to unique identities. As a result, it is unrealistic to assume that, even if Netbill revokes a certificate, it can also effectively prevent the respective user from being granted a ticket. Second, there must be a relationship of trust in order to “bootstrap” long-term keys and, finally, there is a very crucial concern in regards to the use of timestamps and as a result of synchronised clocks in the asynchronous nature of the Internet. It is imperative that the designers efficiently account for clock drifts or time required for message deliveries. Furthermore, we need to establish synchronised clocks between the participants, an, often, unpleasant requirement in terms of practicality of the needed framework and cost.

The security vulnerabilities of Kerberos, which mostly have to do with password guessing attacks, are illustrated in the paper of Thomas Wu (Wu, 1999) and render obvious the fact that Kerberos is an identity management scheme not suitable for the security requirements of a payment system. For all these drawbacks we will modify the protocol in order to avoid any use of Kerberos and timestamps.

We will also propose a way to achieve a stronger sense of fairness for the customer by introducing a way to make sure that the goods sent by the merchant are the ones expected by the customer.

Notation:

C: customer

M: Merchant

N: Netbill Server

K_{CM}: symmetric key shared between C and M

$\{message\}_K$: message encrypted by a symmetric cipher using a key K

$Sign_{Ki}$(message): RSA based Signature on a message issued by entity i using the singing key K_i

P_i, S_i: Public and Private RSA keys owned by the entity i

h(message): A cryptographic checksum of message using an algorithm such as the Secure Hash Algorithm (SHA-256, SHA-512)

TID: Transaction ID is used in order to indicate to the merchant that this is a repeated request. Messages 1 and 2 may be repeated until the users agree on the terms. It is not globally unique. It is used between the customer and the merchant to maintain context.

PRD: Product Request Data: arbitrary stream of application specific data which the customer and the merchant use to specify the goods

RequestFlags: Customer's indication of his request for the disposition of the transaction (delivery instructions)

Bid: Indicates the customer's offer for the merchant's product

EPOID (Electronic payment order ID): a globally unique identifier to prevent replay attacks and which will be used in the NetBill server's database to uniquely identify this transaction.

N_C: stands for nonces (number used once), used by the customer to ensure fairness of message 1 and especially of the key used. The seed for the generation of the session key must include the nonce in the process

ProductID: a human readable description of the product sent by the merchant

Credentials: field that establishes the customer's membership in groups which may make him eligible for a discount

The participants

The “protagonists” of our protocol are: the merchant (M), the customer (C), the trusted third party (Netbill Server) and, optionally, the product verification entity (authority).

Assumptions

Assumption A: We assume that the users use their unique identities which also work as pseudonyms.

Assumption B: Every participating entity holds two pairs of public-private RSA keys, one used for encryption and the other one for signing. I.e. $P_A = (e_A, n_A)$ and $S_A= (d_A, n_A)$, the public-private RSA key pair used for encryption by A, whereas K_A and V_A the pair used for signing and verifying purposes. Thus, by issuing different cryptographic keys for different purposes, we ensure key separation, one of the most important requirements for a successful key management scheme. The customer’s and the merchant’s RSA pairs are certified by the Netbill server and both participants can acquire each other’s public key certificate (public key linked to a unique identity) by contacting the Netbill server.

Assumption C: We assume that the private keys of the players are stored in a secure way and that no other party can impersonate them by obtaining and using their private keys.

Assumption D: Due to their relationships of trust the customer and the merchant both share symmetric keys with the Netbill server. (K_{CN}, K_{MN} respectively).

An overview of the protocol

Figure 6 depicts a high-level exchange of messages between the participants. The customer and the merchant agree upon the price of the product and the customer sends a request message for this product to the merchant. The merchant provides the customer with the product, encrypted with a symmetric key K he generated for this session. Upon receipt, the customer generates a signed purchase request, which can be processed only by the Netbill server, and sends it to the merchant. In particular, the customer at this step submits a payment to the merchant in the form of a signed Electronic Payment Order (EPO) which includes the information of this specific transaction and encrypted instructions for the transaction readable only by the server. If the merchant endorses this order, he forwards it to the Netbill server along with his decryption key. In fact, this is the “point of no return” for the merchant who is, henceforth, not able to abort the protocol. The server verifies all the details of the transaction, transaction id, validity of signatures, account balance and, if everything is valid, it debits the customer’s account and credits the merchant’s account respectively and sends a signed encrypted (readable only by the customer) receipt along with the decryption key to the merchant. The merchant forwards the message to the customer and the transaction is complete. In case the merchant cannot - or chooses not to send the message - the server forwards it directly to the customer.

Protocol flow

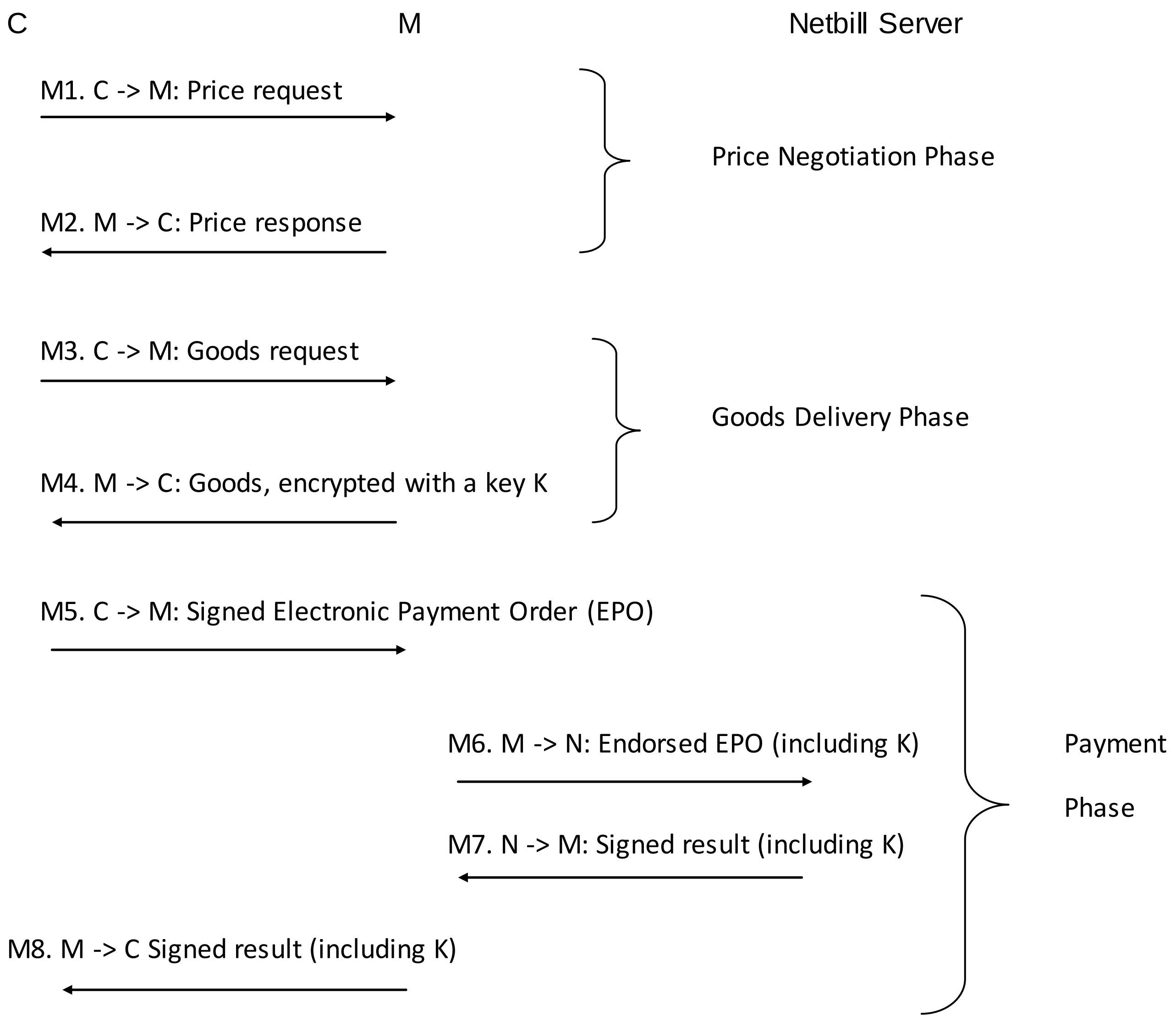

C
M
Netbill Server
M1. C -> M: Price request
M2. M -> C: Price response
Price Negotiation Phase
M3. C -> M: Goods request
M4. M -> C: Goods, encrypted with a key K
Goods Delivery Phase
M5. C -> M: Signed Electronic Payment Order (EPO)
M6. M -> N: Endorsed EPO (including K)
M7. N -> M: Signed result (including K)
M8. M -> C Signed result (including K)
Payment
Phase

Figure 6 Protocol Flow

Detailed description and analysis

The Price Request Phase

The objective of this phase is to allow the users to negotiate and, subsequently, agree on the terms of the transaction (price, delivery). This process can be repeated many times until they reach an agreement.

1. C-> M: $Sign_{KC}(h(Identity, K_{CM} N_C))$, $\{K_{CM}, N_{C,} Identity\}_{PM}$ {PRD, Credentials, Bid, Request Flags, TID}$_{KCM}$

The customer requests the merchant’s certificate from the server as soon as he/she has chosen a product. He is then, able to send the merchant a nonce, his identity and a session symmetric key, all encrypted with the merchant’s public key so that **confidentiality** is maintained.

Moreover, instead of using the Kerberos framework to achieve the authentication of C to M, we use digital signatures. The customer signs the checksum of this information with his signing key and, therefore, **data origin authentication** of the sender is achieved (assurance about the **integrity** of the message, and of the identity of the signer). Data origin authentication is a stronger version of data integrity. By using a nonce in this message we actually combine data origin authentication with freshness and, therefore we provide **entity authentication** of the customer, meaning that the merchant can be sure that he is currently active in the transaction.

Although this message provides non-repudiation because of the signature included, we do not require this service in the negotiation phase. This phase can be repeated many times until the participants reach an agreement.

By obtaining the certificate from the Netbill server and verifying the signature the merchant can be sure that he is communicating with the customer named “Identity”, who is also a legitimate user of Netbill. Moreover, he gains assurance that the message is fresh due to the nonce we used and that the symmetric key K_{CM} generated by C is also “fresh”.

Finally, the original protocol does not provide any integrity check for the second part of the first message. We could include a hash value so that the merchant will know before even decrypting the message that data have been altered in transmission: h({PRD, Credentials, Bid, Request Flags, TID$\}_{KCM}$). Otherwise, the users might start negotiating while having different products, prices or delivery methods in mind.

2. M-> C: $\{$ProductID, Price, Request-Flags, N_C,TID$\}_{KCM}$,

Upon receipt of message 1, the merchant decrypts $\{K_{CM}$, $N_{C,}$ Identity$\}_{PM}$ with his private key S_M and obtains the key, the nonce and the alleged identity of the customer. He/she is now able to verify the signature and also the integrity of the data he has obtained. Subsequently, he sends message 2 to the customer encrypted with their session key and if he, in turn, agrees with the terms they proceed to the goods delivery phase. (**Confidentiality**)

Again the original protocol does not include any **integrity** check for message 2. Data can be modified deliberately or unintentionally causing problems to the participants' communication, stalling their negotiation. Hence, it would be logical to add a form of redundancy: h($\{$ProductID, Price, Request-Flags, N_A,TID$\}_{KCM}$)

At this point, the customer can be sure that this message has been sent by the merchant as he should be the only one capable of decrypting $\{K_{CM}$, $N_{C,}$ Identity$\}_{PM}$ and, as a result, obtaining the session key K_{CM} and the nonce N_C, which provides linkability to the first message. (**Authentication**)

The Goods Delivery Phase

After having reached a consensus on the terms, the customer and the merchant are now ready to deliver the promised goods to each other.

3. C-> M: $Sign_{KC}$(h(Identity, TID, N_C)), $\{$TID, $N_C\}_{KCM}$

We include the nonce in this message so that it is difficult for someone to impersonate the customer by replaying a previously intercepted message (in a previous session). Besides, the TID, as defined by the authors, does not have to be unique.

The merchant verifies the validity of the signature in message 3 after re-computing the hash and, therefore, gains assurance regarding the origin and the integrity of the message (**Authentication, Integrity**). He can be sure that the message indeed comes from the customer with this identity and he can also verify that this message is linked to the 1st message by matching the values of the two nonces. The TID with the nonce, which together are session-unique, indicate that the merchant must send the goods to the customer in the way they agreed in the price negotiation phase.

Confidentiality: The nonce and the transaction ID are transferred encrypted so that disclosure of information about the session is prevented.

4. M-> C: $\{Goods\}_K$, $\{h(\{Goods\}_K), EPOID\}_{KCM}$

The merchant sends the product encrypted with a session key K he generated for this session. He also sends a checksum of the product encrypted with their symmetric key so that the **integrity** of the sent message can be verified by the customer. Finally, a unique electronic payment order id (EPOID) is sent along so that the customer will be protected from any replay attacks by dishonest merchants who might attempt to reuse the customer's old signed payment instructions.

Both parts of the message are encrypted so no information is disclosed to unauthorised parties (**Confidentiality**).

Authentication: Although the customer assumes that this message is indeed sent by the merchant as he is in possession of the symmetric key K_{CM}, he has no way of finding out if this is a malicious party who obtained the key and sends worthless data to the customer. This problem is tackled with the solution we propose later, by establishing a trusted entity which provides assurance about the goods sent. In general, the original protocol does not offer any help or protection to the customer in case of a dishonest advertisement claim.

It should be mentioned that, although the customer receives the goods in this step, the product is encrypted so he is not able to use it without knowledge of the symmetric key K.

The payment phase

The customer has received the encrypted goods and he must now submit the payment to the merchant in the form of a signed Electronic Payment Order (EPO). At any time until this point, the customer can abort the protocol without any risk of completion of the transaction in his/her absence. This is, in other words, the point of no return for the customer.

5. C-> M: $\{Sign_{KC}(EPO, N'_C)\}_{KCM}$, $\{EPO, N'_C\}_{KCM}$

The EPO is a tuple:

Identity, ProductID, Price, M, $h(\{Goods\}_K)$, h(PRD), h(Cacc, AccVN), EPOID, $\{Sign_{KC}(TrueIdentity_c)\}_{KCN}$, $\{Authorisation, CAcc, AcctVN, CMemo\}_{KCN}$

- The pseudonymous Identity, the ProductID, the negotiated Price, the checksum of the encrypted goods h($\{Goods\}_K$), the checksum of the Product Request Data h(PRD), the checksum of the customer's account number with an account verification nonce h(Cacc, AccVN) and finally the globally unique EPOID comprise the clear part of the EPO which is readable by the merchant. The AccVN used here is a pseudorandom number that ensures that the merchants cannot either determine whether different customers use the same account or guess the customer's account. In a way the nonce works like a blinding factor.

- The customer's true identity, encrypted with the symmetric key he shares with the Netbill server $\{TrueIdentity\}_{KCN}$ and again encrypted information about any required authorisation tokens, the customer's account, the verification nonce and a memo field $\{Authorisation, CAcc, AcctVN, CMemo\}_{KCN}$ comprise the encrypted part of the EPO which is only readable by the server.

At this stage we included another nonce generated by the customer and sent along the EPO encrypted under the symmetric key of the participants. The customer's signature on this information, encrypted with the same key as before, provides **confidentiality, data origin authentication of the sender** and **non-repudiation of origin**. Although, usually, the players sign the checksums of messages, it would be wiser in this case and in message 6 to sign the actual message. For legal reasons and in case of any disputes, we prefer the signature to be bound explicitly to the signed data (not an already processed version of the actual data).

Upon receipt, the merchant verifies the signature and reviews all the fields in the first part of the EPO. If the signature is valid and the EPO is formed according to their agreement, he endorses the EPO and forwards it to the NetBill server. The endorsed EPO adds the merchant's account number, the merchant's memo field and the goods decryption key, all signed by the merchant. This is the point of no return for the merchant.

6. M-> N: $\{Sign_M(Sign_{KC}(EPO, N'_C), EPO, N'_C\ Macc, MMemo, K)), Sign_{KC}(EPO, N'_C), EPO, N'_C\ Macc, MMemo, K\}_{KMN}$

The original protocol does not indicate that the merchant can also choose to conceal his identity. However, this protocol could be applied to situations where two users (also M) want to participate in a similar transaction using only their identifiers and not their true identities.

In order to conceal his identity the merchant simply has to send the information about his true identity and his account encrypted and then sign the message with the signing key bound to his identifier. Hence, message 6 becomes:

M-> N: $\{Sign_{KM}(Sign_{KC}(EPO, N'_C), EPO, N'_C$ Macc, MMemo, K, $TrueIdentity_M)), Sign_{KC}(EPO, N'_C), EPO, N'_C$, Macc, MMemo, K, $TrueIdentity_M\}_{KMN}$

With the use of signatures and by encrypting the message with the symmetric key $K_{MN,}$ shared between the merchant and the netbill server, this message offers **data origin authentication**, **non-repudiation** for both players (signed EPO, endorsed EPO) and of course, **confidentiality**.

The Netbill Server is now able to make a decision about the transaction. Then, the result is returned to the merchant who, in turn, forwards it to the customer.

The decision is based on the verification of the signatures, the integrity check on all the communicated data, the privileges of the participants, the account balance of the customer and the uniqueness and freshness of the EPOID. Subsequently, it issues a receipt containing the result, the identities of the participants, the price and the description of the goods, the EPOID and the key K needed to decrypt the goods. The receipt is signed by the server and is denoted:

Result, Identity, Price, ProductID, M, K, EPOID

The receipt is returned to the merchant followed by an indication of the customer's new account balance encrypted. The EPOID is included here so that this receipt is linked to this transaction and therefore to prevent the merchant from replaying data gathered in previous sessions.

7. N-> M: $\{Sign_{KN}(Receipt)\}_{KMN}$, $\{EPOID, Cacc, Balance, Flags\}_{KCN}$

The signature sent here is a RSA digital signature with message recovery, meaning that the merchant can obtain the message by running the verification algorithm with the correct verification key and by, subsequently, processing the resulting value. This enables the server to send the signature alone (without appending the actual message) and, as a result, to avoid message expansion and performance issues. The requirement for such a signature is that the data to be signed must be less than one RSA block (RSA modulus e.g. 2432, 3248) in length. Refer to (M.Martin, 2012) for more information on digital signatures.

Confidentiality: Both parts of the message encrypted.

Data origin authentication: Achieved with the use of the signature.

Integrity: Although the integrity of the receipt is guaranteed by the use of signature, we cannot claim the same about the second part of the message. A checksum or the server's signature on (EPOID, Cacc, Balance, Flags) could be included so that the customer can be certain about its validity upon receipt.

The Flags value is used to communicate small messages from the server to the customer. For instance, it can be used to let him know that his account balance has reached a low level and it should be topped-up soon.

Finally the merchant forwards the necessary information to the customer.

8. M-> C: $\{Sign_{KN}(Receipt)\}_{KCM}$, $\{EPOID, Cacc, Balance, Flags\}_{KCN}$

Again, our usual security goals are all met by the use of the server's signature and encryption of all data.

Communications failure

In case of communications failure after message 5, the customer or the merchant may have no knowledge of the outcome of the transaction. Before this message both participants can abort the protocol as they are not yet committed to the transaction. The authors have tackled this problem with the introduction of a status query exchange phase between the participants. The request includes the EPOID which identifies the session uniquely and the identity of the interested party, both encrypted. The response should include the respective receipt and the second, readable-only-by-the-customer part of message 8. This phase gives the opportunity to the customer to obtain the decryption key K even if the merchant decides not to forward message 8 to him.

Dispute Session

- **Customer**

The original protocol is able to settle complaints and disputes. Many problems can arise for the customer. For example, the product sent may be different than the one specified, the product may arrive damaged-incomplete or the key given by the merchant may be wrong. All these disputes can be settled by an arbitrator who will obtain the registered copies of the customer's signed EPO (which contains also a checksum of the encrypted goods) and the merchant's signed endorsement indicating his agreement with that checksum and attesting to the decryption key. The judge compares these values against the copy of the encrypted goods and the decryption key included in the customer’s complaint. The arbitrator can determine whether the problem is the fault of the merchant or the customer.

Furthermore, the customer may claim that his account was not debited for the expected amount or that, even if he did not agree to the terms of the transaction, the payment went through anyway. The customer’s signed EPO can, again, help the server reach a verdict concerning these claims (an EPO signed by the customer implies the customer’s approval of a payment for a specific price, product etc.). If, however, the Netbill server cannot provide the signed EPOs the customer’s money must be refunded. This process protects the customer from potential fraud by the server.

Enhancing fairness for the customer

We could introduce another trusted entity, called “Authority”, who makes sure that the products advertised match their respective description before being encrypted. Therefore, it could issue a certificate to the merchant for his product.

$\text{Certificate}_{\text{ProductID}}$= ProductID, $\{\text{Goods}\}_K$, N, Sign_{KA}(h(ProductID, $\{\text{Goods}\}_K$, Serial))

Serial denotes: a unique serial number for this specific product

This certificate can be sent to the customer by the merchant in the fourth message.

4. M-> C: $\text{Certificate}_{\text{ProductID}}$, $\{h(\text{Certificate}_{\text{ProductID}})\}_{KCM}$, EPOID

Hence, the customer will have some sense of assurance that the merchant is certified for his services and that the encrypted product he received is the one he should be paying for. Having verified the certificate, he is now confident enough to continue with his payment order.

- **Merchant**

The merchant’s complaints will focus on his payment status. In case his account is credited for less than the amount expected or it is not credited at all after a successful transaction, the merchant can file a dispute. As we discussed earlier, the Netbill server is handling the transactions and the accounts of the players. Hence, if the merchant presents the signed receipt he obtained in message 7, it is very easy for the server to determine the error and take the necessary action. For instance, if the receipt indicates that the transaction linked to a specific EPOID was successful, the server will debit the merchant's account after checking his balance and making sure that no payment has already taken place.

Summary

- **Authentication:** Entity and data origin authentication are achieved by the use of digital signatures, public-private keys as well as freshness indicators added to the protocol.

- **Confidentiality:** Confidentiality requirements are achieved by the use of symmetric and public-key cryptography.

- **Integrity:** The participants gain assurance about the integrity of the communicated messages due to the use of digital signatures and hash values which are transferred encrypted.

- **Non-repudiation:** Provided by the strength of digital signatures. Especially messages 6 and 7 are of significant importance in terms of this service.

- **Fair exchange:** We showed that at the end of the protocol both players will obtain their respective items either by completing the payment process normally or by resolving a dispute afterwards. By using an on-line TTP (Netbill server) and the product verifying entity (Authority) we can achieve stronger fairness meaning that some problems can be solved on-the-fly.

- **Anonymity:** As mentioned in **Definition 6**, true anonymity is hard to accomplish. However, both players can maintain a degree of privacy by using pseudonyms which cannot be linked to their true identities (except for the server who is able to do so).

ii. Identity Management

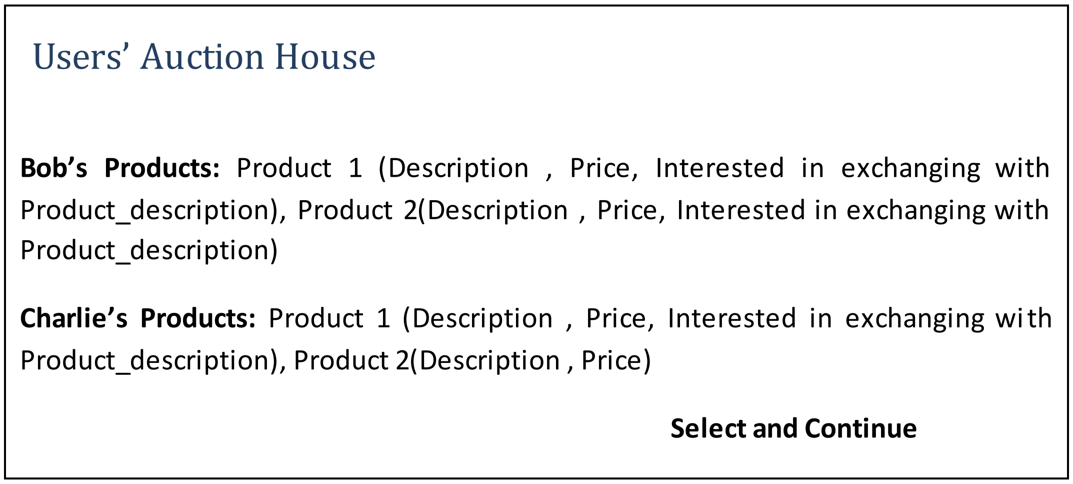
In this final section we are going to propose a way to implement our protocol in the context of social networking services. Our motivation derives from the popularity of these networks and the practical benefits they can offer. A recent study from ITU (International Telecommunication Union) (International Telecommunication Union,

2012) shows that the total number of users actively using social networks in 2012 was just over a billion. The variety of social networks has equally scaled, with the most famous providers being Facebook, Twitter, Google+, LinkedIn, MySpace but also with the development of many others, like The Sphere, Nexopia, Bebo, VKontakte, Hi5, Hyves, Badoo, Skyrock, Mixi, Orkut, Tuenti (the list goes on), whose use is heavily localised. Facebook, in particular, has 900 million active users and, of course, there are overlaps in terms of the users' involvement with other networks. Moreover, more and more users access these services via mobile everyday making these communication mediums rather ubiquitous.

Trust Issues: Our suggestion is based only on the fact that the majority of Internet users are members of these networks and this could help us develop a user-friendly, efficient, widely accepted and ubiquitous payment scheme. However, the assumption that the social network is here the trusted third party contradicts reality. Social networks tend to re-use personal information of their users for advertisement and commercial purposes and one could argue that this is in fact the core of their business framework. Social networks, as we know them today, do not support the requirements of a secure, fair, anonymous payment system and do not inspire trust. Nevertheless, with their wide acceptance and popularity we cannot neglect the fact that they have the potential to offer promising solutions on this sector.

Anonymity and Authorisation: In order to enable the users to maintain their pseudonymity we take advantage of the Identity Management standards used by social networks such as OAuth which is used by Facebook, openID etc. A similar method is being used by e-Commerce businesses that facilitate payments such as Google CheckOut, Amazon Payments, Bitcoin, Paypal and others. Their basic contribution is that it allows a service provider (SP) to gain controlled access to a user's information which is stored at the Identity Provider (IdP, the social network). For example, a user registered on Facebook can access other services provided by SPs without having to enter their credentials again or create new ones for each service. The interactions between the SP and the IdP are managed with http redirects.

Let us examine the following example which is similar to the one presented at (OAuth Protocol Workflow, 2011). We assume that Alice and Bob are users on a social network called Party and there is an "electronic auction house" named User's Auction house where they can advertise their respective products. Let us assume now that Alice finds something interesting that belongs to Bob.


Users’ Auction House

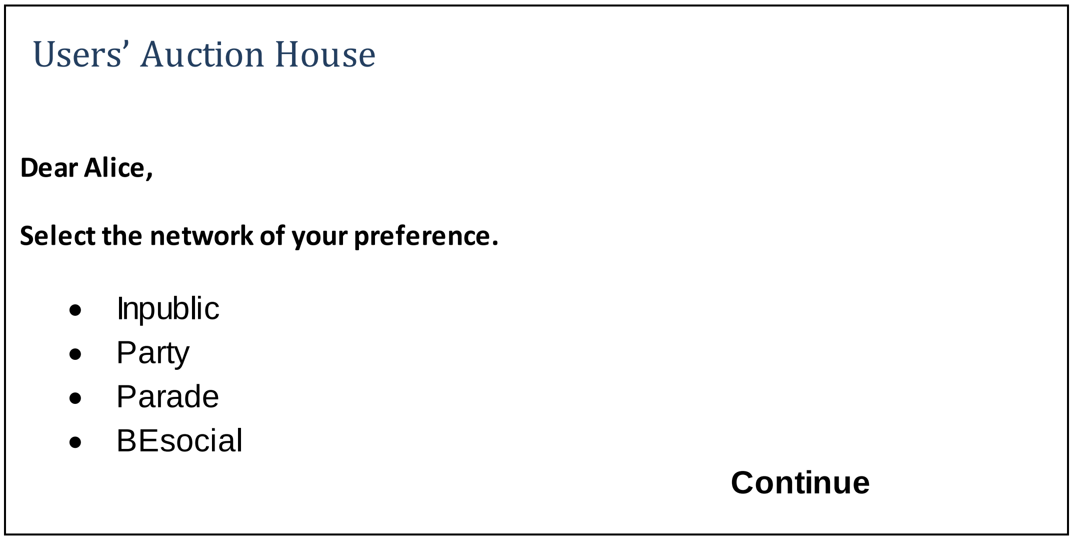
Bob’s Products: Product 1 (Description , Price, Interested in exchanging with Product_description), Product 2(Description , Price, Interested in exchanging with Product_description)

Charlie’s Products: Product 1 (Description , Price, Interested in exchanging with Product_description), Product 2(Description , Price)

Select and Continue

Figure 7 Product selection

Alice selects her product of interest and chooses to proceed with the exchange phase. Then the service provider controlling the auction house will ask Alice which social network service she is registered with and will redirect her to the respective webpage for her authentication phase. The options she is given can be the networks that the merchant is already registered with.


Users’ Auction House

Dear Alice,

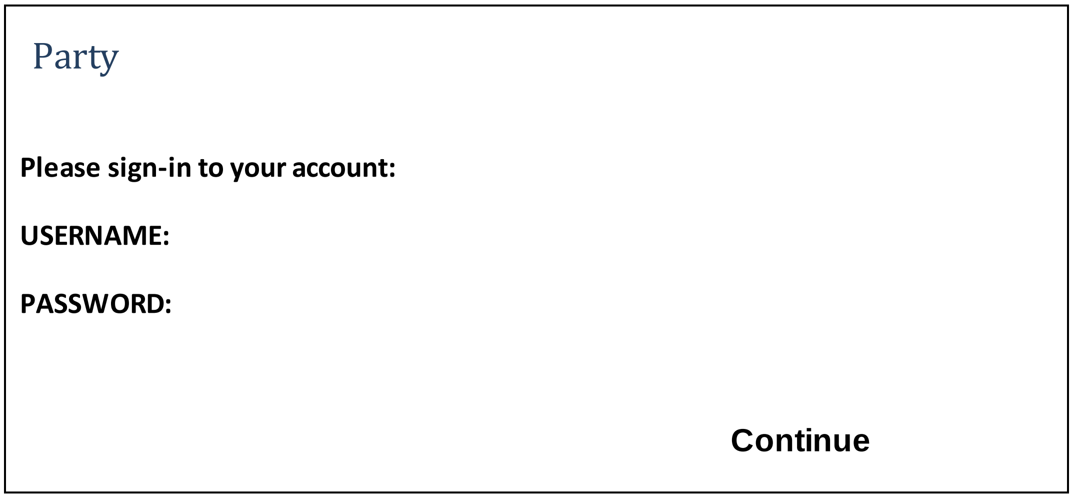
Select the network of your preference.

- Inpublic
- Party
- Parade
- BEsocial

Continue

Figure 8 IdP selection

Alice chooses Party as her choice of preference and the moment Alice clicks on the Continue button, User’s Auction House requests from Party a set of temporary credentials. The SP usually makes an API call to the IdP providing details about the transaction (product_ID, user, price). Alice is redirected to Party OAuth User Authorisation URL with the temporary credentials and the SP asks the social network to redirect Alice back to the auction house page as soon as she is authenticated. Thus we can identify two ways of communication for the service provider, one through the front channel with the resource owner and one through a back channel used by the SP to contact the IdP.



Party

Please sign-in to your account:

USERNAME:

PASSWORD:

Continue

Figure 9 Authentication process

Provided that Alice gets authenticated by Party, she is asked to grant access to User's Auction House which is the client. It should be mentioned that, at no time does Alice enter her credentials into the service provider or reveals her true identity for that matter. Subsequently, Party informs Alice of who is requesting access and the type of access, the privileges being granted, the amount of time etc. Now, we assume that Alice can choose not to disclose her true identity and that she can also use an identifier bound to her account and explicitly used for her transactions like a pseudonym. After Alice has accepted the terms, Party marks the temporary credentials as approved by Alice and redirects her back to the service provider with this Authorised Request Token. User's Auction house can now trade the request token for an authorised token in order to charge Alice's account which is linked to her transactional identifier. However, the transaction is actually handled by the IdP.

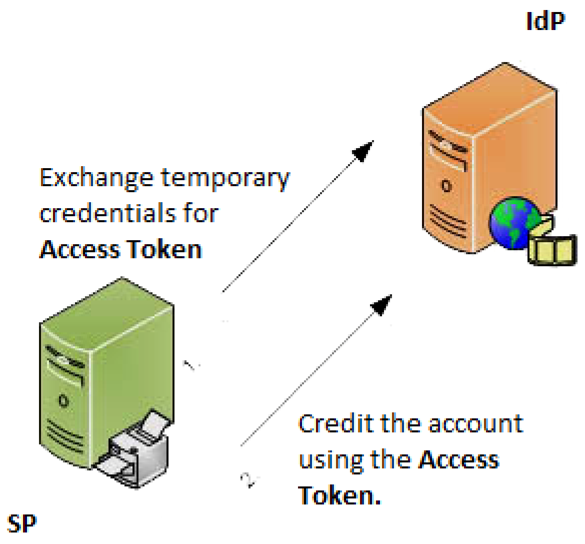

IdP
Exchange temporary credentials for **Access Token**
Credit the account using the **Access Token.**
SP

Figure 10 Data exchange between SP and IdP (OAuth Protocol Workflow, 2011)

Many social networks already enable users to register by choosing their payment method. If the users belong to the same social network, it is possible for them to exchange electronic provider-specific currency by using our protocol. Furthermore, Alice and Bob can stay anonymous to the SP but the transaction can be linked to them by the IdP who handles their account.

However, the anonymity of the users can be compromised when one needs to receive the digital product. Sending it via e-mail to his regular e-mail address could reveal information about his/her true identity. By extending the functionality of the identifier, while bearing in mind that most social networks nowadays assign an e-mail address to each user (usually with the following format username@socialnetwork.com) and by combining these two we could retain a degree of privacy protection. Sending an e-mail to the address identifier@socialnetwork.com should not reveal more information about the user than when he/she uses his identifier for other purposes.

Analogy

Our proposed protocol could be used for a transaction in this context. The Netbill server would be replaced by the social network, the customer and the merchant can be two users registered with this network while the Service provider has to be a trusted intermediary who can perform the same actions as the Authority.

SSL

The initial authentication of the users, illustrated in figure 5, can be realised with SSL. This is in accord with our protocol as SSL, if implemented securely, can provide mutual data origin authentication and confidentiality. SSL uses public key cryptography to enable symmetric key establishment while symmetric cryptography is used for confidentiality purposes. Furthermore, hash functions are used in order to derive session keys and support key separation, elements also met in our scheme. Finally, SSL includes MACs (Message Authentication Codes) and signatures for entity authentication.

There are also practical benefits as SSL makes limited use of the “slow” public key operations (similarly only used in protocol message 1 in order to establish a symmetric key between the participants) and also supports a wide variety of cryptographic algorithms. A comprehensive source of information about SSL/TLS is (Rescorla, 2000).

Chapter 5: Conclusion

i. Conclusion

The importance of electronic transactions in our modern, fast-paced, technology-driven society is undisputable. The foundations are the underlying network protocols that specify the necessary actions, the entities, the processing of messages or even the infrastructure in a payment system.

A comprehensive review of the proposed protocols illustrated how the technology, but also the payment methods, evolved in the last three decades. It becomes obvious that when it comes to electronic payments there are many factors to take into account and there are no one-size-fits-all solutions. By classifying the protocols into broader categories we identified the requirements, the practical costs, the advantages and disadvantages of each respective group of protocols. We focused on third-party based protocols due to the illustrated shortcomings of the obsolete, two-party based schemes which utilize the "unfair" gradual exchange approach. Although in-line schemes can achieve strong fairness, they suffer from practical problems as they, almost certainly, face bottleneck problems due to the involvement of the third party in every communicated message of the protocol. Also, by requiring the TTP to store messages and sensitive information (in some cases even indefinitely), users face a security risk, the risk of the TTP's database being compromised. On-line schemes can also provide strong fairness with less computational and communication issues. However, performance deficits can still be a problem, depending on the structure and the implementation of the system. Finally, a more efficient solution, in terms of performance and privacy protection, was presented with the introduction of the *optimistic*, off-line schemes. However, in most cases they manage to provide only weaker notions of fairness due to the third party's limited involvement.

Because of their importance, network protocols have been the centre of attention for many academics and many automated tools were developed for verification purposes. By modelling and specifying the steps, requirements, malicious entities, honest participants or even the network itself, we can translate the whole process to mathematical notation and then validate it. However, there are some limitations in regards to the finite models these tools use and, as a result, some drawbacks in terms of the findings. We chose the Zhou-Gollmann protocol and conducted a comparison of four different formal analyses in order to illustrate these limitations and to show that each analysis is profoundly dependent on the author's perspective and the language used.

The main contribution of this thesis was to modify a protocol proposed by (B. Cox, D. Tygar, and M. Sirbu, 1995) in order to make it fairer, more efficient and more suitable for payments. We replaced the “inappropriate” Kerberos ticket, used by the original protocol for authentication and key establishment, with digital signatures, we addressed the integrity and freshness of the transferred messages and we suggested the introduction of a product certification entity that enhances the fairness of the system and especially, for the customer. Additionally, we presented a detailed description and analysis of the protocol and proposed a way to use the existing identity management services used by major social or payment networks in order to implement our protocol and achieve our goals within the current, distributed, electronic environment.

ii. Areas of further research

It would be interesting to practically test our protocol within the context we discussed in order to enable low-value transactions among users. Furthermore, it is really important to discuss the issues and concerns of trusting a social network enough to share your payment details and habits with it. The main goal would be to find a way to circumvent these problems and protect the anonymity of users even from the Identity provider.

Another discouraging factor, which should also be tackled, lies in the fact that these networks do not qualify for payments unless they address certain security issues. For instance, session identifiers (cookies, etc.) might not be designed with optimum security in mind as payment services were not their primary concern. Moreover, user authentication could be enhanced and become more reliable by using two-factor authentication for users that want to perform a transaction.

Bibliography

1. Asokan, N., Schunter, M. and Waidner. (1997). *Optimistic Protocols for Fair exchange.* Proceedings of the IEEE Symposium on Research in Security and Privacy, pp 86-99.

2. Asokan, Shoup, Waidner. (1998). *Asynchronous Protocols for Optimistic Fair Exchange.* Proceedings of the IEEE Symposium on Research in Security and Privacy, pp 86-99.

3. B. Cox, D. Tygar, and M. Sirbu. (1995). *Netbill security and transactions protocol.* In First USENIX Workshop on Electronic Commerce.

4. B. von Solms and D. Naccache. (1992). *On blind signatures and perfect crimes.* Computers and Security, 11(6):581–583.

5. Bahreman Alireza, J. D. Tygar. (1994). *CERTIFIED ELECTRONIC MAIL.* In Proceedings of the 1994 Network and Distributed Systems Security Conference.

6. BAO, F., DENG, R. H., AND MAO, W. (1998). *Efficient and Practical Fair Exchange Protocols with off-line TTP.* Oakland, California: In Proceedings of the IEEE Symposium on Security and Privacy.

7. Bella, G., Paulson, L.C. (2001). *Mechanical proofs about a non-repudiation protocol.* In Boulton, R.J., Jackson, P.B., eds.: Proceedings of the 14th International Conference on Theorem Proving in Higher Order Logics. Number 2152 in Lecture Notes in Computer Science, Springer Verlag 91–104.

8. Ben-Or Michael, ODED GOLDREICH, SILVIO MICALI, AND RONALD L. RIVEST. (1990, January). *A fair protocol for signing contracts.* Retrieved July 04, 2012, from IEEE Xplore Digital Library: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=50372

9. Blum, M. (1983, May). *How to exchange (secret) keys.* Retrieved July 02, 2012, from ACM Digital Library: http://delivery.acm.org/10.1145/360000/357368/p175-blum.pdf?ip=134.219.227.11&acc=ACTIVE%20SERVICE&CFID=120908468&CFTOKEN=22152226&__acm__=1341183283_58e2b9444716681491e53e5a7a6cdff9

10. Brands S. (1993). *Untraceable Off-line Cash in Wallets with Observers.* In Douglas R. Stinson, editor, Advances in Cryptology - Crypto '93, pages 302–318, Berlin, 1993. Springer-Verlag. Lecture Notes in Computer Science Volume 773.

11. Bürk, Pfitzmann. (1990). *Value Exchange Systems Enabling Security and Unobservability.* Computers & Security 9/8 (1990) 715-721.

12. Camenisch, L. J., Piveteau, J-M. and Stadler, M. A. (1994). *An Efficient Electronic Payment System Protecting Privacy.* Proc. 3rd European Symposium on Research in Computer Security - ESORICS '94, pp 207-215,.

13. Chaum D. (1988). *Unpublished Manuscript.*

14. Chaum D. (1981). *Untraceable Electronic Mail Return Addresses and Digital Pseudonyms.* Communications of the ACM 24, pp 84.

15. Chaum, D. L. (1982). *Blind Signature for Untraceable Payment.* Advances in Cryptology Proceedings of CRYPTO 82, Plenum Press, New York, pp 199-203.

16. Chaum, D. (1989). *Online Cash Checks.* Advances in Cryptology, Eurocrypt'89, LNCS 434, Springer-Verlag, pp 289-293.

17. Chaum, D. (1989). *Privacy Protected Payment Unconditional Payer and/or Payee Untraceability.* SMART CARD 2000, Elsevier Science Publishers B.V. (North-Holland), pp 69-93.

18. Chaum, D. (1985). *Security without Identification: Transaction Systems to Make Big Brother Obsolete.* Communications of the ACM, 28 (1985), pp 1030-1044.

19. D. Chaum, A. Fiat, and M. Naor. (1988). *Untraceable electronic cash.* In Proceedings of Crypto '88, volume 401 of Lecture Notes in Computer Science pages 319–327. Springer Verlag,.

20. David Chaum, Ivan Bjerre Damgård, Jeroen van de Graaf. (1988). *Multiparty Computations ensuring privacy of each party's input and correctness of the result.* Crypto '87, LNCS 293, Springer-Verlag, Berlin 1988, 87-119.

21. Dorey, P. (2011, November). The Business of Trust, Visiting Professor at Royal Holloway University of London.

22. European Parliament, C. (1999). *Community framework for electronic signatures*. Retrieved from Europa, Summaries of EU legislation: http://europa.eu/legislation_summaries/information_society/other_policies/l24118_en.htm

23. Even Shimmon, Oded Goldreich, and Abraham Lempel. (1985). *A randomized protocols for signing contracts.* Retrieved from laboratoire d'informatique de l'école polytechnique: http://www.lix.polytechnique.fr/~catuscia/teaching/papers_and_books/SigningContracts.pdf

24. Even Shimon, Yacobi Yacov. (1980). *Relations among public key signature systems.* Technical Report 175, Computer Science Departament, Technion.

25. G. Davida, Y. F. (1997). *Anonymity control in e-cash systems.* In Proceedings of Financial Cryptography '97, volume 1318 of Lecture Notes in Computer Science, pages 1–16. Springer Verlag.

26. Gollmann, D. (2011). *Computer Security (3rd edition).* John Wiley & Sons.

27. Gürgens, S., Rudolph, C. (2002). *Security analysis of (un-)fair non-repudiation protocols.* In: Proceedings of the Conference on Formal Aspects of Security.

28. International Telecommunication Union. (2012). *Trends In Telecommunication Reform 2012.*

29. J. Camenisch, M.Stadler, J.M. Piveteau. (1995). *Fair blind signatures.* In Proceedings of Eurocrypt '95, volume 921 of Lecture Notes in Computer Science, pages 209–219, Springer Verlag.

30. J. Camenisch, U. Maurer, and M. Stadler. (1996). *Digital payment systems with passive anonymity-revoking trustees.* In Computer Security - ESORICS'96, volume 1146 of Lecture Notes in Computer Science, pages 31–43, Springer Verlag.

31. Jakobsson, M., Raihi, D., Tsiounis, Y. and Yung, M. (1999). *Electronic Payments: Where Do We Go from Here?* Springer Verlag.

32. Khalid Haseeb , Dr. Muhammad Arshad , Shoukat Ali , Dr. Shazia Yasin , Khalid Haseeb , Dr. Muhammad Arshad , Shoukat Ali , Dr. Shazia Yasin. (2011). *Secure E-Commerce Protocol.* Computer Science Journals.

33. Kremer, S. (2003-2004). *Formal Analysis of Optimistic fair exchange protocols.* Université Libre de Bruxelles.

34. Larry L. Peterson, Bruce S. Davie. (2007). *"Computer Networks, Fourth Edition: A Systems Approach (The Morgan Kaufmann Series in Networking).* Morgan Kaufmann Publishers.

35. M. Jakobsson and M. Yung. (1996). *Revokable and versatile electronic money.* In Proceedings of the 3rd ACM Conference on Computer CommunicationSecurity, pages 76–87. ACM Press.

36. M.Martin, K. (2012). *Everyday Cryptography.* Oxford University Press, USA (May 4, 2012).

37. Matthew K. Franklin, Michael K. Reiter. (1997). *Fair Exchange with a Semi-Trusted Third Party.* In Proceedings of the 4th ACM Conference on Computer and Communications Security.

38. Merriam-Webster. (2012). *Definition of Trust*. Retrieved June 27, 2012, from Merriam-Webster: http://www.merriam-webster.com/dictionary/trust

39. *OAuth Protocol Workflow*. (2011, July 15). Retrieved from hueniverse: http://hueniverse.com/oauth/guide/workflow/

40. Pfitzmann B., Waidner M. (1992). *How to Break and Repair a "Provably Secure" Untraceable Payment System.* Proceedings of Advances in Cryptology (CRYPTO '91), volume 576 of LNCS.

41. Quisquater Jean-Jacques, Guillou Louis, Annick Marie, Berson Tom. (1990). *How to explain zero-knowledge protocols to your children.* Proceeding CRYPTO '89 Proceedings on Advances in cryptology, Pages 628-631.

42. RAY, INDRAJIT RAY and INDRAKSHI. (2002). *Fair Exchange in E-Commerce.* Retrieved July 01, 2012, from http://www.cs.odu.edu/~mukka/cs772s04/slides/Fairexchange.pdf

43. Rescorla, E. (2000). *SSL and TLS: Building and Designing Secure Systems.* Addison Wesley.

44. Robert H. Deng, Li Gong, Aurel A. Lazar and Weiguo Wang. (1996). *Practical protocols for certified electronic mail.* Journal of Network and System Management, Vol. 4, No. 3,.

45. Saidha Puneet, Tom Coffey. (1996). *NON-REPUDIATION WITH MANDATORY PROOF OF RECEIPT.*

46. Sandholm Tuomas, Lesser Victor. (1996). *Advantages of a leveled commitment contracting protcol.* University of Massachusetts at Amherst, Department of Computer Science, Amherst, MA.

47. Schneider, S. (1998). *Formal analysis of a non-repudiation protocol.* In: Proceedings of the 11th IEEE Computer Security Foundations Workshop.

48. Susan Pancho-Festin, Dieter Gollmann. (2005). *On the Formal Analyses of the Zhou-Gollmann non-repudiation protocol.* Proceeding FAST'05 Proceedings of the Third international conference on Formal Aspects in Security and Trust, Pages 5-15, Springer-Verlag.

49. Windley, P. (2005). *Digital Identity.* O'Reilly Media, Inc.

50. Wu, T. (1999). *A Real-World Analysis of Kerberos Password Security.* In Proceedings of NDSS.

51. Y. Frankel, Y. Tsiounis, and M. Young. (1996). *"Indirect discourse proofs": Achieving efficient fair off-line e-cash.* In Proceedings of Asiacrypt '96, volume 1163 of Lecture Notes in Computer Science, pages 286–300, Springer Verlag.

52. You, Zhou, Lam. (1998). *On the efficient implementation of fair non-repudiation.* ACM SIGCOMM Computer Communication Review.

53. Zhang N., Shi Q. (1996). *Achieving Non-repudiation of receipt.* The Computer Journal 39 (10): 844-853.

54. Zhou, Gollmann. (1996). *A fair non-repudiation protocol.* Proceedings of the 1996 17th IEEE Symposium on Security and Privacy, IEEE Computer Society Press, pp.55-61.

55. Zhou, J., Gollmann, D. (1998). *Towards verification of non-repudiation protocols.* In: Proceedings of 1998 International Refinement Workshop and Formal Methods Pacific,Canberra, Australia. 370–380.