FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

EasyPay - A Novel Approach to Mobile Payment

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Wirtschaftsinformatik

eingereicht von

Matthias Trümmel
Matrikelnummer 0925456

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Univ.Ass. Dipl.-Ing. Dr. Marco Zapletal

Wien, 28. Januar 2014

(Unterschrift Verfasser) (Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

EasyPay - A Novel Approach to Mobile Payment

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Business Informatics

by

Matthias Trümmel
Registration Number 0925456

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Univ.Ass. Dipl.-Ing. Dr. Marco Zapletal

Vienna, 28. Januar 2014 _____ _____

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Matthias Trümmel
Audorfgasse 22/1, 1210 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasser)

Acknowledgements

I want to thank my advisor Marco Zapletal for making this thesis possible, for his patience and for always supporting me with useful advice.

Of course I also want to thank my friends all over the world for their support and that they have never stopped believing in me during the last couple of months while working on my thesis and throughout my whole study. Moreover, I want to thank them for the great time I have had with them in Vienna and everywhere else in the world.

Special thanks goes to Alexander Ortner who worked with me on shaping the idea for this thesis and to Thomas Ziegelbecker who always had an open ear when needed. Without our endless motivating study and working sessions I would not be where I am today.

Furthermore, I want to thank my family for their financial and mental support over the last years. Without them I wouldn't have been successful.

Abstract

In the last couple of years the internet and digital money have taken a very big part in everyone's life [29]. Paying bills online and using a wide variety of different apps on mobile phones to conduct financial transactions has become more and more common [2]. Lots of different companies have more or less successfully tried to establish new payment systems that should allow easy money transfers. Especially mobile solutions have become very popular, since mobile phones play a big role in our daily lives [2]. So far most of the mobile solutions have focused on payments between businesses and customers (B2C - business-to-customer) but there is still no solution that provides an easy way to conduct customer-to-customer (C2C) payments.

Most of the payment systems on the market require an active internet connection to carry out a transaction. Unfortunately, cell phone coverage is not available everywhere, especially in rural and abandoned areas. When going abroad, mobile internet may result in high roaming costs which limit the usefulness of online apps. Even in developed countries, such as Austria, where high cell phone coverage can be expected, certain areas are not covered[1] [2]. These areas include for example mountains (not including touristic or skiing areas), tunnels or sometimes public transportation (e.g. airplanes, long distance trains). Therefore, a protocol focusing on offline C2C transactions is proposed which solves the described problems.

This thesis analyses established online and mobile payment systems, investigates pros and cons, and draws conclusions for a novel mobile payment system. The existing solutions and their protocols are analysed and evaluated with a simple framework that is also presented in this thesis. The proposed payment system and its protocol are designed for high security and availability. Moreover, it works without an active internet connection. Therefore, the payment service is always available to the user as long as both transaction parties are carrying their phones. Additionally, Near Field Communication (NFC) is used to make transactions easier and more seamless for the user. NFC has gained a lot of popularity lately, because most of the modern smartphones have an NFC chip. Furthermore, credit card companies like Visa or MasterCard have started to extend their services with NFC to allow customers to pay contactless in shops with a debit or credit card.

[1] `http://www.t-mobile.at/info-und-support/Netzabdeckung/Netzversorgung-in-Ihrer-Umgebung.php`, accessed 2013-12-06

[2] `https://www.a1.net/hilfe-support/netzabdeckung/`, accessed 2013-12-06

A prototype which implements the proposed protocol shows how it works. It also points out a few minor problems and how they can be solved.

Since a majority of Austria's transactions are still made via cash, there is a market potential for mobile payment systems [29]. The user benefits from a tool that can be used everywhere, regardless of country borders, phone signal or WiFi connection. Furthermore, the user is not limited by any external factors apart from the phone's battery. This can especially be an advantage when the user wants to transfer money to a friend, when wanting to transfer small amounts on a regular basis or to split bills between several people.

Kurzfassung

Wenn man die letzten Jahre betrachtet kann man feststellen, dass das Internet sowie digitales Geld unverzichtbare Bestandteile unseres Lebens geworden sind [29]. Der Bankomat und Netbanking sind selbstverständlich geworden und niemand könnte sich heute noch vorstellen ohne diese Hilfsmittel zurecht zu kommen. Digitales Geld hat nicht nur den Vorteil, dass man nicht ständig Bargeld bei sich tragen muss, sondern auch, dass man auf einfache Weise jederzeit Geld ausgeben kann. Viele verschiedene Firmen haben in den letzten Jahre versucht auf diesen Zug aufzuspringen um den Bezahlvorgang noch einfacher und schneller zu gestalten. Gerade durch das Zeitalter der Mobiltelefone sind viele mobile Lösungen entstanden, die jedoch großteils nur verwendet werden können um in Geschäften zu bezahlen (B2C - Business-to-Customer), aber selten um zwischen zwei Personen Geld auszutauschen (C2C - Customer-to-Customer).

Die meisten etablierten Bezahlsysteme benötigen eine Internetverbindung um zu funktionieren. Im Ausland, wenn durch Roaming hohe Kosten entstehen würden, ist die Benutzung eines Online-Bezahlsystems oft mit zusätzlichen Kosten für den Datentransfer verbunden. Des Weiteren gibt es in einigen Gebieten keine vollständige Netzabdeckung durch Mobilfunkanbieter[3,4]. Das kann zum Beispiel abgelegene Gebiete wie Berge (abgesehen von touristisch erschlossenen Gebieten oder Skigebieten) oder Tunnel betreffen. Wenn man nun aus den eben beschriebenen Gründen keine mobile Lösung, die eine Internetverbindung benötigt, verwenden kann, ist man momentan weiterhin auf die Verwendung von Bargeld angewiesen.

Da der Trend jedoch klar Richtung mobiler Bezahlsysteme geht und Bargeld oft nur noch verwendet wird, wenn es keinen anderen Weg gibt, wäre es von Vorteil in allen Lebenslagen eine Möglichkeit zu haben um Geld auch zwischen zwei Personen über das Mobiltelefon zu tauschen. Vor allem in Österreich werden noch viele Transaktionen mit Bargeld durchgeführt [29]. Daher würde es gerade hier einen großen potentiellen Markt für eine Lösung dieser Art geben, von der viele Personen profitieren würden. Das wäre insbesondere dann ein Vorteil, wenn jemand an einen Freund oder Bekannten Geld übertragen will, jedoch kein Bargeld zur Verfügung hat und gleichzeitig keine Verbindung zum Internet möglich ist um ein Onlinesystem zu nutzen. Außerdem könnten so kleinere Beträge einfach und schnell getauscht werden oder Rechnungen zwischen Freunden aufgeteilt werden, ohne sich um etwaiges Wechselgeld kümmern zu müssen.

[3] `http://www.t-mobile.at/info-und-support/Netzabdeckung/Netzversorgung-in-Ihrer-Umgebung.php`, accessed 2013-12-06

[4] `https://www.a1.net/hilfe-support/netzabdeckung/`, accessed 2013-12-06

Diese Arbeit analysiert deshalb bestehende und etablierte Bezahlsysteme in Österreich und in anderen Ländern. Dabei werden Vor- und Nachteile ausgearbeitet, sowie Schlüsse für ein neues Protokoll gezogen, welches das Übertragen von Geld zwischen zwei realen Personen erlaubt. Außerdem wird ein Framework vorgestellt, welches auf bestehenden Kriterienkatalogen für digitale Bezahlsysteme basiert. Dieses wird verwendet um die Bezahlsysteme nach bestimmten Kriterien zu vergleichen und zu untersuchen.

Das Ergebnis der Analyse wird verwendet um die oben beschriebenen Probleme mithilfe eines neuen Protokolls zu lösen. Es erlaubt Geld mit einem Mobiltelefon, welches Near Field Communication (NFC) unterstützt, zwischen zwei Personen zu tauschen. Zusätzlich ist es, da es keine Internetverbindung benötigt, an jedem beliebigen Ort einsetzbar. Daher gibt es abgesehen von der Batterie des Mobiltelefons keine limitierenden Faktoren. Am Ende wird ein Prototyp vorgestellt, der die Funktionalität des Protokolls demonstriert. Die Evaluierung des Protokolls und des Prototyps zeigt, dass das Protokoll auch in der Praxis funktionieren würde. Es wird aber auch gezeigt, dass es noch Probleme mit dem vorgestellten Protokoll gibt und wie diese zu lösen sind.

Contents

CHAPTER 1

Introduction

Means of payment have almost always played an important role in human history. It started more than 5000 years ago with the invention of pre-metallic money, e.g. cowrie or whales' teeth. Another 4400 years passed (600 BC to AD 400) until the first real coins were invented and ported across borders by the Greek and Roman. This was followed by the introduction of different currencies and paper money. Trade expanded rapidly and became a big worldwide ecosystem with a lot of different currencies, the financial market and many different options for exchanging money, e.g. coins, bills and credit cards [7].

The general direction has always been towards more convenient ways to exchange and share money and to make it as easy as possible to spend it. This applies to end-customers, as well as businesses across the world. Currently, the most widespread means of payment are debit and credit cards as well as online banking. For small amounts of money, cash is still the most important way of payment [29].

When the internet became more popular, the need and request for online payment systems providing simple ways to securely exchange money arose. This led to the fact, that many different approaches became available. This accounts for theoretical approaches as well as approaches that have been put in place. Furthermore, electronic payment, security and related topics are among the most popular research topics nowadays [16, 26].

1.1 Motivation

As already mentioned there are plenty of different ways and methods to pay and transfer money between parties. Most of them, besides cash, rely on an active internet connection or one has to go to a bank to initialise the bank transfer. In the last couple of years a lot of different new techniques have arisen that should make the transfer of money easier and more secure against tampering (see chapter 2). This includes Near Field Communication (NFC), which allows for physically transferring data, hence money, over a very short distance. For NFC communication, the following device-combinations are possible [23]:

- two smartphones
- a NFC-enabled bank card and a NFC terminal
- a smartphone and a NFC terminal

However, most of the available NFC solutions only allow to pay for goods and services in a shop. Thus, there is currently a focus on business-to-customer (B2C) applications (see chapter 2). NFC overcomes the problem of transferring the whole transaction over an open channel[1], which makes it more secure against different kinds of attacks.

At certain places or times, when one does not have an internet connection and there is no branch of a bank close by, or it's just too much effort to go to the next bank, one simply can't transfer money between two people besides with cash. This especially occurs developing countries, where there's often internet access available but no working banking infrastructure. This makes money transfers difficult and time intensive.

This is why a novel approach was designed. Focusing on customer-to-customer (C2C) money exchange, it overcomes the stated problems and provides an easy way to securely transfer money offline between two persons when there is no internet connection available.

Especially in Austria a majority of transactions are still performed with cash, as stated by OENB [29]. According to this study 83,12% of all transactions, especially for smaller amounts (only 65,33% of the value), are still conducted with cash. Online payment systems, excluding the use of credit cards and systems alike, are only responsible for 0,13% of the volume and 0,23% of the value. Payments by mobile phone for 0,04% of the volume and 0,01% of the value. This shows that there is a huge potential for all sorts of online and mobile payment systems in Austria.

According to Accenture's Mobile Web Watch 2013 [2], which covers the period from November 2012 to January 2013, mobile payments are already used by 20 % of smartphone users[2]. The study also states, that this number could almost double in the next year. Especially emerging markets like China, India, Brazil, Russia and Turkey play a big role here (currently 29 %). Preferred are mobile payments by people from urban areas, with higher education and social networks users.

In the last couple of years plenty of approaches have tried to establish online or mobile payment systems, especially when the credit card became popular, since the transaction costs were too high at first. The most promising approaches back then were *NetCents* [33], *NetBill* [40], *Agora* [9] and *SET* [46]. Furthermore, mobile solutions like *i-WAT* [38], *PPay* [48] and *Offline*

[1] An open channel is for example an unencrypted connection over the internet or a (wireless) network, which can be eavesdropped.

[2] The study does not state how often and in which timespan mobile payment have to be used to count towards this number. Furthermore, for mature markets such as Austria this number is only 16%. It is also expected, that the number for Austria is even lower, because Austria is only represented by 700 participants in the study, whereas the USA is represented by 1500 participants. Moreover, there is a big variety of mobile payment system available in the USA as chapter 2 shows.

Karma played a role [10]. These examples are only a subset of solutions that have been invented, but most of them remained as a theoretical idea, or did not reach enough people.

As [32] states, *credit cards were designed for the physical world to be seamless - I take out my wallet, pull out my card and swipe.* However, no real credit card substitution for the online and mobile market has reached the critical mass world wide yet.

Therefore, the following conclusions can be drawn. Currently, the use of mobile payment systems, in terms of volume and transaction, is very low in Austria [29]. This and the fact that the usage of mobile phones, mobile internet, and therefore mobile payment systems will grow in the future [2], is among the most motivating facts for this master thesis. There is obviously a need for conclusive systems and research on mobile payment that can replace the credit card and the wallet.

1.1.1 Mobile Payment Market and Extension

After having proposed a protocol which allows offline payments, the next big question is why mobile payment is not already more prominent in Austria and around the world.

The first problem is the **technological heterogeneity** starting with different operating systems on the market including Google's Android and Apple's iOS as the most prominent examples[3]. Another aspect is that a lot of different payment solutions are available on the market, but none of those has really reached a critical mass (see chapter 2). Also the fact, that people have security concerns [2] is a reason why mobile payment is still not very prominent, especially in Austria [29]. T3N [20] argues, that certain requirements need to be met to make it easier to penetrate the market.

- **Customers** The customer base is hard to get, because especially the registration for new customers, as well as a big customer base, is a big step. This is not available for this new approach and a lot of effort is needed to get the customer's trust.
- **Technology** The customer does not care about the technology used as long as it is fast, secure and always available. Most of these requirements are met by the protocol, even though there is potential for improvements regarding the time which is needed to conduct a transaction (see chapter 5.1).
- **Credibility** Being trustworthy is important when it comes to money. With this solution not only a new technology is invented, but also no well known brand is used to promote a product.
- **Benefit** As it has been discussed in chapter 1.1, this approach adds multiple benefits for the user, which is very important to be accepted on the market and to get an own niche on the market. It is mainly given through the offline aspect.

[3] `http://www.forbes.com/sites/louiscolumbus/2013/01/17/2013-roundup-of-mobility-forecasts-and-market-estimates/`, accessed 2013-12-29

- **Points of Acceptance** Finding spots where the mobile payment system is accepted is rather easy in this case, because everyone with a Android Smartphone that supports NFC can use it. This is a major difference to the most common payment systems, because no merchants have to be found who would accept payments.

Reaching a critical mass is a difficult task, because trust in the software and the company by the customer is needed. This is also why [20] argues, that only big players like Amazon, Apple, Google, PayPal, or mobile phone providers would have a realistic chance. This argument has to be considered with care, since there are also startups like Square or Venmo who already have a reasonable number of customers[4] or partners[5].

1.2 Problem Statement

As described in chapter 2, a lot of research has already been done, but there is no conclusive research on the **research question of how to securely transfer money *offline* between mobile phones**.

There are several reasons why offline payment methods will still have an important role in the future:

- Security
  - The offline approach adds an additional security aspect to the money exchange, because a physical presence of the persons and their mobile phones is necessary to complete it.
  - Having the possibility of exchanging money without an internet connection helps preventing man in the middle attacks, eavesdropping on the exchange and other security issues, because a possible hacker does not have access to the transferred data. At a later point, when the payments are synchronised with the server, the tokens that were created cannot be altered anymore. A hacker cannot gain any benefit of eavesdropping or altering a money token.
  - Concerns about the security of mobile payment systems may be the reason why they are not used [2].
- Availability
  - The possibility of exchanging money offline increases the availability of the service, because the internet is only optional and the service can be used seamlessly.
  - The high availability should give the user the feeling, that it works the same way as his or her wallet.

[4] `http://bits.blogs.nytimes.com/2012/08/16/payments-start-up-braintree-buys-venmo-for-26-2-million`, seen 2013-12-29

[5] `http://www.reuters.com/article/2012/08/09/uk-starbucks-square-idUSLNE87800P20120809`, seen 2013-12-29

- Lack of an active internet connection
  - Like stated before, the service can also be used without an active internet connection. This is especially helpful abroad, in developing countries and in rural or abandoned areas.

Accenture [2] put this the following way:

> *The mobile customer is forever changed. Empowered by smartphones and tablets, savvy consumers have come to expect immediacy at their fingertips.* ***They want everything, everywhere, now.*** *The device and the network are simply the means by which they manage and control the communication and entertainment aspects of their lives.*

In other words, a convenient way to exchange money, which is always available needs to be found. To make the problem and the need for a solution easier to understand, the following scenarios are given.

Split Payments Imagine two friends at a cafe drinking a cup of coffee together. One of them pays for both of them, because the other one does not carry any cash. Now the friend without money has two possibilities. The money can either be payed back later in cash or immediately by using a bank transfer. Paying back later may be inconvenient or forgotten about while issuing a bank transfer needs a lot of input data (e.g. bank account number). Both of them take a certain amount of time to conduct and both of them would profit from a solution that solves the problem within seconds.

Abroad When being abroad, free internet, that is not available everywhere can be a problem. This is because one either has to find a Wi-Fi hotspot or pay for the internet usage abroad if there is no special agreement with the phone provider. Not only the lack of internet connection, but also different currencies can be a problem. For example when visiting a friend abroad in the United States, the friend could pay for all bills and gets repaid immediately via mobile phone. This results in the fact that the person who visits the United States only has to convert money once to USD. Therefore, the fee for converting currencies only applies once and only the exact needed amount is converted.

Developing Countries and No Cell Phone Coverage Using mobile phone services in areas without cell phone coverage or internet connections is difficult. With an offline payment system people could conduct their payments without the need to think about cell phone coverage. Furthermore, in areas where crime is very present there is no need to fear if money or the phone gets stolen, because the money is safely stored on the phone and can only be accessed when opening the locked wallet with a code on the phone.

1.3 Aim of the Work

The main focus of this work is to provide a protocol which allows to securely conduct offline C2C payments. A description of the protocol is given and discussed in detail. To give a good idea about how it works, the protocol is discussed from the user's point of view, as well as from the technical point of view. This is necessary to understand the usage, as well as how it can be implemented. Another important part is the description of the payment service and clearing of accounts. This includes a detailed description of how someone can use the system, what steps need to be performed to make a money transaction, as well as a way to conduct the *whole transaction*. An example is to transferring money from one user's bank account to another user's bank account.

In this work existing mobile payment approaches and systems are analysed and conclusions for a new payment system are drawn. As a final result an evaluation in form of a prototype, which implements the proposed protocol is developed for Android. This shows, that the protocol could work in practise. As evaluation the gathered results are discussed and the new payment system is compared to existing solutions.

The results provide a basis and a starting point for further research and development. Since this work only deals with the protocol itself and parts of the environment needed, there is a lot of room for further research in different areas.

This thesis does not aim to provide a ready-to-sell solution for a mobile payment application, but should rather be seen as a discussion for a new approach for a mobile payment application. It should also work as a first approach in a relatively new direction, where not much research has been done. The prototype shows that the introduced protocol works, but does not claim to have all the functions needed to succeed on the market. It also does not aim to be a protocol for business-to-customer (B2C) or business-to-business (B2B) payments but only for customer-to-customer (C2C), hence friend payments.

1.4 Methodological Approach

The master thesis follows the **Design Science** methodology. *The design-science paradigm seeks to extend the boundaries of human and organisational capabilities by creating new and innovative artefacts.* [14, p. 75]. The idea behind this approach is *to solve identified organisational problems* by *creating and evaluating IT artefacts* [14, p. 77] and the goal is utility. Hevner et al. propose seven guidelines for Design Science in Information Systems Research [14, p. 82-90] that will be used to conduct an effective design research.

- Guideline 1: Design as an Artefact
  - The designed IT artefact will be the protocol as well as the prototypical implementation of the stated problem. But not only the protocol, but also the procedures how to use the protocol will be discussed in detail.

- Guideline 2: Problem Relevance
  - As already described in chapter 1.2, no conclusive research about offline money exchange has been done. This, and the fact that no comparable payment system is available on the Austrian market led to a problem which needs to be solved. This is why the development of a protocol and its implementation will be addressed by this master thesis.
- Guideline 3: Design Evaluation
  - The design will be evaluated in terms of functionality, security, availability and anonymity with a simple framework that is derived form other evaluation frameworks.
- Guideline 4: Research Contributions
  - The contribution is a proposed secure protocol for offline money exchange as well as an analysis in form of an evaluation.
- Guideline 5: Research Rigour
  - The research will be based on state of the art technology as well as clear defined and tested literature. Therefore, the protocol of the existing payment systems will be analysed as good as possible and conclusions for the new protocol will be drawn out of it.
- Guideline 6: Design as a Search Process
  - State-of-the-art technology and products will be observed and possible problems identified. Furthermore, solutions for the problems will be proposed and implemented in the protocol and the prototype.
- Guideline 7: Communication of Research
  - The master thesis provides information about state of the art, the proposed protocol, as well as an evaluation of the protocol and the implemented prototype. The protocol is discussed from the user's and from the system's point of view, to make the context as well as how to implement it easy to understand.

1.5 Structure of the Work

The work is structured as follows: First, a **comparison and analysis** of the current state of the art is done, emphasising the advantages and disadvantages of the most important and relevant solutions providing a mobile payment service. In order to make the existing approaches easier comparable, a simple framework is introduced.

The results of the comparison and research that has already been done on that topic is the basis for the second part. This is the **development of a protocol to exchange money *offline*** between mobile phones. The protocol itself shows how to securely transfer money between phones. It also shows how the conducted transactions could be synchronised with the user's payment system account and bank account. This also includes a solution for connecting the payment service to the users' bank accounts (e.g. via a reference account or credit card).

Third, a **prototypical implementation** for Android is created as a proof of concept for the developed protocol to show that it works and how it's implemented.

Fourth, an **evaluation** is done to see if the proposed protocol and implementation could work in practise. The framework introduced in the first part is applied to the novel payment system to see if the requirements are fulfilled and to compare it to existing approaches.

1.6 Classification of Payments

A brief overview about different terms that are relevant for the entire thesis are given in this chapter. It classifies payments in terms of the number of participants, the actual payment time and the amount.

1.6.1 Classification by Participants

Payments can be conducted between different types of actors. These actors can either be real people, hence customers, or businesses on the other hand. Out of these two types of actors three different constellations can occur as follows:

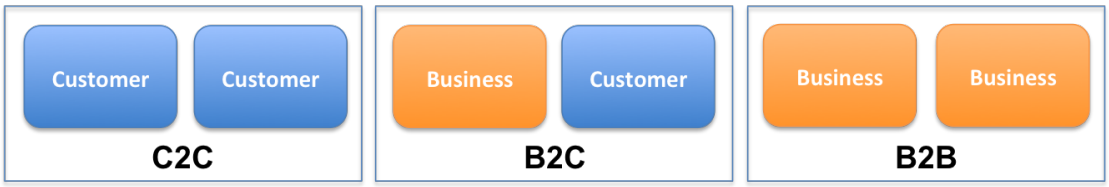


Figure 1.1: C2C - B2C - B2B

- **C2C** stands for customer-to-customer payments and deals with payments that are conducted between two real people.
- **B2C** means business-to-customer. This type of payment involves a customer who typically wants to buy goods from a business.
- **B2B** business-to-business payments are payments that are conducted between two businesses and do not involve any customers.

1.6.2 Classification by Time

Payments can also be categorised by the actual time of payment [44]. Figure 1.2 shows the different forms on a timeline and when they are due for payment. Furthermore, some examples for each category are stated below but they are described in more detail in the next chapter.

Figure 1.2: Past - Now - Future

Pre-Paid Pre-paid means that the user has to pay upfront, hence at some point in the past. Popular examples are *Quick* and *Paysafecard*. On the upside the user can stay anonymous and still be able to pay for goods. In case the card or token is lost, the damage is limited to the remaining pre-paid amount. On the downside the user has to make sure that there is always money available to spend. For most pre-paid systems there is no date of expiration. Thus, the money can be kept on the payment system for as long as the user wants before the actual payment is done, as figure 1.2 indicates.

Pay-Now When the money is instantly transferred between the parties involved it is called a pay-now system. This requires a connection (e.g. via the internet) between the involved parties to execute the transfer. Popular examples are the *online credit transfer* via the online banking service provided by most banks, *cash on delivery*[6], or all kinds of debit cards, e.g. Maestro for Central Europe[7] . Pay-now solutions provide in most cases an easy way to transfer money instantly[8]. In order to conduct an online money transfer the International Bank Account Number (IBAN) and the Business Identifier Codes (BICs, sometimes also known as SWIFT code) are needed [18].

Pay-Later In most cases pay-later systems accumulate the conducted payments and bill the customer at a later point of time at once. This can also be seen in figure 1.2, indicated by the *Pay-Later* shape. The most popular example is the credit card. Another not so well known example is Paybox. This solution charges the customer's phone bill or bank account when used in a shop or when money was transferred between Paybox customers[9].

[6] Cash on delivery is sometimes used for goods that are delivered by mail. The receiver has to pay for the good when it is delivered to him. If he or she does not pay the good is returned to the sender.

[7] `http://www.maestrocard.com`, accessed 2013-11-10

[8] In most cases it takes a couple of hours or days until the payment is visible on one's bank account

[9] `http://www.paybox.at/`, accessed 2013-11-10

1.6.3 Classification by Amount

The last type of payment classification is by amount, hence by the quantity of money paid in a transaction [16]. Depending on the amount different levels of security and restrictions are normally applied.

The higher the amount paid, the more secure the system should be. Since the frequency of payments with a higher amount is lower than the frequency of payments with a lower amount [29], people are more likely to accept that they have to follow certain security protocols, e.g. entering user credentials, a Personal Identification Number (PIN) or a Transaction Authentication Number (TAN).

For payments with low amounts, the same applies the other way around. The less the amount, the more often such payments are carried out [29]. Therefore, the easier it should be to conduct such a transaction. Besides, the damage is lower if a low value transaction is hacked or tampered in any way.

Micro Payment Using sophisticated security procedures for micro payments would not pay off, because they require a lot of computational power which costs money. Therefore, the risk of being a victim of fraud is accepted, since only a small amount of money could potentially get lost or stolen. *"The cost of fraud is made more expensive than the possible value to be gained by cheating"* [16]. Furthermore, creating a hash from data is 100 times faster than signing the same price of data with RSA [36]. This shows that the need for a simple solution is important when it comes to micro payments. [16] also specifies requirements for a good micro payment system. Some of them are:

- **Efficiency** The transaction should be carried out quickly.
- **Low cost** The computational load, the storage costs, as well as the administrative load should be low.
- **Security** It still should have a certain level of security. Especially regarding ownership of the money and integrity.

Moreover, anonymity and multiple transactions with different service providers are stated but those can be considered optional. They do not change anything at the core of the payment system but can be seen as additional factors.

Macro Payment Macro payments use a variety of different security mechanisms to make them as secure as possible. According to [16] especially the following techniques are used to detect fraud and other dangers instantly. First, public key cryptography (see chapter 2.1.2) is used to securely transfer data. Second, online broker activities are used to validate transactions on the customer's and the vendor's side and also to detect abnormalities in the transactions like double spending (see chapter 2.3.1.1). [16] argues that the effort needed to build and maintain such systems is worth it, because with only a small number of fake transactions a lot of damage could be done. Therefore, more security precautions have to be made when dealing with big amounts of money.

CHAPTER 2

State of the Art and Existing Approaches

This chapter gives a brief overview of the most important state of the art solutions concerning payment systems. The solutions described are well established worldwide payment systems but also more regional solutions available on the European and Austrian market, e.g. *Quick*[1]. Furthermore, necessary technology to provide a secure and simple solution is discussed.

Figure 2.1 shows that this chapter is divided into four main parts. The first section is called *Fundamentals of Encryption*. It gives an overview of the state of the art encryption technologies that are needed to make a payment system secure and trustworthy but also to protect against active and passive attacks. The technological aspects of digital payments and virtual currencies are very important, since they heavily depend on cryptographic methods. Among these methods are *Symmetric and Public Key Encryption*, *Public Key Infrastructure (PKI)* and *Hashing*.

The second part deals with the directly relevant technology *Near Field Communication (NFC)*, since it is part of the proposed payment protocol.

Third, the different *Existing and Established Solutions* are divided into four different pillars. Each of these pillars contains the most important solutions of its category and a diversity of different approaches that are available on the market and have a certain degree of popularity.

In order to be able to compare the different solutions, a *Comparison and Evaluation of Existing Approaches* is conducted in the last part. In this chapter, a comparison framework is proposed and applied to the solutions. This helps to get a better understanding of the features that each system supports and their problems. It is also used to compare the existing approaches

[1] `http://www.quick.at/`, accessed 2014-01-26

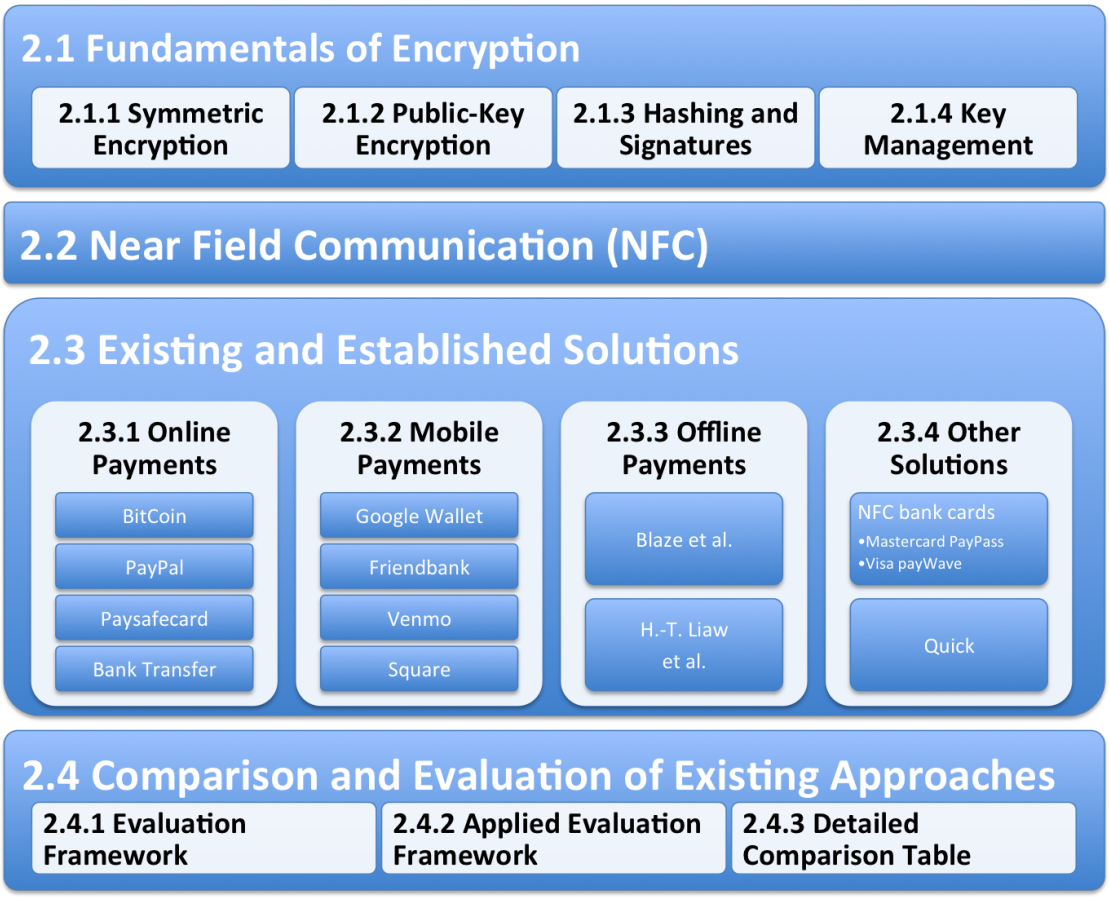


Figure 2.1: State of the Art Overview

with the novel approach of this work. The chapter concludes with a detailed comparison table which makes it easier to compare the introduced solutions.

2.1 Fundamentals of Encryption

Cryptography plays an important role in ensuring that data and messages are securely exchanged between two parties [43]. Encrypting data does not prevent hacking but rather makes it impossible for hackers to read the encrypted data. Currently, there are two main types of encryption in use. *Symmetric Encryption* uses a single shared key to encrypt and decrypt a message. The second type is *Public Key Encryption* where a key pair containing different keys for encryption and decryption is used.

Encryption provides very important functionality to ensure confidentiality, integrity, authenticity and accountability. The mentioned categories are especially important for performing financial transaction [43].

- **Confidentiality** is important to ensure that only the sender and the receiver of the message (e.g. a financial transaction) can read the information that shall be transferred.
- **Integrity** ensures that the information being transferred is not altered by a third party.
- **Authenticity** is needed to be sure that the message and the sender are valid entities and none of the parts in the transaction is altered or forged.
- **Accountability** helps to guarantee that the message that was sent is really from the person who pretends to be the sender of the message.

2.1.1 Symmetric Encryption

Symmetric encryption is the most commonly used type of encryption. It only needs a single key for encryption and decryption. According to [43] symmetric encryption consists of the following parts:

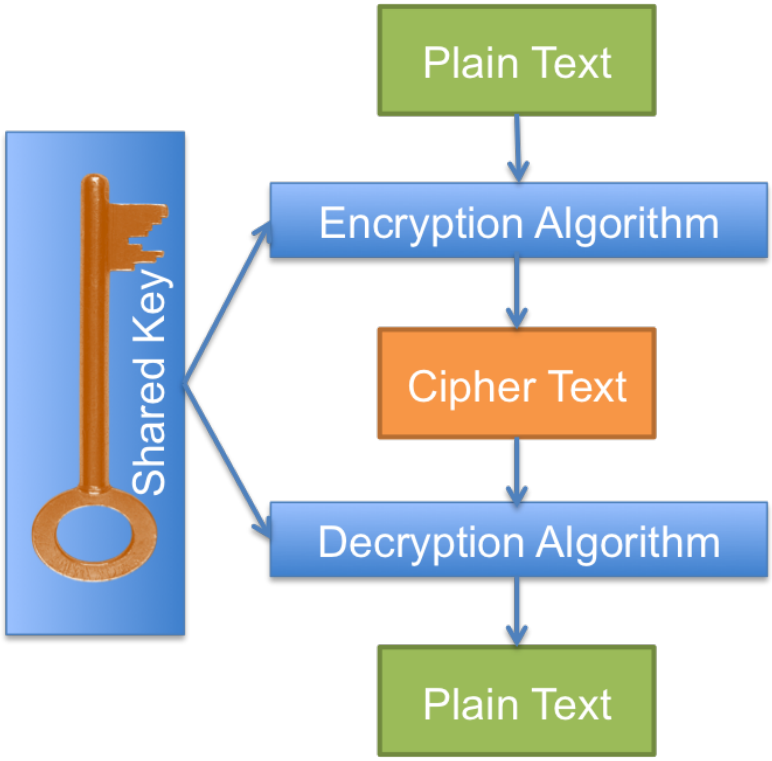


Figure 2.2: Symmetric Encryption

- A **plain text** which is the message that is to be transferred securely.
- The **encryption algorithm** is used to encrypt the message by substituting and transforming the plain text.

- The **key** is used to make the encryption unique, hence the encryption depends on the key as an additional input.
- The **decryption algorithm** does exactly the opposite of the encryption algorithm. This also means this algorithm has to be able to run in the reverse direction.
- The **cipher text** is the encrypted message. It is not possible to read the message without the appropriate key. The cipher text and the key are used to decrypt the message, hence to get the plain text.

Figure 2.2 shows how the symmetric encryption works. *Plain text* is encrypted with a *shared key*. This results in a *cipher text*. This generated *cipher text* can be decrypted with the same *shared key* to get the original *plain text* again. [43] argues that symmetric encryption is only as secure as the algorithm. Moreover, the length and complexity of the key or passphrase is relevant for the extent of security in symmetric encryption.

2.1.2 Public Key Encryption Algorithm

Public key encryption was developed by Ron Rivest, Adi Shamir and Len Adleman in 1977 [43]. This type of encryption is based on splitting prime numbers and uses two keys called public and private key. The public key, as the name indicates, is publicly available to everyone. The private key has to be kept by the owner and may not be given to anyone else. Figure 2.4 shows a simplified version of how such a key pair is created. First, a random number based on two prime numbers is used as an input for the *Key Generation Algorithm*. This algorithm transforms a set of input parameters to a *Key Pair* consisting of a *Private Key* and a *Public Key*.

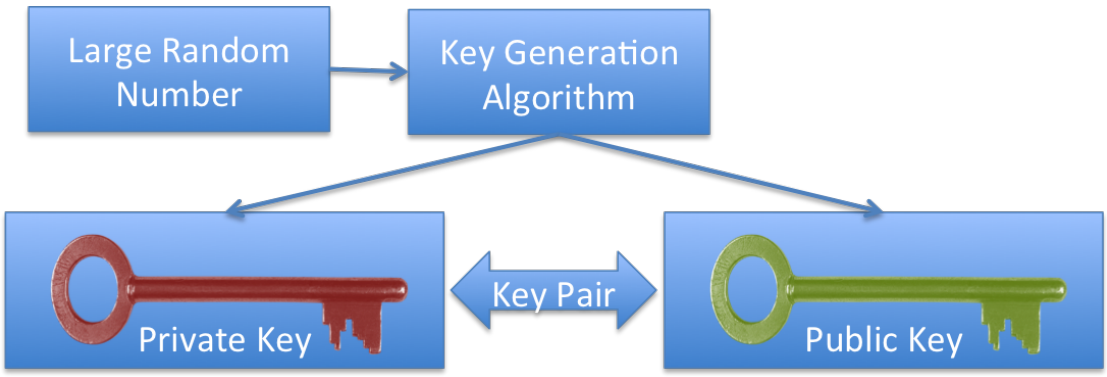


Figure 2.3: Public and Private Key Generation

The fact that there are two keys can be applied in two different ways, as it is shown in figure 2.4. Either one of these keys can be used for encrypting the message and the other one for decrypting it. Therefore, this technique can be used to **establish trust**, as the upper part of the figure shows. The private key is used to encrypt the clear text and the public key is later used to decrypt it. Hence, everyone who is in possession of the appropriate public key can decrypt

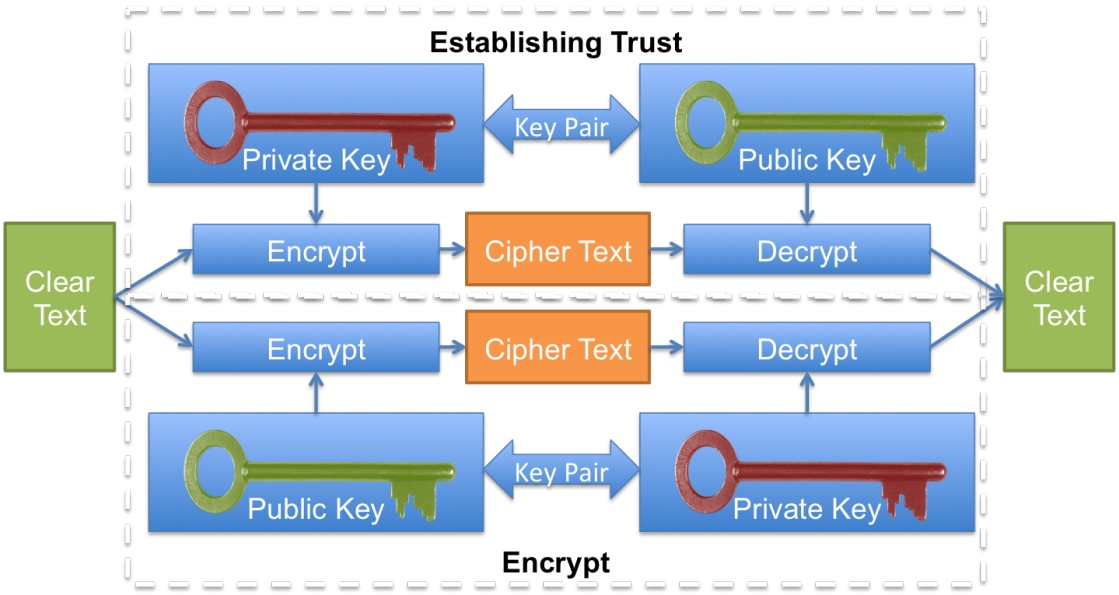


Figure 2.4: Public Key Encryption Usage

the message. One can also be sure that the message was created by the right person, because the sender is the only one who has the private key.

On the other hand a message can be **encrypted** with the public key. Therefore, only the intended recipient can decrypt the message with the private key. This technique is used to prevent eavesdropping attacks.

Public key encryption heavily depends on a large key space to prevent, or at least make it more difficult, to attack the encryption. On the downside a larger key implies a slower system, because it requires more computational power for encryption. By time of writing, the most common key size is between 1024 and 4096 bits [43].

2.1.3 Hashing and Signatures

Any type of encryption helps to prevent *passive* attacks, because a hacker cannot read the data anymore. On the other hand techniques such as hashing, message authentication and signatures can be used to prevent *active* attacks, as these techniques are used to detect altered messages [43].

Hash A hash is simply a checksum of the message being sent. It is computed with an algorithm to ensure the integrity of the message. Hashing itself can therefore be used to provide validation of the data, ensuring that the message that was sent can not be altered without notice. In case the hash is altered as well, the receiver unfortunately would not notice the change. This is why a hash can only be used to validate the integrity of a message. Hashing is also part of the authentication process, since hashing, as well as symmetric and public key encryption, are used to prevent attacks on messages (e.g. altering or falsifying).

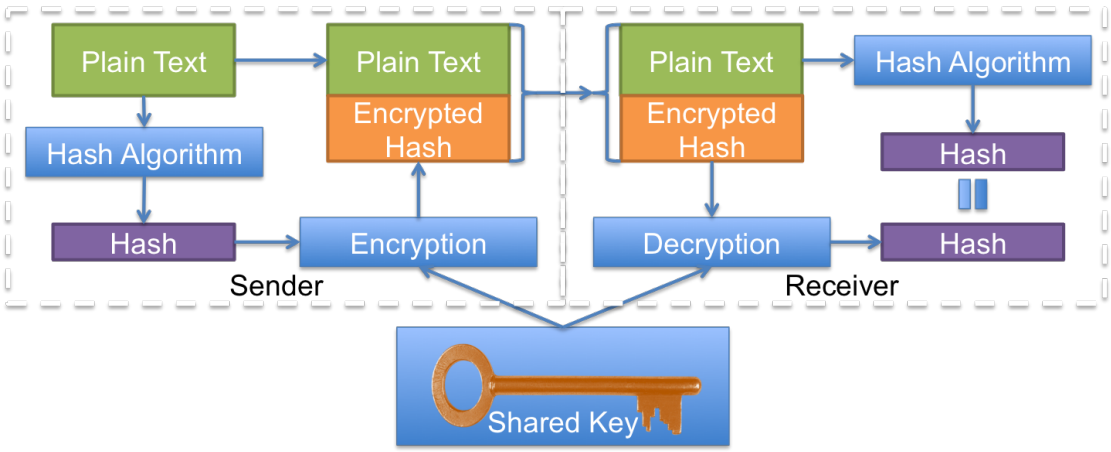


Figure 2.5: Message Authentication Code - Shared Key [43]

Message Authentication Code (MAC) The MAC encrypts the hash with a shared key and ensures that neither the message nor the hash can be altered. This has the advantage that the message can be sent in plain text. The standard way of using the MAC is with a single shared key as shown in figure 2.5. Using this approach the hash algorithm computes the plain text's hash. This computed hash is afterwards encrypted and sent to the receiver together with the plain text. The receiver can decrypt the encrypted hash with the same shared key to obtain the original hash. Furthermore, the receiver can recompute the hash with the same hash algorithm as the sender did. If both ways yield the same hash, the message has not been altered and is therefore valid [43].

Signature Another technique of the same category is using signatures. A digital signature is similar to the above introduced MAC. The only difference is that a public-private key pair is used instead of a shared key to encrypt and decrypt the hash, as shown in figure 2.6. First, the private key of the key pair is used to encrypt the hash that was created from the message being sent. Everyone who is in possession of the public key can check whether the message was sent from the rightful sender and was not altered. This works since no one except for the private key owner is able to modify the encrypted hash that is sent along with the message. Therefore, signing a message means that the plain text can not be altered without notice, but it does not protect against eavesdropping [43].

2.1.4 Key Management

Besides of signing and authentication of messages, public key encryption can also be used for key management and distribution. More precisely, public keys can be used for three purposes [43]:

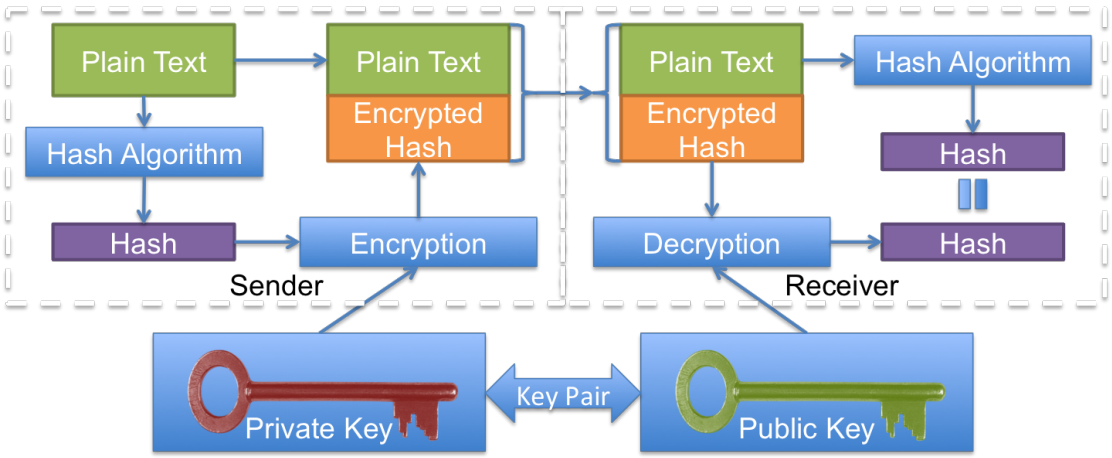


Figure 2.6: Signature [43]

- To securely distribute public keys, hence to ensure that the key came from the rightful owner.
- To utilise public key encryption to distribute secret keys, e.g. to encrypt a secret key with the recipient's public key.
- Or to create temporary keys which are only used once.

Public Key Infrastructure (PKI) The PKI is basically a set of hardware, software, people, policies and procedures needed to create, manage, store, distribute and revoke digital certificates [43]. It controls the distribution of public keys among users and consists of the following parts.

- The **Certification Authority (CA)** issues and revokes the certificates.
- Optionally, a **Registration Authority (RA)** can be responsible for ensuring that a certificate is linked to a certain authority, such as a real person.

Having a central entity that takes care of certificates is especially very important in the financial sector. It provides a central repository where the certificates of all customers are stored and also for access to their public keys.

2.2 Near Field Communication (NFC)

Near Field Communication (NFC) is a technology for contactless exchange of data over a very short distance (up to a couple of centimetres [37]). This allows for transferring data between two mobile phones without the need of plugging in a cable or without an internet connection. A brief introduction about the history and a discussion about technical aspects that are relevant for the purpose of this work are elaborated in this chapter.

History NFC was developed by NXP Semiconductors (formerly Philips Semiconductors) and Sony in 2002. It is based on the well established technology Radio Frequency Identification (RFID) as well as on smart cards [23].

RFID is a technology that allows to automatically recognise and identify RFID-chips. This can be used in a variety of different fields, e.g. for friend-enemy recognition in the military area, to track products and to protect goods in shops so they cannot get stolen [23].

The spread of smart cards started in the early 1950s when the first smart cards were released by Diners Club as well as by Visa and MasterCard shortly after. They placed a magnetic stripe and later a chip on the cards to allow automatic data exchange [23]. With the magnetic stripe and the chip on the card it was possible to identify a user with a simple Personal Identification Number (PIN). An online connection is always required to verify the entered PIN and the data.

To ensure a worldwide standard for NFC, the NFC Forum was founded in 2004[2]. It almost counts 200 members as of 2013[3]. Since this technology is very important for the mobile phone sector, a lot of mobile phone companies are amongst those members, e.g. Google, Samsung, Sony and even Broadcom Corporation or QUALCOMM Inc.

The first field-tests were conducted in 2005. During this test 200 people were provided with an NFC-enabled phone which allowed them to pay in shops and for parking. Only a year later the second test was started in the city Hanau close to Frankfurt in Germany with 100 participants. This test allowed the participants to pay for public transportation tickets with their NFC-enabled phone. After only 10 months the test was extended to a commercial project, which allowed everyone in the city to buy tickets with NFC. The first commercial usage in Austria took place in 2007 where a payment system for public transportation tickets was enrolled with over 1000 NFC tags across the city of Vienna. This system is still in use as of today [23].

NFC has recently got a lot of attention and a boost in popularity for mobile payment applications[4]. Furthermore, the number of NFC-enabled devices has constantly been rising since 2011. It is also expected to continue rising rapidly in the future as shown by figure 2.7 .

[2] `http://www.nfc-forum.org/`, accessed 2013-12-31

[3] `http://www.nfc-forum.org/member_companies/`, accessed 2013-12-31

[4] `http://www.telecomlead.com/smart-phone/nfc-device-shipments-to-grow-118-to-320-million-units-in-2013/`, accessed 2013-12-31

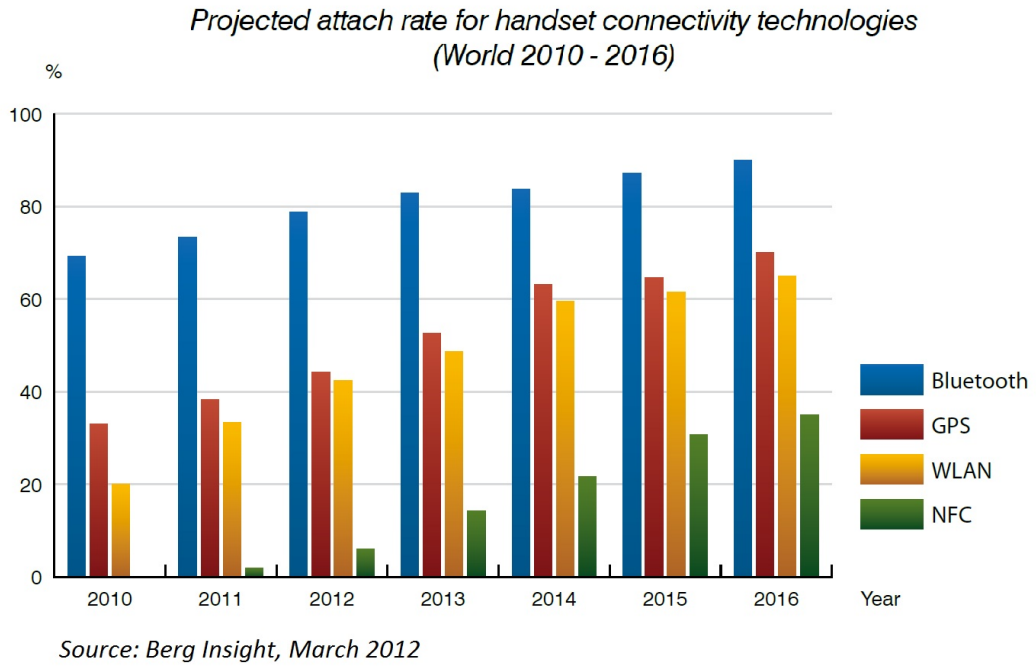


Figure 2.7: Worldwide Phone Connectivity Technologies 2010 - 2016 [47]

Technology NFC on mobile phones basically provides three modes [37] which are shown in figure 2.8.

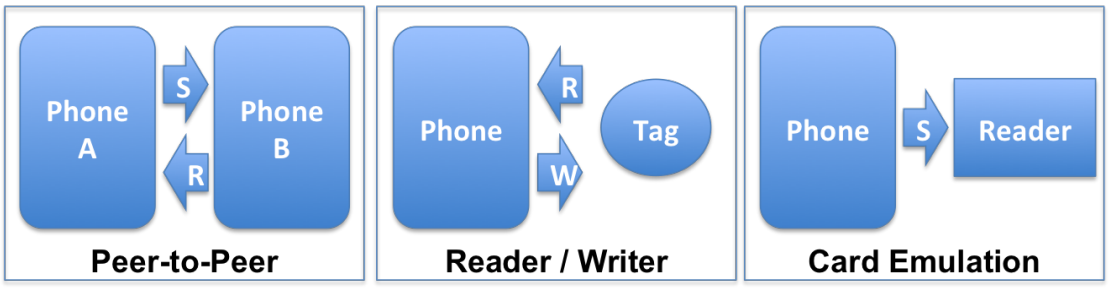


Figure 2.8: NFC Modes

- The **peer-to-peer** mode allows to exchange data between to active devices. Both of the phones can either send (**S**) or receive (**R**) data from the other phone.
- The **reader/writer** mode enables an NFC device to read (**R**) or write (**W**) the content of existing passive RFID and NFC tags.

- The **card emulation mode** allows to emulate NFC cards. Therefore, the phone can communicate with existing NFC readers and send (**S**) data to it. This mode is especially popular for payments with a mobile phone in a shop.

Compared to RFID, NFC solves the problem of only supporting passive tags, because it is also possible to have one active and one passive part (the active reader/device and the passive tag). This works because NFC supports both ways. Therefore, each NFC device is a reader and a tag simultaneously.

The technology was standardised by Ecma (ECMA-340[5] / ECMA-352[6]) and is now an ISO/IEC Standard (ISO/IEC 18092[7] / ISO/IEC 21481[8]). NFC is also backwards compatible with legacy standards [37]. It uses open encryption standards, such as 3-DES, AES and Elliptic Curve, since they are considered to be more secure than proprietary standards [23].

2.3 Existing and Established Solutions

The following chapter takes a closer look at existing and already established solutions on the market. Since from the technical point of view a lot of similar payment systems are available, not all of them are analysed. Therefore, a representative subset was chosen to cover all different types of solutions. The solutions are divided into several groups. The first group contains *online solutions*, such as Bitcoin and PayPal. This group is followed by *mobile solutions* which require a mobile phone, such as Google Wallet. The third group are *offline soultions* which are rather theoretical solutions since they have not been proven to work in practise. The chapter concludes with *other solutions*, hence solutions that do not fit in any other category, such as Quick.
Most of the solutions are based on proprietary protocols. This makes it impossible to investigate how the transactions are carried out in detail. The only exception is Bitcoin. This is why Bitcoin's protocol is analysed in more detail. Therefore, it is necessary to make assumptions for all other solutions and draw conclusions from literature.

Furthermore, a simple framework is introduced to make the solutions comparable. This framework takes a set of attributes and requirements for payment systems into account to make them easily comparable.

2.3.1 Online Solutions

The first section covers all products that are mostly used for online transactions. One of the main facts that distinguish them from other solutions is that they always require an active internet connection and are mostly used on a PC.

[5] `http://www.ecma-international.org/publications/standards/Ecma-340.htm`, accessed 2013-12-31

[6] `http://www.ecma-international.org/publications/standards/Ecma-352.htm`, accessed 2013-12-31

[7] `http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=56692`, accessed 2013-12-31

[8] `http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=56855`, accessed 2013-12-31

2.3.1.1 Bitcoin (BTC)

Bitcoin[9] is a decentralised digital currency that allows anonymous electronic transaction. It uses digitally signed coins that are sent through the internet. Instead of a central bank a peer to peer network is used, hence everyone who uses Bitcoin is part of the bank. It is based on the following principles[10]:

- **Decentralised** Bitcoins are transferred directly from person to person and do not need a third party like a bank or a clearing house to conduct a payment. Each person can have multiple personal Bitcoin addresses which are combined to a wallet. A transaction is stored in a block which contains multiple transactions. This block is used, based on a peer-to-peer proof-of-work system, to validate a transaction and to provide security and to prevent double-spending [31]. This proof-of-work system uses a public history of transactions. A so called block chain, as shown in figure 2.9, is used to implement the history. The coins that are created and exchanged do not exist in form of physical coins, but rather are virtual coins which are proven through their history in the block chain.

- **Low Fees** Since no third party is involved, the fees are much lower compared to other currencies or payment systems.

- **Worldwide** Bitcoin can be used everywhere in the world, hence it can be used regardless of country borders.

- **No Frozen Accounts** A user account cannot be disabled by anyone. As long as the user has a Bitcoin address he or she can send and receive money.

- **No Prerequisites** Everyone can use Bitcoin and there are no prerequisites to use it except for a freely available client.

- **Open Source** Most Bitcoin clients are open source and nobody owns it, which helps ensuring that there are no backdoors and no bugs in the software.

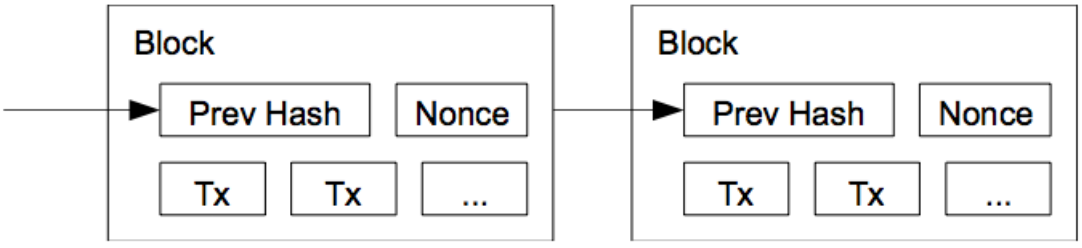


Figure 2.9: Bitcoin Proof-of-Work [31]

[9] http://bitcoin.org/en/, accessed 2014-01-07
[10] http://www.weusecoins.com/en/, accessed 2014-01-07

Block Chains In order to validate and store transactions that have been made, block chains are used. A block chain is a chain of independent blocks, where each block is connected through the previous block's hash as shown in figure 2.9. The connection of the block's hashes leads to a chain. Each block in a chain contains multiple transactions conducted by different users and the whole block chain contains all transactions ever made.

As shown in figure 2.9, each block contains the following information:

- **Prev Hash** This is the SHA-256 hash code from the previous block which is used to link the blocks together.
- **Tx** Each block contains a set of transactions that were made. How a transaction is conducted is discussed later.
- **Nonce** The nonce is a 32-bit number that starts at zero and is increased until the SHA-256 hash value of the whole block starts with a certain amount of leading zero bits (e.g. `000000000000000085ddc1aa2a8296....`). This proofs that a certain amount of computing time has been invested in validating the block, since each time the nonce is increased, the hash code needs to be recomputed. This task needs to be done over and over again until an appropriate hash code is found. When a hash code with the desired number of leading zeros is found, the block is permanently added to the block chain.

This validation process is done by so called *Bitcoin miners*. They are responsible for creating new bitcoins and for validating transactions. The sender has to pay the miner a certain amount of bitcoins as an incentive to conduct the validation.

The created block chains are used to provide a history of transaction, but also to prevent hacking and altering of transactions. If any of the blocks in the block chain is altered, the whole chain after the changed block needs to be recomputed, which would require a lot of computational power. This is described in more detail later.

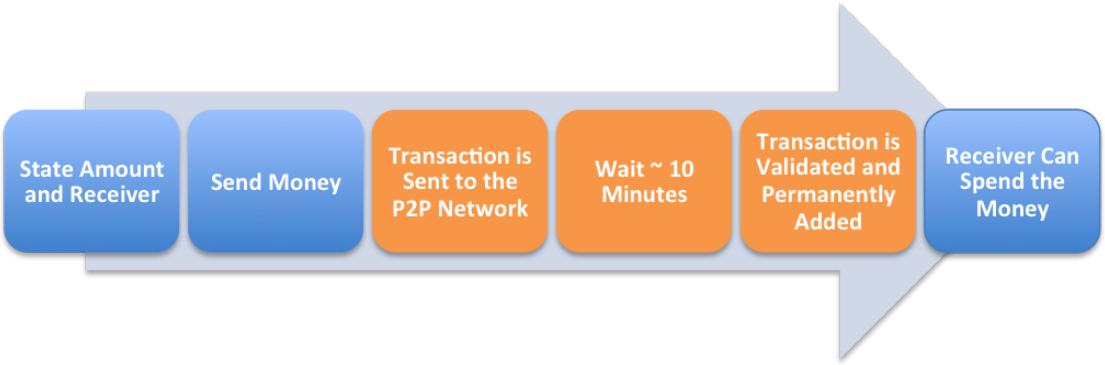


Figure 2.10: Bitcoin User Workflow

User's Point of View From the user's perspective there are a couple of steps that need to be done to be able to use Bitcoin. Once the user has installed one of the various Bitcoin clients[11], which are available for all important operating systems, a unique *Bitcoin address* is generated. Optionally, an existing address can be imported instead. This address is a unique identifier and is used to send and receive money. The workflow for the user to send money is shown in figure 2.10.

In order to conduct a payment the receiver's address as well as the amount needs to be stated. This transaction is signed with an electronic signature. Afterwards the transaction is sent to the P2P network and as soon as it is verified by a Bitcoin miner the money is added to the block chain and available to the receiver and permanently and anonymously stored in the decentralised network. On average, once every 10 minutes a new block is generated but the sender and the receiver do not need to be online for the entire time [41].

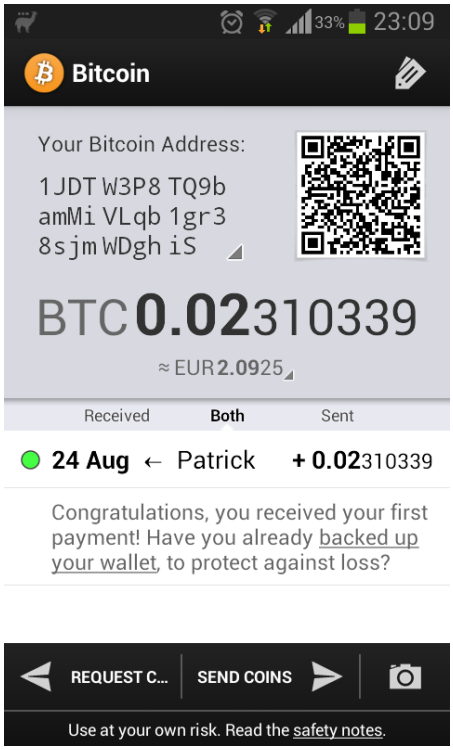


Figure 2.11: Bitcoin Android App

One can increase the probability for faster verification by providing a higher transaction fee, since the miner who solves the challenge will get the entire fee of all transactions in the block [41]. Figure 2.11 shows the Bitcoin address in the upper part of the screen of the Android App Bitcoin[12]. The process is rather self explaining for the user. One can choose between requesting and sending coins by stating the Bitcoin address of the transaction partner. The lower part of the screen shows a successful transaction with a small amount of money that was transferred.

Creating Bitcoins Unlike paper money, where the government decides when to print new money, an application called *Bitcoin miner* is used to create new bitcoins. This application can be freely used by anyone. Everyone who participates in the creation process is rewarded with bitcoins as an incentive to mine bitcoins and there is no central bank that issues new money. The mining process is based on a proof-of-work challenge that requires a certain amount of computational work for each coin as discussed before. This is a simple way to increase the amount of bitcoins available and creates an incentive for more and more users to mine bitcoins. The amount of work needed to create a new coin is adjusted automatically by the network. It takes the increasing CPU power of PCs into account in order to keep the amount of bitcoins

[11] `http://bitcoin.org/en/choose-your-wallet`, accessed 2014-01-07

[12] `https://play.google.com/store/apps/details?id=de.schildbach.wallet`, accessed 2014-01-07

created always at a **predictable** rate. According to [41] the number of bitcoins is **strictly limited to 21 million** and the reward for creating new bitcoins is halved roughly every four years.

Bitcoin Exchange As it has been described, mining bitcoins needs a certain amount of knowledge and computational power. Since Bitcoin has become more and more popular, people started to buy servers and design hardware with the only purpose to mine bitcoins[13]. This made it almost impossible for a regular user to mine a decent amount of bitcoins with standard hardware. Therefore, it is also possible to trade bitcoins against real money in different currencies. Trading bitcoins can be compared to a stock exchange. If one wants to buy bitcoins the user has to state the amount as well as a price he or she is willing to pay for the bitcoins. If the offer corresponds to a sell order by someone else, the exchange is executed. In order to conduct such exchanges different exchange platforms can be used. Mt. Gox is the most popular Bitcoin exchange at the moment[14], but there are many other websites that offer similar services. To use Mt. Gox one has to create an account and transfer money in a currency of choice to that account. This amount can later be used to trade it against bitcoins. Trading bitcoins for real money works the other way around.

Practicality and Popularity Bitcoin has gained a lot of popularity in the last couple of years as the increasing Bitcoin exchange rate on Mt. Gox Live shows[15]. Bitcoin is especially popular when anonymity is important. One case that recently made it into news was the shutdown and the seizure of Silk Road, an illegal market place for drugs that was closed by the FBI[16]. Ross Ulbricht, the founder of the platform, is thought to be holding around 600.000 bitcoins, which is about 5% of the total amount of available bitcoins[17].

The fact that it is distributed and there is no central authority, makes it very practical for anonymous transfers and for international payments. The amount of time needed for a transaction (around 10 minutes) always stays the same, independent from the distance [41].

Since the protocol and the implementation are publicly available, it is relatively easy to develop clients for every operating system, regardless if it is a mobile phone or a PC. Recently, the first internet shops started accepting bitcoins[18]. Furthermore, the first Bitcoin ATM was delivered[19], which allows for anonymous transfer of real money in exchange for bitcoins.

Bitcoin Transaction Transactions are stored in a block chain. Each block consists of multiple transactions (see figure 2.9) and each transaction consists of multiple inputs and outputs (see figure 2.13).

[13] `http://www.weusecoins.com/en/mining-guide`, accessed 2014-01-07

[14] `https://www.mtgox.com/`, accessed 2014-01-07

[15] `http://mtgoxlive.com/orders`, accessed 2014-01-08

[16] `http://gu.com/p/3jbag`, accessed 2014-01-07

[17] about €61 million as of 2013-10-09

[18] e.g. `https://www.spendbitcoins.com/places/`, accessed 2013-10-13

[19] `http://www.coindesk.com/bitcoin-atm-to-be-demonstrated-at-bitcoin-london/`, accessed 2013-10-13

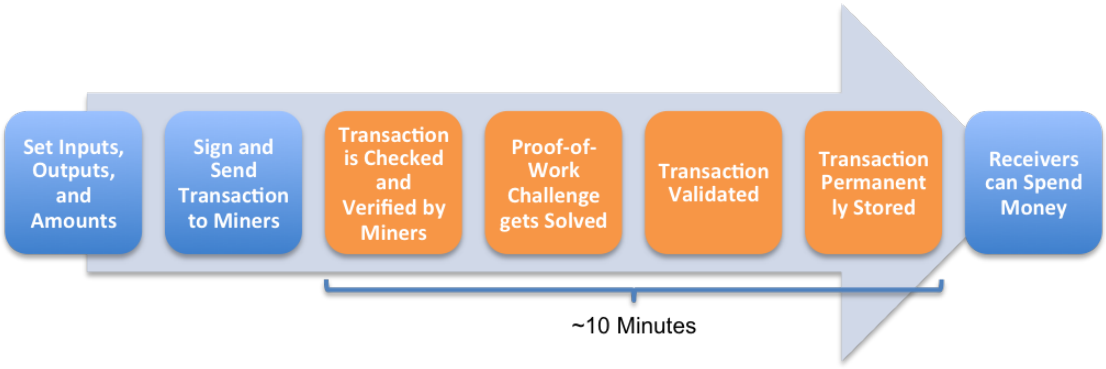


Figure 2.12: Bitcoin Transaction Process

According to [41] a transaction works the following way. It is assumed that Alice wants to send 50 bitcoins to Bob. She previously received 30 bitcoins from Charlie, 15 bitcoins from Dan and 10 bitcoins from Erin during three different transactions. Therefore, she has a total of 55 bitcoins in her wallet. When conducting a transaction, the sender needs to make sure that the amount of inputs is higher than the amount that wants to be sent. Figure 2.13 shows the actual transaction. Furthermore, the whole transaction process is also shown in figure 2.12.

$$\left(\underbrace{\begin{pmatrix} txHash_{CA} & txOutIndex_{CA} & sig_{CA} \\ txHash_{DA} & txOutIndex_{DA} & sig_{DA} \\ txHash_{EA} & txOutIndex_{EA} & sig_{EA} \end{pmatrix}}_{\text{Inputs}}, \underbrace{\begin{pmatrix} pk_B & 50 \\ pk_A & 3 \end{pmatrix}}_{\text{Outputs}}, \ldots \right)$$

Figure 2.13: Bitcoin Transaction [41]

The three inputs are proof that enough money is available. For each input a reference to the previous transaction is stored. This reference consists of the transaction's hash (**txHash**) and the transaction's index (**txOutIndex**) as well as a signature (**sig**) that verifies that Alice is allowed to spend the money.

On the right part of the transaction two outputs can be seen. These entries state where the money of this transaction should go to. Each line shows the amount as well as the public key of the receiver (**pk**). Bob receives his 50 bitcoins and Alice will receive three bitcoins in return. The remaining two bitcoins are the transaction fee.

Afterwards, the entire transaction is signed and sent to all Bitcoin miners that the user is aware of, as figure 2.12 shows. As soon as the first miner has included this transaction in a block by solving the proof-of-work challenge, the transaction is validated and all other miners informed that the transaction has been included in a block. The miner who successfully added

the transaction to a block also receives the whole transaction fee.

A newly mined Bitcoin is a special type of transaction with no inputs and the current reward as well as the miner's Bitcoin address as an output.

Double-Spending and Proof-of-work A well known problem when it comes to digital money is double-spending. This means that a coin is spent more than once. When using Bitcoin this would only work for the very first moment, because as soon as a transaction is included in a block it cannot be spent again [41]. This works because each transaction is checked and verified by the block creator as well as by everyone who receives the block. Furthermore, the block is sent to all Bitcoin users after it has been computed. Therefore, everyone is aware that a transaction has been included in a block, hence that the money has been used and cannot be spent again. Editing old blocks would not work either, because this would destroy the hash and all blocks would have to be recomputed, which would take a big amount of time [41].

Anonymity When talking about the anonymity of Bitcoin two things are important. First, the user does not need to create a validated account, but just needs a Bitcoin address to be able to transfer money. Second, the whole transaction history is publicly available in form of the block chain. The first fact increases the anonymity, the second decreases it. As it has been shown, even though the transaction data is anonymous, anonymity can be attacked with different but related datasets, e.g. by sniffing network traffic and finding patterns [34].

The only thing that can be seen by analysing the transaction data is that a specific amount was transferred from person A to person B. However, this is not entirely true, since multi-input transactions can create links between transactions [31]. According to [34] the following factors have been identified that can be used to compromise anonymity.

When exchanging bitcoins with real money, sometimes identifying information, such as an e-mail address or credit card information, is mandatory. This information can be used to link transactions to users. Apart from these factors, information looking at the context (e.g. a reported Bitcoin theft) can be gained and used to identify users.

Conclusion Bitcoin clearly has potential to be accepted by the broader public for online money exchange. This is especially because of its independence from centralised banks and country borders. These facts give Bitcoin a lot of possibilities regarding international money transfer and also regarding anonymity. On the other hand, this system still has some issues especially in terms of simplicity and trustworthiness. For example, it may be difficult to understand for a non-tech person how this system works and why it is secure.

Moreover, a couple of Bitcoin forks have been developed in the last couple of years [41]. Those include Litecoin (LTC), Peercoin (PPC) and Primecoin (XPM).

According to [41] there are a lot of turbulances on the digital currencies market at the time of writing. Even though a lot of new currencies have been released, they only differ slightly from Bitcoin and there hasn't been a real innovation lately.

2.3.1.2 PayPal

PayPal was founded in 1998 and is another solution which allows to easily transfer money between two parties [11]. Money can be transferred by simply entering the e-mail addresses of the recipient and the amount that should be sent. Even if the receiver does not have a PayPal account, he or she can receive money by creating an account with the e-mail address the money was directed to. The fact that receiving money is rather easy was also the reason that PayPal grew so quickly - almost virally [11]. The customer can choose whether he or she likes to keep the money on the PayPal account or transfer it to a bank account. Another feature of PayPal is that apart from the e-mail address no further contact or bank account details are exchanged. Therefore, anonymity and security is ensured as stated on their website[20]. However, anonymity is only ensured between the users since PayPal stores contact information and bank account details to process payments.

The business model is also rather simple. Sending money is free to allow customers to easily buy goods and don't have to pay additional fees for it. However, when receiving money charges are applied. They vary from country to country (e.g. 3,4% + €0,35 for each transaction in Austria[21]). Because of its ease of use PayPal has become the most successful online payment system serving 110 million customers all over the world[22]. This huge success inspired eBay, which is the world's biggest auction website[23], to buy PayPal in 2002. This was done to help in providing a simple way to pay for auctioned goods. PayPal also provides an app for Android and iOS (see figure 2.14) to exchange money among friends and to pay for goods. This requires an active PayPal account, the e-mail address of the receiver as well as an internet connection to carry out the transfer. They also provide in-store payments with the provided apps [24]. Therefore, one can pay in stores or at the table in a restaurant.

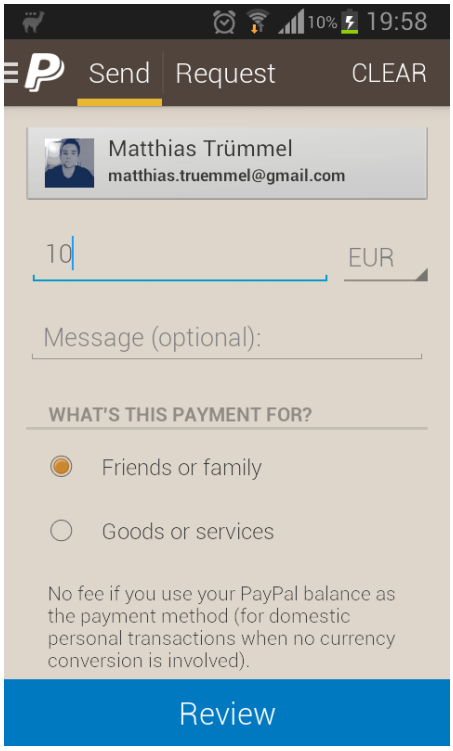


Figure 2.14: PayPal Transaction

[20] https://www.paypal.com/, accessed 2013-12-19
[21] https://www.paypal.com/at/webapps/mpp/paypal-fees, accessed 2013-12-19
[22] https://www.paypal.com/webapps/mpp/ent-online-attract-shoppers, accessed 2013-12-19
[23] http://www.ebay.com/, accessed 2013-12-19
[24] https://www.paypal.com/uk/webapps/mpp/use-paypal-in-stores, accessed 2013-12-19

PayPal may be easy to use to pay for goods.
However, when it comes to customer-to-customer payments with the mobile app, the service still lacks in usability, because a lot of information needs to be entered manually, e.g. the receiver's e-mail address. Furthermore, it only works when an internet connection is available.

2.3.1.3 Paysafecard

The Paysafecard works with the pre-paid principle (compare to chapter 1.6.2). One can buy so called *Paysafecards* at local sales outlets and choose between €10, €25, €50, or €100[25] . Each card contains a 16-digits PIN which has to be entered each time to pay for goods and services. The user can use each card for as many transactions as he or she wants as long as it has a positive balance. The current balance for each card can be checked online. However, if there is still money on a card after 12 months, €2 will be subtracted as a fee each month. Paysafecard also provides *my paysafecard*[26] where multiple Paysafecards can be stored and used. The maximum amount per payment is €1.000 and the maximum balance depends on the status of one's account (either €2.500 or €5.000). Furthermore, there is also an app available which allows for finding points of sale and checking the balance. Customer-to-customer transfers are not possible.

One of the advantages of this system is that if a card is lost, only the remaining amount on the card is lost. This implies that the PIN on the card is not saved somewhere else. Therefore, the risk is very limited. Another plus of this system is its anonymity. Since Paysafecards need to be bought in a shop, anonymity is ensured and the card cannot be linked to a real person. However, this is also one of the downsides, because one can't easily buy Paysafecards online and print them out. Instead one needs to go to a shop to buy them.

2.3.1.4 Bank Transfer and Netbanking

Sending money to friends and to other people via bank transfer is very common [29]. This method requires to exchange name, the International Bank Account Number (IBAN) and the Business Identifier Codes (BIC) upfront to know to which bank account the money should be sent. Furthermore, in the last couple of years banks all around the world have picked up on the mobile phone trend and started to release apps for the most common smartphone platforms. This allows customers to check their bank account balance and start transactions with their mobile phone. For example the Austrian bank *Erste Bank* implemented a feature for their iPhone app[27] to exchange money between two iPhones. This feature requires both users to enable Bluetooth, log onto the app and choose the *iPhone 2 iPhone* option. This wirelessly exchanges the bank account data between the two parties. However, this is limited to a few people, because one has to have a bank account at *Erste Bank*, an iPhone and an internet connection to carry out the transaction.

[25] `https://www.paysafecard.com/`, accessed 2013-12-19

[26] `https://www.paysafecard.com/de-at/produkte/my-paysafecard/`, accessed 2013-12-19

[27] `https://itunes.apple.com/at/app/erste-bank-sparkasse-osterreich/id437840915`, accessed 2013-11-30

2.3.2 Mobile Solutions

This chapter deals with different mobile solutions. This type of solution requires applications to be installed on mobile phones. Furthermore, an active internet connection is needed to execute a transaction.

2.3.2.1 Google Wallet

Google Wallet is a service and an app by Google, which allows for paying with NFC-enabled Android devices. It can be used to pay in shops, to send money to friends in the United States and to buy goods online [12]. Furthermore, it also allows for adding loyalty programs and making use of special offers. The phone owners can add their credit or debit card information on the phone, take a picture of the card and store it in the app for later usage. Google's big advantage is that it allows Android *tap and pay* users within the United States to pay everywhere where Visa payWave and Mastercard payPass[28] (see chapter 2.3.4.1) are accepted. This is done by putting the phone instead of the card to the contactless terminal. *Tap and pay*[29] is an easy way to pay at shops that allow contactless payments. It works on devices that support Android KitKat (Version 4.4) and above. Google Wallet does not need an online connection for B2C transaction, since the phone simply emulates the NFC card. However, the merchant's terminal is connected to the internet to transfer the payment information.

Google stores all credit and debit card information on the phone [12]. Therefore, the user does not need to carry bank cards any more. Being able to pay your friends with Google Wallet is only available in the US [12]. Further, to send a payment to a friend an internet connection is mandatory[30]. Moreover, only a few mobile phones[31] and mobile phone carriers[31] are supported at the moment, hence the usefulness is limited.

2.3.2.2 Erste Bank's Friendbank

Since Erste Bank claims to provide the *most modern bank account in Austria*[32], one of their products is analysed in this section. The bank provides a service called *Friendbank*, which allows the customers to track payments which is lent or owed to friends. This helps to keep track of money transfers between friends. This may sound like a good idea at first, but someone may argue that it is nothing more than a notepad. Therefore, the user still needs to make sure that the bills are cleared correctly and the information in the app is up to date.

[28] `http://www.engadget.com/2011/09/20/polyamorous-google-wallet-adds-visa-to-its-arsenal/`, accessed 2013-12-10

[29] `https://support.google.com/wallet/answer/2466137?hl=en`, accessed 2013-12-10

[30] `http://www.google.com/wallet/send-money/`, accessed 2014-01-01

[31] `https://support.google.com/wallet/answer/1347934?hl=en`, accessed 2014-01-01

[32] `http://www.sparkasse.at/erstebank/Innovations-Zone`, accessed 2014-01-01

2.3.2.3 Venmo

Venmo is another mobile payment service which is only available in the United States[33]. It provides an app for iPhone and Android. Users can send money online to other Venmo users by stating either the receivers phone number, the e-mail address, or choosing the Facebook contact. Receiving money is free while sending is only free when the amount that needs to be sent is part of the balance on Venmo or is transferred from a debit card. For topping up an account with a credit card, a 3% fee is charged. The sender can either choose to pay or to receive money. Either way, a recipient, the amount and a message needs to be entered. Optionally, the location and privacy of the transaction can be set. This allows for making a transaction either private or visible to everyone.

2.3.2.4 Square

Square is another company providing different types of payment services. Figure 2.15 shows their main products, which can be used independently. *Square Wallet* allows customer to pay in shops by saying their name and *Square Register* is a replacement for the regular cash register. Their newest product *Square Cash* allows for sending money via e-mail.

Figure 2.15: Square Products

Square Wallet The iPhone app **Square Wallet** is yet another tool which should help to make the payment process more seamless[34]. The app offers the possibility to search for places nearby that support paying with Square. One has to link the account with a credit card and upload a photo prior to the first usage. The payment process itself can be completed in a few steps, as shown in figure 2.16.

Figure 2.16: Square Wallet Payment Process

[33] `https://venmo.com/info/product`, accessed 2013-12-03
[34] `https://squareup.com/wallet`, accessed 2014-01-01

Once a store is found, the user can either check in with the app manually (**1**), or automate that process with the phone's GPS position for places that are visited on a regular basis. As next step the user has to say his or her name at the cash register (**2**). Then the employee checks whether the customer's name appears on the cash register's list. If the name is on the list it is also checked whether the picture matches (**3**). As soon as the waiter has confirmed that the customer is who he or she claims to be the payment is approved and the checkout is completed (**4**).

Square Register Square also provides a cash register replacement called **Square Register**. This product can be used by companies to sell items in stores. The store can either use an iPad, iPhone or Android phone to run the software[35]. Further, to make the process easier, the customer can swipe his or her card through the *Square Reader* to carry out the payment. The reader is plugged into the phone's audio jack to simplify the process.

Square Cash Another service by Square was released in October 2013 and is called **Square Cash** [36]. It allows for sending money to anyone via e-mail. The process is shown in figure 2.17.

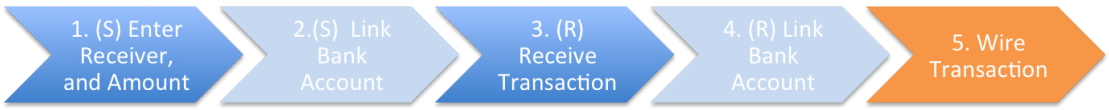


Figure 2.17: Square Cash Payment Process

First, the sender (**S**) has to put the receiver's e-mail address in the recipient field. Further, a copy of the e-mail (CC) needs to be sent to Square and the amount needs to be stated in the subject line. Optionally, the e-mail body can be used for a message (**1**). This is illustrated in figure 2.18. When the sender uses Square Cash for the first time, debit card information needs to be entered (**2**). The receiver (**R**) will get an e-mail with a link where he or she can accept the money (**3**). If the receiver hasn't used the service before, he or she also needs to enter his or her debit card information (**4**). If the payment is accepted by the receiver, Square wires the transaction correctly (**5**). As of now Square Cash allows transactions with up to $2.500 per week [37].

According to Square's website they have an advanced auditing system, they check every transaction for fraud and they also work together with banks to transfer back the money if it was transferred unrightfully. At the moment Square is only available in the United States, Canada and

[35] `https://squareup.com/features`, accessed 2014-01-01
[36] `https://square.com/cash`, accessed 2014-01-01
[37] `http://www.youtube.com/watch?v=PTqxMBwnieo`, accessed 2013-11-27

Japan. As of 2012 they teamed up with Starbucks[38] to allow customers to pay in 7.000 stores. Starbucks also invested $25 million in the startup. The business model is easy and similar to PayPal. They charge 2,75 % per swipe or online payment (as of October 2013).

Conclusion All of the products mentioned above have a clear focus on usability and disregard security. Especially *Square Cash* provides a seamless and easy way to transfer money between customers. However, they don't provide any security features at all. This allows everyone who has access to an e-mail address which has square cash enabled to transfer money to someone else. This has also been discussed in the news[39]. Moreover, *Square Wallet* has it's problems when it comes to automatically checking in at a store, as a waiter could either unintentionally or purposely bill the wrong customer without notice.

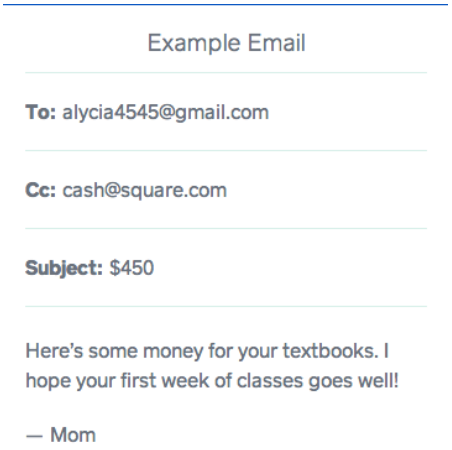


Figure 2.18: Square Cash E-Mail [42]

2.3.3 Offline Solutions

Offline solutions, which do not require an active internet connection to carry out the transaction of money, are discussed in the following section. The solutions discussed are only theoretical experiments described in papers and have never been tried out in practise.

2.3.3.1 Offline Micropayments

In the last years a couple of different offline payment systems have been invented [5, 22, 26] but neither of those systems has made a mentionable impact. Therefore, the following two approaches were chosen to give an idea about how offline payment systems work.

Blaze et. al [5] introduced an offline payment system with an emphasis on risk management. Hence, they allow offline transaction where fraud is unlikely to happen and the costs for backing a transaction with an online check is more expensive than the transaction value. They also proposed to use a cellular phone or PDA device to manage the user's data. Furthermore, a regular communication with the issuer is necessary to update the credentials that are only valid for certain amount of time. This system only works between a payer and a merchant.

The big plus of this method is that the credentials can contain certain encoded information. For example, this could be information on a person's driving license and his or her age. Since

[38] `http://www.nytimes.com/2012/08/08/technology/starbucks-and-square-to-team-up.html`, accessed 2013-11-27

[39] `http://venturebeat.com/2013/10/16/square-cash-lets-you-send-money-over-e-mail-cool-but-who-will-trust-it/`, accessed 2013-01-01

this is only a theoretical payment system, the authors did not describe who would accredit this information. On the downside a lot of manual steps are required to perform this type of transaction, because a so called *offer* needs to be sent to the payer in the beginning. This offer is signed by the payer with a *KeyNote Micro check*. The merchant then sends the micro check to a local *KeyNote compliance checker* that can authorise the transaction. The transaction can later be settled with the so called *Clearing and Settlement Center (CSC)*.

The most important part for the author is the encoded limit in credits, so that thieves would only have a limited use of the system. Therefore, no secure hardware or online transaction authorisation is used and losses are accepted instead.

H.-T. Liaw et al. [26] proposed a similar system called the *electronic traveler's check*. This check has similar properties as electronic cash [39] and the electronic check [3]. First, the customer needs to buy a traveler's check from the bank which can be used for transactions at a later point of time. There are two options available to use it.

The first option is the so called online scheme. It requires an internet connection and the merchant can instantly check the identity of the user.

The other one is the offline scheme. It does not include the bank and the clearing house at first. The customer simply sends the merchant a digital check containing the traveler's check, the face value (amount), a serial number and the timestamp encrypted with the merchant's public key. The check was obtained from the bank in an earlier stage. Either instantly or in a later phase the merchant sends the check to the bank to obtain the money. The bank checks for double-spending and whether the check is valid. If no fraud is detected, the amount is transferred to the merchant.

Even though lost or stolen traveler's checks can be reported, the check can still be used until it is successfully reported at the bank. When the offline scheme is used, the check can be used until the merchant deposits the payments that he or she got at the bank.

2.3.4 Other Solutions

Some of the solutions that are used on the market or have been released recently do not belong to the categories above. Therefore, another category which should address these solutions is introduced.

2.3.4.1 NFC-enabled Debit and Credit Cards

Debit and credit card companies have started to roll out NFC-enabled cards in 2011 [23]. Visa released the product *Visa payWave* and MasterCard the product *MasterCard PayPass*. They allow to pay for goods in shops for up to €25 without entering a PIN or a signature[40]. Therefore, the payment process only takes a couple of seconds. After five times a PIN needs to be entered again[41]. It is also necessary to enter a PIN if one wants to pay higher amounts.

[40] `http://www.cardcomplete.com/complete-karten/services/kontaktloses-bezahlen/`, accessed 2013-11-29

[41] `http://derstandard.at/1388650479190/Ueberweisungstrick-NFC-Bankomatkarten-lassen-sich-hacken`, accessed 2014-01-01

Even if this is a good example of how NFC can be used to create an advantage for the user, it also has some drawbacks. Since NFC works on a distance of up to a couple of centimeters [23], the card reader only needs to get close to the device without the need for physical contact. In areas where this happens a lot (e.g. in the metro or an elevator) it is relatively easy for thieves to get as close to the card as needed. This can be abused to charge cards with up to €25 at any store without any need for confirmation[42]. This can be repeated up to five times in row before the PIN needs to be entered again, hence the maximum damage could be €125[43]. Moreover, it was shown at the Defcon Hacker Conference that the data from a NFC-enabled card can easily be stolen by reading the data on it with a NFC-enabled mobile phone by using the app NFCProxy[44]. Visa and MasterCard said that the customers should not worry because they use multiple security layers and fraud detection to protect customers if credit card data is abused[45].

2.3.4.2 Quick

The smart card, as it was described in chapter 2.2, was used to create the first electronic wallet called Quick in Austria in 1995. Therefore, Austria was the first country with a countrywide electronic wallet system [23].

Quick is used to perform offline transactions with rather small amounts of money. Since this is an offline system, losing a smart card containing money results in the fact that the money cannot be restored. The smart card can be topped up at every ATM machine in Austria with up to €400 by authorising it with a PIN[46]. When the money is loaded onto the card, it is also stored on a *central clearing host* [27]. When one pays with Quick at a merchant's terminal, the amount is stored on the merchant's card, which collects and stores all transactions. The transactions are processed by the central clearing host on a regular basis and the merchant gets the money transferred from the clearing host [27].

Quick can also be used via NFC. Therefore, the same attacks as for NFC-enabled credit or debit cards can be applied. Also, the data and transaction details from the card can easily be read with a standard smartphone that supports NFC, as it has been shown[47]. This has also been tried during writing of this thesis. It was possible to retrieve the balance of the card as well as a list of previously conducted transactions within a matter of seconds.

---

[42] `http://www.visaeurope.at/at/newgs/neue_zahlungsm%C3%B6glichkeiten/kontaktlose_zahlungen`, accessed 2014-01-01

[43] `http://derstandard.at/1388650479190/Ueberweisungstrick-NFC-Bankomatkarten-lassen-sich-hacken`, accessed 2014-01-14

[44] `http://www.forbes.com/sites/andygreenberg/2012/07/27/hacker-demos-android-app-that-can-read-and-use-a-credit-card-thats-still-in-your-wallet/`, accessed 2014-01-01

[45] `http://www.cbc.ca/news/canada/newfoundland-labrador/credit-card-data-can-be-stolen-with-a-wave-and-an-app-1.1386262`, accessed 2014-01-01

[46] `http://www.quick.at/`, accessed 2014-01-08

[47] `http://derstandard.at/1388650296717/Smartphone-App-liest-Bankomatdaten-aus`, accessed 2014-01-08

According to [29], Quick is used for 0,16% of all transactions and for 0,06% of the transaction volume in Austria. This also shows that this service is mostly used for small amounts.

2.4 Comparison and Evaluation of Existing Approaches

Having introduced the most important payment systems on the market, a comparison between them is made. This helps to understand the context, their similarities and differences as well as their advantages.

2.4.1 Evaluation Framework

In order to make payment systems comparable, a set of attributes based on [4, 24, 33] was identified. The defined attributes, which are shown in figure 2.19, represent requirements that should be fulfilled from a very general point of view. The attributes in orange circles represent requirements for the system itself and the attribute in a blue circle is the demand on a payment system that a typical user has. For simplicity, only four main attributes were chosen. Some of those four attributes contain sub attributes, which are used to refine the main attributes.

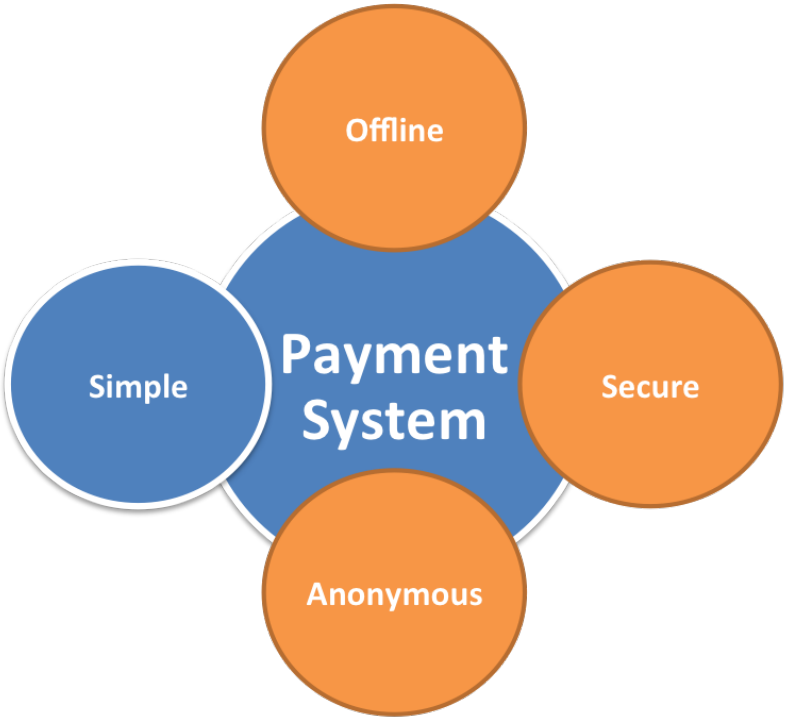


Figure 2.19: Payment System Attributes

- **Offline** Being able to use a payment system offline helps in making it more easy to use and also to increase **availability**. Thus, a payment system that is only useable with an

internet connection narrows down the area where it can be used (see chapter 1.2).

  - **Availability** The availability is one of the sub categories in terms of offline availability, since a 100% availability can only be given when a payment system can also be used offline without the need for a connection to a server.

- **Secure** One of the most important requirements for a payment system is its security. If it does not provide sufficient security features it will not be used [24]. This attribute is especially important when it comes to money related products or software. A secure payment system needs to fulfil the following requirements:

  - **Double-Spending** A secure payment system does not allow double-spending, hence it is not possible to spend one money unit more than once without notice.
  - **Validation** The parties of the payment system must be legitimate users of the system, which needs to be identifiable by both parties in order to prevent fraud.
  - **Fraud Detection and Prevention** Fraud should not be possible in any way. If it is not possible to prevent fraud, measures against it need to be stated and should not be disadvantageous for the user (e.g. risk management).
  - **Non-Repudiation** Once a payment is conducted, the participants may not be able to deny the commitments they have made, hence it needs to comply with non-repudiation. This is especially important for the receiver to claim his or her money.
  - **Atomic Transaction** A conducted transaction needs to be carried out either to its full extent or not at all.
  - **Integrity** The system must ensure that no money can be taken from the user without him or her knowing.
  - **Confidentiality** No one except for the participating parties can gain knowledge about the transaction details.

- **Anonymous** Users don't always want to be traceable so that everyone knows what they are up to. This has become very important since the recent NSA surveillance scandal [48]. On the other hand, anonymity also has its downsides, because as [22] states anonymous payment systems can easily be used for illegal payments or money laundering.

- **Simple** It is important to the user that a payment system stays as easy to use as possible, because if the entry and usage barrier is too high it may not be used. A simple payment system needs to fulfil the following requirements:

  - **Easy to Use** This sub attribute is the most important one for the user. If it is too much effort for the user to use the payment system it may not be used and will therefore never reach a critical mass.

[48] `http://www.independent.co.uk/life-style/gadgets-and-tech/news/it-firms-lose-billions-after-nsa-scandal-exposed-by-whistleblower-edward-snowden-9028599.html`, accessed 2014-01-04

– **Self-Explanatory** A user does not want to read a manual before he or she is able to use the payment system. This is why it also needs to be self-explanatory.

– **Low Entry Barrier** To get users to use a payment system it needs to have a low entry barrier.

The main problem is that a payment system is not able to fulfil all of those requirements. It is only possible to fulfil a maximum of three out of those four attributes, because especially when it comes to the system attributes (offline, secure, anonymous) conflicts exist. Offline and secure do not go along with an anonymous transaction, because there is no way to prevent double spending or a hacked software without an association to a person who can be blamed for double spending. This is also the reason why an completely offline and anonymous system can never be secure. Moreover, a system can only be secure and anonymous when not used offline as Bitcoin shows [31].

Even though certain approaches claim to fulfil all those attributes [21, 28], none of them has reached a critical mass or even got past a theoretical stage. Some of them claim to only fulfil those requirements as long as the software they need will not be hacked [1]. Therefore, it is impossible to use them in practise since it could only be used by honest users who do not double spend [6].

2.4.2 Applied Evaluation Framework

The framework applied on the discussed payment systems can be seen in table 2.1.

Table 2.1: Applied State of the Art Evaluation Framework

	Offline	**Secure**	**Anonymous**	**Simple**
Bitcoin		✓	✓	
PayPal		✓		✓
Paysafecard		✓	✓	
Bank Transfer		✓		
Google Wallet		✓		✓
Friendbank	✓			✓
Venmo		✓		✓
Square				✓
Blaze et al.	✓			
Traveler's Check	✓			
NFC Cards				✓
Quick	✓		✓	✓

When analysing the table, the first notable thing is that most of the payment systems fulfil the security requirement. This is the most important requirement for a system to be accepted

by the users. The offline attribute is only fulfilled by a small subset of the analysed payment systems. Furthermore, all analysed systems are either offline or secure but not both. Anonymity is also not a very widespread feature for payment systems. Bitcoin which is by far the most prominent example of an anonymous payment system provides this feature [34], but is because of its complexity not easy to use. Moreover, simplicity is another reason for a payment system to become popular. Simplicity has been defined in the proposed framework in chapter 2.4.1. According to table 2.1, most of the discussed systems are simple to use. The following conclusions for each payment system can be made:

Bitcoin (see 2.3.1.1) Bitcoin is secure because of its distributed P2P structure but needs an internet connection to ensure the security and to fight double spending. This is also one of the reasons why it is not simple. Another reason is that one needs to take care that his or her own Bitcoin address does not get lost or stolen, because without the address all money is gone as well.

PayPal (see 2.3.1.2) PayPal is probably the first prominent example of a payment system that reached its popularity because it is very easy to use. However, PayPal can only be used online and is not anonymous.

Paysafecard (see 2.3.1.3) The Paysafecard relies on vouchers that can be bought in shops. This vouchers make it anonymous and secure, because the vouchers can be bought with cash. Therefore, no relation to the user can be drawn. It is also secure because the vouchers are printed out and one can only lose as much money as there is still left on the voucher. On the other hand handling of vouchers is difficult. One needs to buy the vouchers upfront but they cannot be bought everywhere. Furthermore, one always needs to be aware of the current balance.

Bank Transfer (see 2.3.1.4) The popular bank transfer is secure because it uses multiple security features such as mobile TANs, digital signatures and PINs. This is why it takes a certain amount of time to conduct a transaction, which does not go along with simplicity. Since bank transfers are mostly used for high volume payments, this is accepted by the users [29].

Google Wallet (see 2.3.2.1) Google Wallet is similar to PayPal and requires certain credentials to be entered. Furthermore, the fact that an internet connection is needed[49] makes it secure and likely to be considered simple. This also explains why it cannot be used offline.

Friendbank (see 2.3.2.2) Friendbank can only be used offline and does not allow to transfer money at all.

Venmo (see 2.3.2.3) Venmo only requires an e-mail address or a phone number to conduct a payment, which may be seen as simple to use. It is also secure because a user can protect his or her account with a PIN or password. However, it can only be used online.

[49] Either on the merchant's side for a B2C transaction or for both parties for a C2C transfer

Square (see 2.3.2.4) When it comes to C2C payments Square's only upside is its simplicity. Payments can be sent via simple e-mails without any need for authentication, hence it is not secure. Moreover, Square Wallet lacks security, because it depends on a check-in and the waiter who is responsible for the check out.

Blaze et al. (see 2.3.3.1) Blaze et al. discussed an idea for an offline payment system, which depends on risk management to keep fraud on a manageable level. Even though stolen credentials can only be used for a certain amount of time and only up to a certain amount, it is not entirely secure. Furthermore, they made no attempt to provide anonymity. Since a certain number of manual tasks need to be performed, it is likely not to be considered as simple either.

Traveler's Check (see 2.3.3.1) Another theoretical offline approach is the traveler's check. The customer needs to buy the checks upfront, which makes it not simple to use. Security is not sufficient since those checks can get stolen and used by the thief until they are reported.

NFC Cards (see 2.3.4.1) When using an NFC-enabled debit or credit card, the merchant needs to have an internet connection to complete the transaction. Due to the existence of the fraud problems discussed in chapter 2.3.4.1 it is also not 100% secure, even though stolen money gets refunded by the credit card company. Another factor is that a credit or debit card is always linked to a real person, so it is not anonymous either. However, especially with the contactless payment feature it is very easy and simple to use.

Quick (see 2.3.4.2) Quick can be used online as well as offline, which makes it always available. Since no PIN or any other type of authentication is needed, it is not secure. However, this is what makes it simple to use. Quick is anonymous because it can also be used with anonymous cards[50].

2.4.3 Detailed Comparison Table

The table in Appendix A gives a more detailed insight of important indicators for each payment system. The indicators are partly based on the attributes for the evaluation framework introduced in chapter 2.4.1. However, the table is a more detailed comparison focusing on the user's point of view. The table can be used to get a quick overview about each payment system, as well as its pros and coms. The indicators used for the table are mostly based on [24] but with additional indicators for digital and mobile payment systems.

- **Online/Offline** Describes if an active internet connection is needed to conduct the payment.
- **Clearing Only** Specifies if payments are accumulated or just passed on to another entity, e.g. the bank account.

[50] `http://www.quick.at/`, accessed 2014-01-08

- **Payment Time** The payment time defines when the user is billed, like already described in chapter 1.6.2.
- **Anonymity** Some of the payment systems do not require any form of authentication or link to a real person. These systems are marked as anonymous.
- **Small Payments** Most of the systems charge for using their system. This indicator tells if using it for small amounts is advisable.
- **Degree of Popularity** This indicator tells in which countries the payment system can be used.
- **Double-spending Protection** Double spending is especially important for payment systems without a central authority, as discussed in chapter 2.3.1.1.
- **Generating/issuing New Money** Describes if it is possible to generate new money or how to transfer money into the payment system.
- **Security** Tells which security features help the user to keep their money safe. The are described in more detail in each payment system's section.
- **Fee Receiving** Fees that apply when receiving money with this system from the user's view.
- **Fee Sending** Fees that apply when sending money with this system from the user's view.
- **Users** This factor indicates the base of potential users for the payment system.
- **Transactions** Gives information about how much money can be spent and if there are any limits.
- **Type** Can be one or many of the following options: customer-to-customer (C2C), business-to-customer (B2C), or business-to-business (B2B).
- **Risk** Risk the customer, the company or the bank could have to deal with when using each system.
- **Pros** Summary of positive features.
- **Cons** Summary of limiting features.

2.5 State of the Art Conclusion

There are already a lot of different online and mobile payment systems on the market that are used by a wide range of customers on a daily basis. The payment systems that were analysed cover a wide variety of different approaches and fields of application. This ranges from tools for friend payments to tools which are used to pay in shops. One can easily get the impression that there is already a payment system for each purpose because of the vast variety of tools. But as already discussed in chapter 1.6.3, no offline payment system apart from Quick has reached the critical mass.

One of the conclusions that can be drawn is that the companies try to make **spending money as easy as possible**. For example *PayPal* only requires a valid e-mail address to use it. They also try to make it as cheap for the end user as possible and try to bill the merchant if possible. That is the reason why the merchant and not the end customer has to pay the fee if fees occur. Therefore, sending money is mostly free and charges apply for receiving. This is one thing that is different for C2C money transfer, since there is no merchant involved where the money could be taken from, hence of the involved customers has to be billed instead.

Security is the biggest challenge when it comes to digital payment system and also the most important as the evaluation framework showed. The services use different approaches when it comes to protecting the user's money. The range starts at no security at all (*Square*, *Cash*), continues with the need to *possess* something (e.g. *Paysafecard*, *Bitcoin*), up to the need to enter a four digits PIN or user credentials (e.g. *PayPal*, *Google Wallet*, or *Venmo*). The conclusion regarding security is, that almost all payment systems offer at least a certain amount of security features to protect the user's money.

As it was discovered, most of the established systems require a **credit or debit card** to be linked to the user's account to use it. This enables the payment service provider to be able to withdraw money from the customer's credit card. This transfers the risk of the payment service provider to the credit card company, since they have to make sure that they get their money. This also means that each customer needs to have a credit card, or the customer won't be able to use the service in most cases. Another notable aspect is, that there are a lot of different payment services available, but most of them are only available in the US and not in Europe.

The analysis of the state of the art showed what was already assumed in the introduction. There is no offline payment system available on the market, which allows offline C2C payments.

Offline Extensions for Existing Payment Systems One question that arises when looking at state of the art solutions that do not support offline is whether they could extend their service. For solutions like NFC-enabled credit cards or Quick the problem is, that the customers would need to buy hardware to allow offline C2C transactions. Moreover, in some cases it is likely not to work with the system's current design, since for example Quick uses a reference account in the background where the money is taken from when the merchant claims it. For mobile solutions like Google Wallet or Venmo an extension that allows offline payments would be easier to implement since they already offer similar online friend payment functions.

CHAPTER 3

Implementation Design

Having analysed existing and established solutions, the next step is to use this knowledge to describe the protocol, that allows *offline* money transactions. Therefore, this chapter deals with explaining the proposed solution, which allows two people to do so. It is divided into three sections as figure 3.1 shows and describes the protocol in a top-down way. It starts with a more general point of view and goes into more detail later.

The *Overview* deals with a general instruction helping to understand why and how the problem was approached.

The *Use Cases* covers general implementation details to get an understanding of the involved actors and their use cases within the payment system.

The last and most important part is the *Protocol* itself which is divided into two parts. It is first described from the user's point of view to make it easier understandable how the user is supposed to use the system. The last part is the description of the protocol from the system's and process' point of view, hence how it is supposed to be implemented. It deals with how user accounts are created, how money is exchanged and how money is wired between the users.

Figure 3.1: Solution Overview

3.1 Overview

As described in the introduction, situations can occur in which there is a need to exchange small amounts of money between two people. In some of those situations, when there is no internet connection available, the need to exchange money still continues to exist. Moreover, online and wireless transactions can always be risky because of several reasons discussed in chapter 2. This brought up the idea to enable the actual transaction process without the need of an internet connection. Hence, the transaction cannot be eavesdropped or altered without the knowledge of the user. On the other hand the lack of an internet connection creates problems, since no online credit check can be performed in real time. Therefore, the user has to top-up his or her phone with money from the online payment service first in order to be able to transfer it to other users.

The actual money exchange is performed offline between two phones that run the payment service client. In order to transfer the data necessary to conduct the transaction between two users, NFC is used. This helps to make the transaction more secure and more convenient for the user at the same time. Each transaction creates a so called money token which contains all necessary payment information (e.g. sender, receiver, amount, currency ...). This token can later be sent to the payment service to redeem the money, hence the clearing process needs to be performed online.

3.2 Use Cases

This section deals with the proposed solution from a general point of view. First, an overview of the actors and the use cases involved in such a payment system are discussed briefly. This aims to help in understanding the context of the system and to explain the components that are needed. Finally, this chapter concludes with requirements that need to be fulfilled by the system.

Actors There are two different kinds of actors involved in the whole payment process. Even though the actual transaction is only performed between the **users**, the **payment service** needs to take care of the accounts being settled and cleared correctly.

- **User** Each transaction requires two users, namely a sender and a receiver. The sender can choose how much money in which currency is sent to whom. The receiver on the other hand can accept the payment by signing it. Each transaction creates a money token, which is stored on both phones. As soon as an online connection is available again, the tokens can be transferred to the payment service or the clearing service agent. Only one of the involved users has to send the token to the payment service, as they are identical.

- **Payment Service** The payment service works as an intermediary and issues money to the users. It also accepts the tokens which were created when carrying out a transaction. These money tokens contain all information about the transaction. The payment service uses this information for further processing of the payments, hence wiring the transfers.

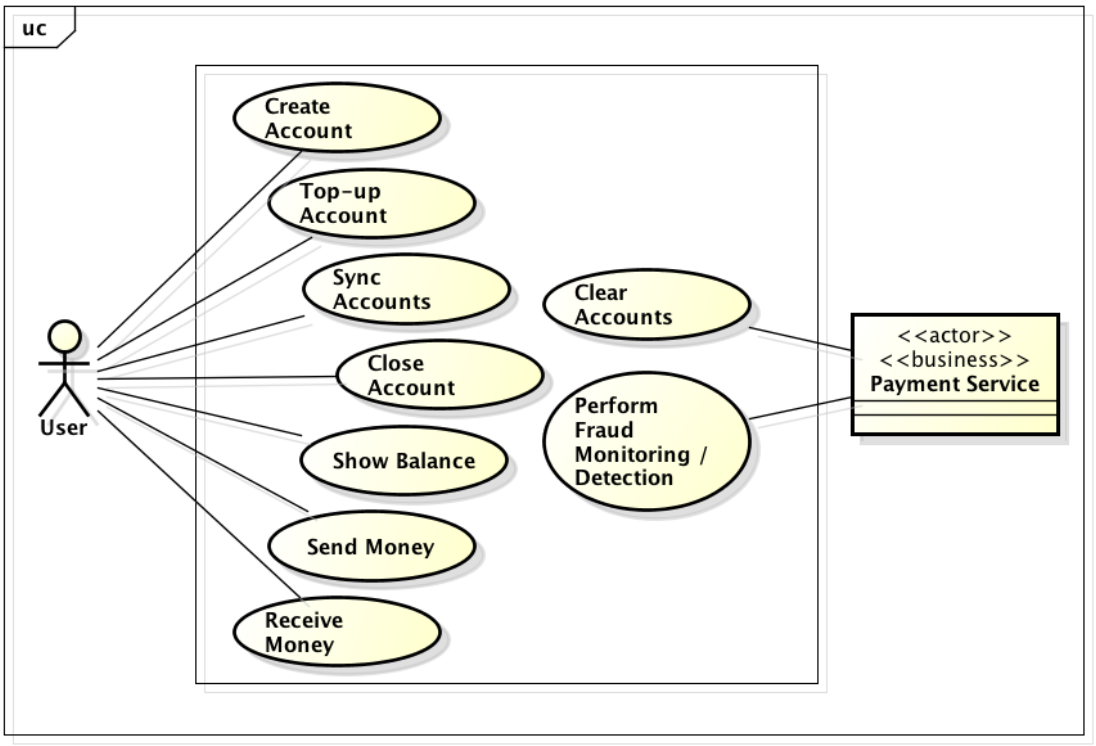


Figure 3.2: Usecase Diagram

User Use Cases The user is able to perform a set of different tasks with the payment system, as it can be seen in figure 3.2.

- **Create Account** The user can create an account with the app and link it to a debit and/or credit card. Linking the account is necessary to allow the user to top-up his or her account with money and for validation purposes. As soon as the account is created and verified, a public-private key pair is created. Additionally, a Personal Identification Number (PIN) is set, which is used to sign the payments and to protect the private key.
- **Top-up Account** When the account is created, the user can top-up his or her account up to a certain amount on his or her mobile phone, which then can be sent to other users. This can be done in two ways:
  - The user can use the payment service's website to top up his or her account. As soon as the user synchronises the account with the payment service, the money is available on the phone.
  - The user can directly top up the balance from the phone. This works only if the user's phone is online.
- **Sync Account** Since the whole transaction is carried out offline, the user has to sync the phone with the online reference account to redeem the payments he or she received at a

later point in time. Money that is topped up using the website is sent to the user's phone during synchronisation.

- **Close Account** When the user does not want to use the service any more, the possibility to close down an account is given.

- **Show Balance** The user is able to see how much money is still left to spend on the phone and online on the reference accounts. This also includes showing a detailed list of all transactions (from, to, when ...).

- **Send Money** The user can choose the amount and currency depending on the phone's account balance. Optionally, a location where the transaction took place can be set. This information is signed with the PIN and sent to the receiver. More details about the transaction and its contents can be seen in chapter 3.3.2.2.

- **Receive Money** The receiver can check a transaction's details and accept it by entering his or her PIN.

Payment Service Use Cases The payment service is passively responsible for two important tasks (see figure 3.2).

- **Clear Accounts** After a transaction was carried out, the money needs to be transferred to the right accounts according to the money tokens received.

- **Perform Fraud Monitoring / Detection** Fraud monitoring could be used by the payment service to find unusual and illegal payment streams and revert them if necessary.

Requirements In order to make such a tool useable and secure certain requirements need to be met. The difficult part is to make it secure and still easy to use at the same time, so that the users are more likely to use it frequently and recommend it to others. The requirements include:

- **Usability** In order to get users to use a tool for offline transactions, it simply must be simple and straight forward. As [45] argues, usability is an important prerequisite to success. Without a good experience, the user does not want to use the tool over and over again. An example for bad user experience is the requirement of conducting too many steps in a transaction.

- **Security** In order for the money to be safely stored and securely transferred, security is an important issue. Therefore, security is also an important issue that needs to be met.

- **Availability** High availability is needed so the tool can be used everywhere, even without phone signal or any other type of connection. This is ensured by enabling offline transactions. Offline in this case means that neither of the involved phones needs to have an active internet connection to a server.

3.3 Protocol

Following the above overview on involved actors and use cases the protocol itself is described. This is done in two ways. First, from the user's point of view and second, from the system's point of way.

3.3.1 User View

The user's point of view helps to understand how the user interacts with the system and how the user is supposed to use it. Therefore, this chapter uses a more general way to describe how the system is used.

In this section the term *Personal Identification Number (PIN)* is used. It can be compared to the PIN that needs to be entered when turning on a mobile phone or when one wants to authorise a money withdrawal at an ATM. For the proposed protocol the PIN is used for a similar purpose as when using an ATM. It is used to authenticate the user without the need to enter username and password each time a payment is conducted.

3.3.1.1 Account Structure

A first step is to give an overview of the relation of all accounts that are involved and how they work together. Figure 3.3 shows the linkage between the accounts of a single user.

The upper part, with the green background and the *ONLINE* label represents the information kept at the payment service. Starting from the top, the user can link several different types of credit or debit cards to an online **reference account** of the service provider (**1**). One reference account can be linked to several accounts for different currencies (**reference money accounts (2)**).

At least one money account needs to be created and at least one card needs to be linked to the account at all the time. The reference account tracks all transactions online and can also be seen as an online backup in the background. The reference account stores all transaction information as soon as the user has synchronised the transactions on the phone. Now that the accounts are linked, the user can top-up one or multiple of the reference currency accounts with money from one of his or her bank cards. He or she can also accept money from other users or transfer money between his or her reference currency accounts.

The lower part of figure 3.3 is highlighted in red and is labelled *OFFLINE*. This indicates that this is the part of the payment system that works offline. Once the online reference account is linked with the account on the phone (**3**), the linked phone and the reference currency account are synchronised (**5**). This enables the user to make the money spendable on the phone (**4**). As an alternative, the user account can be created on the phone instead and the online reference account is created as soon as the user hits the register button and the credentials are verified.

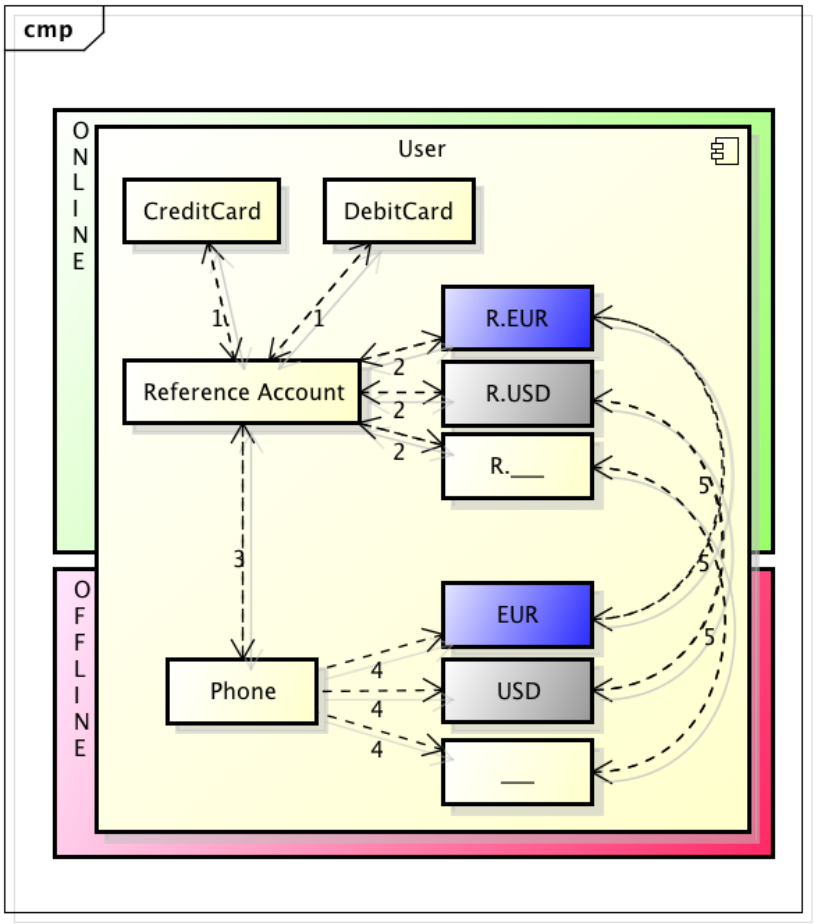


Figure 3.3: Account Structure

3.3.1.2 Sending and Receiving Money

The diagram in figure 3.4 shows the interaction between all involved parties, hence two users and the payment service.

1. **Phone A** creates an account and a public-private key pair which is used to sign payments. In return the user receives a server-signed public key. The public key needs to be signed by the payment service to verify that user **A** is a valid user.

2. **Phone A** has its balance topped-up by the user. This can either work via the payment service's website or directly on the phone.

3. **Phone B** also creates an account and a public-private key pair which is used to sign payments. In return the user receives a server-signed public key. The public key needs to be signed by the payment service to verify that user **B** is a valid user.

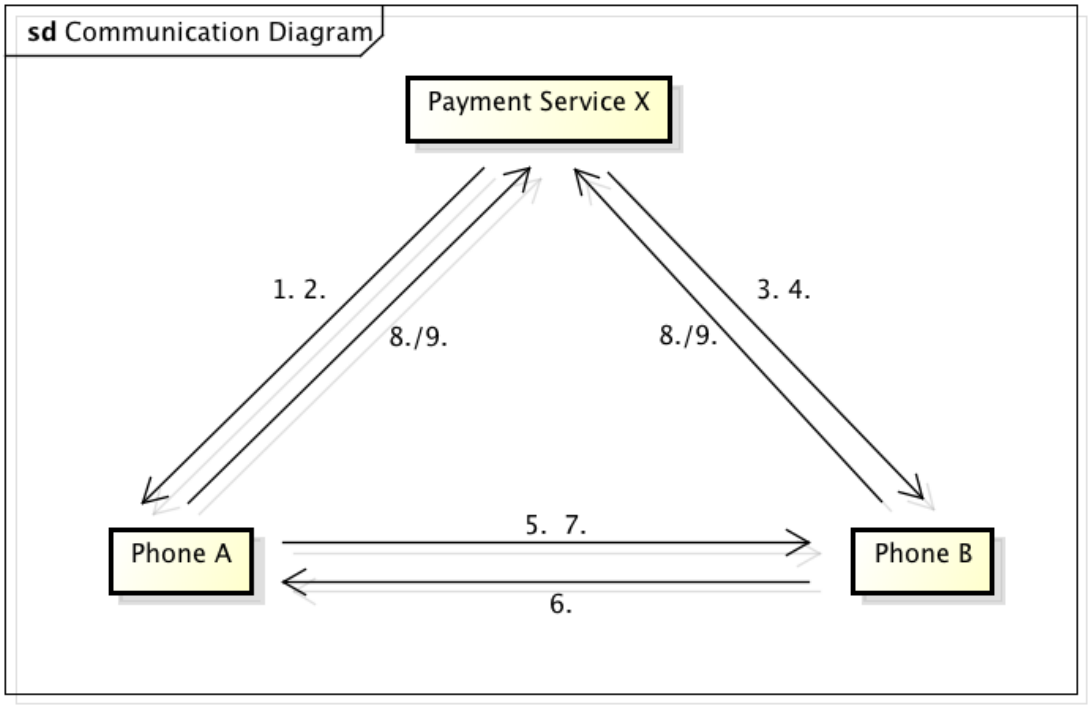


Figure 3.4: User View: Transaction

4. **Phone B** has its balance topped-up by the user.

5. **Phone A** enters the data required to initialise a payment (e.g. *currency* and *amount*), enters the PIN, selects if he or she wants to set the *location* where the transaction took place and touches his or her phone with **Phone B** to send the signed token and user credentials.

6. **Phone B** checks the transaction details. If the transaction is okay and the credentials from user **A** are valid, user **B** signs it with his or her own PIN. Then user **B** touches his or her phone again with **Phone A** to send the signed token back. This time his or her own credentials are sent along as well.

7. **Phone A** now also has the chance to check whether the credentials from user **B** are valid. If this is the case, the token is signed again and with a third touch of the phones sent back to **Phone B**. This step is required to ensure that both parties are able to check each other's credentials and that both have the exact same key. With this step the transaction is complete.

8. and 9. **Phone A** or **Phone B** can send the token to the payment service to trigger clearing between the users. Since both parties signed the token, there is not necessarily a need for the other user to also send the token to the payment service.

3.3.2 Process View

This chapter deals with the technical perspective of the proposed protocol. It shows which steps are performed during a money exchange, which data is transferred and why. The chapter is divided as shown in figure 3.5. It is structured in a way in which the user would use the payment system. It starts with how accounts are created, continues with the actual money exchange and ends with the clearing and synchronisation of payments.

Figure 3.5: Process View: Chapter Structure

- **Account Creation and Certificate Issuing** The first part deals with how a user account is created. The essential parts are the certificates and how they are structured.
- **Money Token** The next part is about *what* kind of data is exchanged.
- **Money Transaction** This part deals with *how* the data is exchanged offline, hence the actual protocol.
- **Online Synchronisation and Clearing of Accounts** The chapter concludes with the synchronisation of accounts with the payment service and how the money is actually cleared between the users.

The process view also has to deal with the issuing, distribution and usage of certificates to sign transactions and to validate user identities. The term *pub* is used to describe a public key and *priv* for a private key. For example `X.priv(A.pub)` means, that the **Payment Service X** signed the public key from **User A** with its private key. Therefore, everyone who knows `X.pub` can check the signature and validate the key.

3.3.2.1 Account Creation and Certificate Issuing

When a user creates an account, certificates that are needed to sign transactions are created. Figure 3.6 shows how the certificate structure works.

- **Payment Service X** is the certification authority (see chapter 2.1.4) which issues certificates to the users.
  - `X.pub` is the payment service's public key, which is distributed to all users and therefore publicly available. This key is required to validate the signed public keys of all users.
  - `X.priv` is the payment service's private key. This key signs the user's public keys.

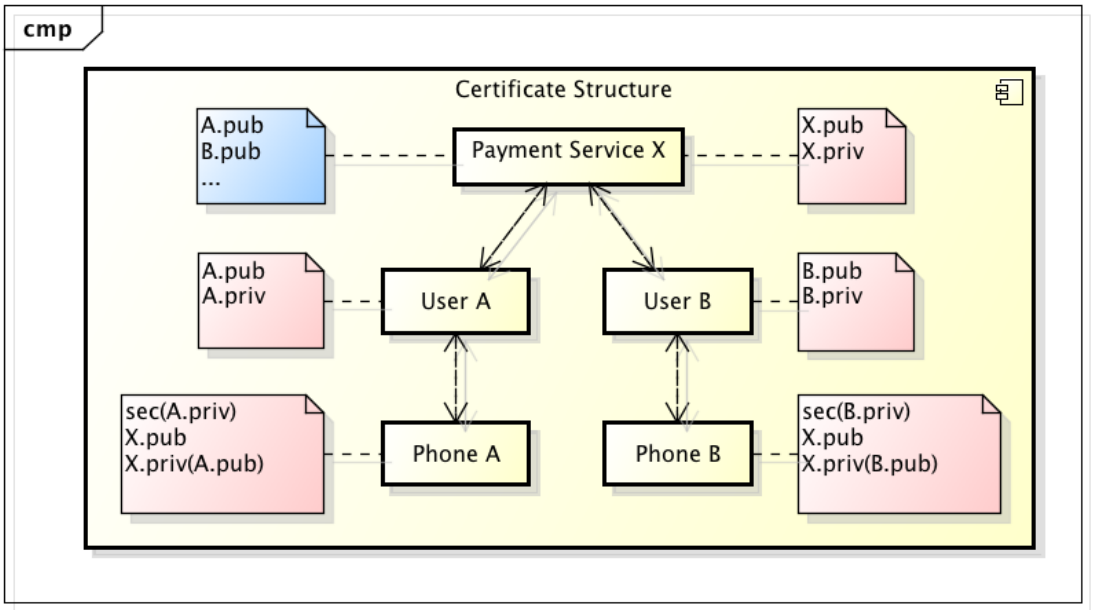


Figure 3.6: Process View: Certificate Structure

  - `A.pub, B.pub,...` represent the users' public keys, which are sent to the server upon registration of the user. These keys can be requested by everyone for validation purposes.

- **User A** is the account of the first user. The following keys are issued on account creation.
  - `A.pub` is the user's public key.
  - `A.priv` is the user's private key.
- **Phone A** is the phone of user **A**. It holds the following keys after the user logged in:
  - `sec(A.priv)` is the user's securely stored private key. This key is used to sign transactions.
  - `X.pub` is the **Payment Service X**'s public key. This key is installed on all phones by default to validate the signed public keys from other users.
  - `X.priv(A.pub)` is the user's public key which is signed by the server. This is done so that**A** can identify himself as a valid user when conducting transactions.
- **User B and Phone B** have the same types of keys as **User A and Phone A**.

3.3.2.2 Money Token

When money is exchanged virtually it has to contain more than just the amount and the currency. A lot of different attributes and techniques are used to describe such a money token. Therefore,

this chapter deals with the money token content. Since the authenticity of those *tokens* cannot be established with physical security characteristics, other techniques have to be used. This is the reason why cryptography has to be used to accomplish this problem when it comes to virtual money, as stated in chapter 2.1.

Money Token Attributes As described in the electronic check architecture [3], a check or token needs to contain certain data fields. This paper is used as a basis to identify the attributes needed for transferring money.

- **Version Number** The version number indicates which version of the protocol is used. This is necessary to ensure that updates can be made.
- **Sender** The sender information needs to be included. A unique ID, as well as a human readable name is used to make the sender uniquely identifiable for the payment service and still readable to humans. The sender's public key, which is signed by the server as shown before, is also stored here.
- **Receiver** The same as for the sender also applies to the receiver.
- **Amount** This field contains the amount that the sender wants to transfer to the receiver.
- **Currency** The currency indicates in which currency the amount shall be transferred.
- **Date** The date and time when the transaction took place.
- **Location** This optional field contains the location where the transaction took place, e.g. to help the user to remember why the payment was made.
- **Message** The message is also an optional field with information entered by the user. This could e.g. contain the reason for the money transfer or other necessary personal information from the user.

Hierarchical Structure The token is organised in a hierarchical structure and each phase of the protocol adds a new layer to the token structure. This results in a structure that can be represented in a simplified version the following way: `A.priv(B.priv(A.priv(Money)))`. The phases are described in more detail in the next chapter (3.3.2.3). Figure 3.7 shows how such a hierarchical money token structure looks like. Starting from the inside it has the following layers.

- **`A.priv` (1st Phase)** Contains most of the real transaction information. This data is created by the person who wants to send money and contains the *Money Information* and *RA (**R**andom value **A**)*. Both parts are signed in the first phase by A's private key. Afterwards it is sent to B.
  - **Money Information** This part of the token contains the actual transaction information that was previously discussed.

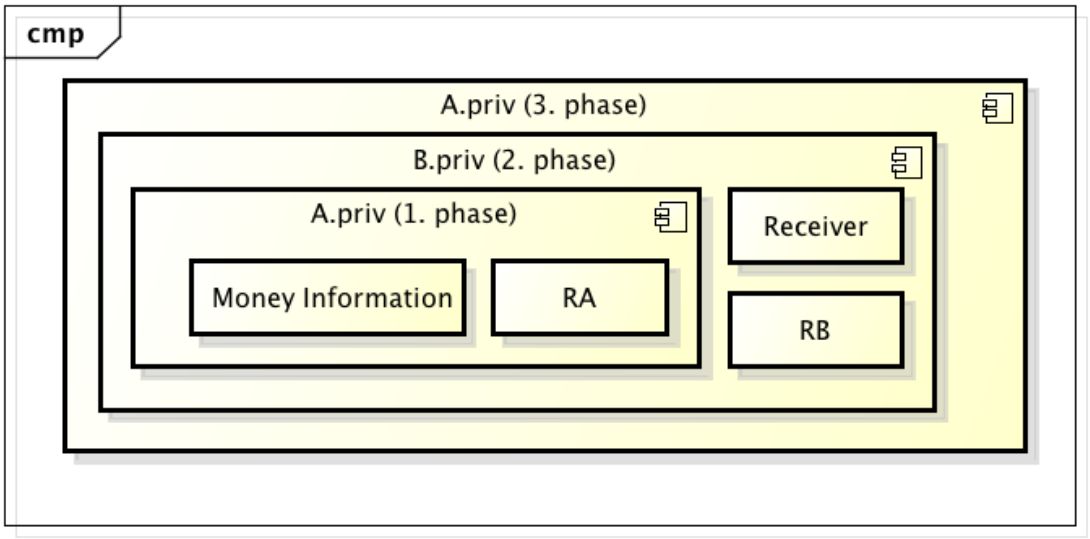


Figure 3.7: Process View: Money Token

  - **RA** `RA` (**R**andom value **A**) is a random value created by A. This value is added to make each money token unique, so it cannot be used twice without notice. This works because for each transaction a new `RA` is created and it can easily be detected if the same value is used more than once.

- **`B.priv` (2nd Phase) B** receives the information created by **A** in the first phase. The *Money Information* is displayed to the user. The information from the first phase, as well as the information added in this phase, is signed by B's private key and sent back to **A** for a final review.

  - **Receiver** After accepting the payment, **B** adds the *receiver* information, which contains who the money will go to.
  - **RB** `RB` (**R**andom value **B**) is B's equivalent to `RA` and is needed to make each token completely unique.

- **`A.priv` (3rd Phase)** User **A** makes a final review and signs the transactions again. This phase creates a valid money token. The token is again sent back to B, so both parties share the exact same token.

Token Representation To make the token as independent from any form of software or hardware, Extensible Markup Language (XML) is used to describe such a money token. As [25] states, many industry standards have been proposed in XML. Since those money tokens only use a subset of functionality or differ from existing standards like *UN/EDIFACT*[1], a new XML

[1] `http://www.unece.org/trade/untdid/welcome.html`, seen 2013-12-12

schema is introduced to transfer money.

The exchange format is based on the three phases of the proposed protocol. Each phase has its own tag in the XML data file that is transferred. After the completion of each phase, which is indicated by an existing signature, the next one is added.

Since each phase encapsulates the data from the previous one, each time all data would have to be signed again. Instead, inspired by [30], a different approach is used. Due to the fact that only a minor set of new data is added in each phase, it would not make sense to always sign the whole set of data over and over again. Therefore, only the newly added data, as well as the signature from the previous phase, is signed. This ensures that no data can be changed without notice, because if data would be changed the previous hash would change as well. Not even **A** is able to change data from phase one after receiving the token for the second time in phase three, because **B** has already signed **A**'s signature from phase one (see figure 3.7).

The XML code example 3.1 shows how a complete transaction looks like. After **A** has entered the transaction information, line 3 to 11 is signed and the information is stored in the tag `sig` in line 26. The signing key from phase one is placed in a separate tag. Then again line 14 to 19 is signed in phase two and the key is stored in line 27. This ensures, that the validity of the whole transaction can always be checked and ensured. Furthermore, it is still easier to deal with the data, because it is not stored nested. The same procedure is applied in phase three.

Listing 3.1: XML Payment Message

```
1  <payment id="" version="">
2       <!-- PHASE ONE -->
3       <phase id=1>
4             <sender id="" name="" pubkey="" />
5             <when date="" time=""/>
6             <amount value="" currency=""/>
7             <location name="" lat="" long=""/>
8             <message> </message>
9             <serverauth></serverauth>
10            <ra></ra>
11      </phase>
12
13      <!-- PHASE TWO -->
14      <phase id=2>
15            <receiver id="" name="" pubkey=""/>
16            <rb></rb>
17            <serverauth></serverauth>
18            <phase id=1>KEY1</phase>
19      </phase>
20
21        <!-- PHASE THREE -->
22      <phase id=3>
23            <phase id=2>KEY2</phase>
24      </phase>
```

```
25
26          <sig id=1></sig>
27          <sig id=2></sig>
28          <sig id=3></sig>
29
30  </payment>
```

The rest of the token is straightforward. Both the sender and the receiver have the public key of the **Payment Service X** stored on their phones. This is needed to validate the authenticity of the sender and the receiver. The tag `serverauth` (line 9 and 17) contains the signed sender tag (line 3) and the signed receiver tag (line 14). If the signature, which is created when the user registers for the service, corresponds to the information in the tags, the user is authenticated. This ensures that the `id`, the `name`, as well as the used `pubkey` belong to the user in question. Hence, if the sender signs the first phase, the receiver can be sure that the sender is a valid and registered user of the service and that the information was entered by him. This is important for the receiver to claim the money that was sent to him.

3.3.2.3 Money Transaction

The money transaction itself is the most important part of the whole payment system. As figure 3.8 shows, it is divided into three phases. This process, which can be seen in detail in figure 3.9, describes how a financial transaction is conducted and which data needs to be sent to whom. It is only carried out between the phones. The payment service is not part of the actual transaction and only responsible to clear the transactions at a later point as described before.

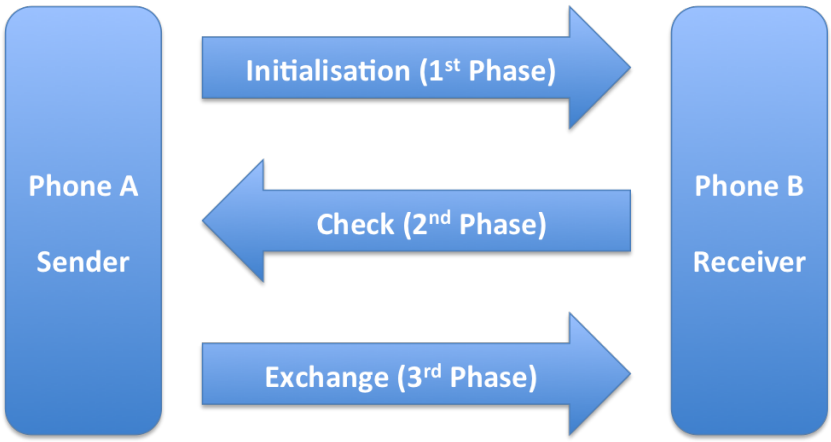


Figure 3.8: Money Transaction Phases

- In the *first phase* **A** sends the transaction information, which includes the payload, a random value RA, as well as **A**'s signed credentials by the payment service. All this is signed with **A**'s private key and sent to **B**. This phase is called *initialisation phase*. **B** can validate the data received from **A** and check if **A** is a valid user.

- In the *second phase* **B** has to check the transaction data. Then the random value RB as well as the by the payment service signed credentials are added. This again is signed with **B**'s private key and returned to **A**. This stage is called the *check phase*. Here **A** can check whether **B** is a valid user and if this is the case start the last phase.

- The *third phase* is the *exchange phase*. This stage is used to bring both parties on the same denominator, hence to make sure that both users share the same money token. This stage cannot be omitted, because otherwise **B** would not have the clarity that **A** identified **B** as a proper user.

A more detailed look at the protocol, which is shown in figure 3.9, reveals that the following steps are necessary for a successful transaction. The left side represents the **Sender A** and the right side the **Receiver B**. On each side three parts are involved. **User A an B** are real people who control their phones. They are responsible to enter and review the transaction data. **Phone A an B** are the actual phones with their NFC interfaces and access to the phone's storage. Their task is to send, receive and compute information. **Phone Storage A and B** are the phone storages responsible for persisting data.

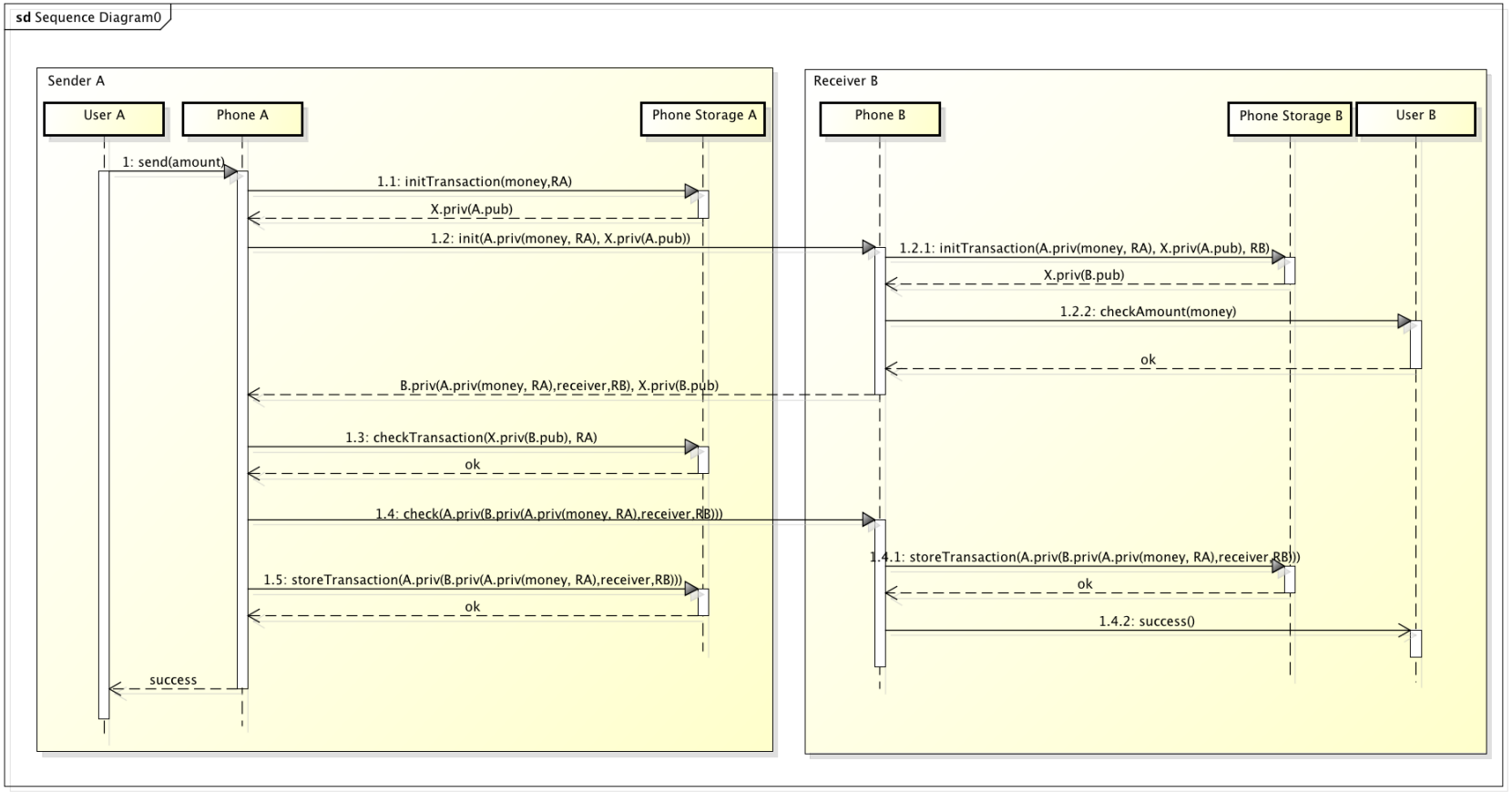


Figure 3.9: Sending and Receiving Money

1 `send(amount)` First, the `amount` and the currency needs to be entered by **Person A**. This information is processed and signed by the phone with the user's private key. Afterwards it is stored together with the *random value RA* on the phone's storage.

1.1 `initTransaction(money,RA)` This command initialises the transaction on the phone. The transaction details (`money`, see chapter 3.3.2.2) and `RA` are stored for later comparison.

1.1 return `X.priv(A.pub)` In return **A**'s public key is loaded from the phone's storage. This public key was signed by the payment service **X** before.

1.2 `init(A.priv(money, RA), X.priv(A.pub))` This command starts the *first phase*. The signed information (`money` and `RA`) is sent to **B**, together with **A**'s signed credentials. The transaction is approved with **A**'s PIN. **B** first checks whether **A** is a legitimate user by validating the authenticity of **A**. This is done by checking `X.priv(A.pub)`. If this succeeds **B** has the validity that **A** is a legitimate user and can also use **A**'s public key to validate the first part of the transaction.

1.2.1 `A.priv(money, RA), X.priv(A.pub), RB` After receiving the first message from the sender, the contained information is stored on **B**'s phone storage for later usage. A *random value RB* is created, which is assigned to the transaction. This step is similar to **1.1**.

1.2.1 return `X.priv(B.pub)` Again, the signed public key of **B** is retrieved from the phone's storage.

1.2.2 `checkAmount(money)` Next, the user is provided with the transaction details (`money`). This includes the sender, the amount, the currency and optionally the location (see chapter 3.3.2.2).

1.2.2 return `ok` If the user accepts the transaction, he or she also enters the PIN. This step signed the transaction details and prepares them to be sent back to user **A**.

1.2 return `B.priv(A.priv(money, RA),receiver,RB), X.priv(B.pub)` **B** now returns the signed transaction, which also contains the part of the transaction that **A** sent. Furthermore, it contains who will receive the money in the field `receiver`, as well as the computed random value `RB` together with **B**'s signed credentials.

1.3 `checkTransaction(X.priv(B.pub), RA)` **A** validates the authenticity of **B** and checks whether the transaction has not been altered.

1.3 return `ok` If the transaction has not been altered all details are displayed again to **A**. This time the receiver's information is included as well.

1.4 `A.priv(B.priv(A.priv(money, RA),receiver,RB))` If person **A** agrees that the details and authenticity of **B** is proven to be correct, **A** can initialise the third and last phase by simply transferring the signed transaction again to **B**. This step completes the transaction.

1.4.1 `checkTransaction(A.priv(B.priv(A.priv(money, RA),receiver,RB)))` **B** can store the transaction on the phone and redeem the money as soon as the phone is online again.

1.4.1 return `ok` Returns whether storing the transaction was successful.

1.4.2 `success()` **B** is now provided with a message that the transaction was successful.

1.5 `storeTransaction(A.priv(B.priv(A.priv(money, RA),receiver,RB))` Simultaneously, **A** can also store the transaction details on the phone's storage.

1.5 return `ok` This also returns whether storing the transaction was successful.

1 return `success` A final success message is shown to the user. This marks the transaction as complete.

3.3.2.4 Online Synchronisation and Clearing of Accounts

As already mentioned, it is necessary to go online with the phone to send the money tokens to the payment service at some point. This is needed for the receiver to redeem the money from the person who sent it. Since both parties share the same money token, it does not matter who of the two involved users goes online first.

Clearing After a successful money transaction, the sender as well as the receiver are in possession of a money token. This token contains all relevant information for wiring the payment, hence sending the right amount of money to the rightful receiver. The user has the impression that the money is exchanged between the two involved mobile phones. From the technical point of view it is only a cheque which allows clearing between the two involved accounts at a later stage.

Figure 3.10, which is an extension to figure 3.3 in chapter 3.3.1.1, shows how clearing works from a more general point of view. It can be seen that information is exchanged *OFFLINE* between the phone's accounts as highlighted by its red background. Moreover, information is exchanged by the payment service *ONLINE* between the reference accounts as indicated by the green background. The exchange on the phone level, which occurs offline (highlighted in red), is the first step, since that is where the transaction with the money tokens happens. The second part occurs between the reference accounts where the clearing takes place (highlighted in green). The payment service is responsible for wiring the money according to the transferred money tokens. According to figure 3.10 two possible cases can occur:

- When the sender **A** synchronises the tokens first, the payment service can send the money to the rightful receiver **B** (e.g. **A**.R.EUR *sends* to **B**.R.EUR.)

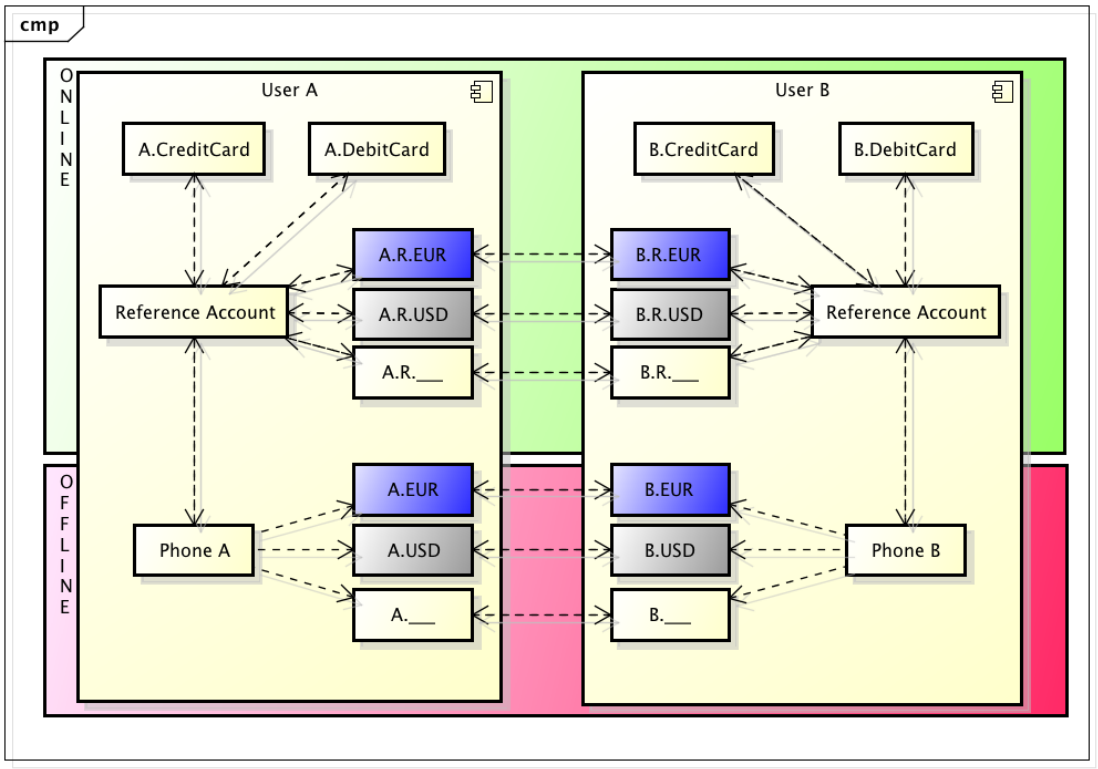


Figure 3.10: Account Synchronisation and Clearing

- The other possible case is that the receiver synchronises the tokens first. In this case the receiver can request the money he or she owns to be transferred to his or her accounts (e.g. **B.**R.EUR *claims* from **A.**R.EUR.).

Need for a Reference Account One may ask why it is necessary that every user has a reference account, which is assigned to a phone. There are several reasons for the need of such an account. First, it is necessary that users cannot simply *create* new money, e.g. by cloning the tokens on the phone (see chapter 2.3.1.1). Even if this happened, there would be the need for a place where the money that was created with cloned tokens, could be taken from. Second, it is necessary to overcome the need to store credit or debit card information directly on the phone. Instead, they are stored online at the payment service. Third and most importantly, the reference account is needed for the receiver to claim his or her money. This is possible because the money token contains the sender's and the receiver's information which is signed by both.

CHAPTER 4

Prototype Implementation

As a proof-of-concept, the proposed protocol was implemented for smartphones running Android. Since NFC is a new technology, not all phones offer an NFC interface. Therefore, two phones from the Google Nexus series, namely a **Nexus 5**[1] with Android 4.4.2 (KitKat) and a **Galaxy Nexus**[2] with Android 4.3 (Jelly Bean) were used to test the implemented software.

This chapter is structured according to figure 4.1. First, *Transferring and Signing of Data* is described. This explains how client and server handle the data. It also explains how the system's components are connected with each other. Second, the implementation of the *Server* is described. This includes how the data is stored and maintained in the database as well as how data is received from the client. The last part is the *Mobile Client* itself. Again, the implementation and details about handling and storing data is described. It also includes a detailed description and screenshots about how a transaction is carried out by two users.

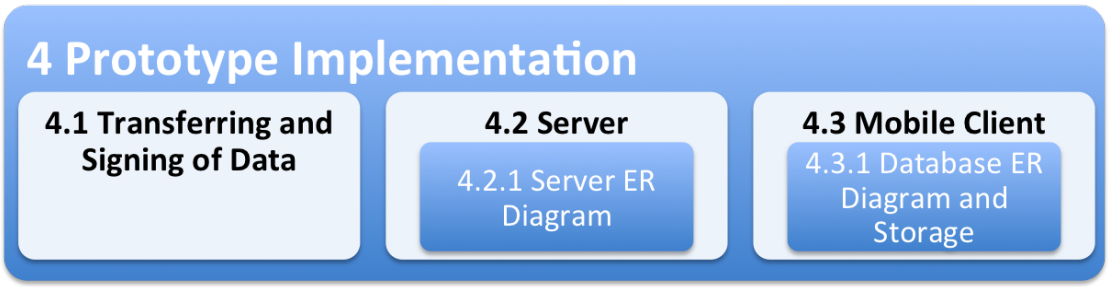


Figure 4.1: Solution Overview

The general structure of the implemented prototype can be seen in figure 4.2. The server in the upper part, which represents the payment service, uses a MySQL Database to store data

[1] `http://www.google.com/nexus/5/`, accessed 2013-12-20
[2] `http://www.android.com/devices/detail/galaxy-nexus`, accessed 2013-12-20

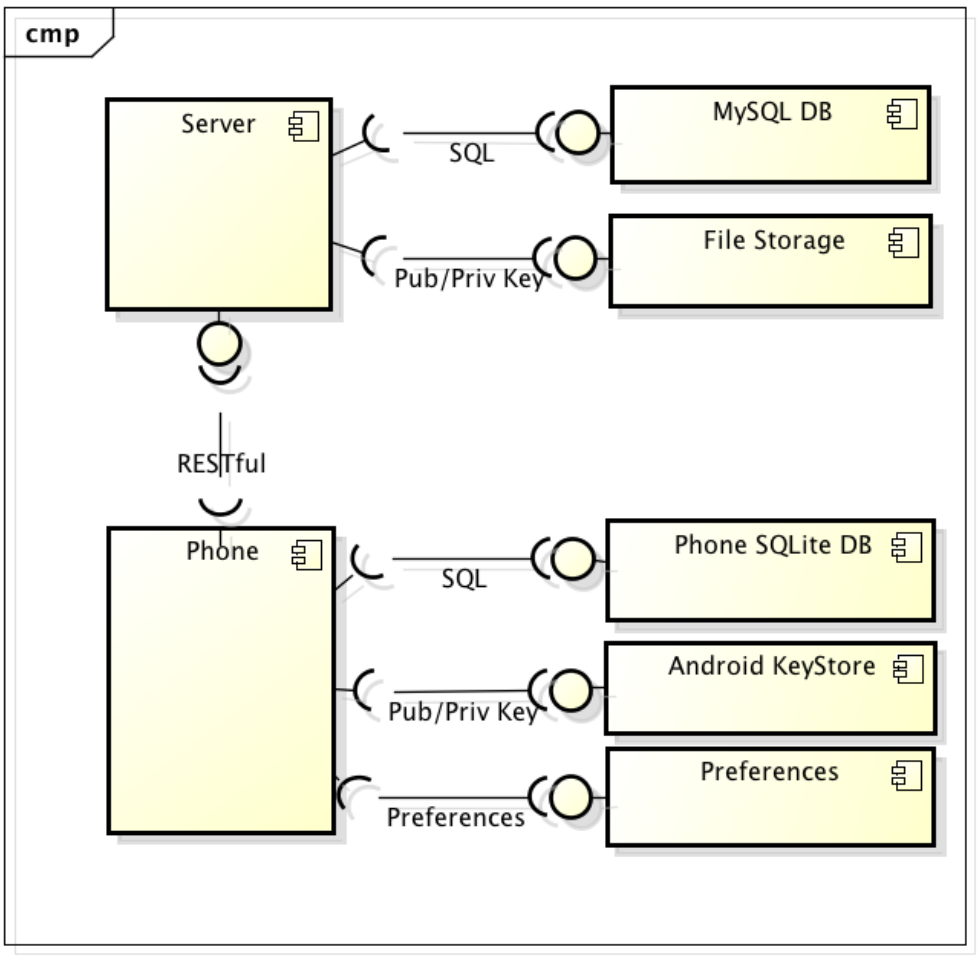


Figure 4.2: Prototype Component Diagram

about the users and the transactions made by the users. Furthermore, the server's public-private key pair needs to be stored to sign the user's public keys. For reasons of simplicity, the keys are stored in two separate files to ease the process to distribute the server's public key certificate to the user's phone.

On the phone three different types of storage were used. First, the included SQLite Database to store a list of the transactions conducted as well as the transferred money tokens. This is discussed in more detail in chapter 4.3.1. Second, Android's KeyStore[3] is used to securely store the user's generated public and private key for signing money tokens. Finally, the Shared Preferences are stored. This includes the access token and the server URL.

[3] `http://developer.android.com/reference/java/security/KeyStore.html`, accessed 2013-12-20

To connect client and server, a RESTful webservice is used [8]. REST is used because it provides an easy way to seamlessly interchange data between the phone and the server, while being simple and with good performance at the same time. In order to test features such as synchronisation, registering of new accounts and balance top-up, different resources were created to provide the requested service. They are discussed in chapter 4.2.

4.1 Transferring and Signing of Data

As seen in chapter 3.3.2.2, the protocol uses XML to exchange data between all entities to keep it as extendable as possible. Since reading and writing plain XML text may be considered cumbersome, POJOs[4] were used to store all transaction information on the phone. Using POJOs also allows for retrieving single parts of the transaction instead of the whole data structure. This is needed since not the whole transaction token is signed but rather just parts of it, as shown in figure 3.1. As soon as all the information has been encapsulated inside a POJO, they are serialised into a string and later deserialised by an XML serialisation framework[5]. There are two reasons for working with text strings rather than Java objects:

- It allows for platform independence.
- Creating and verifying signatures is made easy.

Serialising is mostly used on the phones to exchange data between them while conducting a money transaction. Furthermore, it is used to send the complete money token as text stream to the server for validating and clearing.

Java has a relatively good support for security and for signing and encrypting data. Android also provides example source code of how to implement and use the signing features as part of the SDK (Software Development Kit)[6]. Signing needs the serialised money token as well as the private key stored in the Android KeyStore. A signature string is the result of this process. In order to validate the data the receiver needs the serialised money token, the generated signature string as well as the correct public key.

4.2 Server

The server's main task is waiting for actions triggered by the user on the phone. Examples for such actions are registering of a user, checking of the online balance and synchronising of the accounts. Since most tasks are trivial, only one is explained in detail. The server uses the **Spring Framework**[7] to provide the RESTful service to the client, since this framework provides an easy way to publish resources so that the client can access them.

[4] Plain Old Java Objects. POJOs are Java-Objects that do not contain any logic and are used to simply store and retrieve data.

[5] XML Simple Framework was used: `http://simple.sourceforge.net/`

[6] `http://developer.android.com/tools/samples/index.html`, accessed 2013-12-21

[7] `http://projects.spring.io/spring-framework/`, accessed 2013-12-22

Listing 4.1: HTTP POST-Request to Sync a Payment

```
https://192.168.1.4:8080/sync
?userid=<uid>
&accesstoken=<access>
&dataid=<did>
&data=<payload>
```

Listing 4.1 shows the details of a client request. It accepts a money token including the user's ID and the client's access token[8] for user authentication. Furthermore, the phone's local ID for the money token as well as the data itself (encoded as Base64 string) are sent to the server. The server first checks whether the user is allowed to sync this token. If the user is allowed to the token is decoded and the validity of it checked. If it is valid the hash code of the serialised XML string is used to check whether the same token has already been synchronised by the other party of the transaction. If this is not the case, the token is added to the transaction table. If yes, the token is dropped but a success message is returned, since the same token can be transferred once by the sender and once by the receiver. All other implemented resources work in a similar way:

- `/register` This resource is needed for user registration at the server.
- `/balance` This returns the current balance of the account specified by the user.
- `/topup` This is a method for test purposes. It allows the user to top-up an account with money.
- `/pubkey` This returns the public key of the specified user. It allows for validating any public key.

4.2.1 Database ER Diagram

Figure 4.3 contains the server's database model.

- **User** The user table contains all relevant information for user management.
- **BankAccount** This table contains all linked bank accounts according to the account structure in figure 3.3 in chapter 3.3.1.1.
- **Transaction** All transaction information is stored in this table. For each transaction the sender and the receiver are stored so it can be seen which accounts take part in a transaction (e.g. **`A.`**`R.EUR` sends to **`B.`**`R.EUR`). Since an offline currency exchange is not possible, there is no need to save the account information twice to avoid redundancy.
- **Accounts** Is a relation table between users and transactions. It is needed to allow for multiple currencies because for each currency a new entry in this table is created.

[8]The access token is sent to the client upon registration or login to ensure the user's authentication. Since REST is stateless [8], each request needs to contain the authentication information.

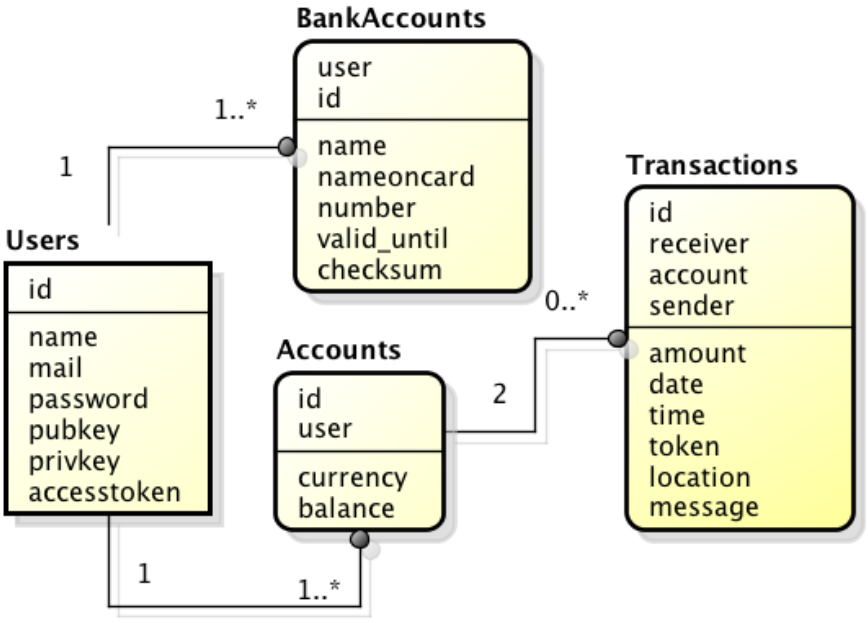


Figure 4.3: Server Database ER Diagram

4.3 Mobile Client

The most time intensive task of implementing the prototype was programming the client. It was necessary to implement the handling and signing of the money token and also the communication between client and server as well as between the clients via NFC.

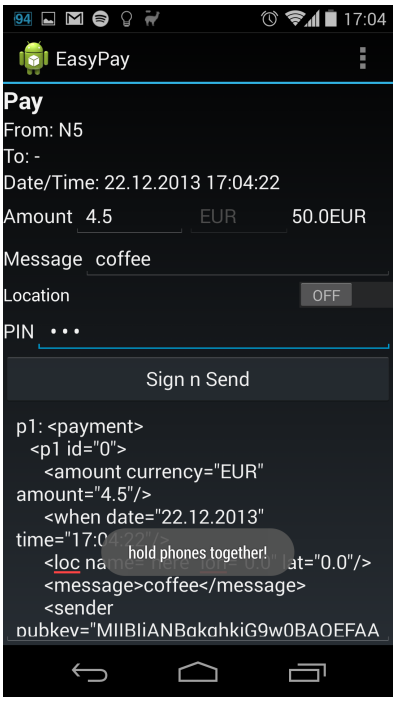
For money token handling a component that works on client and sever side was developed. This is due to the fact that Android is based on Java[9]. Therefore, XML serialisation and deserialisation as well as POJOs were used for dealing with money tokens. Signing and validating of a token also works in a similar way. The only difference is that the own key pair is stored in a dedicated place for keys: the Android Key Store. This feature allows for protecting the keys so that they cannot be accessed by anyone else except for the phone owner[10]. Additionally, a PIN is required to be entered by the user to ensure that only the rightful user is able to sign a transaction.

Figure 4.4 shows screenshots of how a transaction between **N5** and **GS** is conducted.

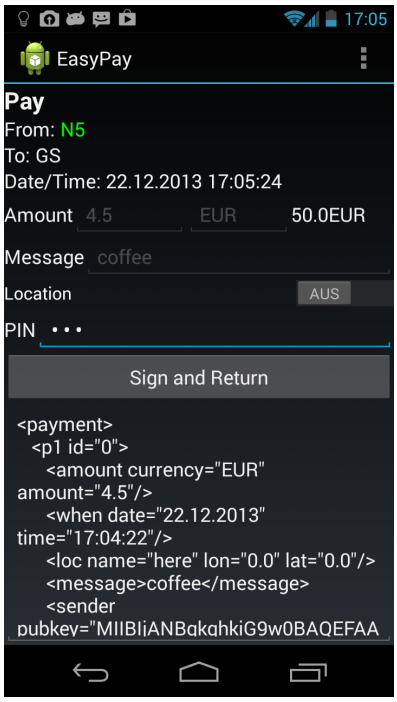
- **Phase 1** Screenshot 4.4a shows the data entered by the sender. The transaction's sender is the user **N5**. Furthermore, the current date and time is stated. The user has a balance of

[9] `http://developer.android.com/tools/index.html`, accessed 2013-12-22

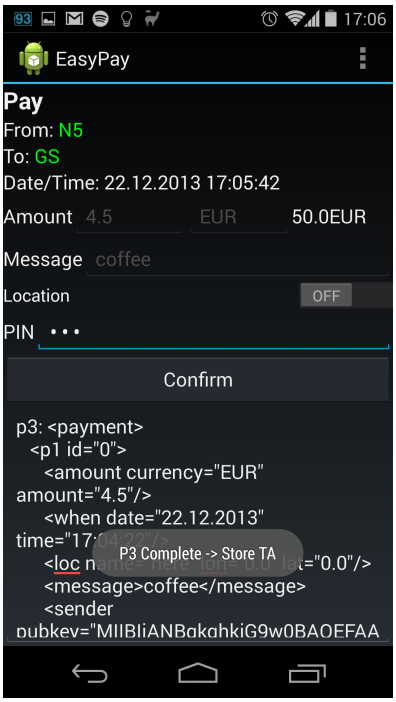
[10] `http://developer.android.com/reference/android/security/KeyPairGeneratorSpec.Builder.html#setEncryptionRequired()`, accessed 2013-12-22



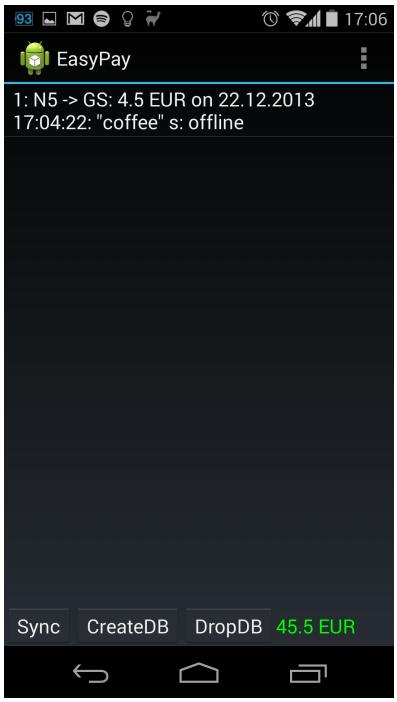
(a) Phase 1 (Sender **N5**)



(b) Phase 2 (Receiver **GS**)



(c) Phase 3 (Sender **N5**)



(d) Transaction List (Sender **N5**)

Figure 4.4: Android Client Screenshots

€50 of which he or she wants to transfer €4,5 to a friend. The transaction message (*coffee*) indicates the purpose of the transaction. After entering the PIN and pressing *Sign n Send*, the user is instructed to press the phone against the phone of **GS** who is the recipient of the money. The lower part of the screen shows information about the plain text money token.

- **Phase 2** After the phones have touched and the data was transferred, all transaction information is shown to the receiver, as it can be seen on screenshot 4.4b. **GS** can validate the data and if it is valid sign them with his or her own PIN. It is to be noted that the sender **N5** is now highlighted in green. This indicates that the data sent from **N5** was correctly signed by the sender and that the sender is a rightful user of the system. This is checked by validating the sender's information with the server authentication information, which is also part of the transaction. In order to send the data back to the sender, the *Sign and Return* button needs to be pressed before the phones can be pressed together again.

- **Phase 3** The receiver now gets to see the whole transaction again, as shown on screenshot 4.4c. It is to be noted that now the sender **N5** as well as the receiver **GS** are highlighted in green. This tells the sender that also the receiver **GS** is a rightful user of the system and can therefore be trusted. Furthermore, it shows that the transaction has been properly signed and validated. After pressing the *Confirm* button the transaction is completed. Once again the phones need to be pressed together so that **GS** also receives the final money token. When the third step of the transaction is completed, the transaction data is stored in the phone's database, as a message in the lower part of the screen indicates.

- **Transaction List** After the transaction is completed, the result can be seen on the transaction list. This is shown in screenshot 4.4d. Since this is **N5**'s transaction list, the balance has dropped to €45.5, as indicated in the lower right part of the screen. The transaction list shows all transactions conducted that have not been synced with the server, as indicated by the status *offline*. As soon as the button *Sync* is pressed, the transaction does not show up anymore unless the app preferences are configured otherwise.

4.3.1 Database ER Diagram and Storage

On the client's side different types of storage are used, as shown in figure 4.2.

SQLite The database that is used for the prototype is kept simple. There is only one table which is used to store a list of all conducted transactions. The information that needs to be shown to the user is stored in a single dataset. The rest of the information, such as the signature, is only stored in the field `token` which contains the whole money token. This setup enables to quickly generate a list that can be shown to the user upon request but it can also be easily transferred to the server. The field `sync` contains the synchronisation status, which indicates whether or not the token has already been transferred to the server.

Key Store Android's KeyStore provides an easy and convenient way to store and retrieve public-private key pairs. The advantage of using the KeyStore is that it provides a secure way to

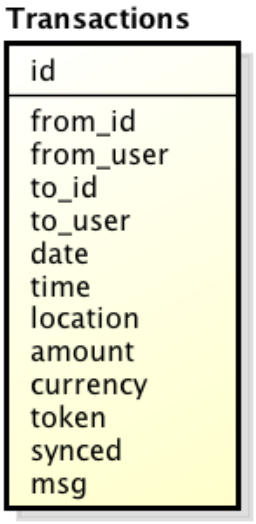


Figure 4.5: Android Database ER Diagram

store the keys so that no other application can access them. Furthermore, it is possible to encrypt them in the key store. However, this forces the user to either enter a PIN or an alphanumeric password as phone lock security[11].

Stored Properties The *Stored Properties* provide a convenient way to store data that does not need to be protected and does not need to fulfil any security requirements. Therefore, they are used to store non-sensitive user related information. This includes the username, the user's public key and the server's IP address.

[11] `http://developer.android.com/reference/java/security/KeyStore.html`, accessed 2013-12-24

CHAPTER 5

Evaluation and Critical Reflection

This chapter justifies the way of implementation and deals with issues that arose. Finally, the implemented prototype is evaluated using the proposed framework.

5.1 Prototype Implementation Results and Problems

During implementation of the protocol a few simplifications were made in order to keep the prototype simple and to focus on the main purpose, which is the exchange protocol itself. Therefore, only the protocol and parts of the surrounding environment were implemented.

One of the first things encountered was that **NFC only supports a uni directional transfer of data and only one data package at a time in P2P (Person-to-Person) mode** on Android[1]. Therefore, it was not possible to implement the protocol in a way which allows the whole money exchange in a single continuous transaction. This led to the implementation of three different phases as introduced in chapter 3.3.2.3. For each phase an own data exchange was performed. Twice from **A** to **B** (phase one and three) and once from **B** to **A** (phase two). Hence, three data exchanges are carried out while performing a transaction.

This was considered a disadvantage at first. However, it allows the users to only press the phones together when data is exchanged rather than during the whole transaction. Furthermore, it increases security and usability, since it is easier to enter and review the data as well as enter the PIN hidden from the other participant.

In chapter 3.3.2.2 it was argued that XML is used by many different industry standards. Therefore, XML was used to encapsulate the data that needs to be transferred. This allows for a well structured data format in a self-descriptive way. This allowed for transferring the object data as a text string between the devices and also to the server. Since **manipulating**

[1] `http://developer.android.com/guide/topics/connectivity/nfc/nfc.html#p2p`, accessed on 2013-12-25

XML without a framework is difficult, Java object serialisation was used to make this process easier. However, some problems were encountered during the implementation. The problems arose while implementing the signing and validating part of the prototype. Since not the whole transaction and only parts are signed, the part that needs to be signed has to be extracted from the whole transaction. This first step is a simple task in Java, because the XML hierarchy is represented in a hierarchy of Java objects. Therefore, the parts can simply be retrieved by using *getter*-methods. However, signing the data was difficult, because for signing and verifying the data needed to be exactly identical. Even though all data was in the transmitted text string, the serialisation and deserialisation API (Application Programming Interface) changed the data order in the output string. This led to the strings not being exactly identical and therefore the data validation failed. After shifting the order of the attributes within the POJOs it was possible to validate the tokens.

Therefore, even though the designed representation of the transaction token worked as expected, a simpler, text based representation close to the EDI standard [17] or based on Google's Protocol Buffers would make this task easier. The heterogeneity advantages of XML described by C. Huemer [15] do not apply here because non-textual information will never be transferred with this protocol. Furthermore, the data will only be used to transfer payments with homogenous systems[2] and not between different e-commerce solutions.

During a transaction a lot of different signatures are created and validated, which needs computational power. During a single transaction **three signatures are created**[3] **and at least five validations of signatures**[4] **are performed**. The expectation, that this computational effort would challenge state-of-the-art phones was proven wrong during the testing. The only part of the process that takes a relatively considerable amount of time[5] is the creation of the user's signature on the phone when registering for the service. However, since this has to be done once, it can be neglected. Nonetheless, there are still many performance improvements that can be made. Potential areas of improvement are signing, transmitting and validating of data.

Simplicity implies that only a short amount of time is needed to conduct a transaction. This is an important factor for a new system to get accepted by potential users as it has been discussed. Therefore, the duration to conduct a transaction was measured. The **amount of time needed for a transaction was around 25 seconds**. The time was measured with the current prototype implementation of the protocol. It is not caused by a low app performance, but by the time needed for manually entering PINs on the devices. The time could be improved with user interface tweaks or if the PIN only needs to be entered once per person (e.g. when starting the app) and the send button does not need to be pressed.

Handling the complete transaction token was another difficult task. In the current imple-

[2] It is more likely that only a homogenous system, hence only one provider, for this protocol will be available.

[3] One in each phase.

[4] Two before the second phase starts (signature phase one and A's authenticity) and three before phase three (signatures from phase one and two, and B's authenticity).

[5] Around one to two seconds, depending on the phone model.

mentation of the prototype only one transaction at a time can be sent to the server. Therefore, as soon as the *Sync* button is pressed, a loop sending all tokens to the server that have not been synchronised is started. For each received token a hash code is generated and it is checked whether the token has already been sent by the other participant to the server. This was the **first step towards improving the performance** of the system, because the existence of the same hash code in the database implies that the data in the token has been validated before. Therefore, the whole token does not need to be split into its parts again for detailed analysis.

5.2 Known Limitations

Proposing a protocol which allows for exchanging money offline between two persons has some limitations compared to other systems that always have a server connection.

One of the main issues compared with other online solutions is that **offline currency exchange is not possible** with this protocol. This is the case because without an internet connection the current exchange rate cannot be retrieved and money cannot be *generated* offline. However, this does not imply that a user cannot retrieve money in a foreign currency and send it to someone else afterwards.

The protocol **only works as pre-paid system** (see chapter 1.6.2). The is because the systems needs to make sure that the user really owns the money he or she is about to spend. Without an internet connection this can only be ensured when there is proof that the user owns the money. This is done by transferring the money to the phone in advance.

This implies that if money was loaded onto the phone from the payment service directly (e.g. a bank account or similar) it can be lost under certain circumstances. Examples for such circumstances are losing the device, resetting it to factory defaults (see chapter 5.3 for further details) and theft. The reason for the possibility of losing money is that in each of the above cases the user cannot proof if the money was still on the phone or if it has already been transferred to another user. However, if **B** received the money from another user, the money is not necessarily lost, because **A** still has the exact same money token (see chapter 3.3.2.3). Should the case occur, that **A** as well as **B** lose their phones the transaction information, hence the money, is irreplaceably lost. This behaviour can be compared to Bitcoin (see chapter 2.3.1.1) because if the user's private Bitcoin address is lost, all the bitcoins associated with it are also irreplaceably lost. This has happened before with big amounts of money in Great Britain[6]. Also Quick (see chapter 2.3.4.2) has to deal with the same problem, because when the smart card is lost, the money is irretrievably lost as well.

As soon as the money was transferred to the phone, the server cannot track where the money is as long as the phone has not been synchronised. Therefore, the server cannot be sure where the money is and if its still in possession of the user who lost his or her phone. For example, if the user puts €10 on his or her phone, sends the entire amount to someone else and loses the phone afterwards, he or she is not in possession of the money anymore. If the money is still on the phone when losing it, he or she would still be in possession of it but the server does not

[6] *Suche nach Festplatte mit 4,8 Millionen Euro Bitcoins:* `http://derstandard.at/1385169320372/Suche-nach-Festplatte-mit-48-Millionen-Euro-an-Bitcoins`, accessed 2013-12-27

know that. This is because there is no proof that the user still has it and thus the money cannot be refunded.

5.3 Handling Errors, Overspending, Fraud Detection, and Prevention

Losing the possibility to always be able to check the data available on the phone creates drawbacks and limitations. Concerning fraud prevention many different factors have to be considered or taken care of upfront to prevent or limit them. Therefore, this chapter covers fraud on purpose and also handling errors by users which could lead to money loss. As figure 5.1 shows, it is divided into five main cases, namely *PR - Phone Reset*, *PS - Phone Stolen*, *PH - Phone Hacked*, *IT - Identity Theft* and *PKT - Private Key Theft*. *PH* is split into four different sub cases indicated by *PH:xx*.

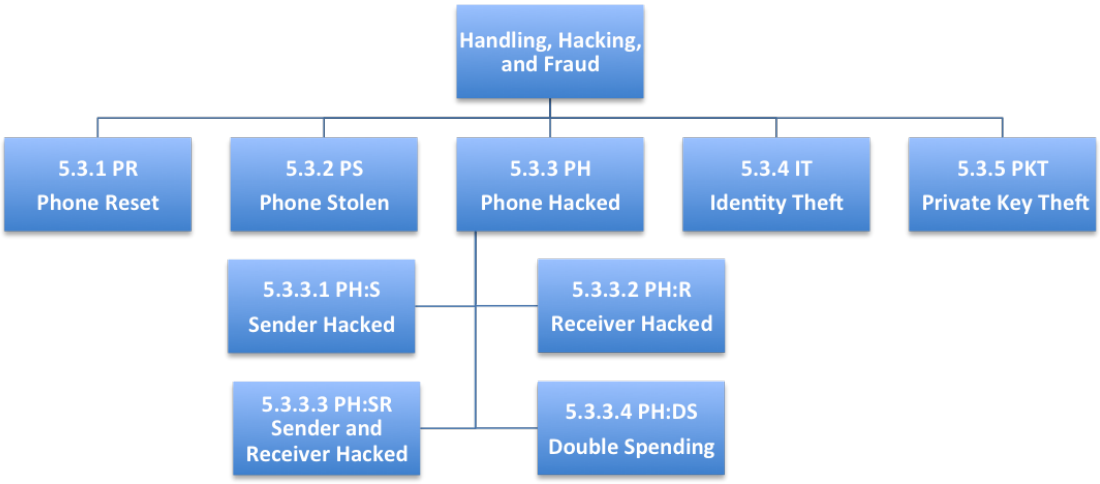


Figure 5.1: Different Ways of Hacking

5.3.1 PR - Phone Reset

This case covers the circumstances when a phone is reset by the user, hence all the data on it is lost. Since the origin purpose of a phone's factory reset function is to get rid of all data on it a data recovery is likely not possible at a later point. As it was argued in chapter 1.2, the proposed protocol should work like a wallet. However, if a wallet gets lost, which is similar to a phone reset, all information is irretrievably gone. This is also the case for this proposed protocol, as it has already been argued before (see chapter 5.2). The created money tokens can be stored anywhere on the phone and are therefore lost as well.

On the upside, since each transaction requires two participants, the transaction partner also holds the exact same token on the phone. Therefore, the money is only lost if either the second participant also erases or loses his or her phone or if the second participant never syncs his or her

phone with the reference account. Money that was transferred directly from the online reference account to the phone cannot be retrieved when the phone's data is lost, as it was argued before.

5.3.2 PS - Phone Stolen

Another big issue is if a phone gets stolen and someone else spends the money using it. The only problem with a stolen phone is that the money that was topped up from the reference account is gone. For money that was received from other users of the system the same applies, as discussed in chapter 5.3.1. Since the user account is protected with a password and a PIN, the money that is still left on the phone cannot be spent by a third party. Therefore, the user's private key cannot be used by anyone else either.

5.3.3 PH - Phone Hacked

When dealing with a hacked phone several cases can occur. A phone that was stolen before can be hacked. This case is covered in chapter 5.3.4. Furthermore, a hacker could use his or her own account to accomplish different tasks. An example is the hacker pretending to have more money on the phone than he or she has topped up before. This could be achieved by manipulating or reverse engineering of the client software and also by trying to take away more money from someone than agreed as well as trying double spending (see chapter 2.3.1.1).

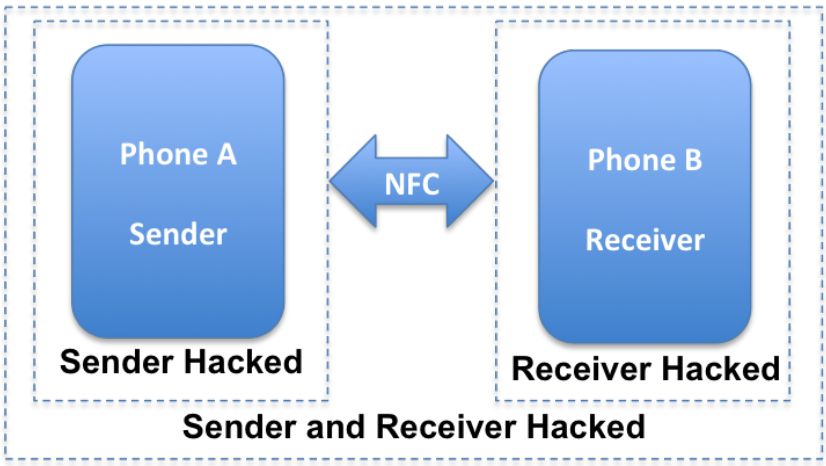


Figure 5.2: Hacked Phone Posibilities

Figure 5.2 illustrates the three possibilities that can occur when someone tries to hack the payment system. If someone takes over either **A**'s phone (**Sender Hacked**) or **B**'s phone (**Receiver Hacked**), not the whole system is compromised and fraud can easily be detected. This is shown in sections 5.3.3.1 and 5.3.3.2. A more severe case, when both **A** and **B** are hacked (**Sender and Receiver Hacked**), is also covered in this section.

The fraud prevention component of the proposed system relies on user authentication and linking a legitimate bank account to the user[7]. Therefore, the user needs to be online at least once before being able to use the payment service. If the necessary registration steps have been taken, the user get the signed public key (see chapter 3.3.2.1). **Therefore, all actions taken by a user can be tracked back to a real person who is responsible for the actions that were taken**.

5.3.3.1 PH:S - Sender Hacked

If only **A** has been hacked two possible scenarios can occur:

- If **A** tries to send different information in the plain text part of the token than he or she signed with his or her private key to change them later. For example, the amount stated in the plain text can be set to a lower value than originally agreed on. Since **B** checks whether what he or she signed corresponds to the plain text, such an attack is be detected immediately.

- **A** tries to spend more money than he or she has on his or her phone, because he or she managed to change the application code of the app. In this case, the user is able to spend more than he or she owns. However, since **A**'s verified information is included in the money token, it can always be tracked back to the user and billed from his or her bank account or credit card.

5.3.3.2 PH:R - Receiver Hacked

A similar situation can occur when **B** tries to forge a money token. For example, **B** stores the first part of the previous transaction (phase one) and reuses it. He or she could do this multiple times by signing it over again with his or her own private key. By doing this, **B** could create fake money transaction from **A** as accredited sender and therefore harm **A**.

However, this would not work because of two reasons. First, **A**'s part of each transaction (phase one) is partly unique because of its random value (see chapter 3.3.2.2). This part would always stay the same and can therefore easily be detected. Second, this type of fraud was also the reason that the third phase was introduced because the last phase ensures, that both participants have signed the transaction, which is again checked and signed by **A**.

5.3.3.3 PH:SR - Sender and Receiver Hacked

This paragraph explains what would happen if both **A** and **B** try to trick the payment service or the server. This could be the case when two people try to make money by faking entire transactions, which could be possible when both parties use an altered version of the app. This type of fraud attempt can also easily be detected and dealt with, since those transactions do not use the properly signed user keys. It can be detected by the server when the users try to synchronise the accounts, since no reference accounts for the used keys can be found by the

[7] This is not part of the current prototype implementation.

payment system. If valid user keys are used, this could also be solved because the payments can be assigned to users and therefore bank accounts.

5.3.3.4 PH:DS - Double Spending

The last case is double spending. This type of fraud has been addressed in many different research papers and books [10, 16, 19, 26, 33, 36, 41]. It deals with the question if it is possible to spend or claim the same token more than once, since this would lead to a severe problem.

The proposed protocol does not allow double spending. Since each created money token needs the contribution from **A** and **B** as well as random random data, each transaction creates a unique token. Even if the same amount is transferred multiple times between the same parties a different unique token is created. Therefore, if **B** tries to claim the amount he or she received more than once, double spending is detected. This is because all parameters of the token are identical, which implies that the token is the same.

In order to speed up the process of double spending detection, a hash code from the complete token is created (see chapter 5.1). Therefore, detection is even possible if only a single character of a money token is changed by a hacker. To achieve this, the random values of all transactions that have the same sender and receiver can be searched for. If this also does not lead to a result, the token were not forged or altered, as the signature checks would have failed otherwise.

5.3.4 IT - Identity Theft: Stolen and Hacked Phone

Another question is what would happen if a hacker or theft gets access to a phone with all credentials needed to use the payment service. This is especially relevant in offline payment systems because the account on a stolen phone cannot easily be deactivated. However, if the thief does not know the PIN or the user's password he or she won't be able to spend the money. A stolen and hacked phone is similar to the situation described in chapter 5.2.

5.3.5 PKT - Private Key Theft

If against all odds (compare to chapter 4) a user's private key is stolen from a phone, the hacker could potentially sign as many tokens as he or she wants to. Certain security measures need to be taken to prevent or at least to limit this, which has only been done partly in the protocol's currents design. This problem is similar to a stolen credit card, where a thief could also use someone else's credit card. Private key theft works because a hacker could install the certificate on a phone with a manipulated version of the client software and act like he or she is someone else. This could lead to money transactions in someone else's name. However, this threat implies that the hacker is able to retrieve the user's private key from the KeyStore.

This problem could be solved as follows:

- Assigning the keys a limited time to live. However, this is a usability problem since the users would have to get new keys on a regular basis. This conflicts with the proposed need of simplicity.

- A rejection list on the phones could be implemented. This list would contain a list of identities that have been stolen and should not be trusted anymore.
- All money that would be stolen from a user should be reimbursed by the payment service as soon as the fraud was reported, as it is the case for most banks[8].

5.4 Evaluation

Chapter 2.4.1 introduced a simple framework for payment systems to make them easily comparable. This framework is now used to evaluate the novel approach and to compare it to the existing solutions described in chapter 2. Therefore, table 5.1 shows an extended version of the table shown in chapter 2.4.1. The highlighted line represents the novel approach.

Table 5.1: Extended Applied State of the Art Evaluation Framework

	Offline	**Secure**	**Anonymous**	**Simple**
Bitcoin		✓	✓	
PayPal		✓		✓
Paysafecard		✓	✓	
Bank Transfer		✓		
Google Wallet		✓		✓
Friendbank	✓			✓
Venmo		✓		✓
Square				✓
Blaze et al.	✓			
Traveler's Check	✓			
NFC Cards				✓
Quick	✓		✓	✓
EasyPay	✓	✓		✓

EasyPay is the only payment system that allows offline and secure payments at the same time while still providing a simple way to conduct transactions. The following list evaluates and analyses EasyPay in more detail.

- **Offline** The proposed solutions supports offline transfers, as it uses NFC to transfer data between the phones.
  - **Availability** Since this solution does not rely on any type of internet connection or a phone signal, the availability is given at all time.

[8] `http://www.raiffeisen.at/eBusiness/01_template1/1006637000974-NA-154480155273655657-NA-30-NA.html`, accessed 2013-12-29

- **Secure** Security is given because it uses different public-private key pairs to sign the money token and to provide authenticity for others.
  - **Double-Spending** Double spending cannot occur because each token has a unique signature. Therefore, it would be noticed when someone tries to use a token more than once.
  - **Validation** The parties of the payment system are legitimate users of the system, as their authenticity is ensured by the payment service.
  - **Fraud Detection and Prevention** Different types of PINs and passwords as well as encryption are used to prevent fraud and to detect people that are not rightful users of the system.
  - **Non-Repudation** Since a transaction is signed with PINs by both its participants and a money token contains the credentials of both users, it can always be assigned correctly. Therefore, it cannot be denied.
  - **Atomic Transaction** Only if all three phases of the transaction are conducted a complete money token is created. If a transaction would be interrupted in an earlier stage, a half completed token would be created that cannot be used.
  - **Integrity** Integrity is ensured because the money and the certificates to sign transactions are securely stored and can only be accessed when providing the right PIN or password.
  - **Confidentiality** Since the whole transaction is carried out over NFC, eavesdropping is not possible. If the complete money token is transferred via a secure HTTP channel to the server (HTTPS), no one can read the transaction.
- **Anonymous** Anonymity is not given but this approach does not claim to provide anonymity.
- **Simple** Since only a PIN or password as well as tapping the phones together to conduct a transaction is needed, it is simple. The user does not need any type of other information to successfully carry out a transaction.
  - **Easy to Use** No additional information is required to conduct a transaction.
  - **Self-Explanatory** There is no complex procedure to use the payment system as it is the case for Bitcoin (see chapter 2.3.1.1).
  - **Low Entry Barrier** Since only a debit or credit card as well as an Android smartphone is needed and no additional software or hardware needs to be bought the entry barrier is low.

As it has been shown, all requirements are fulfilled by the payment approach. Therefore, the claimed attributes *offline*, *secure* and *simple* can be assigned to it.

CHAPTER 6

Summary and Outlook

6.1 Summary

The first part of the thesis focused on analysing existing payment approaches. After an analysis of important payment systems on the market, a comparison between them was conducted. The comparison is based on a simple framework taking system and user requirements (offline, security, anonymous and simplicity) into account. Applying the framework on the analysed payment system showed that it is not possible to create a perfect payment system that fulfils all requirements. It is only possible to fulfil a maximum of three out of four requirements. Security was identified as the most important requirement, which is fulfilled by almost all analysed payment systems. However, the problem is that the more secure a payment system is the more complex it gets (see table 2.1). Furthermore, only one of the analysed payment systems was able to provide anonymity, offline usage and simplicity at the same time, namely the B2C system Quick. The most interesting fact was that there is no payment system for friend payments that is secure and offline at the same time.

The observations that were made when applying the framework, as well as the problem statement discussed in section 1.2, led to the solution described in chapter 3. It allows to conduct payments offline and uses online reference accounts which are needed to keep track of the payments. Therefore, the payments can be seamlessly conducted offline and the payment information is sent to the server when an internet connection is available again.

The solution shows a workflow that is capable to transfer money offline while being secure and simple for the user at the same time. The data that is transferred during the execution of the workflow contains all relevant transaction information for the server to wire the transaction correctly at a later point in time.

When conducting an offline payment a so called money token is created. Each transaction creates a unique money token which can only be used once. This prevents double spending as further explained in chapter 5.3.3.4. For simplicity and expandability the token is a XML formatted string which contains the information and signatures according of the proposed protocol.

The money tokens are signed in three phases by both participants resulting in non-repudiation payment information which can be redeemed when the information is synchronised with the server. The three phases are needed to ensure that both participants have the possibility to verify each other's credentials and to review and sign the token.

The solution also deals with the environment necessary to run such a payment system, hence the account and server infrastructure. The payment server uses the money tokens to correctly wire the money between the sender and the receiver. Since both parties share the same money token after a conducted transaction, it does not matter which of the participants synchronises the token with the server first. This fact also adds additional security, since even if one of the participants loses his or her phone, the transaction can still be wired. Furthermore, all payment system users are accredited by the payment server with a server signed signature, which can be verified offline by every user with the server's public key.

This shows that there is a way to conduct secure and offline C2C payments without the need for trusted hardware.

In order to prove that the idea also works in practise, an implementation in form of a prototype was developed for smartphones running Android Jelly Bean or higher. The client uses the proposed solution as guideline. Apart from a drawback encountered in terms of handling the data exchange with NFC, it was possible to implement the prototype as planned. The resulting client in its current version consists of three separate data exchanges via NFC and each data exchange represents a phase according to the protocol.

In addition of the Android client some parts of the environment, such as the server structure were implemented as well. However, only the parts necessary to show how wiring of transactions works were implemented.

This novel approach focuses on security, availability and simplicity. As shown in chapter 2.4 it is not possible to create a payment system that is offline, secure and anonymous at the same time. This also applies to the novel approach as shown in chapter 5.4, but it is also shown that this is the first approach which combines security and simplicity while still being easy to use without a phone signal or an internet connection. This was achieved by using NFC to transfer the payment information. This also makes transferring money easier and more convenient at the same time. Since NFC is a technology that is supported by more and more smartphones (see chapter 2.2), a big number of smartphone users can be reached. This is possible because a standard NFC-enabled smartphone is sufficient to be able to use this solution and there is no need for any kind of special hardware like a secure element or trusted hardware.

6.2 Outlook and Open Issues

Although this thesis provided a conclusive design of a C2C payment protocol which allows for conducting payment offline between friends, a lot of research can still be done in different directions. The following ideas can be seen as starting points for further research. The ideas do not claim to be conclusive.

User Authentication In the current implementation the user is accredited as legitimate user by the payment service. However, since there is quite some effort needed to accredit a user, different improvements could be made here. This could be achieved for example by making the user enter his or her bank account information and send his or her passport or drivers license to complete the registration.

- **Mobile Phone Provider** Since in most cases a mobile phone user has a contract with a mobile phone provider, the phone number can be used as a unique identifier and for settling payments.
- **PayPal** EasyPay could be used as an extension to PayPal and cleared not with a bank account or credit card directly but rather via a PayPal account. This would make it simpler for the user to enter his or her credentials since only a username and a password are needed. However, the downside with this approach is that PayPal takes a small amount of money as fee, which would have to be paid by the users as well.
- **National PK** M. Hassinen et al. [13] proposed a different way to authenticate users. Since it is cumbersome to authenticate oneself via a mobile phone, especially when more than a credit card number and a name are needed, the national public-key infrastructure (the *Bürgerkarte* in Austria) could be used to authenticate oneself.

Advanced Security Features The reason why no advanced security features have been used for this approach is that as many people as possible should be able to use the payment system. Therefore, for people owning phones that provide advanced security features, simplifications in the protocol could be made to make it even easier to use.

- **Secure Element** Most NFC controllers provide a secure element to store sensitive data. Since this is not part of every phone with a NFC controller, it is not part of the current protocol and Android KeyStore is used instead. Parts of the application code could be executed on the secure element [35] to prevent some of the issues discussed in chapter 5.3.
- **Trusted Hardware** A feasibility study that tries to find out if it would make sense to use trusted hardware would be another starting point towards hardware related security features.
- **Fraud and Money Laundry Analysis** Each new payment system is a potential candidate for illegal actions. Therefore, another security feature that could be implemented on the

server side is an automated analysis for abnormalities in the synchronised transactions taking place.

Protocol Design Extensions In chapter 3 the first version of the protocol was introduced. This protocol could be extended and improved in different ways.

- **Encoded Limits** Blaze et al. [5] proposed a solution that encodes certain limits in the checks that are provided by the bank. This would for example allow to encode the age in the check so children cannot buy alcohol and to set a maximum amount of money that can be spent and received. A similar system could be developed for this novel approach in order to make it more convenient for children and their parents.
- **One Way Hashes** The current solution uses signatures to verify the money tokens. As it is proposed by [26] one way hash codes could be used to lower the amount of data that needs to be sent. For example, the sender's and receiver's information could be stored in hash format.

Usability There is also still a lot of improvement in terms of usability that can be done, since the prototype focused on showing that the proposed protocol works. Therefore, the following ideas that came up during implementation could be further analysed.

- **PIN** At the moment the PIN needs to be entered three times to accredit the transaction. Twice by the sender and once by the receiver. When the PIN only needs to be entered once when starting the application, improvements in terms of the time needed to conduct a transaction could be made. The same applies for the send button, which is not necessary.
- **Online Mode** If a connection to the internet is available a simplified version of the protocol could be used, which only needs one NFC data exchange.

System Extensions Apart from the protocol the whole environment has potential for improvement.

- **B2C** One of the next steps would be to see if B2C payments would also be possible with this protocol and payment system.
- **Wallet** Since NFC is very important for this protocol and its implementation, other features that require NFC could be implemented as well. The application could for example be extended by a digital wallet which allows to emulate debit or credit cards. This would be the next step to get rid of the real wallet.

List of Figures

List of Tables

APPENDIX A

Comparison Table

Table A.1: Comparison: EasyPay, Bitcoin, and Erstebank Friendbank

	EasyPay	**Bitcoin**	**Friendbank**
Offline/Online	offline	online	offline
Clearing Only	yes	no	-
Payment Time	pay-now/later	pay-now	-
Anonymity	no	yes	-
Small Payment	yes	yes	-
Degree of Popularity	-	worldwide	AT only
Douple-Spending Protection	ensured	ensured	-
Issueing New Currency	debit/credit card	mining or debit/credit	-
Security	pin, nfc	mining	-
Fee Receiving	free	free	-
Fee Sending	free	0,0001 BTC[1]	-
Users	bank account / CC	PC / smartphone	iOS / Android
Transaction	balance/credit	balance	-
Risk	losing debit amount	losing virtual coins	-
Type	C2C	C2C, B2C, B2B	-
PROs	offline	worldwide, P2P protocol	-
CONs	only C2C	own currency	-

[1] `https://bitcointalk.org/index.php?topic=219504.0`, seen 2013-12-13

Table A.2: Comparison: PayPal, Quick, and PayPass/PayWave

	PayPal	**Quick**	**PayPass/PayWave**
Offline/Online	online	offline	online
Clearing Only	yes	no	no
Payment Time	prepaid/pay-now	pay-now	pay-later
Anonymity	no	no	no
Small payment	yes	yes	yes
Degree of Popularity	wordwide	AT only	worldwide
Douple-Spending Protection	ensured	ensured	ensured
Issueing New Currency	debit/credit card	bank account	credit card
Security	email/pw	none	depends
Fee Receiving	1,9% to 3,4% + 0,35 €[2]	free	free
Fee Sending	free	free	free
Users	E-mail address and CC	bank card	enabled CC
Transaction	balance/debit/credit	balance	credit
Customer's Risk	account freezing	loosing card	stolen CCs
Type	C2C, B2C	B2C	B2C
PROs	worldwide	offline, easy usage	little payment time
CONs	online	money only on card	only B2C

[2] `https://www.paypal.com/at/webapps/mpp/paypal-fees`, seen 2013-12-13

Table A.3: Comparison: Google Wallet, Paysafecard, Square Cash/Wallet

	Google Wallet	**Paysafecard**	**Square Cash/Wallet**
Offline/Online	online	online	online
Clearing Only	yes	no	no
Payment Time	pay-now	pre-paid	pay-now
Anonymity	no	yes	no
Small payment	yes	yes	yes
Degree of Popularity	US only	30 countries	US, Canada, Japan
Douple-Spending Protection	ensured	ensured	ensured
Issueing New Currency	via debit or credit card	at a store	via bank account
Security	PIN, NFC	coupons[3]	location, photo, name
Fee Receiving	free	free	-
Fee Sending	2.9%[4]	free	-
Users	multiple[5]	everyone	CC owners
Transaction	balance/debit/credit	balance	credit
Risk	hackers[6]	lost vouchers	email address, inattentive waiter
Type	B2C, C2C	B2C	B2C, C2C
PROs	combines multiple CCs	anonymity	no barriers to use
CONs	US only, stolen phone	little convenience	not spread, little security

[3] `https://www.paysafecard.com/en-gb/security/`, seen 2013-12-27
[4] `http://www.google.com/wallet/faq.html#tab=faq-fees`, seeon 2013-12-27
[5] `https://support.google.com/wallet/answer/1347934?hl=en`, seen 2013-12-13
[6] `http://cybersecurity.mit.edu/2012/10/google-wallet-overview-threats-and-security-measures/`, seen 2013-12-27

Table A.4: Comparison: Venmo, Bank Transfer / Netbanking, Traveler's Check, Offline Payment

	Venmo	**Bank Transfer / Netbaking**	**Traveler's Check**	**Offline Payment [5]**
Offline/Online	online	online	online/offline	offline
Clearing Only	no (venmo balance)	no	no	
Payment Time	pay-now	pay-now	pay-later	pay-later
Anonymity	no (public possible)	yes	no	yes
Small payment	yes	yes	yes	yes
Degree of Popularity	US only	worldwide	none	none
Douple-Spending Protection	ensured	ensured	ensured	ensured
Issueing New Currency	via debit or credit card	bank transfer	buying checks	later via debit/credit
Security	pin	PIN/TAN/mobileTAN/password	the check itself	risk mgmt.
Fee Receiving	free	free	-	-
Fee Sending	depends[7]	flatrate	-	-
Users	iPhone owners	bank account owners	-	-
Transaction	balance/debit/credit	balance/credit	credit	-
Risk	lost phone	hacked account	phising, fraud	lost account number
Type	C2C	C2C, B2C, B2C	B2C	B2C
PROs	usability	worldwide	online/offline	offline
CONs	US only	only B2C	only theoretical	only theoretical

[7] `https://venmo.com/info/fees`, seen 2013-12-27

Bibliography

[1] Fadi Abdulhamid and Ezz Hattab. A Model for Person-to-Person Electronic Payment System. 2008.

[2] Accenture. Mobile Web Watch 2013. *URL: http://www.accenture.com/SiteCollectionDocuments/PDF/Technology/accenture-mobile-web-watch-2013-survey-new-persuaders.pdf*, 2013.

[3] Milton M. Anderson. The Electronic Check Architecture. *URL: http://www.echeck.org/library/wp/ArchitectualOverview.pdf*, 2013-12-04.

[4] Nadarajah Asokan, Phillipe A Janson, Michael Steiner, and Michael Waidner. The State of the Art in Electronic Payment Systems. *Computer*, 30(9):28–35, 1997.

[5] Matt Blaze, John Ioannidis, and Angelos D Keromytis. Offline Micropayments without Trusted Hardware. In *Financial Cryptography*, pages 21–40. Springer, 2002.

[6] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *Advances in Cryptology–EUROCRYPT 2005*, pages 302–321. Springer, 2005.

[7] Glyn Davies. History of Money, A. *University of Chicago Press Economics Books*, 2005.

[8] Roy T Fielding and Richard N Taylor. Principled Design of the Modern Web Architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2):115–150, 2002.

[9] Eran Gabber and Abraham Silberschatz. Agora: A Minimal Distributed Protocol for Electronic Commerce. In *The Second USENIX Workshop on Electronic Commerce Proceedings*, pages 223–232, 1996.

[10] Flavio D Garcia and Jaap-Henk Hoepman. Off-line Karma: A Decentralized Currency for Peer-to-peer and Grid Applications. In *Applied Cryptography and Network Security*, pages 364–377. Springer, 2005.

[11] Andrés Guadamuz González. PayPal and eBay: The Legal Implications of the C2C Electronic Commerce Model. In *18th BILETA Conference: Controlling Information in the Online Environment, April, QMW, London*, 2003.

[12] Google Inc. Google Wallet. *URL: http://www.google.com/wallet/faq.html*, accessed 2013-11-29.

[13] Marko Hassinen, Konstantin Hyppönen, and Elena Trichina. Utilizing National Public-Key Infrastructure in Mobile Payment Systems. *Electronic Commerce Research and Applications*, 7(2):214–231, 2008.

[14] Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. Design Science in Information Systems Research. *MIS quarterly*, 28(1):75–105, 2004.

[15] Christian Huemer. XML vs. UN/EDIFACT or Flexibility vs. Standardization. 2000.

[16] Min-Shiang Hwang and Pei-Chen Sung. A Study of Micro-payment Based on One-way Hash Chain. *IJ Network Security*, 2(2):81–90, 2006.

[17] International Organization for Standardization (ISO). *Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT)*, 1988. ISO9735:1988.

[18] International Organization for Standardization (ISO). Financial Services - International Bank Account Number (IBAN). *ISO 13616-1:2007*, 2007.

[19] Ghassan O. Karame, Elli Androulaki, and Srdjan Capkun. Double-Spending Fast Payments in Bitcoin. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, pages 906–917, New York, NY, USA, 2012. ACM.

[20] Maik Klotz. Mobile Payment: Die Big Five und warum Startups keine Chance haben, 12 2013.

[21] Heinz Kreft, Manfred Schimmler, and Achim Walter. *FairCASH based on Loss Resistant Teleportation*. Shaker, 2011.

[22] Dennis Kügler and Holger Vogt. Offline Payments with Auditable Tracing. In Matt Blaze, editor, *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 269–281. Springer Berlin Heidelberg, 2003.

[23] Josef Langer and Michel Roland. *Anwendungen und Technik von Near Field Communication (NFC)*. Springer DE, 2010.

[24] Zon-Yau Lee, Hsiao-Cheng Yu, and Pei-Jen Ku. An Analysis and Comparison of Different Types of Electronic Payment Systems. In *Management of Engineering and Technology, 2001. PICMET'01. Portland International Conference on*, pages 38–45. IEEE, 2001.

[25] Haifei Li. XML and Industrial Standards for Electronic Commerce. *Knowledge and Information Systems*, 2(4):487–497, 2000.

[26] Horng-Twu Liaw, Jiann-Fu Lin, and Wei-Chen Wu. A new Electronic Traveler's Check Scheme based on One-Way Hash Function. *Electronic Commerce Research and Applications*, 6(4):499–508, 2008.

[27] Martin Manninger and Robert Schischka. Adapting an Electronic Ourse for Internet Payments. In *Information Security and Privacy*, pages 205–214. Springer, 1998.

[28] Gennady Medvinsky and Clifford Neuman. NetCash: A Design for Practical Electronic Currency on the Internet. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 102–106. ACM, 1993.

[29] Peter Mooslechner, Helmut Stix, and Karin Wagner. The Use of Payment Instruments in Austria. *Monetary Policy & the Economy*, (4):53–77, 2012.

[30] Mohamed Nabeel and Elisa Bertino. Secure Delta-Publishing of XML Content. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 1361–1363. IEEE, 2008.

[31] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. *URL: http://www.bitcoin.org/bitcoin.pdf*, 2009.

[32] Ayo Omojola. Why Mobile Retailers Should Ditch Credit Cards, 12 2013.

[33] Tomi Poutanen, Heather Hinton, and Michael Stumm. NetCents: A Lightweight Protocol for Secure Micropayments. In *USENIX Workshop on Electronic Commerce*, pages 25–36, 1998.

[34] Fergal Reid and Martin Harrigan. An Analysis of Anonymity in the Bitcoin System. In Yaniv Altshuler, Yuval Elovici, Armin B. Cremers, Nadav Aharony, and Alex Pentland, editors, *Security and Privacy in Social Networks*, pages 197–223. Springer New York, 2013.

[35] Marie Reveilhac and Marc Pasquet. Promising Secure Element Alternatives for NFC Technology. In *First International Workshop on NFC*, pages 75–80, 2009.

[36] Ronald L Rivest and Adi Shamir. PayWord and MicroMint: Two Simple Micropayment Schemes. In *Security Protocols*, pages 69–87. Springer, 1997.

[37] Michael Roland. Software Card Emulation in NFC-enabled Mobile Phones: Great Advantage or Security Nightmare? In *Fourth International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use (IWSSI/SPMU 2012)*, page 6, 2012.

[38] Kenji Saito. Peer-to-Peer Money: Free Currency over the Internet. In *Web and Communication Technologies and Internet-Related Social Issues HSI 2003*, pages 404–414. Springer, 2003.

[39] Kim Sangjin and Oh Heekuck. Efficient Anonymous Cash Using the Hash Chain. *IEICE Transactions on Communications*, 86(3):1140–1143, 2003.

[40] Marvin Sirbu and Douglas Tygar. NetBill: An Internet Commerce System Optimized for Network-Delivered Services. *Personal Communications, IEEE*, 2(4):34–39, 1995.

[41] Simon Sprankel. Technical Basis of Digital Currencies. 2013.

[42] Square Inc. Square Cash. *URL: https://square.com/cash*, accessed 2013-11-27.

[43] William Stallings. *Computer Security: Principles And Practice*. Pearson Education India, 2008.

[44] Karsten Stroborn, Annika Heitmann, Kay Leibold, and Gerda Frank. Internet Payments in Germany: A Classificatory Framework and Empirical Evidence. *Journal of Business Research*, 57(12):1431–1437, 2004.

[45] Viswanath Venkatesh, Venkataraman Ramesh, and Anne P Massey. Understanding Usability in Mobile Commerce. *Communications of the ACM*, 46(12):53–56, 2003.

[46] A Visa and MasterCard Electronic Commerce Collabaration. SET: Secure Electronic Transactions. 1997.

[47] Nick Wood. NFC Handset Shipments Reached 30M in 2011, 12 2013.

[48] Beverly Yang and Hector Garcia-Molina. PPay: Micropayments for Peer-to-Peer Systems. In *Proceedings of the 10th ACM conference on Computer and communications security*, pages 300–310. ACM, 2003.