We hereby certify that this dissertation, submitted by **Mohammad Vahidalizadehdizaj** satisfies the dissertation requirements for the degree Ph.D. in Computer Science and has been approved.

Lixin Tao — 5/10/17

Dr. Lixin Tao — Date
Chairperson of Dissertation Committee

5/10/17

Dr. Charles Tappert — Date
Dissertation Committee Member

M. Badi — 5/10/17

Dr. Mehdi Badir — Date
Dissertation Committee Member

Seidenberg School of Computer Science and Information Systems
Pace University

An Efficient Decentralized Mobile Payment Protocol With Improved Security and Privacy

A DISSERTATION PRESENTED
BY
MOHAMMAD VAHIDALIZADEHDIZAJ
TO
THE DEPARTMENT OF COMPUTER SCIENCE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN THE SUBJECT OF
COMPUTER SCIENCE

PACE UNIVERSITY
NEW YORK, NY
MAY 2017

Thesis advisor: Dr. Lixin Tao Mohammad Vahidalizadehdizaj

An Efficient Decentralized Mobile Payment Protocol With Improved Security and Privacy

Abstract

The exponential growth of mobile devices makes them a suitable computing platform for electronic payment. However, there are serious challenges in e-commerce transactions, such as privacy protection, security, bandwidth limitations of mobile networks, and limited capabilities of mobile devices to handle excess or indirect computational time. The traditional e-commerce payment protocols that were originally designed to keep track of the traditional flows of data from desktop computers are vulnerable to attacks, and because they were not designed for mobile platforms, have excessive engineering overhead. In this thesis, a new private mobile payment protocol is introduced that is designed specifically for the mobile platform. It is based on a client-centric model that utilizes symmetric key operations. The protocol reduces the computational cost (the engineering overhead) of Diffie-Hellman key agreement protocol by using the algebra of logarithms instead of the algebra of exponents. The protocol achieves proper privacy protection for the payer by involving mobile network operators and generating temporary identities. It avoids replay attacks by using random time-stamp generated numbers.

Contents

Listing of figures

I dedicate this thesis to my mother and father.

Acknowledgments

I would like to express my grateful appreciation and thanks to God. You gave me strength to never give up even in the hardest situations. A special thanks to my family. Words cannot express how grateful I am to my mother and father for all of the sacrifices that they made on my behalf. Their prayers for me sustain me always. I would also like to thank all of my friends who supported me in writing and encouraged me to strive towards my goal of contributing to the body of knowledge of Computer Science.

Lovers find secret places inside this violent world where they make transactions with beauty.

Jalaluddin Rumi

1

Introduction

E-commerce is commerce via the internet. E-commerce is any financial transaction over Internet like ordering a book from an online bookstore. Most of the time payer uses his credit card in this process. An e-commerce transaction involves purchaser or cardholder, merchant, purchaser's credit card issuer (bank), merchant's acquirer (bank), and certification authority for supporting secure transaction execution. Most of these protocols are using Diffie-Hellman for establishing a secure connection between the engaging parties. Most important challenges in this field are security and privacy. Mobile commerce or m-commerce is electronic commerce conducted via the mobile platform. An m-commerce transaction involves all e-commerce parties plus mobile network operators. [37].

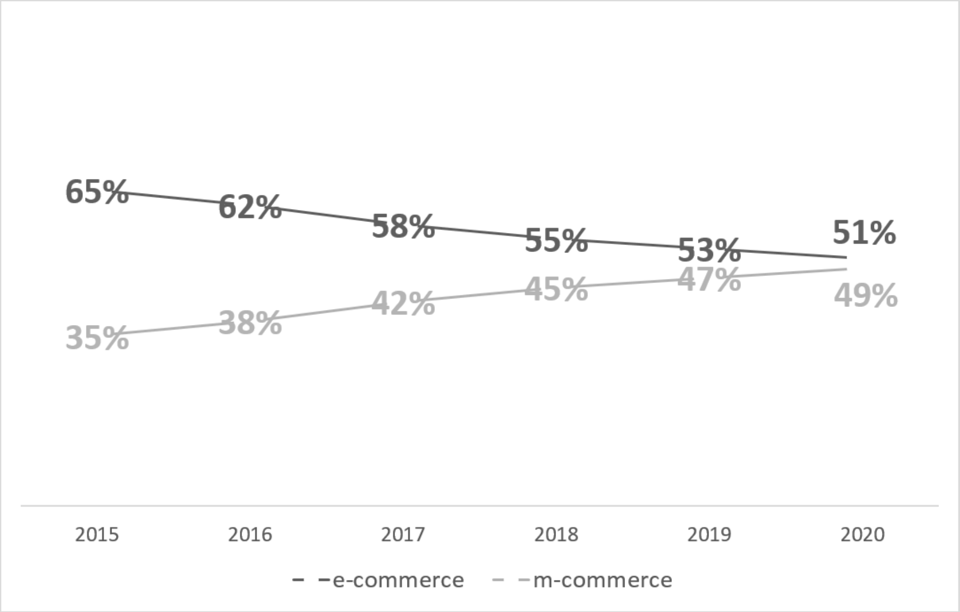


Figure 1: Mobile commerce share of e-commerce

In a typical payment scenario, the customer should open a credit card account in a bank that supports electronic payment. The payer may receive a digital certificate signed by the bank (based on the protocol that his card issuer is using). Then, the customer selects his items or services by browsing merchant's website and order them. The customer should send a message that includes two parts. The first part that includes order information is for the merchant. The second part is for merchant's bank that includes payment information. Merchant's bank should check this payment information with the issuer of the credit card for authorization. After a successful authorization, merchant's bank informs the merchant that the payment information is acceptable. Then, merchant completes the order, sends confirmation and invoice to the customer, and captures the transaction from his bank. Note that, for m-commerce, the mobile network operator may be involved in this process.

Mobile commerce (m-commerce) is e-commerce activities conducted via the mobile platform. Principals of m-commerce are the same as e-commerce plus mobile network operator. M-commerce inherits the challenges of e-commerce. Moreover, most of the e-commerce protocols are based on public key cryptography that is not efficient in mobile and wireless networks. Some of these protocols are keeping credit cards information on mobile devices or using this information in transactions without proper protection. Therefore, they are vulnerable to attacks [35].

Mobile devices like smartphones and tablets are becoming very popular among people worldwide. People carry these devices and use them in different situations and locations. There are more mobile devices in the world than there are computers. Also, these mobile devices are the most accessible computer in daily lives. Most of these devices are light, easy to carry, and convenient to use. Mobile devices are compatible with networks like 4G LTE that is available in outdoor spaces. More than half of the internet access in the world is through the mobile devices (instead of personal computers). Also, people would rather order their needs (goods or services) online through their smartphone [3].

The growth of m-commerce sales continues to be rapid even with the challenges that m-commerce face like slow download times. Forrester predicted 11 percent (of whole e-commerce) growth in m-commerce between 2016 and 2020. Currently, m-commerce has 35 percent of e-commerce transactions. Forrester predicts that m-commerce will be 49 percent of e-commerce in 2020. This amount is 252 billion dollars in sales. You can see m-commerce growth forecast in figure 1 [5]. The amount of Chinese mobile payment (m-commerce), has exceeded Japan's GDP. This is because of reliable mobile payment services in China. China smartphone payments have been doubled to about USD5.44 trillion in 2016. It seems going out without cash and wallet is becoming reality for the Chinese people by the help of mobile payment [1].

1.1 Mobile Payment Challenges

M-commerce has its own challenges like security and privacy. Mobile devices have limited computational power, less stable network, limited memory, limited storage, and etc. Most of the e-commerce protocols are based on public key cryptography that is not efficient for mobile devices and wireless networks. The client may need to do heavy calculations in this type of cryptography. So, this type of cryptography is not suitable for mobile platforms.

Some of these protocols are using a mechanism to authenticate the certificate of the engaging parties. However, this step may become costly for a mobile device via a network that is not as stable as a wired network connection. Some of these protocols are keeping credit card's information on mobile devices or using this information in transactions without proper protection. These issues make the existing protocols vulnerable from the aspect of security. Most of these protocols were designed to keep track of the traditional flow of data. This flow should be carried between client and merchant as a transaction. These protocols are vulnerable to attacks like transaction or balance modification attack. There is no proper notification in these protocols after a successful transaction as part of the protocol.

1.2 Desired Mobile Payment Protocol Attributes

A mobile payment protocol should be defined that is suitable for the mobile platform. This protocol should decrease the computational cost of payment process in order to make it suitable for the mobile platform. The computational power, communication speed, and variable size are the most important mobile platform limitations. Also, customer's privacy protection should be another feature of this protocol. Note that, privacy protection is a significant challenge especially in a mobile platform. Non-repudiation can be provided by using digital signature. However, non-repudiation should be an optional feature, since it may have an extra computational cost for the mobile devices.

Furthermore, to avoid replay attacks in the protocol, random time-stamp generated numbers should be generated in its steps. Cloud messaging should be utilized to provide an extra layer of security for the payment protocol to prevent card not present fraud. Cloud messaging is available in all mobile operating systems like GCM for Android, BBPS for Blackberry, APNS for Apple, and MPNS for Microsoft. So, the customer can easily see the origin of the message when it comes as a cloud message. This protocol should help people to make their payment via their mobile devices in a secure and efficient way.

The recommended protocol should be based on the client-centric model. Temporary payer's identity, involving mobile network operators, and utilizing random time-stamp generated numbers should be used to provide suitable privacy protection for the payer and avoid replay attacks. Without this temporary identity, the payer's privacy will be more vulnerable to attacks. One of the goals is to hide payer's identity from merchant to protect his privacy (optional feature). The digital signature could be utilized in the payment protocol to provide non-repudiation. An improved key agreement protocol should be used instead of Diffie-Hellman. The protocol should be suitable for mobile devices from aspect of computational and communication cost.

1.3 Limitations of Current Solutions

There are many payment protocols for e-commerce like SET, iKP, KSL, and etc. Existing payment protocols are not designed for mobile devices and mobile or wireless networks. These protocols have heavy computations for the engaging parties. These computations are not suitable for the mobile platform. The cost of communication may be large in these protocols, since certificate authentication may be required. These protocols are using PKI encryption that is not proper for the mobile platform because of its heavy computational and communication cost.

The mobile platform has limitations in computational power and network stability and network bandwidth. So, the existing protocols are not suitable for this platform. These protocols don't protect customer's privacy. Also, many people abandon online payment because of the issues in 3D Secure implementations [35]. Currently, 3DS implementations like Verified by Visa are a pioneer in online payment. However, none of these protocols are widely accepted by people so far.

1.4 Problem Statement

There are different types of limitations in m-commerce and mobile platform in comparison with other common e-commerce platforms. Another significant challenge in m-commerce is identity protection. There are several situations that people can use m-commerce protocols instead of e-commerce protocols. However, m-commerce protocols are not suitable for mobile platform. So, a payment protocol suitable for mobile platform is needed for m-commerce.

There are some assumptions in this protocol. It is assumed that the payer has a credit card that can be used in online transactions. It is assumed that the payer is using a mobile device through a mobile network like 4G LTE. It is assumed that their mobile network operators are supporting the recommended protocol. It is assumed that there exists a payment gateway named pay-center. This payment gateway is the medium between the banks and the other parties. The payment gateway will commit the transaction between the payer and the merchant.

In this m-commerce protocol, the focus should be on decreasing the computational cost and communication cost in the transactions. Decreasing the computational cost of the secure channel establishment protocol (Diffie-Hellman) will be suggested. The goal is to design a payment protocol for mobile platform.

This mobile payment protocol should have lower computation cost in mobile devices. This protocol should use symmetric encryption instead of PKI encryption. This protocol should provide privacy protection for the customer. Using temporary ID for the customer will be suggested. This ID should not be reusable. It should be only for that transaction. Replay attack should not be possible in this protocol. Using random and time-stamp generated numbers in the transaction will be recommended.

1.5 Contribution

Contributions in this research are:
1) Introducing a new private mobile payment protocol based on client-centric model that is suitable for mobile devices via mobile and wireless networks.
2) Suggesting an improved version of Diffie-Hellman key agreement protocol for establishing secure connections among engaging parties. Logarithm will be used instead of powering in the process of the recommended key agreement protocol. This change will make the computation cost less and make the process proper for mobile platform even from aspect of required variable size.
3) Providing proper privacy protection for the payer by involving mobile network operators in the payment process and generating temporary identities.
4) Decreasing the risk of replay attacks by using random time-stamp generated numbers in the payment process.

1.6 Dissertation Roadmap

Related materials will be reviewed in chapter 2. The recommended key agreement protocols will be described in chapter 3. Chapter 4 describes the improved mobile payment protocol. Chapter 5 describes the experiments. The research will be concluded in chapter 6.

You think of yourself as a citizen of the universe. You think you belong to this world of dust and matter. Out of this dust you have created a personal image, and have forgotten about the essence of your true origin.

Jalaluddin Rumi

2

Literature Review

A key agreement protocol is a protocol to generate a shared session key between two parties. In this chapter, the Diffie-Hellman protocol will be reviewed. Diffie-Hellman is the most dominant key agreement protocol for e-commerce transactions. Group Key agreement protocol is to generate a shared key between more than two parties. Then, DL08 and KON08 will be reviewed. These protocols are the most important group key agreement protocols. A payment protocol helps customers to do the payment via e-commerce. Then, current best payment protocols for e-commerce transactions will be reviewed. These protocols are SET, iKP, KSL, and 3DS [10, 14, 22].

2.1 Key Agreement Protocol

In this section, Diffie-Hellman key exchange protocol will be reviewed. This protocol is very successful in the market and it is the most common key agreement protocol among the existing key exchange protocols.

2.1.1 Diffie Hellman Key Agreement Protocol

Diffie and Hellman in their seminal work developed a key agreement scheme between two parties over an insecure channel. Before the actual key exchange begins, both parties agree on a prime number p and its primitive root g. You can see the steps to generate the shared session key in below.

1) Alice chooses her secret random number and computes her middle number. Then, she sends her middle number to Bob.

2) Bob picks his secret random number, computes his middle number and sends the result to Alice.

3) On receipt of the transmission, Alice calculates the shared key by her private number and Bobś middle number. Bob calculates his shared key by using his private number and Aliceś middle number [29].

The security of the Diffie-Hellman protocol is based on how difficult it is for an eavesdropper, Eve, to construct the key using the public information exchanged between Alice and Bob. Eve has to find Alice's secret number and/or Bob's secret number using the prime number that she can obtain by intercepting their communications. This is also known as the discrete logarithm problem. Discrete logarithm is a hard problem to solve.

Step	Action	Description
1	Alice and Bob agree on two numbers p and g	p is a large prime number and g is called base or generator
2	Alice picks a secret number a	Aliceś secret number = a
3	Bob picks a secret number b	Bobś secret number = b
4	Alice computes her public number $X = g^a \bmod p$	Aliceś public number = X
5	Bob computes his public number $Y = g^b \bmod p$	Bobś public number = Y
6	Alice and Bob exchange their public numbers	Alice knows p, g, a, X, Y Bob knows p, g, b, X, Y
7	Alice computes $k_a = Y^a \text{mod}\, p$	$k_a = (g^b \text{mod}\, p)^a \bmod p$ $k_a = (g^b)^a \bmod p$ $k_a = (g^{ba}) \bmod p$
8	Bob computes $k_b = X^b \text{mod}\, p$	$k_b = (g^a \bmod p)^b \bmod p$ $k_b = (g^a)^b \bmod p$ $k_b = (g^{ab}) \bmod p$
9	By the law of algebra, Aliceś k_a is the same as Bobś k_b, or k_a = k_b = k	Alice and Bob both know the secret value k

Table 2.1.1: Diffie-Hellman Key Agreement Protocol

It is considered infeasible to solve using current computing technology, for example, where the prime has more than 300 decimal digits, and a and b have more than 100 decimal digits [20, 21]. Heavy computation is needed to generate a shared session key in Diffie-Hellman key agreement protocol. It is not good for a mobile device with limited resources to do such a heavy computation for generating a shared session key [11, 24].

The fundamental math of Diffie-Hellman includes the algebra of exponents and modulus arithmetic. For this discussion, Alice and Bob will be used as a conventional example. The goal of this process is to agree on a shared secret session key for Alice and Bob. Alice and Bob will generate a shared key based on their selected private numbers. These are symmetric encryption algorithms that will be used to encrypt the data stream between them.

There are requirements on the numbers that the parties can pick. These requirements can be found in the references. This algorithm has been reviewed and discussed many times in a variety of papers. This algorithm has some problems that make it unsuitable for mobile devices [25, 26].

Note that, some of the numbers that each side may pick, may be very large for example if p is a 512-bit binary number, the minimum allowed in the standard, would be a number with up to 150 plus digits expressed in decimal notation. Implementation details are very important as typical mobile variables that cannot hold numbers as big as this, for example, a 512 bit (=64 byte) the number will not fit into a 4-byte integer field. In addition, these computations are too heavy for a mobile device with limited resources. You can see Diffie-Hellman steps in table 2.1.1.

2.2 Group Key Agreement Protocols

Group key agreement protocol generates a shared session key for a group of parties (more than two parties). In this section, DL08 and KON08 will be reviewed. These protocols are the best group key agreement protocols based on [12].

2.2.1 DL08

In order to evaluate and compare the recommended group key agreement protocol, DL08 should be reviewed in this section. DL08 is a group key protocol that was developed by Desmedt and Lange proposed a three-round group key agreement protocol in 2007. This protocol is proper for groups of parties with different computational capabilities. In this protocol, a balanced group of n parties should have approximately $n/2$ more powerful parties.

The number of computations for calculations of signatures and verifications are important factors in computing the complexity of this protocol. The assumption is that a Digital Signature algorithm is used by the signing scheme. I can assume that a signature generation has the cost of one exponentiation and a signature verification has the cost of two exponentiation.

According to this assumptions, the complexity of this protocol includes total number of $(9n/2) + 2n\lg_3\lceil n/3\rceil$ multiplications, $3n/2$ pairings and $3n/2$ exponentiation. The parties will have to transmit $7n/2$ messages and also receive $3n + n\lg_4\lceil n\rceil$ messages. DL08 is a three-round protocol based on the Burmester-Desmedt scheme that achieves the best performance from aspect of cost based on [12].

2.2.2 KON08

KON08 is a cluster-based GKA protocol proposed by Konstantinou in 2010. It is based on Joux's tripartite key agreement Protocol. It has two variants: contributory and non-contributory. In the lower level nodes belong to only one cluster. In upper levels nodes belong to two clusters. Authentication can be provided by the use of an authenticated version of Joux protocol. Authentication method does not influence the number of rounds or the communication cost of this protocol. In particular, the protocol has $\log_2 n/3$ rounds and $4n$ messages should have been transmitted. In the authenticated version, the group has to perform no more than $5n$ scalar multiplications and $11n/2$ pairing computations.

2.3 Payment Protocols

In this section, the most important payment protocols will be reviewed. These protocols are iKP, SET, KSL, and 3D Secure [27, 30].

2.3.1 Internet Keyed Payment Protocols (iKP)

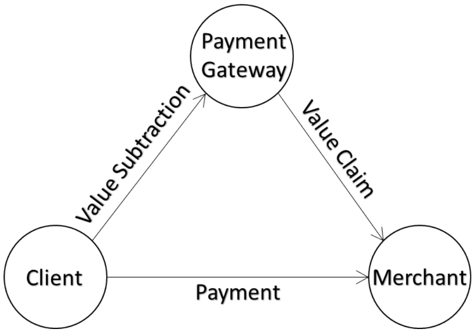


Figure 2: High level steps of SET and iKP

IBM developed iKP $(i = 1, 2, 3)$ family of protocols. These protocols are designed to implement credit card transactions between customer and merchant. They use existing financial network for payment clearing and authorization. These protocols utilize public key cryptography. These protocols can be implemented with hardware or software. Each member of the family differs from the others from the aspect of the level of complexity and security. If the number i (ex. 3kp) is larger, the security will be stronger [9, 17, 34, 36, 40].

These protocols have been used on the Internet since 1996. iKP protocols are unique because of longevity, security, and relative simplicity of the underlying mechanisms. These protocols are based on public key cryptography. The number of the participants who possess public key pairs is different in each one of these protocols. Names of these protocols are showing this number like 1KP, 2KP, and 3KP. Participants of these protocols are customer, merchant, and payment gateway (acquirer) [15, 16].

As you see in figure 2, SET and iKP protocols have three high-level steps [32]. One of these steps is value claim. In this step, payment gateway requests the monetary value from the merchant. In value subtraction, customer request from payment gateway to subtract the monetary value from his account. The last step is payment. In this step, the customer pays the monetary value to the merchant. iKP is a direct ancestor of SET. SET is similar to 3KP. In both SET and iKP all the participating parties are required to have their own certificates [18, 23].

2.3.2 Secure Electronic Transaction (SET)

SET defines an open encryption and security specification. This protocol is designed to protect credit card transactions over the internet. The initial version emerged in a call for security standards by MasterCard and Visa in February 1996 [4, 11, 29]. Many companies such as IBM, Microsoft, Netscape, RSA, Terisa, and

Verisign were involved in the development of its initial specification.

The first generation of SET-compliant products emerged in 1998. E-commerce success depends on secure payment systems. Authentication, encryption, integrity, and non-repudiation are four essential security requirements for safe electronic payment. SET is defined as an automated dynamic scheme that let credit card holders order items over the Internet in a secure way [7, 8, 13, 33, 39].

This protocol is composed of request and response message pairs. All participants in a typical transaction in SET should acquire their own public key certificates. A typical SET transaction includes initialization, purchase order, authorization, capture payment, and card inquiry. For example, if you want to order a book from an online store, you need to first put the book in your basket. Then, you need to initiate the payment. Then, the website will authorize you for the payment. If the authorization result is a success, the website will do the transaction in cooperation with the bank [19, 28, 37, 38].

Different roles in this protocol are cardholder, merchant, issuer, acquirer, payment gateway, and certification authority. In a typical scenario in SET, the first customer will open a credit card account in a bank that supports electronic payments and receives a digital certificate that is signed by his bank. Merchants have their own certificates.

Then, the customer places an online order. The merchant should be verified by the customer based on merchant's certificate. Then, the customer sends order information, payment information, and his certificate to the merchant. Then, merchant requests payment authorization. After successful authorization by the payment gateway, the merchant provides requested goods or services. Finally, merchant requests his payment from payment gateway.

SET has some problems. In this protocol, the cardholder is not protected from dishonest merchants that have a tendency to charge more than their advertised price or are hackers who put up an illegal website to collect credit card information. The suggested payment protocol doesn't have this issue. Besides, the merchant is not protected from dishonest customers who provide invalid credit card numbers or claim a refund without any real cause. In this protocol, the merchant is more vulnerable to fraud since legislation protects customers in most of the countries in the world. This issue will not being addressed in the suggested payment protocol. In the protocol both customer and merchant will be protected from fraud.

Transport Layer Security (TLS) and Secure Sockets Layer(SSL)

TLS and SSL are known as SSL. SSL is a cryptographic protocol to secure the connections in computer networks. Its main usage is web browsing, email, instant messaging, voice over IP, and etc. For example, web browsers use SSL to secure the communications between the client and server. SSL is widely accepted in technology environment [20].

SET guarantee the security of a payment transaction. On the other hand, SSL is not a payment protocol. SSL simply encrypt/decrypt all the communications between the cardholder and merchant website. SSL is not backed by any financial institution. SET is backed by Visa and MasterCard. So, SET can't guarantee the security of a payment transaction. SSL can't authenticate all parties, since SSL certificates are not backed by any financial institution. Because of these reasons, SSL will not be compared with the recommended payment protocol [6, 20, 27, 32].

3D SET or 3-Domain SET

3D SET is a hybrid SSL/SET architecture. In this architecture merchant's bank and payment intermediary are associated. Intermediary is a technical agent from the bank that secure the exchange messages and originate the exchanges from SET. In 3D SET, intermediary agent acts in the middle of three domains. These three domains are client, merchant, and SET proper [17, 19, 20].

In this protocol, the payment gateway plays three major roles. It may play the role of SSL server between the clinet and the server. The payment gateway may also play as the webhost that initiates the SET transaction for the merchant. Finally, it may play the role of the certification authority for the customers [20].

The main drawback of this method is that client can't control the certificate that is issued for him, since it will be saved by the intermediary. Also, if the customer and merchant runs their transactions with one intermediary most of the time, the intermediary may be able to construct their marketting profiles [20].

2.3.3 KSL Protocol

KSL is a payment protocol for e-commerce in fixed networks like the Internet. This protocol is not proper for the mobile platform because of its heavy computation and communication costs. The idea is to reduce the number of people who possess their own key pair in the SET protocol. In this protocol, all participants except customer should have their own certificates. So, client-side computation is lighter [22]. This protocol has two sub-protocols that are merchant registration and payment. The client should share two sets of keys before payment starts. He should share symmetric keys X_i with merchant and Y_i with issuer [6].

KSL protocol is a nonce-based protocol that is introduced by Kehne, Schonwalder, and Langendorfer. KSL is an alternative of Kerberos. The drawback of Kerberos is that it uses timestamps. So, it requires at least one loose synchronization between timestamp generator and participants. Participants of KSL protocol are customer, merchant, payment gateway, and financial service provider. All of them except customer should have their own certificates. So, the computational cost, in customer device, is lighter [25, 26, 31].

In a typical transaction in KSL, customer and merchant exchange necessary information at the beginning. Then, the customer sends a payment request. Merchant deciphers the message and sends a value claim request that is signed by the customer to the payment gateway. After authorization of merchant, payment gateway does the transaction by the help of issuer. Then, payment gateway notifies merchant and customer. Note that, the cost of communication in KSL is high. Finally, customer retrieves the result of his request [8, 9, 28].

2.3.4 3D Secure Protocol

3D Secure is an extra security layer for online credit and debit card transactions that is based on XML. Arcot Systems developed this protocol for the first time. The name 3D came from 3 domains. It means that this authentication is based on three domains: acquiring domain (merchant and his bank), issuer domain (customer credit card issuer), interoperability domain (the internet, merchant plug-in provider, and Access Control Center).

The 3DS transaction involves request/response pairs. Visa and MasterCard don't allow merchants to send these requests to their own servers because of security concerns. Banks isolate their servers by licensing software providers that are called merchant plug-in provider (MPI) for this purpose. Access Control Center (ACS) is on the issuer's side (bank). Most of the banks, contract third parties to provide their ACS. However, it is possible that bank provides ACS.

This protocol utilizes XML messages that are sending over SSL connections. In this protocols, a pop-up may interfere the payment process and try to authenticate the customer before the authorization process. The intention is to prevent card not present fraud. These implementations utilize text messaging for sending one-time passwords to clients if needed. There are some benefits in this extra security level. The card holder is confident that his card will not be used without his authorization, the merchant will be protected from fraud, and banks see that the customer is already authenticated and they will be more willing to approve the transaction (the authorization step). Note that, this approach needs heavier computations. This approach also needs more communications between the engaging parties, but it is safer.

Visa implemented 3DS as Verified by Visa. Verified by Visa is a protocol that protects customers against unauthorized use. This protocol has two steps to authenticate the purchase. The first step is to show the customer a personal message. Note that, only the customer and his bank know this message. The customer sets this message when he signs up for Verified by Visa. The second step is the customer will be asked to enter his Verified by Visa password [35, 37, 39].

MasterCard SecureCode is another implementation of the 3D Secure protocol. They have an easy 3 step enrollment process. They claim that this implementation protects the privacy of the transaction. It gives the customer and extra layer of online shopping security. In this approach, only customer and his financial institution know the code. So far, more than one million merchants in 122 countries support this implementation.

American Express implemented 3DS as SafeKey. They defined it as an authentication tool that reduces online fraud by confirming cardholder's identity with an extra password. They have three different sections for merchants and acquirers, issuers, and 3DS providers. They have a registration process that allows customers to choose their MPI service provider among the possible options.

JCB International has its own implementation of 3DS as J/Secure. J/Secure defined as a payer authentication service. They claim that they made online shopping more secure with this method. It was fully launched in April 2004. They have a registration step for their customers. In this approach, customers should register with their credit card issuers.

There are some issues in these implementations of 3DS. Firstly, there is a pop-up window or inline frame that is coming from a source that is not a familiar domain (ex. shopping site, bank, Visa, or MasterCard). It will be very hard for a customer to find out if this window is a phishing scam or it is coming from the bank. Customer will not be able to check the iframe certificate because new versions of browsers don't allow users to do this.

Also, a man in the middle attack and phishing scam is possible in this step. Mobile users may experience some issues to see the pop-up or iframe since the issuer may not be aware of the mobile users and the mobile browser may not support these technologies. Sometimes a 3D Secure confirmation code is required in this process. If the code is planned to be sent as a text message, the customer may be unable to receive it based on the country that he is in. Also, this may cause trouble for the people who change their cell phone number regularly (for example due to travel). This issue is solved in the proposed 3D protocol in this research.

I sent my Soul through the Invisible, Some letter of that After-life to spell: And by and by my Soul returned to me, And answered: 'I Myself am Heaven and Hell.

Omar Khayyam

3

Efficient Secret Key Agreement Protocols

In this chapter, proposed key agreements protocols and group key agreement protocols will be described. Key agreement protocol defines a shared key between two parties. Group key agreement protocol establishes a shared key among more than two parties.

3.1 Problem Description and Desired Solution

There are many exchange protocols available. However, they are not built for the mobile platform. The mobile platform has its own limitations like computational power limitation. Existing protocols are built for a computer with a stable network connection.

The desired exchange protocol should be lightweight. It should have less computational cost and communication cost in comparison with the existing exchange protocols. This protocol should prevent sending the private keys via a network. The desired exchange protocol in this chapter will be a protocol that is proper for mobile platform.

3.2 Improved 2-party Secret Key Agreement Protocol

3.2.1 Problem Description and Desired Solution

Key agreement protocol is to define a shared key between two parties. The most common key agreement protocol is Diffie-Hellman. However, Diffie-Hellman is using algebra of exponents that is not suitable for the mobile platform. The mobile platform has limited memory and computational power. Algebra of exponents produces larger numbers. These large numbers are not suitable for mobile platform.

The mobile platform needs to deal with smaller numbers. Calculations are lighter in smaller numbers in comparison with larger numbers. Also, saving the smaller numbers will be more convenient for the mobile platform. The desired key agreement protocol in this chapter is a protocol that is using algebra of logarithm. It produces smaller middle numbers. The mobile device deals with smaller numbers for calculations.

3.2.2 Algorithm Explanation

The improved key agreement protocol is based on the algebra of logarithms and modulus arithmetic. The intention is to make shared key generation process properly for mobile platform by reducing computational cost. Also, the size of temporary results in the computations has been decreased, since memory and size of variables are limited in mobile devices. The improved algorithm has the same function as the original Diffie-Hellman secret key agreement algorithm. For describing the key agreement protocol, Alice and Bob are used as a conventional example. The goal of this process is to generate a shared key between Alice and Bob.

3.2.3 Algorithm Definition

In this section, you can see the steps of the suggested protocol. These steps are calculating the shared key between two parties twice as fast as the original Diffie-Hellman protocol. Please refer to chapter 5 section 1 to see the results of the performance experiments. You can see the steps of the proposed key agreement protocol and its example in table 3.2.1 and table 3.2.2. In this example: $n = 2, p = 2,$ and $q = 7(a = p^n = 4 = 2^2)$.

	Action	Description
1	Alice and Bob agree on three numbers a, p, and q	q is a large prime number $a = p^n$ $p = \{2, 3, \ldots, n\}$ n, u, and $v = 1, 2, 3, \ldots, n$
2	Alice picks a secret number u	Aliceś secret number = u $u \bmod q$ is not zero
3	Bob picks a secret number v	Bob's secret number = v $v \bmod q$ is not zero
4	Alice computes her public number $A = ((u \bmod q) \times \log_p a) \bmod q$	Alice's public number = A $= (\log_p a^{(u \bmod q)}) \bmod q$
5	Bob computes his public number $B = ((v \bmod q) \times \log_p a) \mathrm{mod}\, q$	Bob's public number = B $= (\log_p a^{(v \bmod q)}) \bmod q$
6	Alice and Bob exchange their public numbers	Alice knows u, a, p, q, A, and B Bob knows v, a, p, q, A, and B
7	Alice computes $k_a = ((u \bmod q) \times B) \bmod q$	$k_a = ((u \bmod q) \times (\log_p a^{(v \bmod q)})) \bmod q$ $k_a = (\log_p a^{(v \bmod q) \times (u \bmod q)}) \bmod q$ $k_a = (\log_p a^{((u \times v) \bmod q)}) \bmod q$
8	Bob computes $k_b = ((v \bmod q) \times A) \bmod q$	$k_b = ((v \bmod q) \times (\log_p a^{(u \bmod q)})) \bmod q$ $k_b = (\log_p a^{(u \bmod q) \times (v \bmod q)}) \bmod q$ $k_b = (\log_p a^{((v \times u) \bmod q)}) \bmod q$
9	By the law of algebra, Aliceś k_a is the same as Bobś k_b, or $k_a = k_b = k$	Alice and Bob both know the secret value k

Table 3.2.1: The Key Agreement Protocol

Step	Action
1	Alice and Bob agree on $n = 2, p = 2, a = 4,$ and $q = 7$
2	Alice picks a secret number $u = 3$
3	Bob picks a secret number $v = 4$
4	Alice computes her public number $A = ((3 \bmod 7) \times \log_2 2^2) \bmod 7 = 3$
5	Bob computes his public number Bob's public number = B $A = ((4 \bmod 7) \times \log_2 2^2) \bmod 7 = 1$
6	Alice and Bob exchange their public numbers Alice knows u, a, p, q, A, and B Bob knows v, a, p, q, A, and B
7	Alice computes $k_a = ((u \bmod q) \times (\log_p a^{(v \bmod q)})) \bmod q$ $k_a = ((3 \bmod 7) \times 1) \bmod 7 = 3$
8	Bob computes $k_b = ((v \bmod q) \times (\log_p a^{(u \bmod q)})) \bmod q$ $k_b = ((4 \bmod 7) \times 6) \bmod 7 = 3$
9	By the law of algebra, Aliceś Alice and Bob both know the k_a is the same as Bobś k_b secret value k or $k_a = k_b = k = 3$

Table 3.2.2: The Key Agreement Protocol Example

Algorithm 1- Original Diffie-Hellman Secret Key Agreement Algorithm

```
Algorithm 1- DiffieHellman {
    generator = 3;
    primeNumber = 982451653;
    calculateMiddleKey(selectedNumber) {
        payerMiddleNumber =
        power(generator, selectedNumber)
        % primeNumber;
        return payerMiddleNumber;
    }
    calculateSharedKey(selectedNumber,
    otherPersonMiddleNumber) {
        sharedKey = power(otherPersonMiddleNumber,
        selectedNumber) % primeNumber;
        return sharedKey;
    }
}
```

In this section, pseudo-code of the original Diffie-Hellman protocol will be reviewed. As you see in the implementation, the generator is 3 and the prime number is 982451653. There are two procedures here. One of them calculates the middle key and the other one calculates the shared key. To calculate the middle key only the selected number of the user is required. However, to calculate the shared key, the selected number of the user and the middle number of the other user should be available.

Algorithm 2- Improved Diffie-Hellman Secret Key Agreement Algorithm

```
Algorithm 2- ImprovedDiffieHellman {
    generator = 3;
    primeNumber = 13;
    calculateMiddleKey(selectedNumber) {
        double c = logarithm(generator);
        payerMiddleNumber =
        (((selectedNumber % primeNumber) *
        (log(generator) / c)) % primeNumber);
        return payerMiddleNumber;
    }
    calculateSharedKey(selectedNumber,
    otherPersonMiddleNumber) {
        sharedKey =
        (((selectedNumber%primeNumber)*
        (otherPersonMiddleNumber))%primeNumber);
        return sharedKey;
    }
}
```

In this section, the improved key agreement protocol will be reviewed. As you see in the implementation, the generator is 3 and the prime number is 13. There are two procedures here. One of them calculates the middle key and the other one calculates the shared key. To calculate the middle key only the selected number of the user is required. However, to calculate the shared key, the selected number of the user and the middle number of the other user should be available.

Theorem 1

The Improved Diffie-Hellman Secret Key Agreement algorithm (algorithm 2) has the same function as the original Diffie-Hellman secret key agreement algorithm (algorithm 1), and the former is almost twice as fast as the latter.

Proof

As you see, algorithm 2 is calculating the middle key and the shared key by algebra of logarithm instead of algebra of exponents in algorithm 1. You can see the proof of correctness of the improved key agreement protocol below. Suppose, there exist two parties A and B. Two keys will be compared. If they are equal, the key agreement protocol is correct. In this proof, rules of the algebra of logarithms and algebra of modulus arithmetic will be used. Assumptions:

a_i is $a \bmod p_i$ and b_i is $b \bmod p_i$. Integer a is represented by r-tuple $(a_1, \ldots, a_r)$. In this kind of representation, residues should be calculated by multiple divisions. So, $a_i = a \bmod p_i (1 \leqslant i \leqslant r)$. Then, based on theorem 2.1 modular[modular**?**]:

$$(a_1, \ldots, a_r) \times (b_1, \ldots, b_r) = (a_1 b_1 \bmod p_1, \ldots, a_r b_r \bmod p_r)$$

In this case, the prime number is q. In addition, based on algebra of logarithm [2]:

$$\log_b(m^n) = n \times \log_b(m)$$

There are two parties A and B. Letś follow the protocol to generate a key for each one of them. Then, by the rules that just mentioned, the key that is generated for A is equal to the key that is generated for B. Besides, results of the calculations in both sides (A and B) are equal.

A:

$k_a = ((u \bmod q) \times (\log_p a^{(v \bmod q)})) \bmod q$

$k_a = (\log_p a^{(v \bmod q) \times (u \bmod q)}) \bmod q$

$k_a = (\log_p a^{((u \times v) \bmod q)}) \bmod q$

B:

$k_b = ((v \bmod q) \times (\log_p a^{(u \bmod q)})) \bmod q$

$k_b = (\log_p a^{(u \bmod q) \times (v \bmod q)}) \bmod q$

$k_b = (\log_p a^{((v \times u) \bmod q)}) \bmod q$

As you can see the results in both sides are equal. So, Alice and Bob now have a shared key. They can use this key for symmetric encryption. The security strength of Diffie-Hellman is kept and reduced its computational cost. Improved key agreement protocol can be used instead of Diffie-Hellman in the mobile payment protocol in order to make it proper for the mobile platform.

End of Proof.

3.3 Proposed Group Secret Key Agreement Protocol

In this section, proposed group key agreement protocols will be reviewed. These two proposed protocols will be reviewed. These protocols are circular and linear approaches.

3.3.1 Problem Description and Desired Solution

Sometimes it is necessary to define a secure communication channel between n users. This process may be very costly. In the simplest way, the cost will be exponential. If a shared key has been established between each pair of users in the group. This amount of calculations and communications are not suitable for mobile platform.

The desired group key agreement protocol is a protocol that has less computational cost and communication cost. This goal can be achieved in different ways. In this research, two methods will be introduced. The first one is a circular approach that is $O(n^2)$. The second one is a linear approach.

3.3.2 Circular Group Secret Key Agreement Protocol

Problem Description and Desired Solution

As mentioned earlier, the cost of defining a secure channel between n users in a group may be exponential. A naive method may have expensive computational cost and communication cost that is not suitable for the mobile platform. in this section, the desired protocol is a protocol that doesn't have exponential cost. The desired protocol would be a protocol that decreases the cost of defining a shared key for a group of n users.

Algorithm Explanation

In this section, the proposed circular group key agreement protocol will be introduced. Mobile phones have come a long way from just a medium of communication. They are now a helpful assistant that can do a multiplicity of tasks for us these days. Mobile and portable devices are suitable and secure for online payment transactions. Although most users prefer to do an abundance of their on-line activities with their mobile apps via their cell phones, there are several serious issues that hinder the universal acceptance of mobile payment among users worldwide.

This protocol is suitable for portable and mobile devices since it has fewer computations and smaller numbers in comparison with Diffie-Hellman key agreement protocol. As a result of this, this protocol is proper for using in mobile devices with limited hardware and computational resources. This algorithm is based on the algebra of logarithms and modulus. The goal of this protocol is generating a shared secret session key between n users. This method is proper for limited resources of the mobile platform.

This shared key will be used by each one of the parties to generate keys for each side independently. These are symmetric encryption algorithms that will be used to encrypt the data stream between the engaging users. In this algorithm, each K_i has a key a_i. Besides, neither the shared secret key nor the encryption key, do not travel over the network via this process. The design of this protocol is focused on generating a shared session key with considering the limitations in computational power and hardware of portable and mobile devices.

Algorithm Definition

The shared key in this protocol will be generated in a circular approach. First, the person chooses his private number. Then, he calculated his middle number and sends it to the next person in the circle. It will continue till the accumulative middle key reaches the person. The person can calculate his own shared key by using his own private key and the accumulative middle key. Because of this, the proposed protocol was named Golden Circle. The goal of this protocol is to improve the running time of the similar algorithms that were discussed in this section. After the calculations, each party will have the shared session key or GK (Golden Key). You can see the steps of this protocol in table 3.3.1.

Action	Description
All of n parties agree on three numbers $g,\ p$ and a new operator $\sim$: $a \sim b =$ If$(a+b) > n$ then $a \sim b = a + b - n$ Else $a \sim b = a + b$	p is a large prime number g is called the base or generator for each $i\ a_i,\ g,\ p = 1,\ 2,\ 3,\ \cdots,\ n$
Each of them for example K_i picks a secret number a_i	
For each party K_i from $K_{(i \sim 1)}$ to $K_{i \sim (n-1)}$ does in a sequence First computes $A_{(i \sim 1)} = (g^{a^{(i \sim 1)}}$ mod p And sends $A(i \sim 1) to K(i \sim 2)$ Second $A_{(i \sim 2)} = (A_{(i \sim 1)} \times a_{(i \sim 2)})$ mod p ... last $A_{(i \sim (n-1))} = (A_{(i \sim (n-2))} \times a_{(i \sim (n-1))})$ mod p At the end $K_{(i \sim (n-1))}$ sends $A_{(i \sim (n-1))}$ to K_i	
Each of them for example K_i does $K = (A_{(i \sim (n-1))}$ mod $p)^{a_i}$ mod p This is the shared session key	$K = (g^{a^{(i \sim 1)} \cdots a^{(i \sim (n-1))}}$ mod $p)^{a_i}$ mod p $= (g^{a^{(i \sim 1)} \cdots a^{(i \sim (n-1))}\ a^{(i)}})$ mod p $= (g^{a_i \cdots a^{(i \sim (n-1))}})$ mod p
Fortunately computed K of each of them is equal to otherś	Each of them knows the secret value k

Table 3.3.1: Steps and proof of correctness of GC

Theorem 2

The recommended group key agreement protocol has the same function as the naive Diffie-Hellman group key agreement protocol, and former improves the latter's runtime complexity from $O(2^n)$ to $O(n^2)$.

Proof

In this section, you will see that the generated shared group key will be identical for all of the group members no matter where you start generating the key. In this mathematical proof, assume that there are n group members (k_i) and each one of them selects a secret number for himself (a_i). At the end of the process, each member will have the shared group key. p is a large prime number g is called the base or generator. Each party knows $a_i, g, p = 1, 2, 3, \cdots, n$. All of the n parties agree on three numbers g, p, p. First, a new operator should be defined $(\sim)$.

$$a \sim b = \text{If}(a+b) > n \,\text{then}\, \{a \sim b = a + b - n\}\ Else\ \{a \sim b = a + b\}$$

Each of the members for example k_i picks a secret number a_i. For each party K_i

from $K_{i\sim 1}$ to $K(i \sim (n-1))$ does in a sequence:

First computes $A_{i\sim 1} = (g^{a_{i\sim 1}}) \bmod p$ And sends $A_{i\sim 1}$ to $K_{i\sim 2}$

Second $A_{i\sim 2} = (A^{a_{i\sim 1} a_{i\sim 2}}) \bmod p$

...

$A_{i\sim(n-1)} = (A_{i\sim(n-2)})^{a_{i\sim(n-1)}}) \bmod p$

At the end $K_{i\sim(n-1)}$ sends $A_{i\sim(n-1)}$ to k_i.

Each of them for example k_i calculates the shared key like below:

$K = (A_{i\sim(n-1)})^{a_{i\sim 1} \cdots a_{i\sim(n-1)}} \bmod p)^{a(i)} \bmod p$

Note that this key will be equal for all the members.

$K = (g^{a_{i\sim 1} \cdots a_{i\sim(n-1)}} \bmod p)^{a_i} \bmod p$

$= (g^{a(i\sim 1) \cdots a(i\sim(n-1))\, a(i)}) \bmod p$

$= (g^{a_i \cdots a_{i\sim(n-1)}}) \bmod p$

End of Proof.

Complexity

First, let's consider the complexity of the proposed key agreement protocol. For generating a GC, n calculations are required. Each party needs calculations for computing its GC. Furthermore, each party needs a GC that is calculated by the other parties. The computational cost for generating a shared key is constant because of its logic (using algebra of logarithms instead of algebra of exponents). As it can be seen below, the complexity of the algorithm for generating a shared session key for n parties is $O(n^2)$. So, no matter for how many users you want to define a shared session key, you can do it in a polynomial time now. In the equations below, n is the number of iterations and GK is the cost of generating a shared session key with Golden Circle.

$(n \times GK) \times (n - 1 \times \text{calculations})$
$GK : n$
Each $GK \times (n - 1)$
$\Rightarrow GK\,\text{Total} = n \times (n - 1)$
$= n \times (n - 1)$
It is polynomial.
$\Rightarrow$ Complexity : $O(n^2)$

First, imagine the shared session key between n users will be generated by Diffie-Hellman, First, a shared session key should be generated between each of the two possible pairs among the users. Then a shared session key between each two shared session keys should be defined. This should continue until a shared session key between all n users has been acquired. The time complexity of generating the all shared keys is $O(2^n)$. This is super-polynomial for n parties. This chapter suggests a solution define a shared session key between n users with a polynomial time complexity. The traditional ways can be used, but their complexity is not proper for mobile devices. The complexity of the key agreement protocol is a key point in the second protocol.

Furthermore, in the example n is a big number. For example, a company wants to generate a shared session key between thousands of the employees. The complexity of the algorithm is very important here, because of the limited resources of the mobile platform (computational power, network capacity, and etc). These resources are available in different cellphones. People has a wide variety of mobile phones with distinct capabilities and hardware. To see the details of this protocol refer to [33].

3.3.3 Golden Linear Group Secret Key Agreement Protocol

Problem Description and Desired Solution

The recommended circular approach decreases the cost of defining a shared key for a group of n users. However, if the number of users grows, still the cost may be a lot for a mobile device. The desired protocol in this section will be a protocol that has less computational cost than the circular approach. This protocol should have linear computational cost. Two of the main attributes of this protocol are correctness and efficiency. This goal can be achieved by selecting a user as the coordinator and following the steps of the algorithm. The desired protocol in this section will be suitable for mobile platform with its limitations.

Protocol Explanation

Circular Group Secret Key Agreement Protocol This protocol is linear and the recommended circular approach is quadratic in time complexity. This protocol needs one of the group members as the leader. However, in the circular approach, there is not a leader. All group members are the same in the circular approach. This protocol has been divided into two sub-protocols. The first one is secure channel sub-protocol. The secure channel defines a secure channel between each one of the group users and the leader. The second one is initiation sub-protocol. The goal is to define a shared session key for a group of users by these two sub-protocols. In this step, leader defines a shared key and distribute it among group members. In the secure channel sub-protocol, the proposed key agreement protocol named GKA (Golden Key Agreement Protocol) will be used. Based on GKA protocol, all members and leader of the group should share three numbers a, p, q at the beginning. This protocol is very efficient if there are not a lot of joins and leave in the group.

Protocol Design- Secure Channel Sub-protocol

Leader is a group member who coordinates the shared key generation process. In this part the leader will establish a secure connection to each one of users in the group. Firstly, each user will choose a private key (user i will choose private key p_i). The leader also will choose a private key (p_c). Then, each user will send $A_i = ((p_i \bmod q) \times \log_p a) \bmod q$ to the leader. Also the leader will broadcast $A_c = ((p_c \bmod q) \times \log_p a) \bmod q$ to all the group members.

Then, each one of the users will calculate $k_i = ((p_i \bmod q) \times A_c) \bmod q$. Then, the leader will calculate $k_{c_i} = ((p_c \bmod q) \times A_i) \bmod q$. As a result of this sub-protocol, the leader will have a secure line with each one of the members. It means the leader is ready to share the shared key will all group members securely. The shared key for the group will be generated in the next sub-protocol.

Protocol Design- Initiation Sub-protocol

In this part, the leader will generate a shared session for the users. Firstly, each user chooses his own private key p_i and sends it to the leader through the secure channel. The leader then generates a shared session key for the group based on the equation below and will send it to the members through their secure channels.

$$
\begin{aligned}
&k = (\log_p a^{(p_1 p_2 \ldots p_n)}) \bmod q \\
&a = p^n \Rightarrow k = (\log_p p^{[n(p_1 p_2 \ldots p_n)]}) \bmod q \\
&\Rightarrow k = ((p_1 p_2 \ldots p_n) \times \log_p p^n) \bmod q
\end{aligned}
$$

This shared session key is only for current users of the group. if anyone leaves or joins the group, the shared group key should be updated. The protocol needs to add/remove the contribution of the member who joins/leaves the group. In the next sections, The join and leave sub-protocols will be reviewed. Join protocol is useful when a new member joins the group and leave protocol is useful when a member leaves the group.

Protocol Design- Join Protocol

When a new user ($user_{n+1}$) want to join a group, he should establish a secure channel with the leader of the group. He uses the secure channel protocol (GKA) for this step. Then, he will send his private key with a positive sign to the leader through his secure channel. Then, the leader will update his shared session key by multiplying it to the received number $((p_{n+1} \bmod q))$, since the sign is positive. After that, the leader will broadcast the updated shared session key to the group members. The key will be updated for the leader, members of the group, and the new member Based on the equation below.

$$
\begin{aligned}
&\text{Current } k = \ (\log_p a^{(p_1 p_2 \ldots p_n)}) \bmod q \\
&\text{New } k = (p_{n+1} \bmod q) \times (\log_p a^{(p_1 p_2 \ldots p_n)}) \bmod q \\
&= (p_{n+1} \times \log_p a^{(p_1 p_2 \ldots p_n)}) \bmod q \\
&= (\log_p a^{(p_1 p_2 \ldots p_n p_{n+1})}) \bmod q
\end{aligned}
$$

Protocol Design- Leave Protocol

When an existing user (for example $user_n$) want to leave the group, he should inform the leader before he leaves. He should send his private key with a negative sign $(-1 \times p_n)$ to the leader through his secure channel. Then, the leader will update his private key by dividing it by the received number. Note that, the leader ignores the negative sign. This negative sign is the only symbol of leaving. After that, the leader will broadcast the new shared group key. After that, the leader and the group members will have the same updated key based on the equation below.

Current $k = (\log_p a^{(p_1 p_2 \cdots p_n)}) \bmod q$
New $k = (\log_p a^{(p_1 p_2 \cdots p_n)}) \bmod q \,/\, (p_n \bmod q)$
$= (\log_p a^{(p_1 p_2 \cdots p_n)/p_n}) \bmod q$
$= (\log_p a^{(p_1 p_2 \cdots p_{n-1})}) \bmod q$

Theorem 3

The recommended group key agreement protocol has the same function as the naive Diffie-Hellman, and the former improves the latter's runtime complexity from $O(2^n)$ to $O(n)$.

Proof

In this section, you see that from whichever party that you start generating the shared key, you will have the same shared group key. There are n members in the group. Numbers p, q, and a are shares among group members and the leader. Member i chooses his private key p_i. The leader chooses his private key p_c. Member i sends $A_i = ((p_i \bmod q) \times \log_p a) \bmod q$ to the leader. Leader sends $A_c = ((p_c \bmod q) \times \log_p a) \bmod q$ to member i. Then, the shared key will be calculated by the leader like this:

$$k = (\log_p a^{(p_1 p_2 \ldots p_n)}) \bmod q$$
$$a = p^n \Rightarrow k = (\log_p p^{[n(p_1 p_2 \ldots p_n)]}) \bmod q$$
$$\Rightarrow k = ((p_1 p_2 \ldots p_n) \times \log_p p^n) \bmod q$$

End of Proof.

Complexity

In this section, the cost of the protocol will be analyzed. Let's start with secure channel sub-protocol. In the secure channel sub-protocol, the leader should establish a secure connection with each one of the members based on the private key of the member. The leader will have two calculations and one broadcast. Each member will have two calculations and will send one message and receive one message. In the initiation sub-protocol, the leader will have one calculation and one broadcast. He will also receive n messages from members of the group. Each member will send one message to the leader and receive one message from the leader in this sub-protocol.

In the join protocol, the leader will have three calculations (two for secure channel sub-protocol and one for updating the shared group key) and will send one message to the new user and one broadcast to all members of the group. Each user will only receive one message. In the leave protocol, the leader will have one calculation and one broadcast to members of the group. The leaving member should send one message to the leader and each member except the leaving member should receive one message. In conclusion, the protocol will send $3n$ messages and receive $3n$ messages. The protocol should do $2n + 2$ calculations for a group of n parties. To see the details of this protocol refer to [40].

A wealth you cannot imagine flows through you. Do not consider what strangers say. Be secluded in your secret heart-house, that bowl of silence.

Jalaluddin Rumi

4

The Mobile Pay Center Protocol

The proposed mobile payment will be introduced in this chapter. The recommended payment protocol is proper for the mobile platform because of its lightweight computational and communication costs. This payment protocol also has some specific features, for example, a feature that provides privacy protection for the customer.

4.1 Desired Properties

The desired payment protocol should be lightweight to be suitable for the mobile platform. This protocol should protect the customer's privacy. This should be an optional feature, since some customers may feel comfortable to reveal their identities for the merchants. This protocol should utilize temporary and timestamp generated identities for the customer to provide identity protection. This protocol should utilize mobile network operators to protect mobile device users' identities. This protocol should be protected against replay attacks by using suitable encryption and timestamp generated numbers in the transaction. People should be able to pay monetary values to other people only by knowing the other person's cell phone number.

4.2 Problem Description

In this research, it is assumed that the customer is using a mobile platform (a mobile device via a mobile network like 4G). There are many payment protocols available. You are using these protocols each time you are involved in an e-commerce transaction. Some of these protocols need a certification authority to verify the certificates of the engaging parties. Some of these protocols are using PKI infrastructure (public/private key). Many people like to use the mobile platform to do e-commerce transactions like ordering a book from Amazon. There are different platforms available for mobile devices these days like Android and iOS that should be utilized for this purpose.

However, these protocols are built for the computers with a stable network connection. None of these protocols are built for the mobile platform. The mobile platform has its own limitations like computational power and network bandwidth. Also, the network stability of the mobile platform is not comparable with a wired network connection. So, these protocols are not suitable for the mobile platform. These protocols don't protect the privacy of the payer. Defining a secure channel in these protocols are expensive.

The desired payment protocol will have a lightweight key agreement protocol to define a secure channel between the parties. This payment protocol should protect the privacy of the payer. The protocol should be protected against replay attacks. The desired protocol could provide non-repudiation. In summary, this protocol should be suitable for limited resources of the mobile platform. the assumption is that the proposed protocol is based on customer centric model. It is using symmetric encryption and decryption. It is not using PKI infrastructure. Certificate verification is not required for the customer on his mobile device.

4.3 Assumptions

The first assumption is that the customer is using the mobile platform. He is using a mobile device in a mobile network. This transaction is also designed for mobile devices on the wireless network. The customer should have a bank account that allows him to do online transaction. This protocol is mainly designed for credit card owners who are interested in making payment through the mobile platform. It is assumed that the customer is willing to protect his identity during the transaction.

4.4 Purpose

The first purpose of this protocol is to make a lightweight payment protocol for the mobile platform. The next purpose is to protect the privacy of the customer of payer. Also, this protocol is designed to be resistant to replay attacks. This protocol involves mobile network operators to protect the customer's privacy. In summary, this protocol is designed to be a secure lightweight payment mechanism for the mobile platform.

4.5 Design Trade-offs

In the proposed protocol, symmetric encryption will be used instead of PKI infrastructure to avoid heavy computations on the customer side.There will be no certificate validation for the customer to avoid extra computation and communication cost. The digital signature could be used to provide non-repudiation. Privacy of the customer will be protected by using temporary identities. Note that this feature is an optional feature that can be removed when performance and speed are more important than security. This feature may have expensive computation for a mobile device since it is base on PKI infrastructure.

The proposed protocol is based on the customer-centric model instead of merchant-centric to avoid full connectivity issue (all parties are connected). In the secure channel protocol, algebra of logarithms is being used instead of the algebra of exponents, since the protocol face smaller numbers in the algebra of logarithms. It means logarithms make bigger numbers small enough for a mobile device to avoid heavy computations. Also, if the numbers are smaller, it will be easy for the mobile device processor to do calculations on it or save it in its cache for fast access.

4.6 The Proposed Protocol

Symbols	Descriptions
MNO	Mobile Network Operator
{payer, payee, payerś MNO, payeeś MNO}	A set of engaging parties, which includes Payee, Payer, Payeeś MNO and Payerś MNO
Pay-Center	Time Stamp and Digital Sign center
PN	Phone Number of Party P
PIN	Party P selected this password identification number
ID	Identity of Party P, which identifies Party P to MNO
AI	Account Information of Party P, which including credit limit for each transaction and type of account
R_1	Random and times-tamp generated number by Payer act as Payer's pseudo-ID
R_2	Random and time-stamp generated number to protect against replay attack
K_1	Shared key between payer and his mobile network operator
K_2	Shared key between payer and payee
$AMOUNT$	Payment transaction amount and currency
$DESC$	Payment Description, which may include delivery address, purchase order details and so on
TID	The Identity of transaction
TID_{Req}	Request for TID
$PayeeID_{Req}$	Request for payee identity
Req	Request
MX	The message M encrypted with key X
$H(M)$	The one-way hash function of the message M
i	Used to identify the current session key of X_i and Y_i
$K_{p\text{-}p}$	The secret key shared between Payerś MNO and Payeeś MNO.
Success/Fail	The status of registration, whether success or failed
Yes/No	The status of transaction, whether approved or rejected
Received	Payment receivable update status, which may include the received payment amount
Pr_P	Private key of party P
Pu_P	Public key of party P
CK	client key: a key that is necessary for decoding X_i and Y_i sets on client side
CK_{Req}	Request for client key
T	Current date and time

Table 4.6.1: Notations

Phase 1: Payment Initialization: Payer ⇒ Payee: R_1, TID_{Req}, $PayeeID_{Req}$ Payee ⇒ Payer: $\{ID_{Payee}, TID, ID_{MNO}\}k_2$
Phase 2: Payment Subtraction Request Payer: Payer ⇒ Payer's MNO: $\{ ID_{Payee}, ID_MNO, R_1, TID, AMOUNT, DATE, R_2,$ $H(ID_{Payee}, ID_{MNO}, R_1, TID, AMOUNT, DATE, R_2), \{R_2, DESC\}K_2\}X_i, i, ID_{Payer}$ Payer's MNO ⇒ pay-center: $H[\{ID_{Payee}, ID_{MNO}, R_1, TID, AMOUNT, DATE, R_2, H(ID_{Payee}, ID_{MNO}, R_1, TID, AMOUNT, DATE, R_2),$ $\{R_2, DESC\}K_2\}X_i, i, ID_{Payer}]$ pay-center ⇒ Payer's MNO: generates *TimeStamp1* and verifies Payer's digital signature
Phase 3: Payment Authorization Request: Payer's MNO ⇒ Payee's MNO: R_1, ID_{Payee}, TID, $AMOUNT, DATE, \{R_1, DESC\}K_2$
Phase 4: Payment Confirmation Request: Payee's MNO ⇒ Payee: $\{R_1, TID, AMOUNT, DATE, \{R_1, DESC\}K_2, R_2,$ $H(R_1, TID, AMOUNT, DATE, \{R_1, DESC\}K_2, R_2), H(K_{pp})\}y_i, i$
Phase 5: Payment Confirmation Response: Payee ⇒ Payee's MNO: $\{Yes/No, R_2, H(K_{pp}, H(R_1, TID, AMOUNT, DATE,$ $\{R_1, DESC\}K_2, R_2), \{Yes/No, TID, AMOUNT, DATE\}K_2\}Y_{i+1}$
Phase 6: Payment Authorization Response: Payee's MNO ⇒ pay-center: $H(\{Yes/No, R_2, H(K_{pp}),$ $H(R_1, TID, AMOUNT, DATE, \{R_1, DESC\}K_2, R_2), \{Yes/No, TID, AMOUNT, DATE\}K_2\}Y_{i+1})$ pay-center ⇒ Payeeś MNO: generates *TimeStamp2* and verifies Payeeś digital signature Payeeś MNO ⇒ Payerś MNO: $Yes/No, TID, AMOUNT, DATE, \{Yes/No, TID, AMOUNT, DATE\}K_2$
Phase 7: Payment Subtraction Response: Payer's MNO ⇒ Payer: $\{Yes/No, R_2, H(K_{p\text{-}p}), H(ID_{Payee}, ID_MNO, R_1,$ $TID, AMOUNT, DATE, R_2), \{Yes/No, TID, AMOUNT, DATE\}K_2\}X_{i+1}$ Payee's MNO ⇒ Payee: $\{Received, R_2, H(K_{p\text{-}p}), H(R_1, TID, AMOUNT,$ $DATE, \{R_1, DESC\}K_2, R_2)\}Y_{i+1}$

Table 4.6.2: The mobile payment protocol

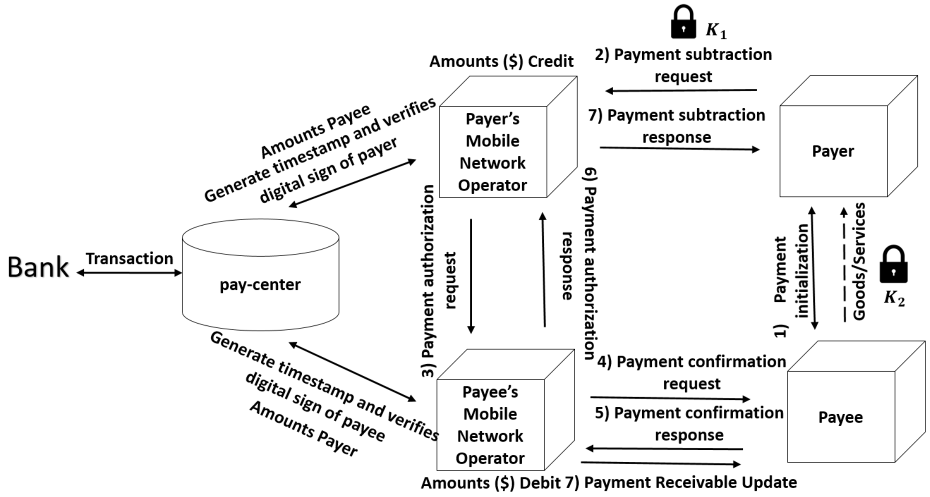

K_1
Amounts ($) Credit
2) Payment subtraction request
7) Payment subtraction response
Payer's Mobile Network Operator
Payer
Amounts Payee
Generate timestamp and verifies
digital sign of payer
Bank
Transaction
pay-center
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies
digital sign of payee
Amounts Payer
4) Payment confirmation request
5) Payment confirmation response
Payee's Mobile Network Operator
Payee
Amounts ($) Debit
7) Payment Receivable Update

Figure 3: Proposed mobile payment protocol

Mobile commerce (m-commerce) is e-commerce activities conducted via the mobile platform. Principals of m-commerce are the same as e-commerce plus mobile network operator. M-commerce inherits the challenges of e-commerce. Moreover, most of the e-commerce protocols are based on public key cryptography that is not efficient in mobile and wireless networks. Some of these protocols are keeping credit card's information on mobile devices or using this information in transactions without proper protection. This is why they are vulnerable to attacks.

Most of these protocols were designed to keep track of the traditional flow of data. This flow should be carried between client and merchant as a transaction. These protocols are vulnerable to attacks like transaction or balance modification attack. There is no proper notification in these protocols after a successful transaction as part of the protocol. These protocols don't have proper privacy protection for payer in their design. In 3DS, a pop-up window may interfere in the payment process that may be considered as man in the middle or phishing scam, since its domain is not Visa, MasterCard, the bank' domain, or the merchant's domain. Also, if the generation of a one-time password is required, these protocols utilize texting as their interaction with the customer. This may cause problem when the customer is in not in his home country.

In a typical payment scenario, the payer should open a credit card account in a bank that supports electronic payment. The payer should receive a digital certificate signed by the bank. Then, the customer selects his items or services by browsing merchant's website and order them. The customer should send a message that includes two parts. The first part is the merchant that includes order information. The second part is for merchant's bank that includes payment information. Merchant's bank should check this payment information with issuer of the credit card for authorization.

After a successful authorization, merchant's bank informs the merchant that the payment information is acceptable. Then, merchant completes the order, sends confirmation and invoice to the customer, and captures the transaction from his bank. Card not present fraud is a type of attacks that may happen when the merchant is not able to check the card physically. Note that in 3DS, the customer may experience a pop-up window in the middle of a transaction that tries to prevent this type of attack by using behavior-based authentication. There are different types of limitations in m-commerce and mobile platform. Also, identity protection is a significant challenge in m-commerce. Because of these restrictions, the focus should be on decreasing the computational cost and communication cost in these transactions.

The recommended protocol is proper for the mobile platform. Firstly, this protocol should decide how to move forward the payment data like payment amount from payer to payee via the network in the protocol. When payer and payee have accounts in the same bank, the situation is simple. Payer should inform his bank to debit the requested amount from his account and credit it into payee's account without any financial clearance.

However, when payer and payee have two accounts from two different banks or they have two different mobile network operators, there will be two possible models available. These models are the customer-centric model and merchant-centric model. Merchant-centric model is the traditional approach. In this method, merchant initiates the transaction. One problem with this approach is full connectivity (all participants can exchange messages with each other without intermediaries). This problem makes the payment protocol more vulnerable to different types of attacks. The second method is the client-centric model. In this approach, payer initiates the payment process and requests to pay and amount to the merchant. The mobile payment protocol is based on the client-centric model. Note that, there is no full-connectivity problem in this approach because there is no direct communication between payee and payer's bank.

The proposed mobile payment protocol is based on the client-centric model. Its principals are the payer, payee, mobile network operator, MPI, ACS, payer's credit card issuer (bank), payee's acquirer (bank), and certification authority for supporting secure transaction execution. The protocol works with two sets of keys. The first set should be shared between payer and his mobile network operator. The second set should be shared between the payee and his mobile network operator. The protocol consists of two sub-protocols that are registration and payment protocols. Payer and payee must register with their own mobile network operators at the beginning. Payer and his mobile network operator should generate a session key by running the improved key agreement protocol.

As you see, the protocol depends on the mobile network operators. People may change their mobile network operator. It is not affecting the customer since mobile network operator contribution in the transaction is only for that transaction. So, if the customer changes his mobile network operator after the transaction. This new mobile network operator will be involved for this customer in the next transaction. The customer needs to get the wallet application from his mobile network operator. By using this the customer becomes dependent on his mobile network operator. However, this is one of the elements that provides privacy protection for the user, since the customer's identity is hidden behind his mobile network operator. His mobile network operator introduces the customer with a temporary identity. Note, that privacy protection is an optional feature. It can be turned off by the customer. You can see the notations of the proposed protocol in table 4.6.1.

There is a party in the proposed payment protocol named pay-center. This party is also known as the payment gateway in m-commerce. The payment gateway is a party that connects banks and financial institutes to the mobile network operators. This party should be defined and managed by financial systems since it is dealing with financial transactions. The rest of this section is defining the new mobile payment protocol that will be implemented in seven steps. At the beginning, payer encrypts registration details such as account information, payer's identity, and his phone number with his shared key. This information should be sent to payeŕs mobile network operator.

payer ⇒ payeŕs MNO:
$\{PN_{Payer}, ID_{Payer}, AI_{payer}\}K_1$

There are several challenges in designing the payment protocol. One of these challenges is to prevent privacy violation of payer. Most of the current payment protocols are providing identity protection from eavesdroppers. But, they don't provide identity protection from the merchant. So, merchants can make the profile of their customers without their permission. One of the goals is to avoid possible identity or privacy violation in the payment protocol. In order to overcome the issue of privacy violation, mobile network operators should be involved in the payment process. Besides, temporary identity for the customers should be generated to provide proper privacy protection for them. This temporary identity should be generated based on the customer's phone number and his password identification number. Note that, this *ID* will be generated after a successful authentication.

During the registration process, payer has to set his password identification number (PIN_{Payer}) in order to access his mobile wallet application. This implementation uses two-factor authentication that is an important principle for mobile device access control. Two-factor authentication identifies the user in two steps. The first factor is the mobile wallet application on his mobile (something that he has). The second factor is a password (something that he knows). Then, ID_{Payer} will be computed by hashing payer's phone number (PN_{Payer}) and password identification number (PIN_{Payer}).

$$ID_{Payer} = PN_{Payer} + Hash(PN_{Payer}, PIN_{Payer})$$

Then, payer's mobile network operator decodes the message with his shared key (K_1). Payer's mobile network operator stores necessary information into its database. If registration process is successful, payer's mobile network operator will send a confirmation message to inform payer about the result. Confirmation message is also encrypted with the session key (K_1).

Payer's MNO $\Rightarrow$ Payer: $\{Success/Failed\}$ Encrypted with K_1

After registration, payer receives mobile wallet application through email or downloads it from his mobile network operator website. Mobile wallet application has symmetric key generation and payment software. After successful installation, a set of symmetric keys ($X = \{X_1, X_2, \ldots, X_n\}$) will be generated. They will be stored in payer's mobile device and will be sent to his mobile network operator. They will be used to make secure channels between the payer and his mobile network operator. The payee must go through the similar registration process with his mobile network operator. This enables him to receive the payment amount. Payee generates a set of symmetric keys ($Y = \{Y_1, Y_2, \ldots, Y_n\}$) with his mobile network operator. These keys will be stored into payee's terminal and his mobile network operator's database.

In the protocol, if a person captures details of a payment transaction, he will not be able to use the message again, since all messages are encrypted in the protocol. Besides, these messages include random time-stamp generated numbers in order to protect the protocol from replay attacks. If someone steals the payment device, he can access (X or Y) the shared keys. Therefore, the thief can decode the payment messages and use them for illegal payment. To address this issue, all keys (X and Y) are encrypted in client device (with his key). Note that, this key is only viewable by his mobile network operator. Client does the following steps in order to obtain the client key.

P $\Rightarrow$ Pś Mobile Network Operator:
$\{PN_P$, Current Date and Time, $CK_{Request}\}Pu_{P's\,MNO}$
Pś MNO $\Rightarrow$ P: $\{CK\}$ Encrypted with Pu_P

Current payment protocols support transaction privacy protection from eavesdroppers. However, they don't support transaction privacy protection from the bank. So, it is obvious that who is paying how much to whom for ordering what items in each transaction. Also, some credit card issuers provide categorized spending charts (ex. merchandise, dining, and travel) for their customers. So, the financial institution or bank knows details of the transaction. transaction privacy protection should be provided in the protocol. For this purpose, transaction's details should be encrypted before sending it to pay-center (payment gateway). So, pay-center can't make profiles of the customers and merchants.

The next challenge could be to support non-repudiation. After a successful payment, payer or payee should not be able to deny the transaction. For this purpose, digital signatures should be utilized (mathematical scheme to demonstrate the authenticity of a digital document). Pay-center is responsible for generating time-stamps and verifying digital signatures in the protocol. The proposed payment protocol has seven phases. In the protocol, digital signatures will be verified twice. In phase 2, pay-center verifies payer's digital signature and generates the first time-stamp. In phase 6, pay-center generates the second time-stamp and verifies payee's digital signature. Because of these two verifications, non-repudiation could be provided in the payment protocol.

It is important to prevent replay attacks in payment protocols. Most of the current payment protocols support this feature. This feature should be supported since it is an essential and fundamental feature of a payment protocol. Without a mechanism to prevent replay attacks, the payment transaction may be used again by an eavesdropper. If an eavesdropper captures one of the transactions, he can manipulate the transaction and use it again for illegal purposes. Another restriction is about the keys that prevent replay attacks. This restriction will be discussed later.

In the protocol, there is a mechanism to prevent replay attacks. there are two random and time-stamp generated numbers. The first one is payer's pseudo-ID.

The second one is to prevent replay attacks. This number will be included in the messages in phases 2, 4, 5, 6, and 7 of the payment protocol in order to prevent replay attacks. In this case, a person cannot use a transaction for the second time, since the time-stamp is not matched with the current time. As mentioned earlier, the proposed payment protocol is composed of seven phases as illustrated in figure 1. You can see these seven phases with their details in table 4.6.2. These steps are designed for the mobile platform. These phases should be implemented properly as a payment protocol.

In Summary, payer sends the subtraction request to his mobile network operator. His mobile network operator sends the request to payee's mobile network operator. Payee's mobile network operator sends the request to the payee and receives his response. Payee's mobile network operator sends the reply to payer's mobile network operator. If the payee accepts the request, payer's mobile network operator will initiate the transaction through the payment gateway (pay-center). If payee rejects the request, payer's mobile network operator will inform the payer about the denial. After a successful transaction pay-center informs mobile network operators about the successful result. Then, they inform their clients about the result of their transaction. This notification is part of the recommended protocol but is not part of the existing payment protocols.

After successful completion of these seven phases, the payee will release or deliver the purchased goods or services. As mentioned earlier, one of the challenges in mobile payment is to prevent replay attacks. To prevent replay attacks, payer's mobile network operator and payee's mobile network operator make sure that symmetric keys (X_i and Y_i) have not been used before processing the current payment transaction. Mobile network operators will keep a list of generated secret keys and expire used symmetric keys from the list. Payer and payee may receive an update notification from their mobile network operators when their key is expired. To update their secret keys, they should connect to their mobile network operator and generate a new session key (K_1) by running the key agreement protocol. Then, they generate a new set of secret keys (X and Y) with a new session key (K_1) in offline mode.

4.7 Use Cases

In this section, a simple use case of this protocol will be explained. Assume that a customer wants to buy and item from a salesperson that he just met in the street. The customer likes the item, but he doesn't want to pay its price by cash. He is also worried about his privacy since he doesn't know the sales person. So, he wants to protect his privacy. He can use the recommended protocol. First, he needs to negotiate over the price with the sales person. This can be implemented as part of the mobile app that is doing the payment or it may be a receipt. Then, the customer needs to know the cell phone number of the salesperson. Then, he sends the payment request to the sales person. However, the merchant only sees the temporary identity of the customer.

So, the privacy of the customer is protected. Then, if the merchant accepts the request, the request will be sent to the payment gateway. After a successful transaction, a push notification will be sent to the mobile device of these two people. So, the merchant will give the item to the customer. Note that since both people are using mobile devices in this use case, their mobile network operators are involved in the transaction. The identity of each person and his bank account is stored in his mobile network operator. These mobile network operators are the parties that communicate with the payment gateway.

The next use case is about a price meal payment. Assume that a person wants to buy a meal from a sandwich maker vehicle in the street. In the protocol. First, two parties need to negotiate over the price and the meal that customer wants to order. It may be implemented as part of the application on the phone or it may be in a type of receipt to show how much should be paid. After this, the customer initiated the payment with the amount that they discussed and to the vehicle owner only by asking his mobile number. It is assumed that both people have the proposed payment application on their mobile devices. Then, the vehicle owner receives a payment request. If he accepts, the transaction will be done in pay-center. Then, the vehicle owner give the meal to the customer.

4.8 Cloud Messaging and 3D Secure

Cloud messaging is a mobile service that allows third-party applications to send data or information as push notification from their servers to the operating system on the device. Note that, device operating system is handling these messages. All mobile devices support this feature nowadays. Apple has APNS (Apple Push Notification Service), Google has GCM (Google Cloud Messaging), Microsoft has MPNS (Microsoft Push Notification Service), and Blackberry has BBPS (Blackberry Push Service). This feature is well supported in mobile devices and it is secure to transfer messages from third-party to the mobile device.

Customer may face a pop-up window in the payment process based on 3DS. This pop-up window may be considered as a man in the middle or phishing scam, since its domain is not Visa, MasterCard, the bank' domain, or the merchant's domain. Also, if the generation of a one-time password is required, these protocols utilize text messages for interaction with the customers. This may cause problem when the customers are not in their home country.

3D Secure is an extra layer of security for payment protocols. 3D Secure implementations are the latest pioneers in e-commerce transactions in the secure connection space. 3D Secure (3DS) is an additional security level for online transactions with either credit or debit cards. The 3D secure protocol is based on XML. Visa introduced this protocol to the customers as ”Verified by Visa”. MasterCard introduced it as ”MasterCard SecureCode”. American Express introduced it as ”American Express SafeKey”. JCB International implemented this protocol as ”J/Secure” [35].

These implementations have some issues. Many customers abandon this way of payment because of the security concerns. This method is using a pop-up window to authenticate the cardholder. The source of this pop-up window is not recognizable by the customer. the customer can't recognize domain of this pop-up window. It is not clear if this pop-up is from the bank, Visa, MasterCard, or trusted domains.

This protocol is dependent on the geographic location of the customer at the time of the transaction. It means if the customer is not in his own country, he may have some problems in receiving notifications, one-time passwords, and etc. Text message may be blocked for different reasons in the country that he is currently in.

This protocol is inconvenient for the people who tend to change their number time to time. Some people need to change number because of different reasons. Current type of implementation may be inconvenient for them during the payment process since they need to change their number more often. Also, mobile users sometimes have a problem with viewing the pop-up window on their mobile device, since mobile browsers are simpler browsers. These browsers may have a problem in showing pop-up windows.

There are different types of attacks that may happen in electronic commerce. Card not present fraud is one type of these attacks. It may happen when the merchant is not able to check the card physically. For example, when the customer is ordering an item from an online store, the store is not able to check customer's card physically. Note that in 3DS, customer may experience a pop-up window in the middle of transaction that tries to prevent this type of attack by using behavior-based authentication.

Cloud messaging will be utilized for transferring one time passwords to customers and as the extra security layer in the protocol. This extra security layer prevents card not present fraud. The extra security layer is improved version of the 3D Secure protocol. Note that, by using cloud messaging instead of text messaging, the customer can see the source of notifications and authentications. This approach is not dependent on the current geographic location of the customer.

In this protocol, cloud messaging will be suggested to prevent card not present fraud and to transfer one time passwords to customers to prevent man in the middle attack and phishing scam in this step. Note that, the recommended approach is based on the 3D Secure protocol.

In the suggested payment protocol, cloud messaging will be utilized to add an extra layer of security to the payment protocol. The intention is to prevent card not present fraud in the payment protocol. For this purpose, 3D Secure features should be improved for authenticating a customer. Note that, 3DS is utilizing behavioral data to understand which transactions are suspicious. This protocol will not prompt all the customers, but it only prompts transactions with risk score shows higher than the threshold. There is a behavior model available that assigns each transaction a score based on different factors. If this risk score is less than the threshold, the system allows the customer to continue the transaction. But, if this score is higher than threshold, customer will be prompted by a pop-up window or inline frame in the middle of transaction.

All 3DS features for this extra security level should be borrowed, but instead of showing the extra layer as a pop-up window or inline frame, cloud messaging should be utilized for this purpose. Several issues will be prevented by utilizing cloud messaging instead of a pop-up window (ex. difficulty in viewing pop-up windows in mobile devices, difficulty in receiving one-time passwords and confirmation codes via text message, and unable to see the source of the pop-up window). A notification will be pushed to the customer to make sure that he is the legitimate customer. Note that, all mobile operating systems are equipped with cloud messaging tools. The benefit of this approach is that the customer recognizes the source of push notification. Also, if sending a one-time password is required, this approach will be better, since it doesn't have the limitations of text messaging or emails. Note that, texting and sending email from one country to the other one may be temporarily unavailable or limited, but cloud messaging only require access to the internet.

4.9 Security Analysis

In this section, possible attacks toward the proposed mobile payment protocol will be reviewed as a payment protocol. The first possible threat is replay attack. There may be two possible replay attack scenarios. The first one is that an eavesdropper captures the temporary identity of one of the engaging parties and he wants to introduce himself with this stolen ID. The solution for this threat is the temporary identities are random and time-stamp generated numbers (R_1). So, the eavesdropper can't introduce himself with the stolen ID, since the temporary ID is already expired.

The second possible threat is that an eavesdropper captures a transaction in the payment protocol. He may alter part of the transaction and use it later for illegal purposes. The solution for this threat is another random and time-stamps generated number(R_2). As mentioned earlier, another random and time-stamp generated number is used in the transaction to protect the protocol from replay attacks. In summary, two random and time-stamp generated numbers are used in different phases of the protocol (R_1 and R_2) in order to protect it from replay attacks.

Another possible attack is repudiation attack. Note that, this feature should be optional, since it will have extra cost. The engaging parties should not be able to deny a successful transaction. In the protocol, timestamps will be generated and verified digital signatures twice. The first one is in phase two. In this phase, the first time-stamp will be generated and will verify payer's digital signature in pay-center. The second one is when the second time-stamp will be generated and verifies the merchant's digital signature in phase six.

Another threat is card not present fraud. This type of fraud may happen when merchant can't examine the credit card physically for example, during an online payment. So, the person who is initiating the transaction should be the card owner or the person who is allowed to use the card. To prevent this fraud, An extra security layer is added that is based on behavioral data like location, how often the card owner buys specific items. Note that, features of the 3D Secure protocol should be borrowed for the extra security layer. However, cloud messaging should be utilized instead of the pop-up window or inline frame. In this case, the customer can recognize the source of the push notification. So, man in the middle attack or phishing scam is not possible. All users should not be prompted with this extra layer. 3DS model should be used to calculate the risk of each transaction based on available history data. If the score is higher than a threshold, the payment process will be prompted to make sure the person is the legitimate card owner.

4.10 Protocol Properties and How They Are Met

This protocol has different properties. Let's discuss how these properties are met in the payment protocol. A new private mobile payment protocol will be described based on the client-centric model. This protocol is suitable for mobile devices via mobile and wireless networks.

Improved version of Diffie-Hellman key agreement protocol will be suggested for establishing secure connections among engaging parties. The Logarithm is used instead of powering in the process of Diffie-Hellman. This change will make the computation cost less and make the process suitable for mobile platform even from the aspect of required variable size. Based on chapter 5 experiments, the proposed key agreement protocol is twice as fast as Diffie-Hellman that is the most popular existing key agreement protocol.

Suitable privacy will be provided for the payer (cardholder) by involving mobile network operators in the payment process and generating temporary identities. These temporary identities will be generated for the transactions and they are timestamp generated numbers. These identities are not reusable. Also, the merchant and the payment gateway will not be able to see the real identity of the payer.

The risk of replay attacks will be decreased by using random time-stamp generated numbers in the payment process. The transactions are not reusable since the timestamps will be checked before doing the transactions. Digital signatures could be utilized in order to provide non-repudiation in the protocol. In the steps of this protocol, payer and payee sign the transaction by their digital signature. So, they can't deny their contribution on the transaction after committing it.

Cloud messaging should be utilized to prevent card not present fraud and to transfer one time passwords to customers in order to prevent man in the middle attack and phishing scam in this step. Note that, the approach is based on the 3D Secure protocol. Push notification is a useful feature of the mobile platform. This feature is widely supported by the mobile platform. Information will be sent to the mobile devices by push notification instead of sending them by text messages. Also, push notification service is a cloud service and only needs the internet. This feature doesn't depend on the current geographic location of the user.

Oh, come with old Khayyàm, and leave the Wise To talk; one thing is certain, that Life flies; One thing is certain, and the Rest is Lies; The Flower that once has blown forever dies.

Omar Khayyam

5

Experimental Validation

Evaluations and experiment results will be reviewed in this chapter.

5.1 Evaluation of the Recommended Key Agreement Protocol

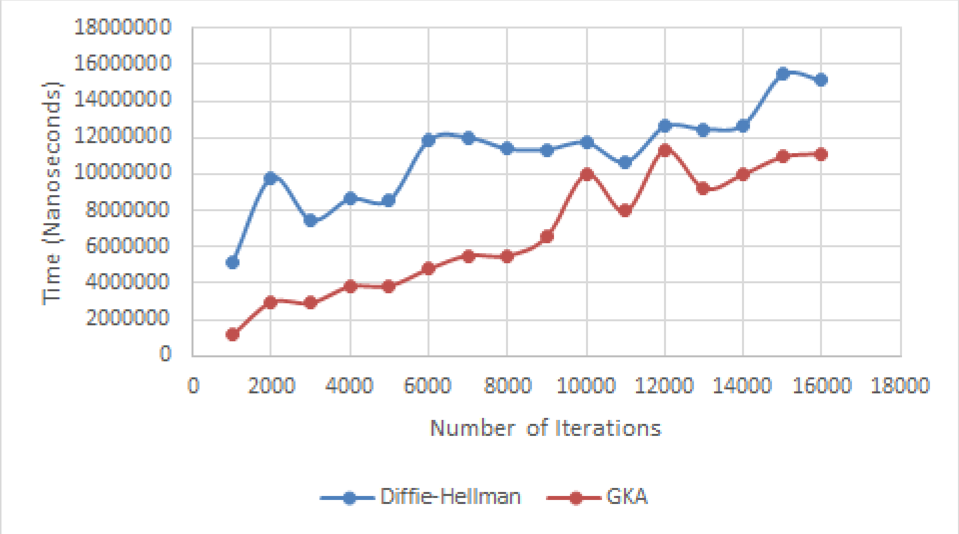


Figure 4: Comparison of Golden Key Agreement Protocol with Diffie-Hellman (more than one iteration)

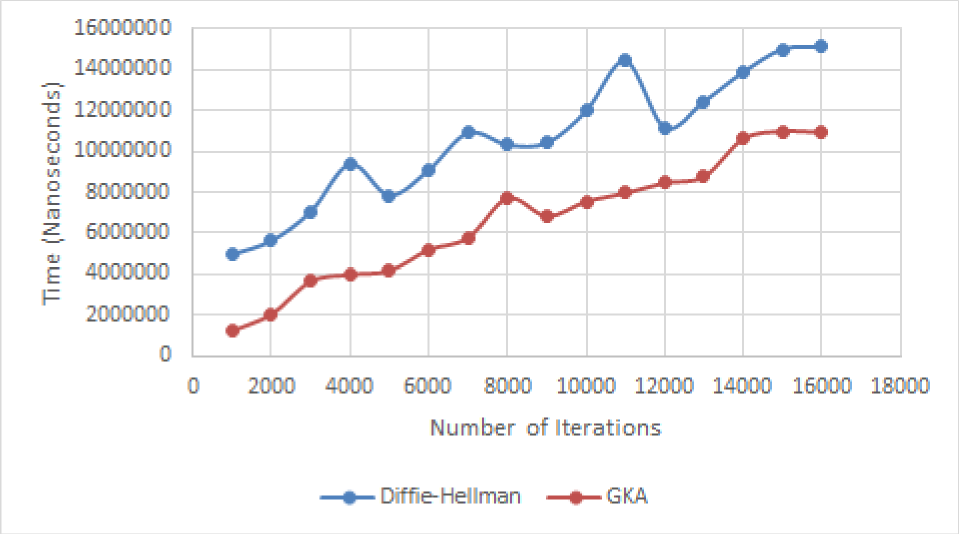


Figure 5: Comparison of Golden Key Agreement Protocol with Diffie-Hellman (more than one iteration)

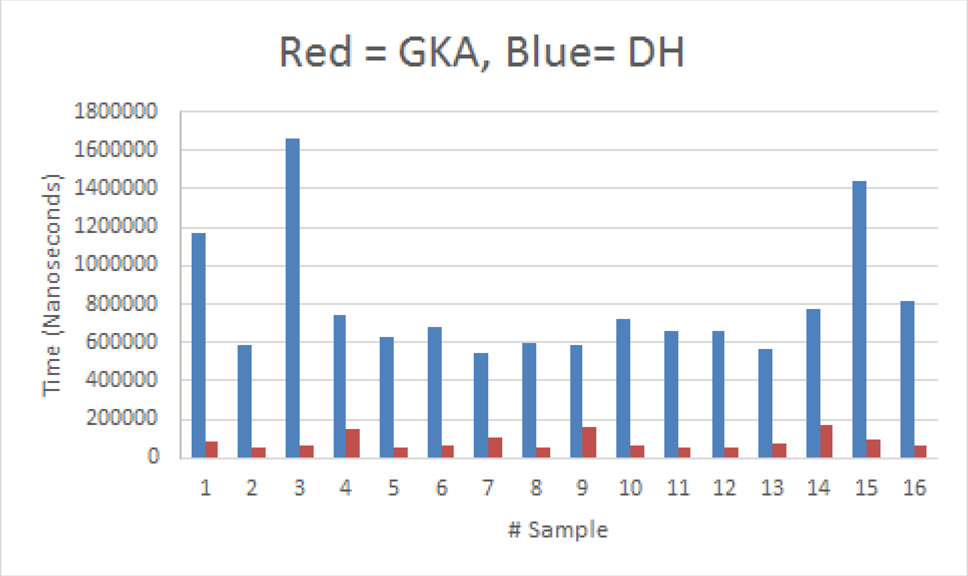


Figure 6: Comparison of Pace protocol with Diffie-Hellman (one iteration)

In this part, Golden Key Agreement Protocol (Pace protocol) will be compared with Diffie-Hellman. These two algorithms are implemented and tested with different parameters. All the experiments are done with a laptop with core-i7 CPU (2670 QM- 2.20 GHz), 8 GB of Ram, and windows seven 64-bit operating system. The first experiment was with the parameters in below.

$q(Pace) = 15485863$
p(DH) = 15485863
p(Pace) = Random(2,10)
g(DH) = Random(2,10)
n(Pace) = 1

You can see the result of these experiments in figure 4. The second experiment was with the parameters in below.

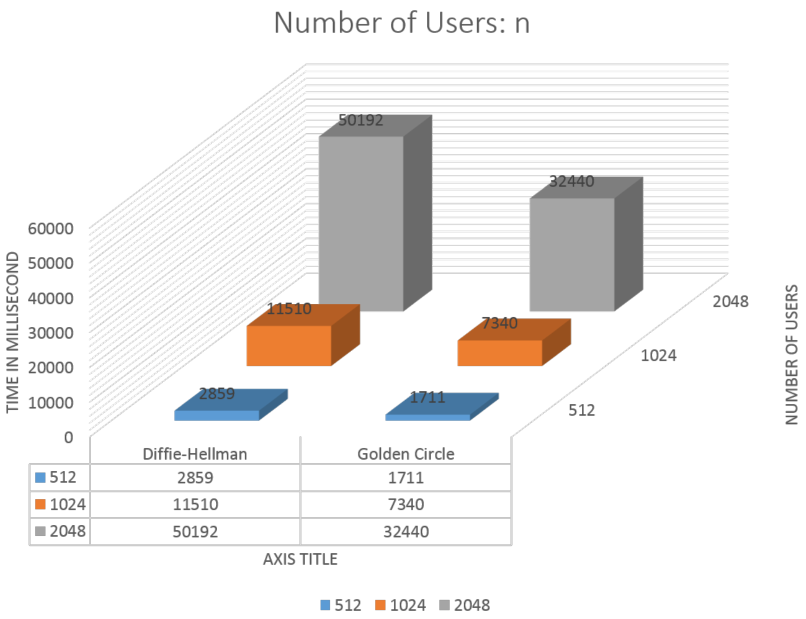
$q(Pace) = 982451653$

p(DH) = 982451653
p(Pace) = Random(2,10)
g(DH) = Random(2,10)
n(Pace) = 1

You can see the result of these experiments in figure 5. In the third and last experiment, The protocol is run against Diffie-Hellman 16 times and the running time of both protocols are recorded. You can see result of these experiments in figure 6. As you see in the figures, the protocol is faster than Diffie-Hellman. In these sixteen experiments, the average running time of Diffie-Hellman was 801268.8125 nanoseconds. The average running time of the protocol in these experiments was 86662.5625 nanoseconds.

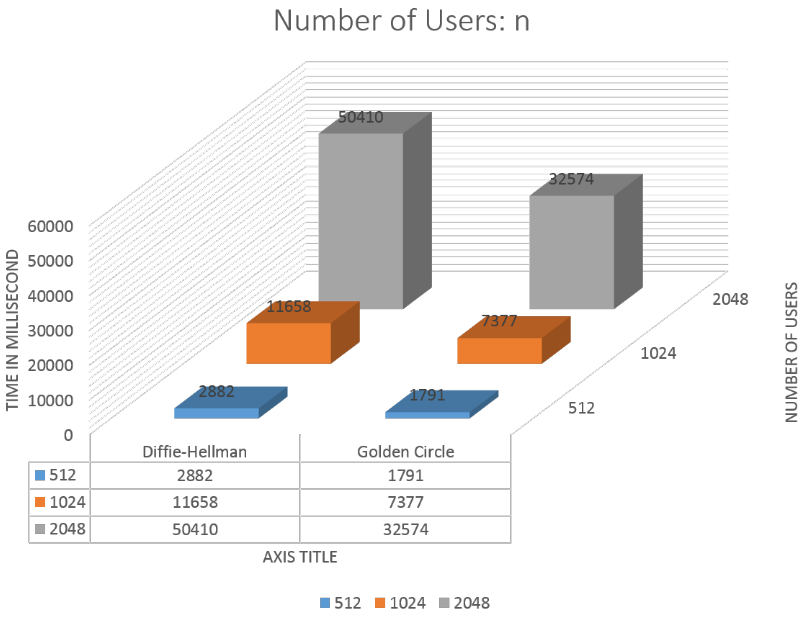
5.2 Evaluation of Proposed Circular Group Key Agreement Protocols

This section presents the experimental results of the GC algorithm. The simulation is conducted with 8 GB of memory. The simulation used Intel Core i7- 2670 QM Processor (6M Cache, up to 3.10 GHz) as processing unit. The simulation was in Windows 7 64-bit and the program was written is Java. The private keys of the users were randomly generated numbers between 1 and 1024. In the Diffie-Hellman algorithm, *p* was a random prime number. In addition, *q* or generator was also a random number. Likewise, in GC, all the private keys of n users were randomly generated numbers between 1 and 1024. In GC, p, q, and *m* were also randomly generated numbers. The test environments were identical for these two protocols.



	Diffie-Hellman	Golden Circle
512	2859	1711
1024	11510	7340
2048	50192	32440

Figure 7: Time Comparison- Experiment 1



	Diffie-Hellman	Golden Circle
512	2882	1791
1024	11658	7377
2048	50410	32574

Figure 8: Time Comparison- Experiment 2

According to the complexity of the GC, it is better than Diffie-Hellman when the number of users grows. The result of the simulation can be seen below. Better running times have been recorded especially when the number of users grows. The algorithm was using mod and logarithm. Diffie-Hellman was using power and mod. The specific type of the mathematics functions that is used, will make running time of the algorithm much better when n grows. The simulation for generating a shared session key between n users was done for three numbers as number of users (n). The result of the simulation can be seen in the figure 7.

In addition, another experiment has been done with another setup. The result can be found in figure 8. The experiment was conducted with 8 GB of memory. The simulation used Intel Core i7- 2670 QM Processor (6M Cache, up to 3.10 GHz) as processing unit. The experiment was on Windows 7 64-bit and the program was written in Java. The private keys of the users were randomly generated numbers between 1 and 2048. p in the Diffie-Hellman experiment was a random prime number (greater than 122949823). In addition, q or generator was a random number. In GC, all the private keys of n users were randomly generated numbers between 1 and 2048. In GC, p, q, and m were randomly generated numbers. The environments of the two algorithms were identical.

As you can see in the figures when the number of users who want a shared session key grow you can see the difference between the two methods. When the number of users is 512 people, the difference between the running time of the two algorithms is 843 milliseconds. When the number of users is 1024, the difference is 4298 milliseconds. Finally, the difference when the number of users is 2048, is 19021 milliseconds. As a result of this, the proposed protocol for generating the shared session key between n users could improve the running time of Diffie-Hellman especially for a group with large number of users.

5.3 Evaluation of the Recommended Payment Protocol

In this section, the payment protocol will be evaluated from the aspect of performance.

5.3.1 Performance Evaluation- Simulation

In this section, the execution time of the improved version of Diffie-Hellman will be compared with the original one. For this experiment, a virtual machine with 7 GB of memory is used. This virtual machine had two CPUs and each CPU had two cores. The operating system was windows seven 64-bit. For this experiment, two Java procedures were developed with random generated selected private keys. One of these procedures was for Diffie-Hellman and the other one was the improved version. The protocol was run against Diffie-Hellman 100 times to see the difference.

In the first experiment, two protocols were run against each other 100 times and stacked time of these experiments was kept. In these experiments, the prime number was 7. You can see the result of these experiments in figure 9. The horizontal axis is showing a number of the experiments and vertical axis is showing the stacked time based on milliseconds. After 100 experiments, average execution time of the protocol was 34,930 milliseconds and average execution time of the original version was 63,116 milliseconds.

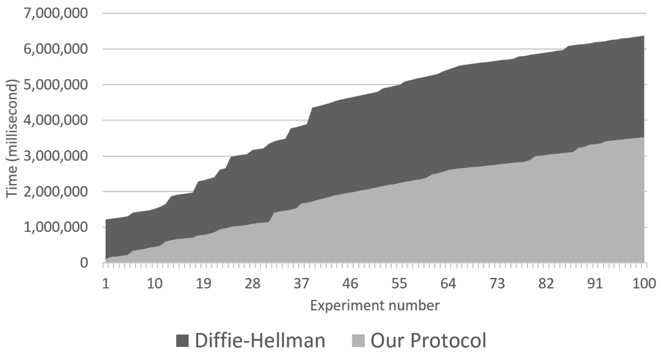


Figure 9: Performance evaluation: prime = 7 and 100 iterations

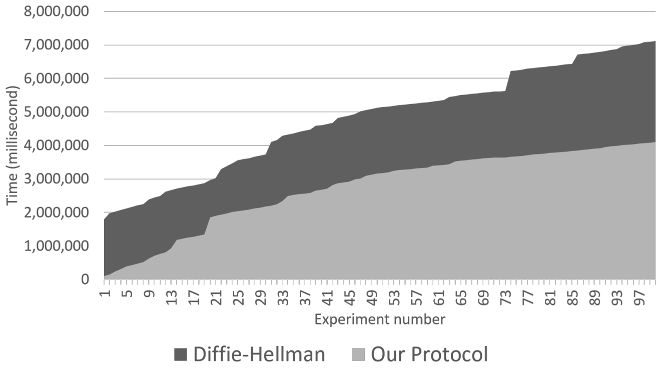


Figure 10: Performance evaluation: prime = 982,451,653 and 100 iterations

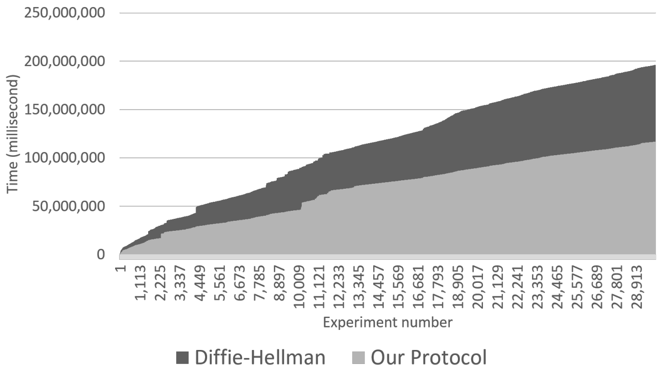


Figure 11: Performance evaluation: prime = 7 and 30,000 iterations

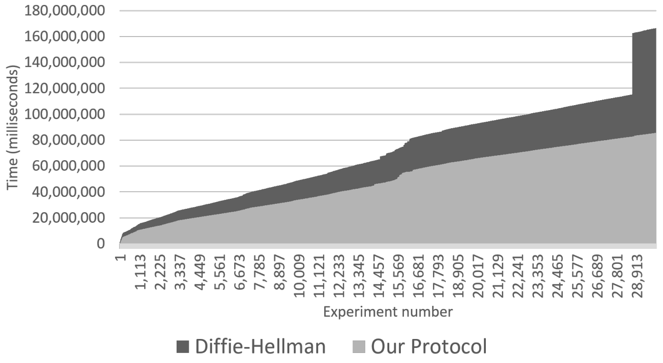


Figure 12: Performance evaluation: prime = 982,451,653 and 30,000 iterations

In the next experiment, a large prime number is selected. The protocols ran against each other 100 times and stacked time is kept of the protocol and Diffie-Hellman protocol for all these experiments. The prime number was 982,451,653. You can see the result of these experiments in figure 10. After 100 experiments, average execution time of the protocol was 40,573 milliseconds and average execution time of the original version was 70,434 milliseconds.

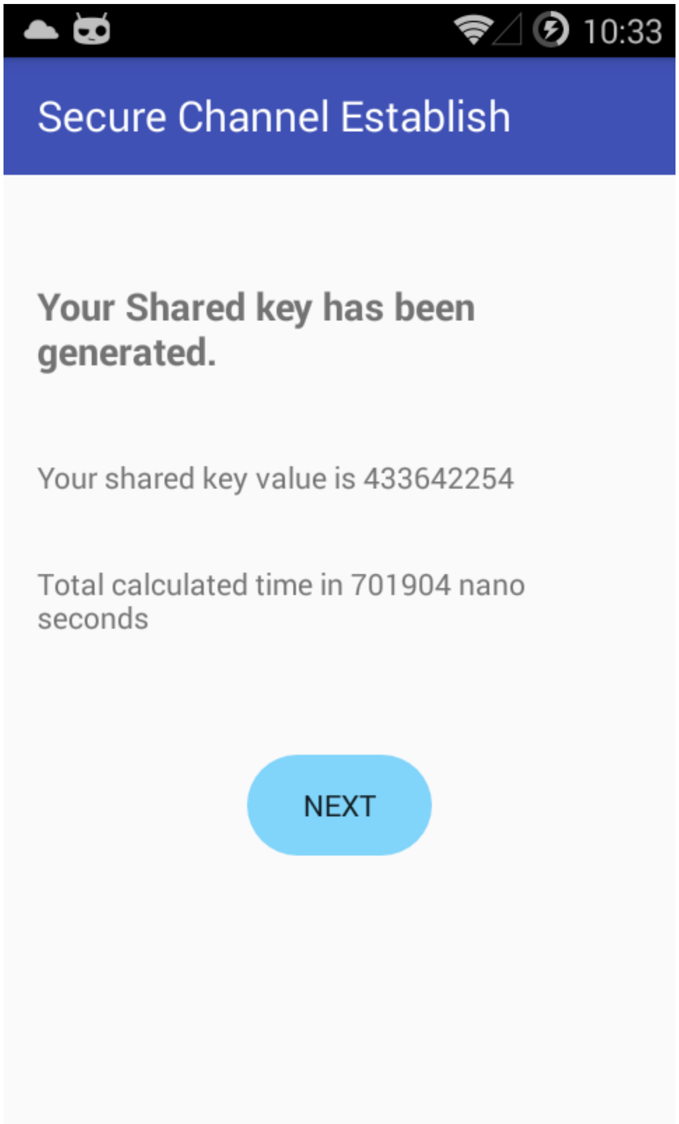
In the third experiment, the protocols were run against each other 30,000 times and the stacked is kept the time of the protocol and Diffie-Hellman protocol for all these experiments. The prime number was 7. You can see the result of these experiments in figure 11. After 30,000 experiments, the average execution time of the protocol was 3,899 milliseconds and average execution time of the original version was 6,545 milliseconds.

Finally, the procedures were run against each other 30,000 times with a big prime number and the stacked time was kept for the recommended protocol and Diffie-Hellman for all these experiments. The prime number was 982,451,653. After 30,000 experiments, the average execution time of the protocol was 2,855 milliseconds and average execution time of the original version was 5,548 milliseconds. You can see the result of this experiment in figure 12.

In summary, the protocol was almost twice as fast as Diffie-Hellman based on the statistics in the majority of the experiments. This is an important improvement since the recommended key agreement has less computational cost than Diffie-Hellman. Therefore, the recommended protocol is faster than D-H. So, the improved protocol is used instead of the original one in order to propose a payment protocol proper for mobile platform.

5.3.2 Performance Evaluation- Real Environment

For this experiment, the recommended Android app (Appendix A) has been installed on two mobile devices. These devices were Samsung Galaxy S3 mini and LG D820 Google Nexus. These mobile devices were connected to another mobile device as the hotspot. The hotspot had Verizon 4G LTE network. The experiment was conducted with two different approaches: Diffie-Hellman and Pace Protocol. As you can see in figure 13 and 14, Pace protocol was more than twice as fast as the Diffie-Hellman protocol. Note that, the time were recorded on Samsung Galaxy S3 mini.



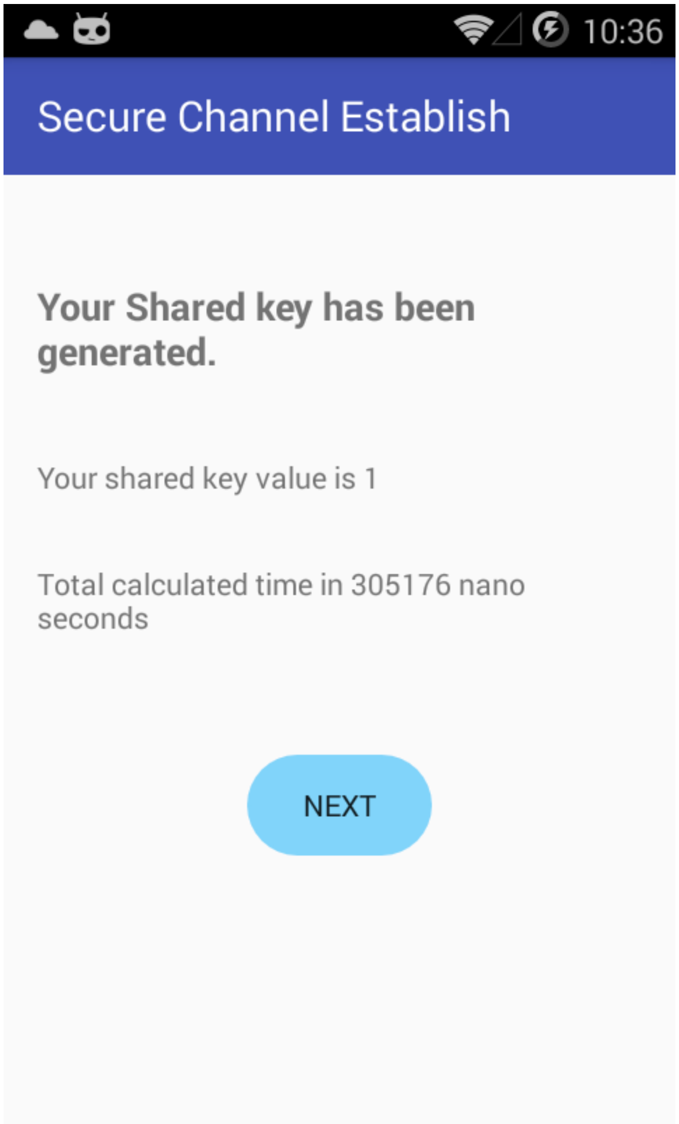
Secure Channel Establish

Your Shared key has been generated.

Your shared key value is 433642254

Total calculated time in 701904 nano seconds

Figure 13: Diffie-Hellman protocol on Samsung Galaxy S3 mini



Secure Channel Establish

Your Shared key has been generated.

Your shared key value is 1

Total calculated time in 305176 nano seconds

Figure 14: Pace protocol on Samsung Galaxy S3 mini

5.3.3 Linear Group Key Agreement Protocol Comparison

In this part we are going to compare the recommended linear group key agreement protocol with DL08 and KON08. This comparison is from aspect of number of rounds, total number of sent messages, total number of received messages, and computations cost. You can see the result of these comparisons in table 5.3.1.

5.3.4 MPCP Comparison

The recommended payment protocol has different features. Some of the features of the recommended payment protocol are measurable like the runtime of the key agreement protocol. In order to see what are these features and how they are met in the recommended protocol, please refer to chapter 4. For example, decreasing the risk of replay attacks, providing non-repudiation (optional feature), and providing privacy protection are not measurable features. However, it has been explained how these features are met in this research in chapter 4. Note that This protocol is compatible with the mobile platform since the computational and communication cost of the protocol to establish a secure channel between the parties is lighter and suggested 3D implementation is by utilizing the cloud messaging that is well supported in the mobile platform.

Protocol	Rounds	Sent Messages	Received Messages	Computations
DL08	3	$7n/2$	$3n + nlog_4n$	$15n/2 + 2n[log_4n]$
KON08	$log_2n/3$	$4n$	$4n$	$21n/2$
GLGKA	2	$3n$	$3n$	$2n + 2$

Table 5.3.1: Group Key Agreement Protocols Comparison Result

In this section, the proposed payment protocol will be compared with existing payment protocols from the aspect of providing privacy protection, being compatible with mobile devices, preventing card not present fraud by an extra security layer, preventing man in the middle attack and phishing scam in the extra security layer. There are two types of identity protection that can be provided for a customer in a payment protocol. They are protection from merchant and protection from eavesdrops. Also, there are two types of transaction privacy protection. They are protection from related financial institution and protection from eavesdrops.

Table 5.3.2 shows the result of these comparisons. As you see in the table, only the proposed protocol protects payerś (customer or client) identity from the merchant (payee). Only the protocol protects transaction's privacy from the bank or related financial institution. Besides, the calculations for generating a shared key between parties is proper for the mobile platform because 3DS and the protocol are providing and extra security layer to prevent card not present fraud.

However, only the protocol is able to prevent man in the middle attack and phishing scam in this extra security level, since cloud messaging is utilized instead of the pop-up window. Note that, cloud messaging is available in all mobile operating systems these days. Also, some mobile devices or their browsers may not support pop-up window or iframe. So, the customer may experience difficulty in this step. But, the approach is compatible with all mobile devices, since push notification is utilized that all mobile devices support. The recommended implementation of the extra security layer is not restricted to the geographic location of the customer. In the recommended implementation, the customer can see the source of authentication.

Key Points	SET	iKP	KSL	MPCP
Identity Protection from Merchant	N	N	N	Y
Identity Protection from Eavesdrop	Y	Y	Y	Y
Transaction Privacy Protection from Eavesdrop	Y	Y	Y	Y
Transaction Privacy Protection from Bank	N	N	N	Y
Compatible with Mobile Devices	N	N	N	Y

Table 5.3.2: Comparison

5.4 Phases of a Usecase- Alice Orders a Guitar

In this use case, Alice wants to buy a guitar from Bob online store. After she chooses the guitar and put it in her shopping basket, Bobś website informs her about the price of the specific guitar. Alice initiates a payment transaction with the description of the guitar that she wants to buy and the Bobś ID. You can see the assumptions of this specific use case in below.

$TID = 1234567$
$R_1 = 217$- Random number by Payer act as Payerś pseudo-ID
$R_2 = 156$- Random number to protect against replay attack
$K_1 = 13$- Shared key between payer and his mobile network operator
$K_2 = 15$- Shared key between payer and payee
Payer = Alice
Payerś MNO = Verizon
Payee = Bob
Payeeś MNO = AT&T
$Amount = 200$
$Date = 5/1/2017$
$K_{p\text{-}p} = 12$
$Desc =$ Alice Address- Payment Details- Item Details
$X = \{7, 77, 777\}$X series of keys

$Y = \{7, 77, 777\}$Y series of keys
$i = 2$
$x_2 = 77$
$y_2 = 77$
$x_3 = 777$
$y_3 = 777$
$TimeStamp1 = 5/1/17 - 13 : 10 : 7$
$TimeStamp2 = 5/1/17 - 13 : 10 : 27$

You can see the seven phases of our protocol for this use case in below. Phase 1 is payment initiation. The payment subtraction request would be sent by Alice in phase 2. In phase 3 Verizon try to authorize the request. AT&T sends the request to the merchant in phase 4. AT&T receives a response from the merchant in phase 5. Payment authorization response will be received by the pay-center from AT&T in phase 6. Verizon and AT&T will send the transaction result (notification) to the customer and the merchant in phase 7.

In phase 1, Alice requests Bobś identity and the transactionś identity from Bob. Alice also sends R_1 (random number by payer act as payerś pseudo-ID) to Bob. Then, Bob sends his identity, transaction ID, his mobile network operator's ID to Alice. Note that Bobś message to Alice is encrypted by K_2 (the shared key between payer and payee).

Phase 1: Payment Initialization:
Alice ⇒ Bob: $217,\ TID_{Req},\ BobID_{Req}$
Bob ⇒ Alice: $\{ID_{Bob},\ 1234567,\ ID_{\mathrm{AT\&T}}\}15$

In phase 2, Alice sends Bobś identity, his mobile network operatorś identity, transaction ID, transaction amount, current date, R_1, transaction details (encrypted), and R_2 (Random number to protect against replay attack) to her mobile network operator. Then, Verizon informs the pay-center about the transaction. Then, pay-center verifies Alice's digital signature and generates the first time-stamp. Note that transaction details are encrypted with K_2.

Phase 2: Payment Subtraction Request:
Alice ⇒ Verizon: { ID_{Bob}, $ID_{\mathrm{AT\&T}}$, 217, 1234567, $\$200$, 5/1/17, 156,
$H(ID_{Bob}, ID_{\mathrm{AT\&T}}, 217, 1234567, \$200, 5/1/17, 156)$,
{156, Alice Address- Item Details } 15} 77, 2, ID_{Alice}

Verizon ⇒ pay-center:
$H[\{ID_{Bob}, ID_{\mathrm{AT\&T}}, 217, 1234567, \$200, 5/1/17, 156,$
H (ID_{Bob}, $ID_{\mathrm{AT\&T}}$, 217, 1234567, $\$200$, 5/1/17, 156),
{156, Alice Address- Item Details}15}77, 2, ID_{Alice}]

pay-center ⇒ Verizon: generates '5/1/17-13:10:7'
and verifies Aliceś digital signature

In phase 3, Verizon as Aliceś mobile network operator communicates with AT&T as Bobś mobile network operator. Verizon sends Bobś ID, transaction ID, current date, transaction details (encrypted), R_1, and the amount to AT&T. Note that transaction details are encrypted with K_2.

Phase 3: Payment Authorization Request:
Verizon ⇒ AT&T: 217, ID_{Bob}, 1234567,
$\$200$, 5/1/17, {217, Alice Adress- Item Details}15

In this phase, AT&T sends R_1, the amount of the transaction, current date, transaction ID, and *i* to Bob.

Phase 4: Payment Confirmation Request:
AT&T ⇒ Bob:

$\{217, 1234567,\ \$200,\ 5/1/17,\ \{217,\ \text{Alice Adress- Item Details}\}15,\ 156,$
$H(217, 1234567,\ \$200,\ 5/1/17,\ \{217,\ \text{Alice Adress- Item Details}\}15,\ 156),\ H(12)\}77,\ 2$

In this phase, Bob replies to AT&T. Bob accepts the payment. Bob sends R_2 and his response to AT&T.

Phase 5: Payment Confirmation Response:
Bob ⇒ AT&T: $\{Yes,\ 156,\ H(12,\ H(217,\ 1234567,\ \$200,\ 5/1/17,$
$\{217,\ \text{Alice Adress- Item Details}\}15,\ 156),\ \{Yes,\ 1234567,\ \$200,\ 5/1/17\}15\}777$

In this phase, AT&T sends Bobś response to the pay-center. Then, pay-center generates the second timestamp and verifies Bobś digital signature. The transaction will be committed in this phase.

Phase 6: Payment Authorization Response:
AT&T ⇒ pay-center: $H(\{Yes,\ 156,\ H(12),$
$H(217,\ 1234567,\ \$200,\ 5/1/17,\ \{217,\ \text{Alice Adress- Item Details}\}15,\ 156),$
{ Yes, 1234567, \$200, ‘5/1/17’ } 15} 777)

pay-center ⇒ AT&T: generates ’5/1/17-13:10:27’
and verifies Bobś digital signature
AT&T ⇒ Verizon:
$Yes,\ 1234567,\ \$200,\ 5/1/17,\ \{Yes,\ 1234567,\ \$200,\ 5/1/17\}15$

Finally, phase 7 is the notification phase. Alice and Bob will be informed about the result of the transaction through their mobile network operators. If the result of the transaction is a success, Bob will send the guitar to Alice address.

Phase 7: Payment Subtraction Response:
Verizon ⇒ Alice: $\{Yes,\ 156,\ H(12),\ H(ID_{Bob},\ ID_{\text{AT\&T}},\ 217,$

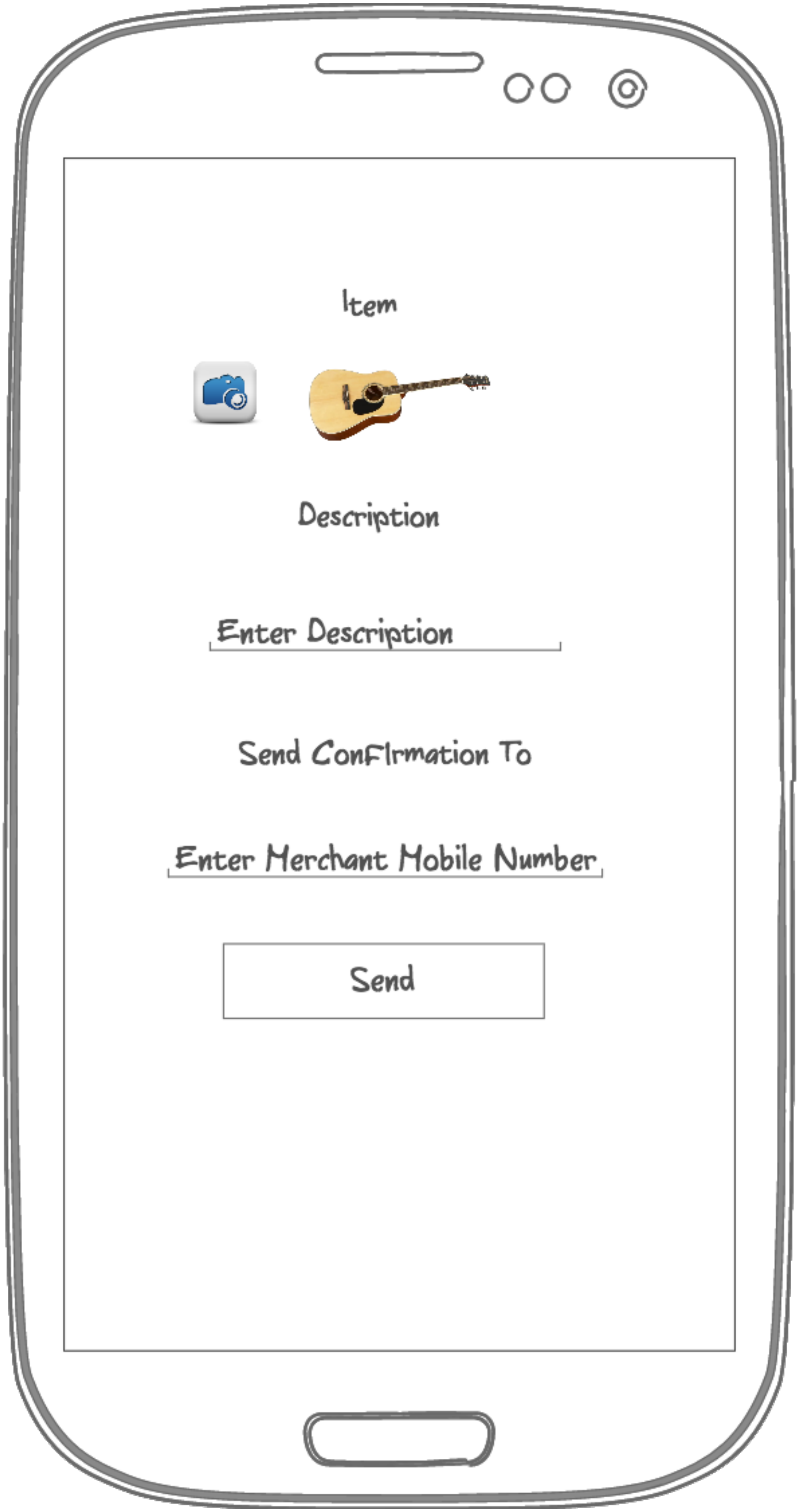
$1234567,\ \$200,\ 5/1/17,\ 156),\ \{\mathit{Yes},\ 1234567,\ \$200,\ 5/1/17\}15\}777$
AT&T ⇒ Bob: $\{\mathit{Received},\ 156,\ H(12),\ H(217,\ 1234567,\ \$200,$
$5/1/17,\ \{217,$ Alice Adress- Item Details$\}15,\ 156)\}777$

5.5 Proposed Mobile Payment Protocol Usecase- Buying Guitar in the Street

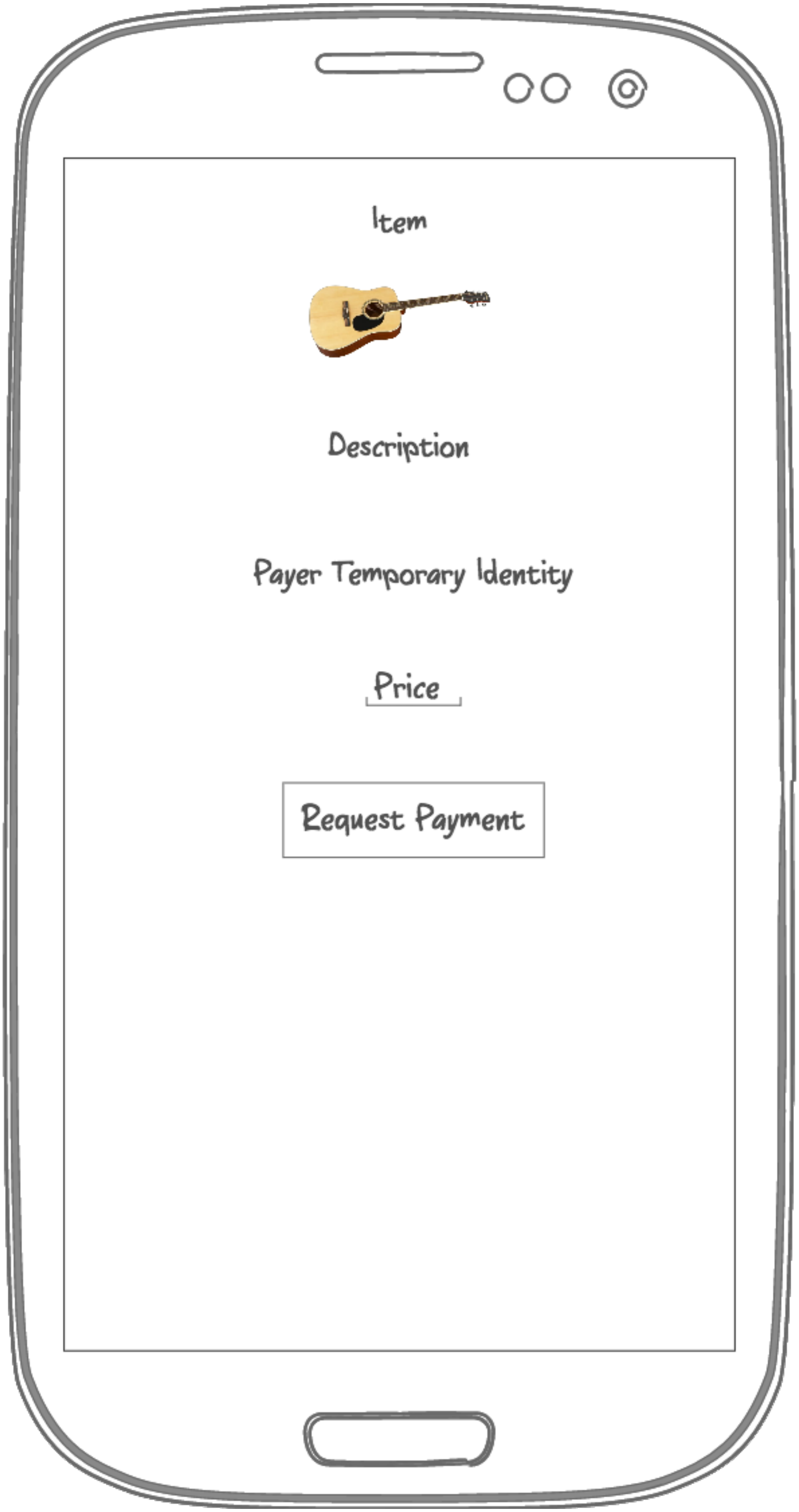
A usecase of the recommended payment protocol will be reviewed in this section. In this usecase, a person (payer) wants to buy a guitar from a person in the street (the merchant). The payer doesn't want to reveal his identity to the merchant. The only information that the payer needs is the mobile number of the merchant. Payer takes a screenshot of the guitar that he wants to buy and all along with a brief description will send the buy request to the merchant. Merchant only sees the temporary identity of the buyer unless the buyer feels comfortable to reveal his identity. Then, the merchant sends the price to the payer. The payer checks the price and if it is acceptable, he will initiate the payment request. Note that at the beginning of the transaction, payer should have a secure channel with his MNO (Mobile Network Operator) and another secure channel with the merchant. To see the details of these steps, read chapter 4. The payer sends the payment request all along with the item description to his MNO.

His MNO sends the payment request to the merchant MNO. Merchant MNO sends the request to the merchant and receives his reply. The merchant's MNO sends the reply to the payer's MNO. The payer's MNO will initiate the transaction through the pay-center if the reply is accepted. The pay-center verifies the payer MNO and merchant MNO. Then, the pay-center will do the transactions with the cooperation of the banks of the two parties. Then, the pay-center will send a success notification to the payer MNO and merchant MNO if the transaction goes through. Then, the MNOs deliver the messages to the payer and merchant. If the transaction succeeds, the merchant will release the item. Then, the payer will receive the item. You can see the mockup of these steps and the transaction flow in the figures below.

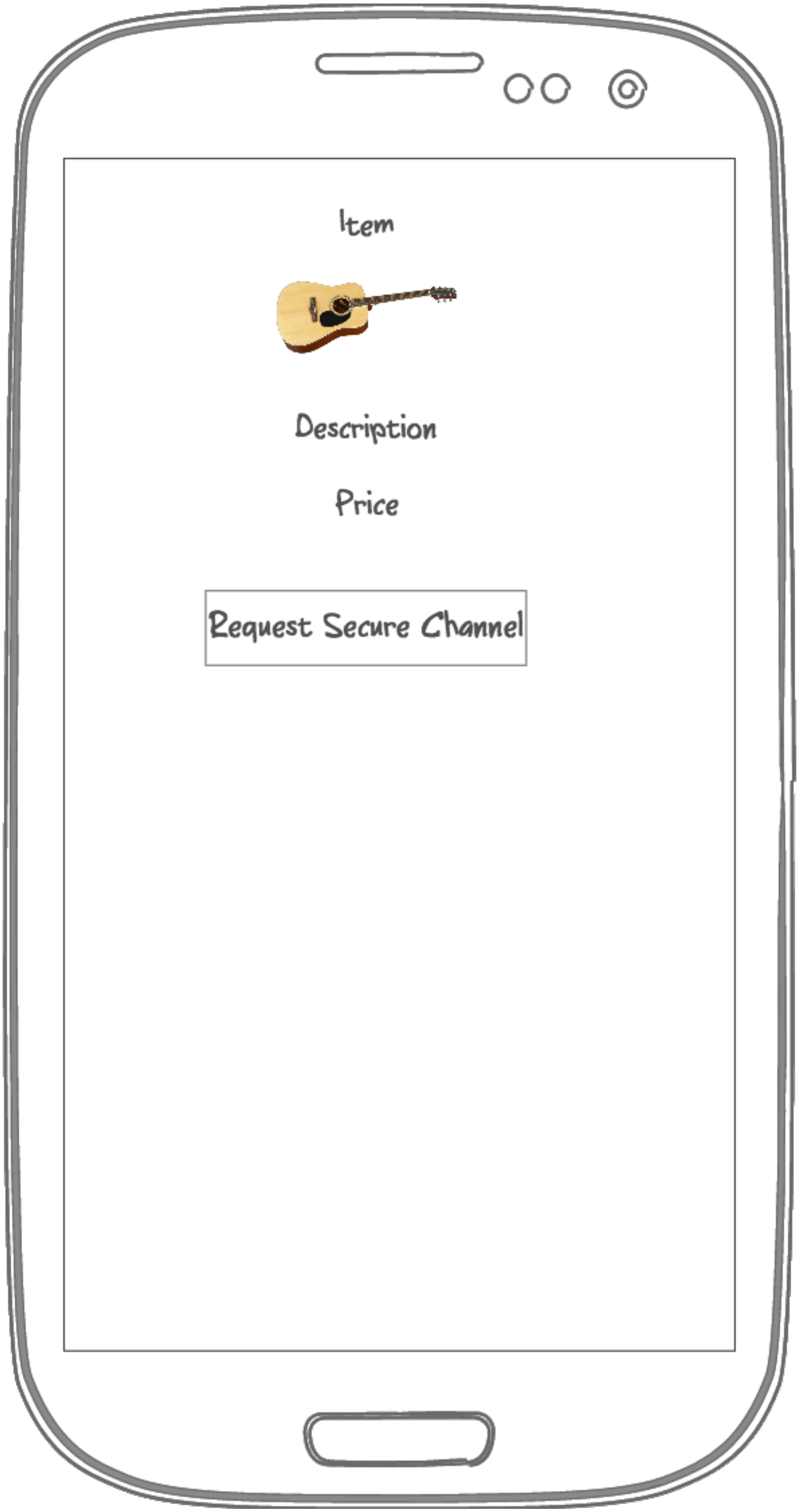
1 - Page 1


Item
Description
Enter Description
Send ConfIrmation To
Enter Merchant Mobile Number
Send

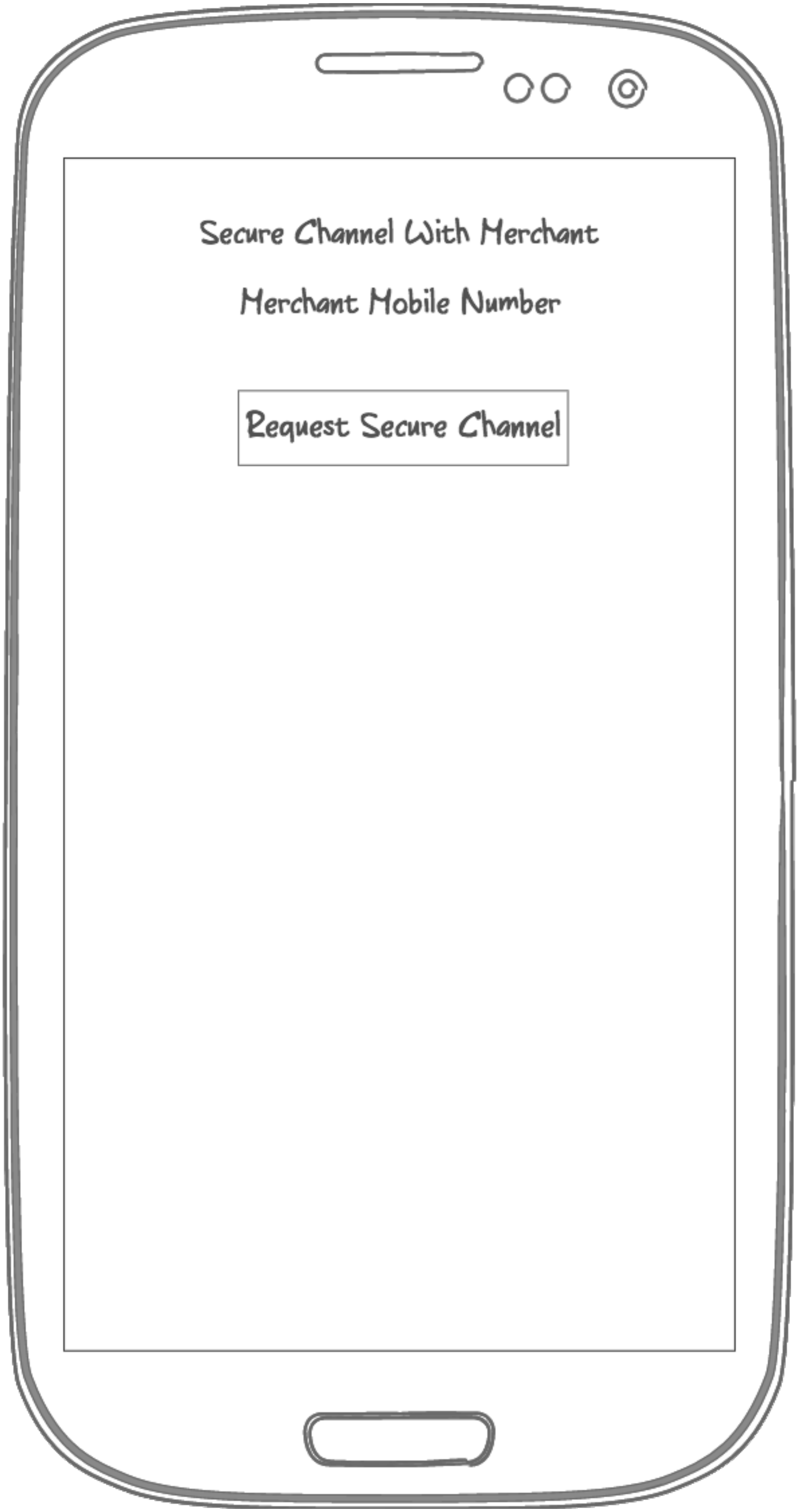
2 - Page 2

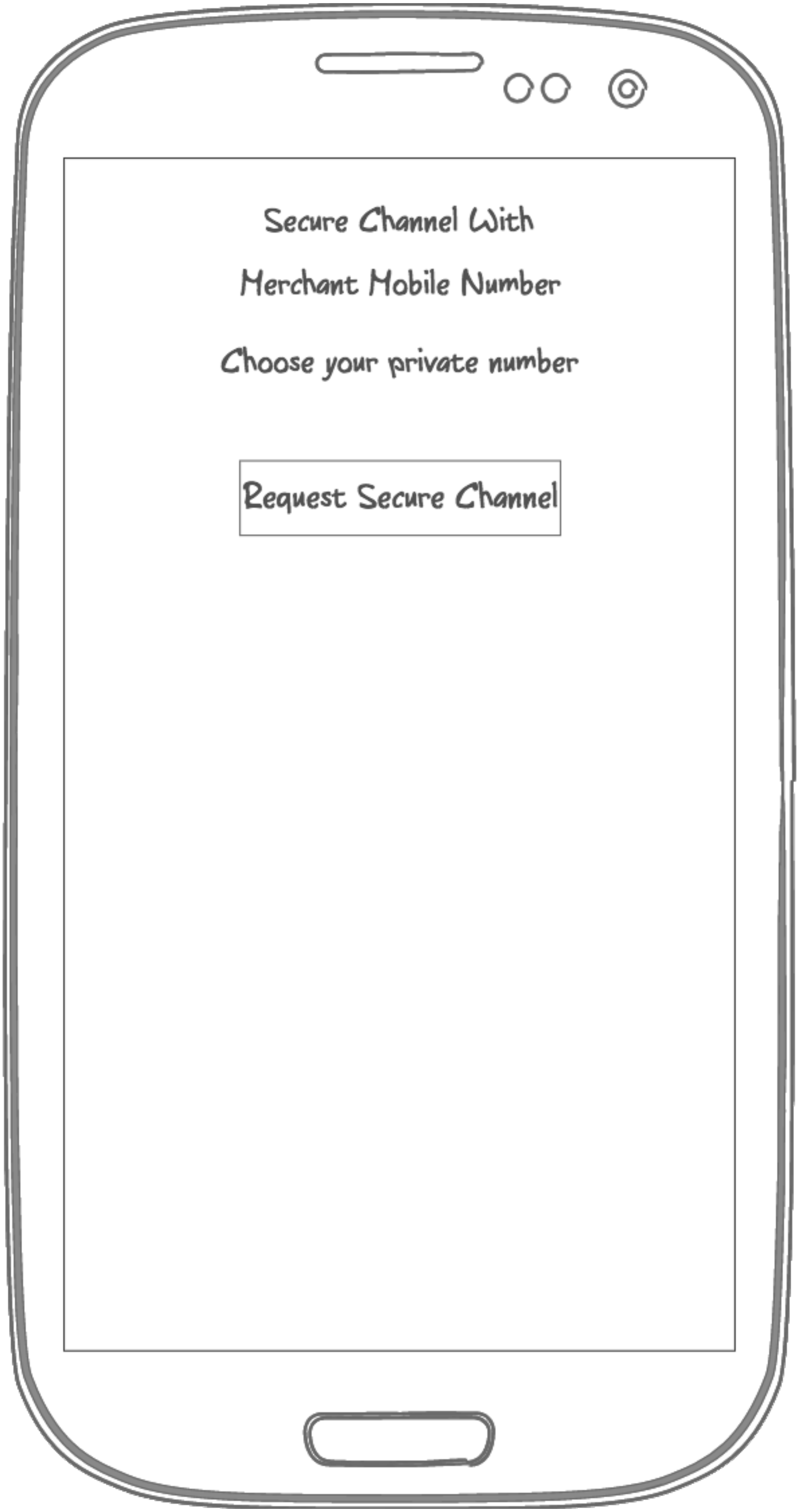

Item
Description
Payer Temporary Identity
Price
Request Payment

3 - Page 3



4 - Page 4


Secure Channel With Merchant
Merchant Mobile Number
Request Secure Channel

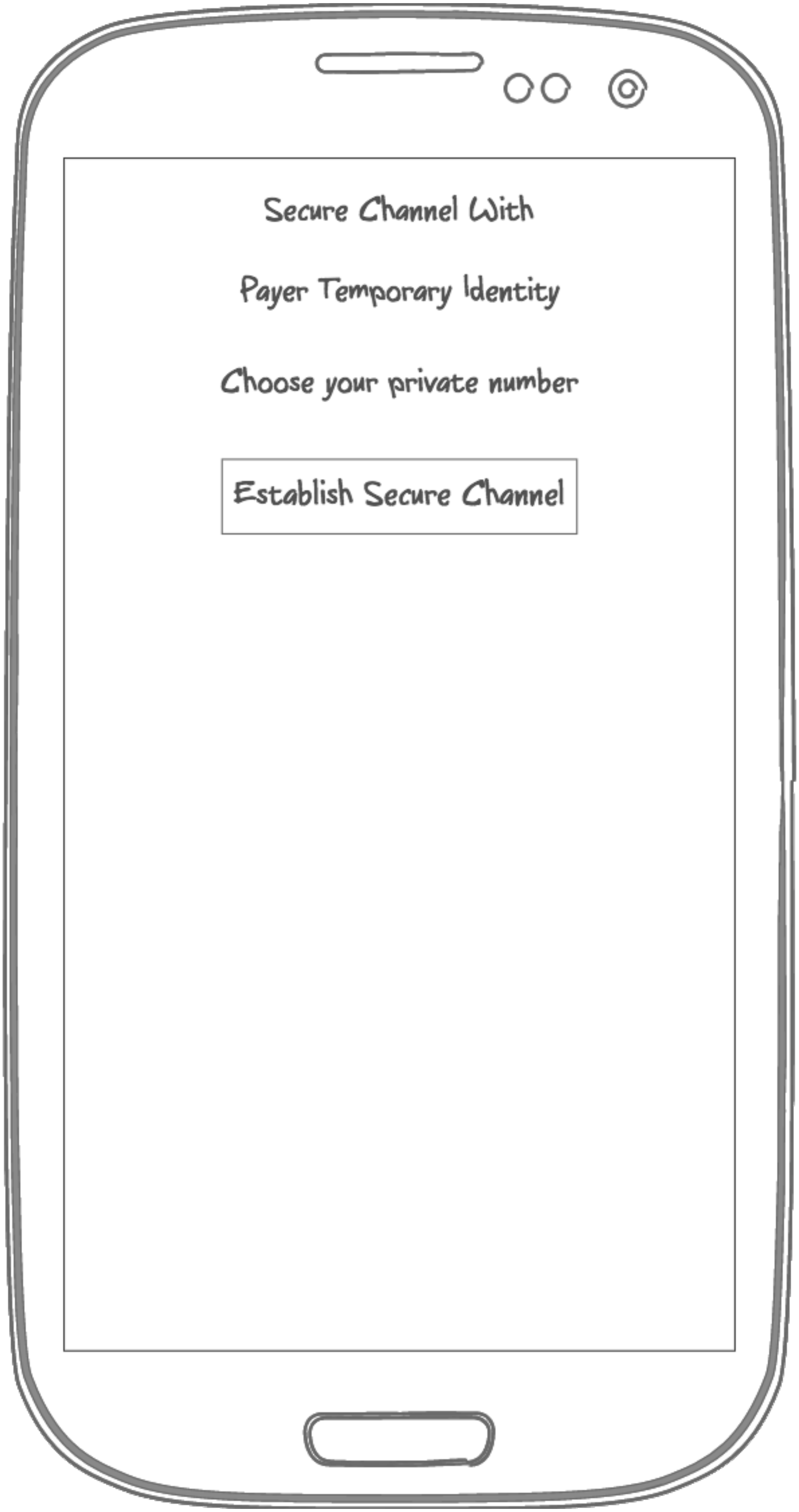


Secure Channel With

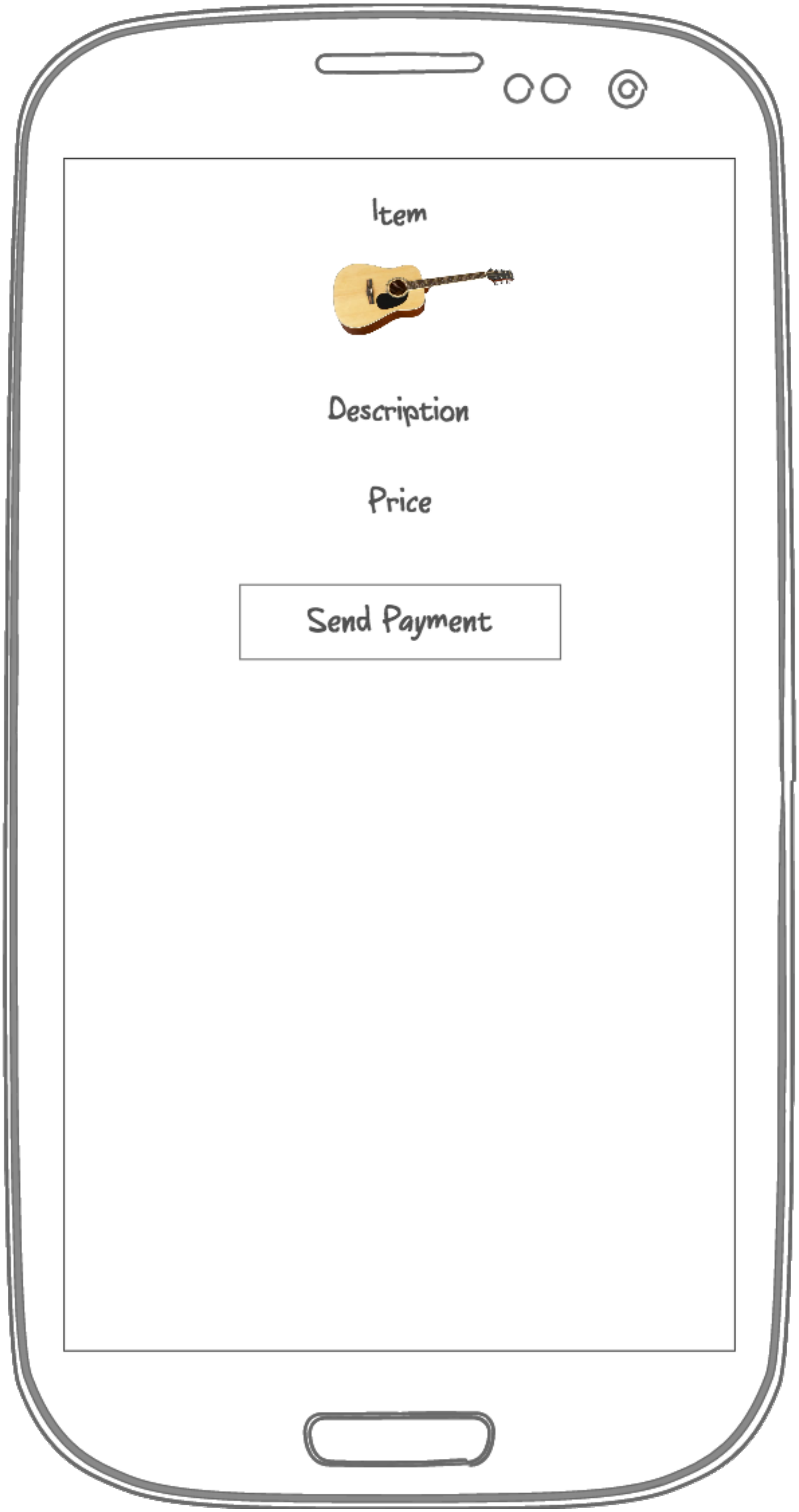
Merchant Mobile Number

Choose your private number

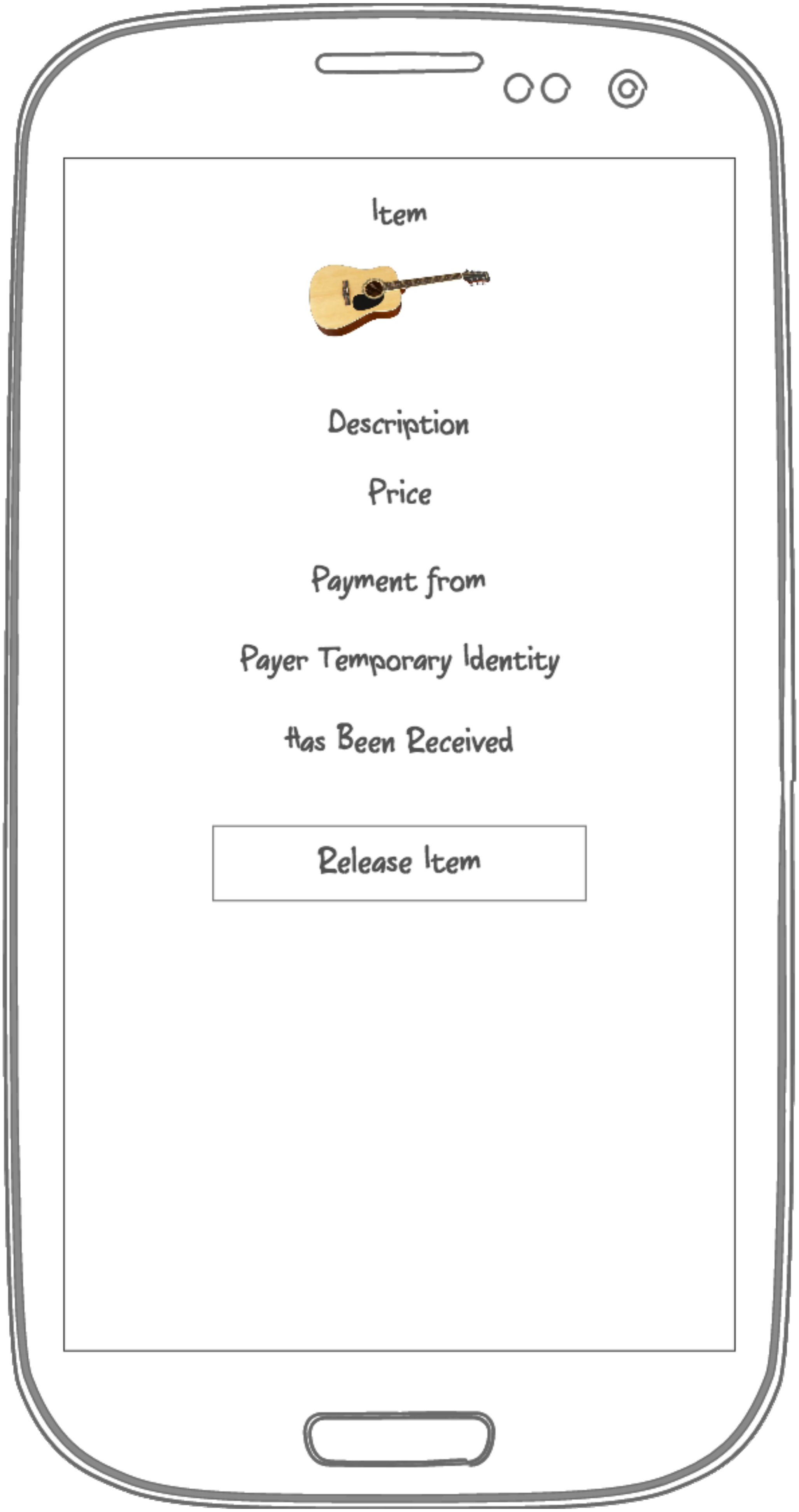
6 - Page 6


Secure Channel With
Payer Temporary Identity
Choose your private number
Establish Secure Channel

7 - Page 7



8 - Page 8



Item

Description

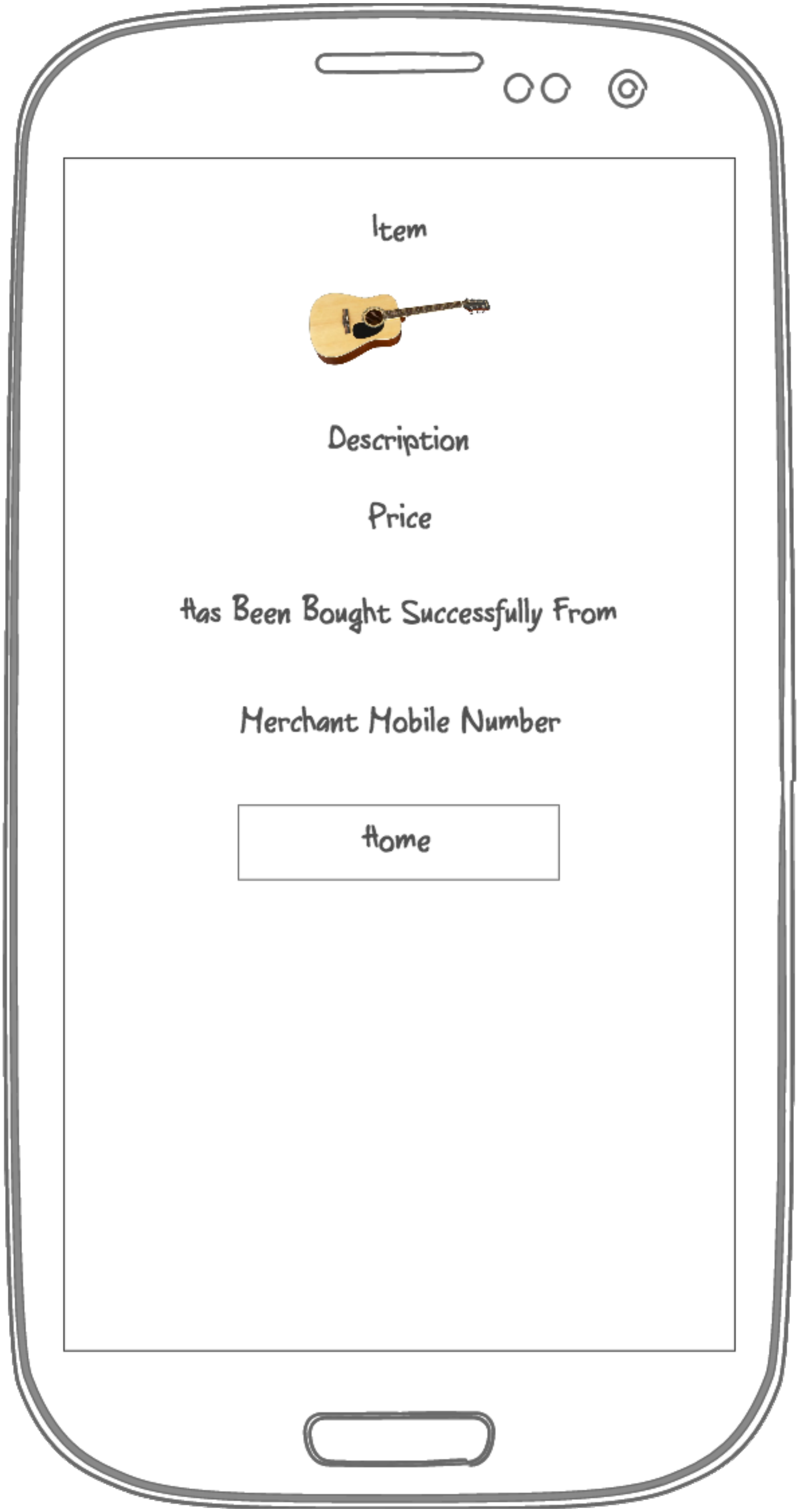
Price

Payment from

Payer Temporary Identity

Has Been Received

9 - Page 9



Item

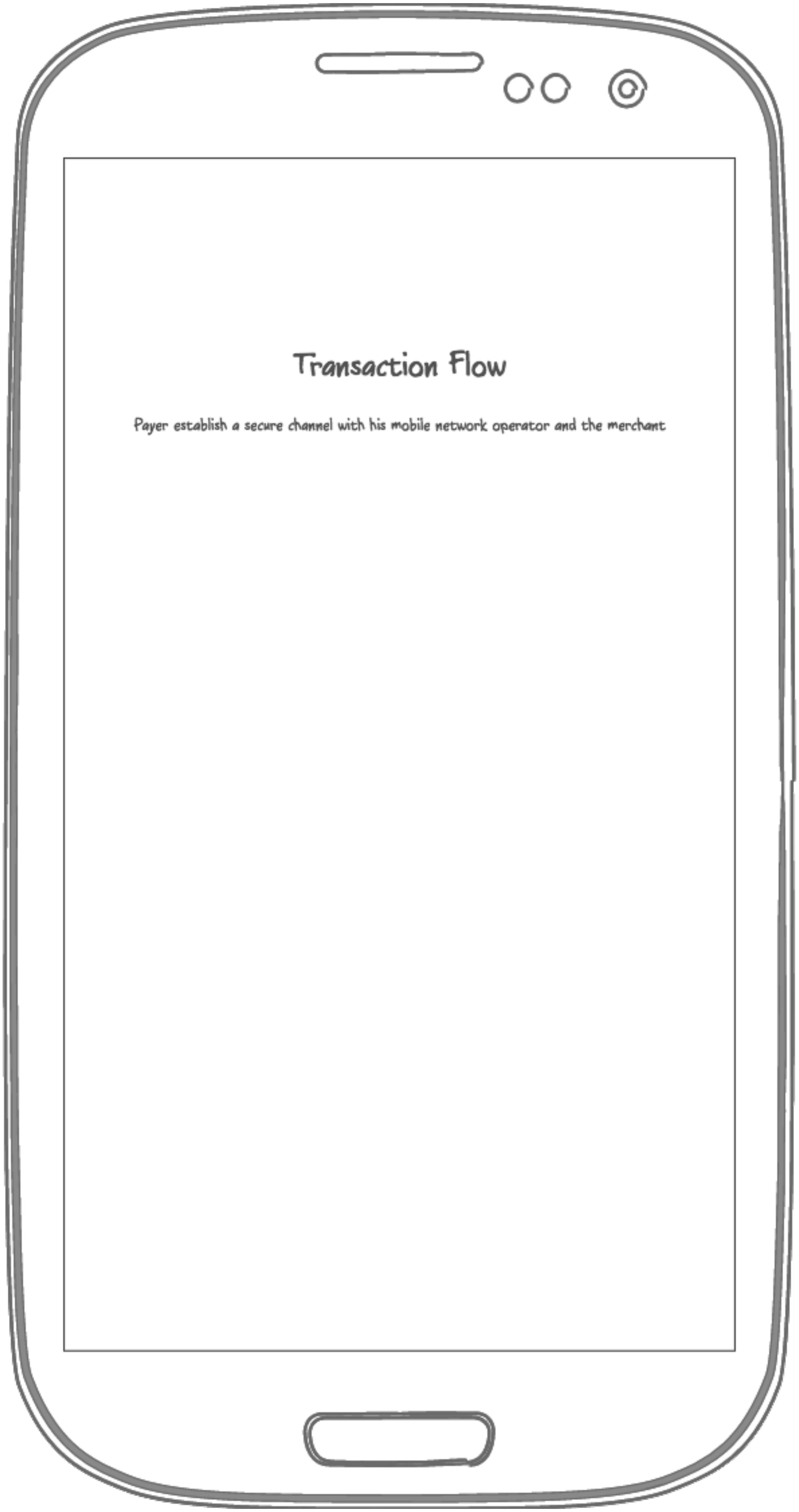
Description

Price

Has Been Bought Successfully From

Merchant Mobile Number

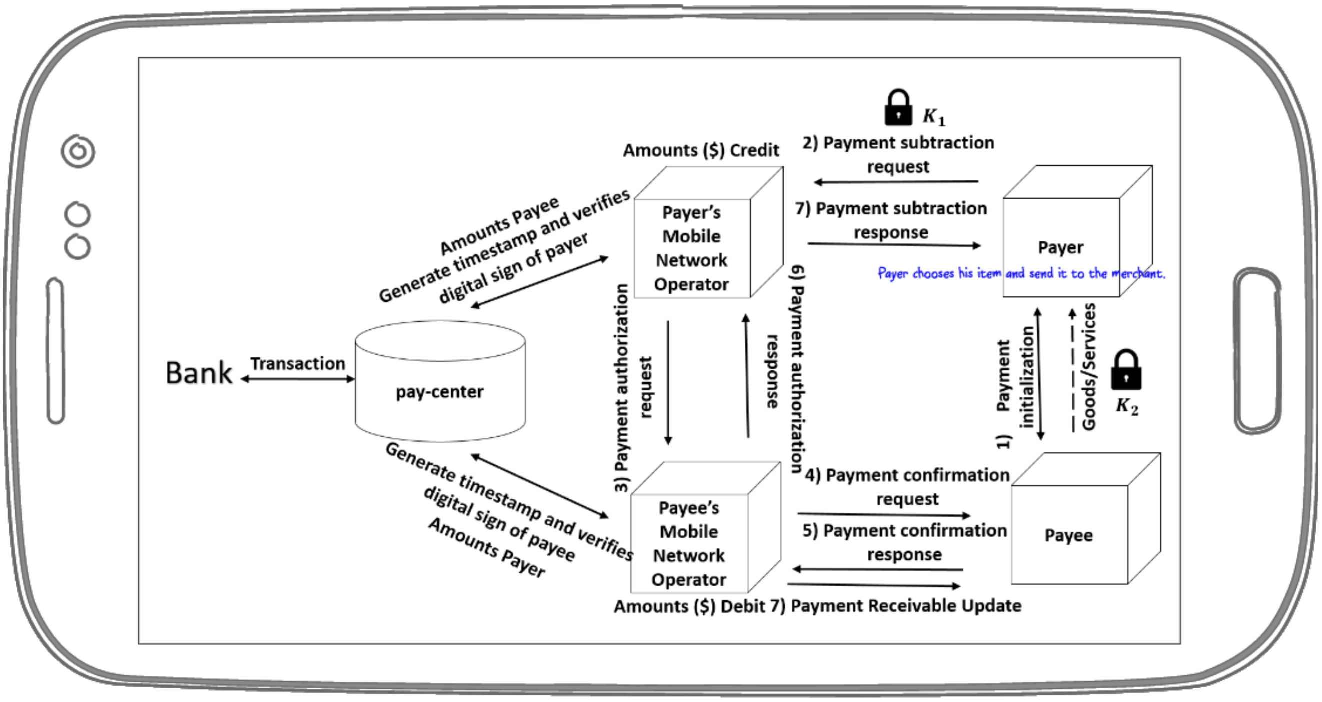
10 - Page 10



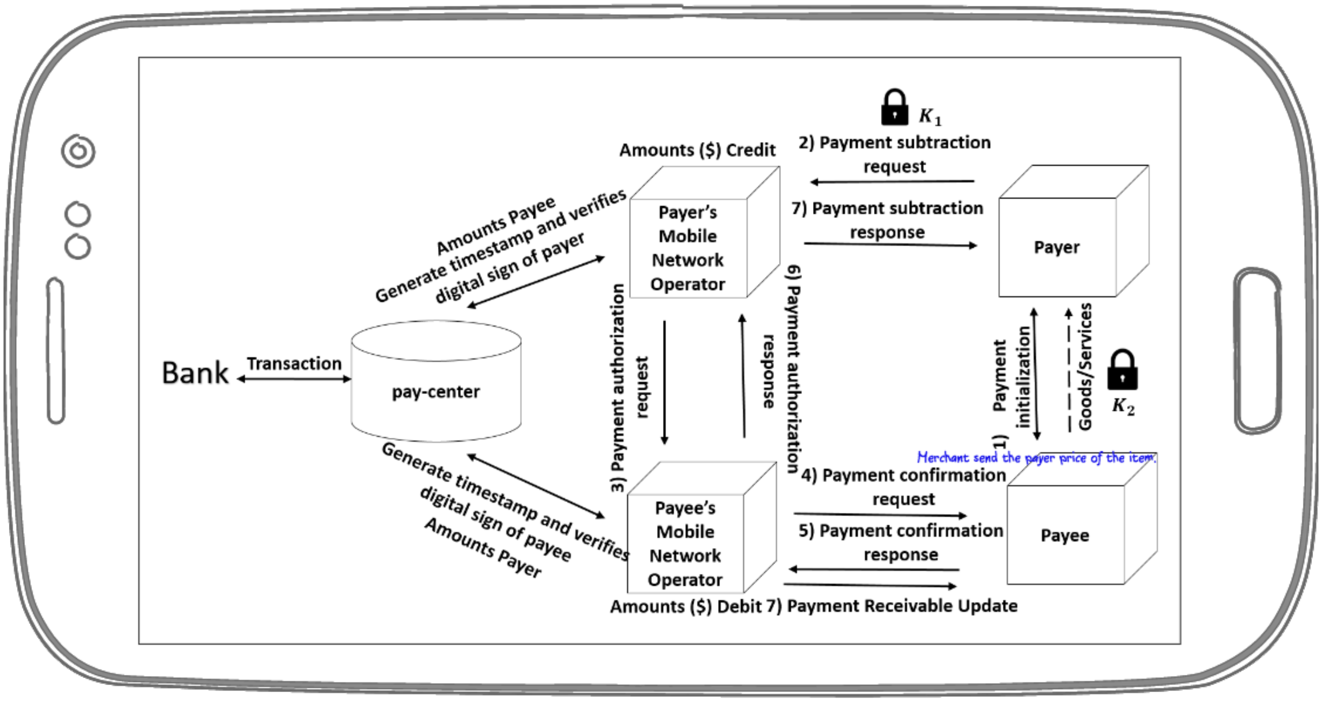
Transaction Flow

Payer establish a secure channel with his mobile network operator and the merchant

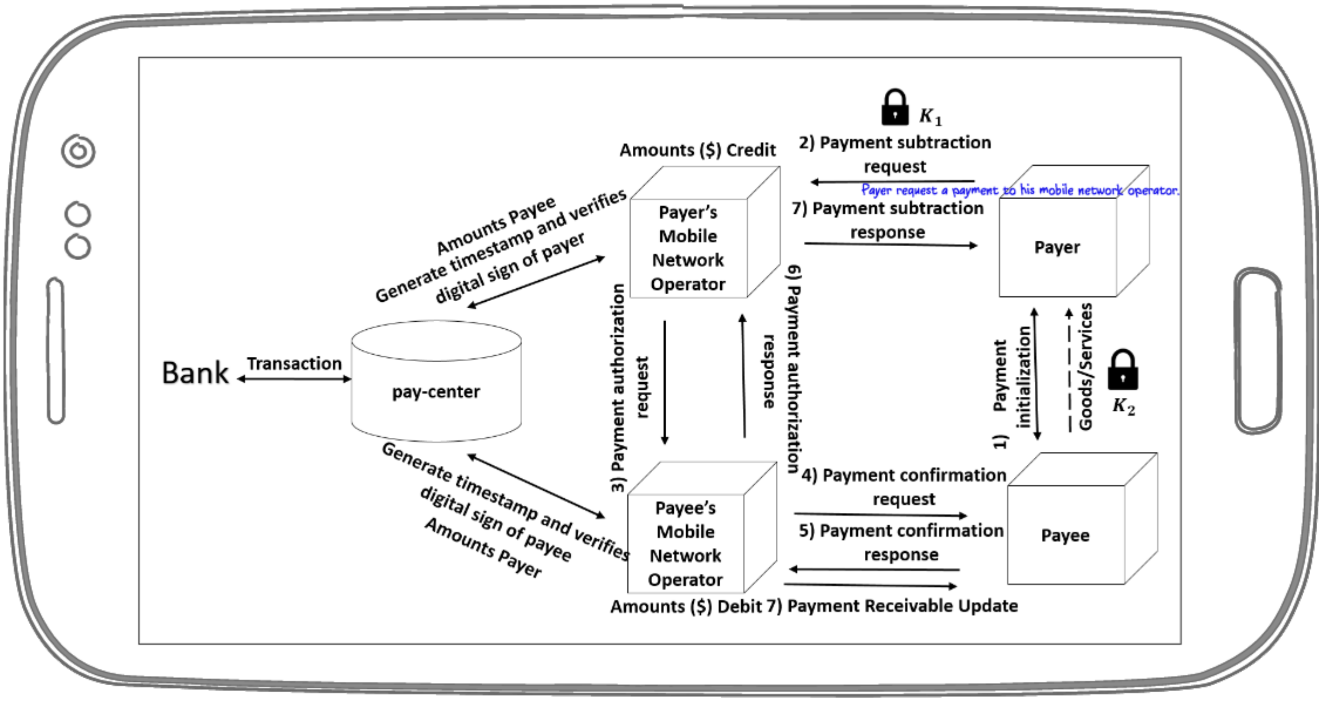
11 - Page 11


K_1
2) Payment subtraction request
7) Payment subtraction response
Amounts ($) Credit
Payer's Mobile Network Operator
Payer
Payer chooses his item and send it to the merchant.
Amounts Payee
Generate timestamp and verifies digital sign of payer
Bank
Transaction
pay-center
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies digital sign of payee
Amounts Payer
Payee's Mobile Network Operator
4) Payment confirmation request
5) Payment confirmation response
Payee
Amounts ($) Debit
7) Payment Receivable Update

12 - Page 12

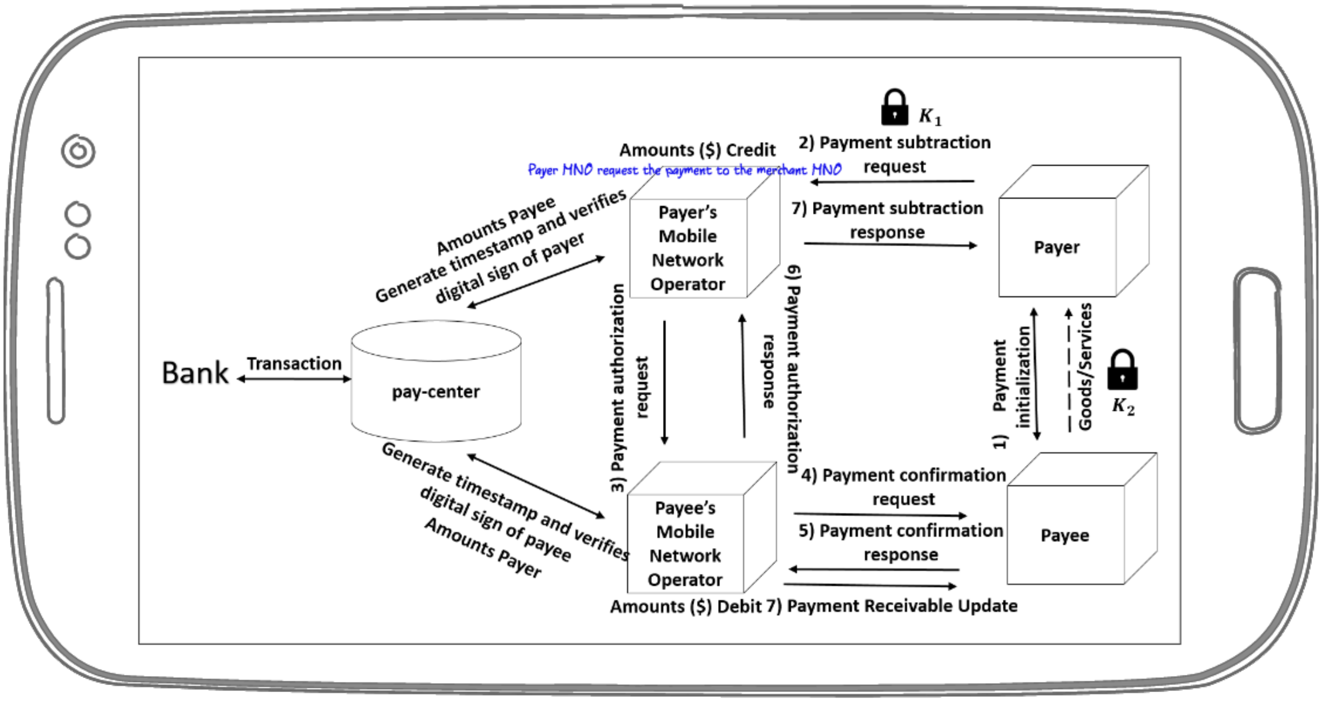

K_1
2) Payment subtraction request
7) Payment subtraction response
Amounts ($) Credit
Payer's Mobile Network Operator
Payer
Amounts Payee
Generate timestamp and verifies digital sign of payer
Bank
Transaction
pay-center
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Merchant send the payer price of the item.
Generate timestamp and verifies digital sign of payee
Amounts Payer
Payee's Mobile Network Operator
4) Payment confirmation request
5) Payment confirmation response
Payee
Amounts ($) Debit
7) Payment Receivable Update

13 - Page 13



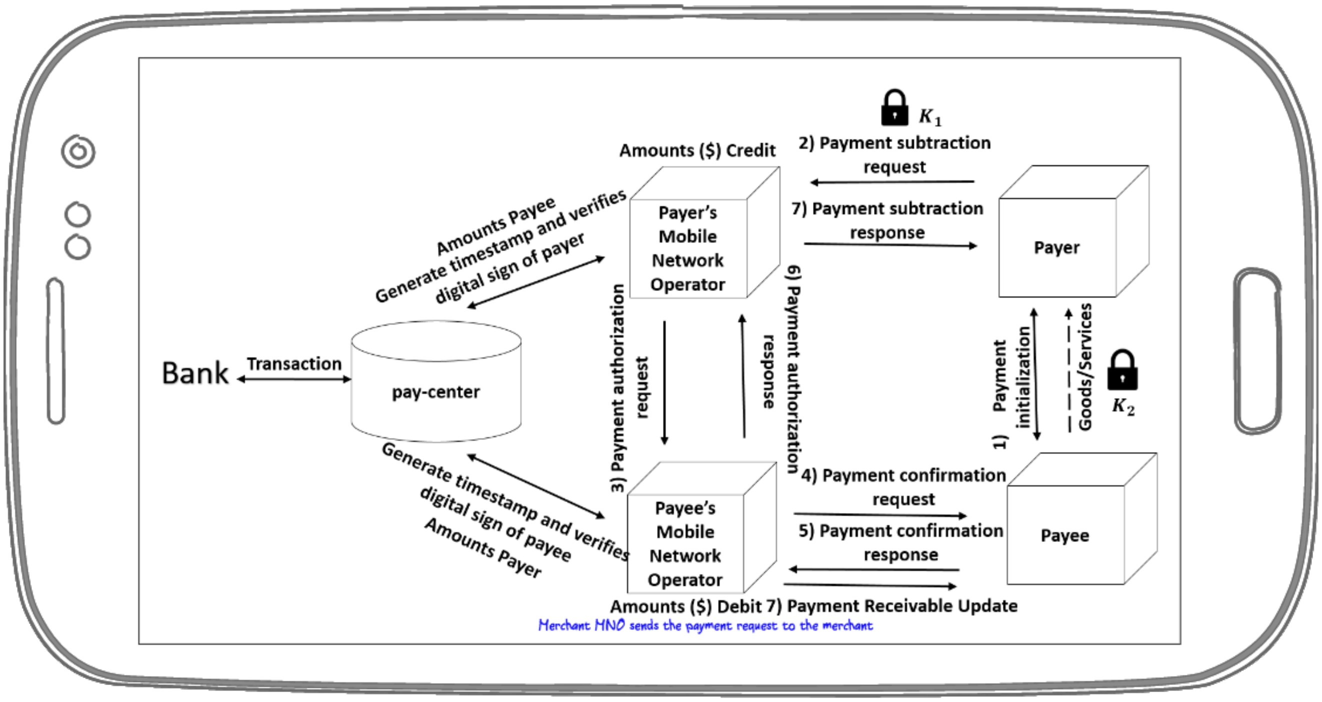
K_1
2) Payment subtraction request
Payer request a payment to his mobile network operator.
7) Payment subtraction response
Amounts ($) Credit
Payer's Mobile Network Operator
Payer
Amounts Payee
Generate timestamp and verifies digital sign of payer
Bank
Transaction
pay-center
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies digital sign of payee
Amounts Payer
Payee's Mobile Network Operator
4) Payment confirmation request
5) Payment confirmation response
Payee
Amounts ($) Debit
7) Payment Receivable Update

14 - Page 14



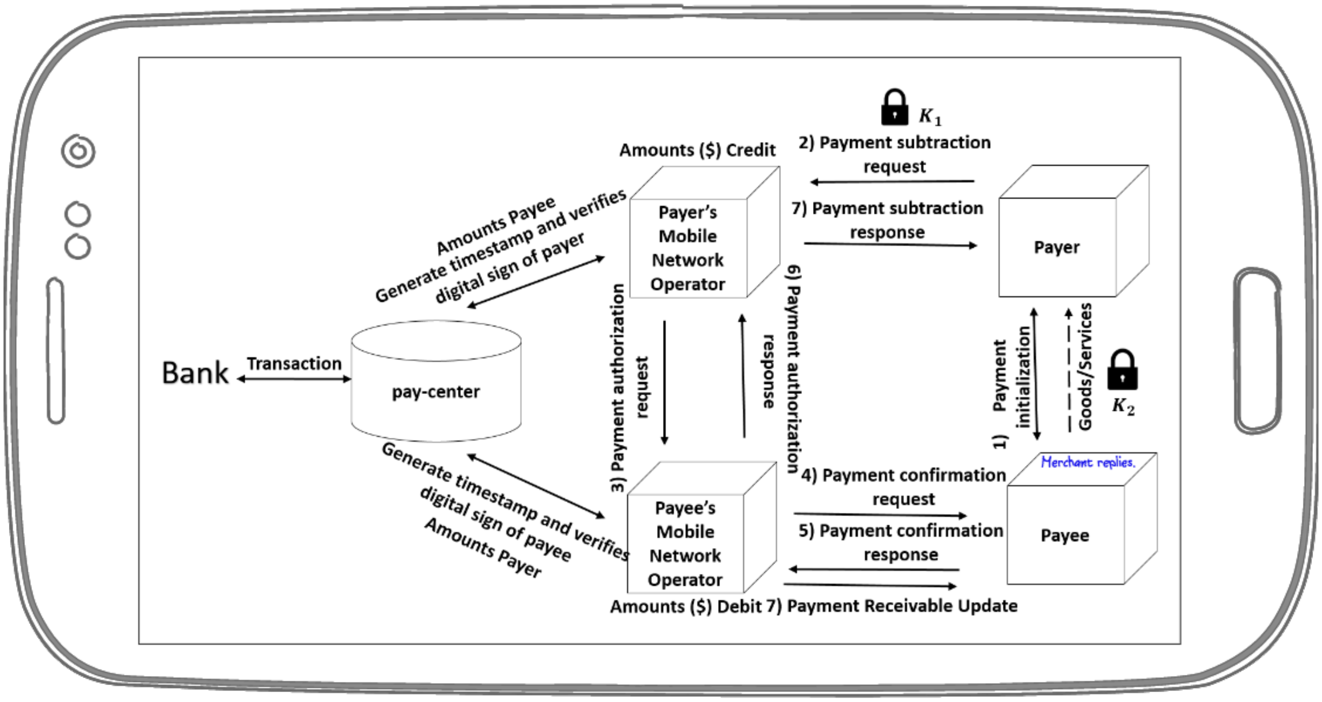
K_1
Amounts ($) Credit
2) Payment subtraction request
Payer MNO request the payment to the merchant MNO
Amounts Payee
Generate timestamp and verifies digital sign of payer
Payer's Mobile Network Operator
7) Payment subtraction response
Payer
Bank
Transaction
pay-center
3) Payment authorization request
response
6) Payment authorization
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies digital sign of payee
Amounts Payer
Payee's Mobile Network Operator
4) Payment confirmation request
5) Payment confirmation response
Payee
Amounts ($) Debit
7) Payment Receivable Update

15 - Page 15

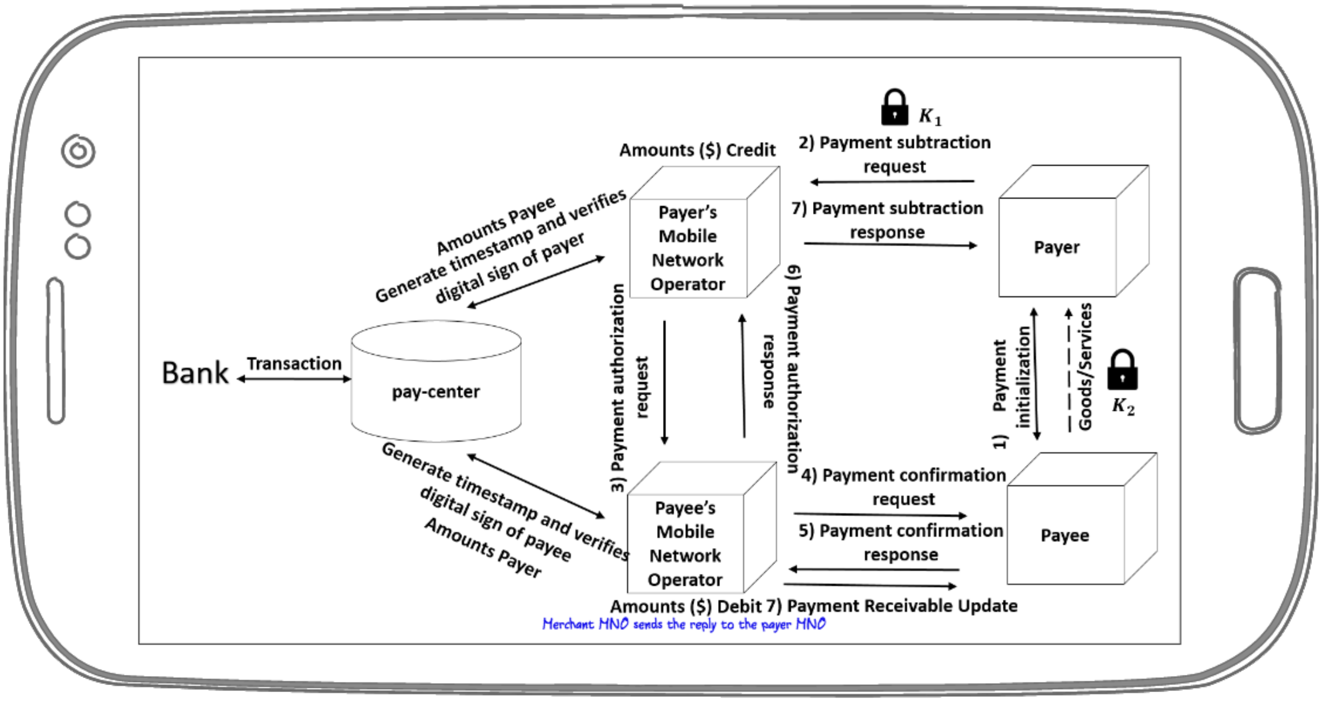


Bank
Transaction
pay-center
Amounts Payee
Generate timestamp and verifies digital sign of payer
Generate timestamp and verifies digital sign of payee
Amounts Payer
Amounts ($) Credit
Payer's Mobile Network Operator
Payee's Mobile Network Operator
Amounts ($) Debit
K_1
2) Payment subtraction request
7) Payment subtraction response
Payer
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Payee
4) Payment confirmation request
5) Payment confirmation response
7) Payment Receivable Update
Merchant MNO sends the payment request to the merchant

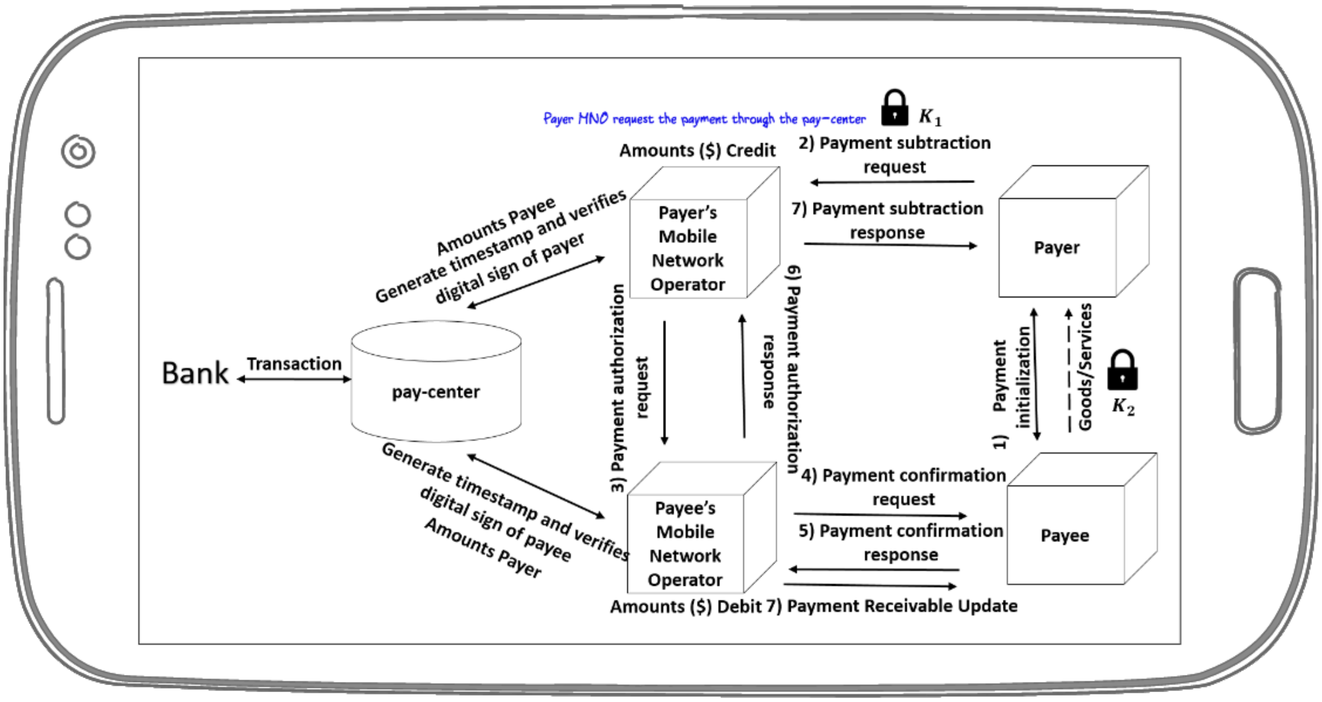
16 - Page 16


Bank
Transaction
pay-center
Amounts Payee
Generate timestamp and verifies
digital sign of payer
Generate timestamp and verifies
digital sign of payee
Amounts Payer
Amounts ($) Credit
Payer's Mobile Network Operator
Payee's Mobile Network Operator
Amounts ($) Debit
3) Payment authorization request
6) Payment authorization response
K_1
2) Payment subtraction request
7) Payment subtraction response
Payer
1) Payment initialization
Goods/Services
K_2
Merchant replies.
Payee
4) Payment confirmation request
5) Payment confirmation response
7) Payment Receivable Update

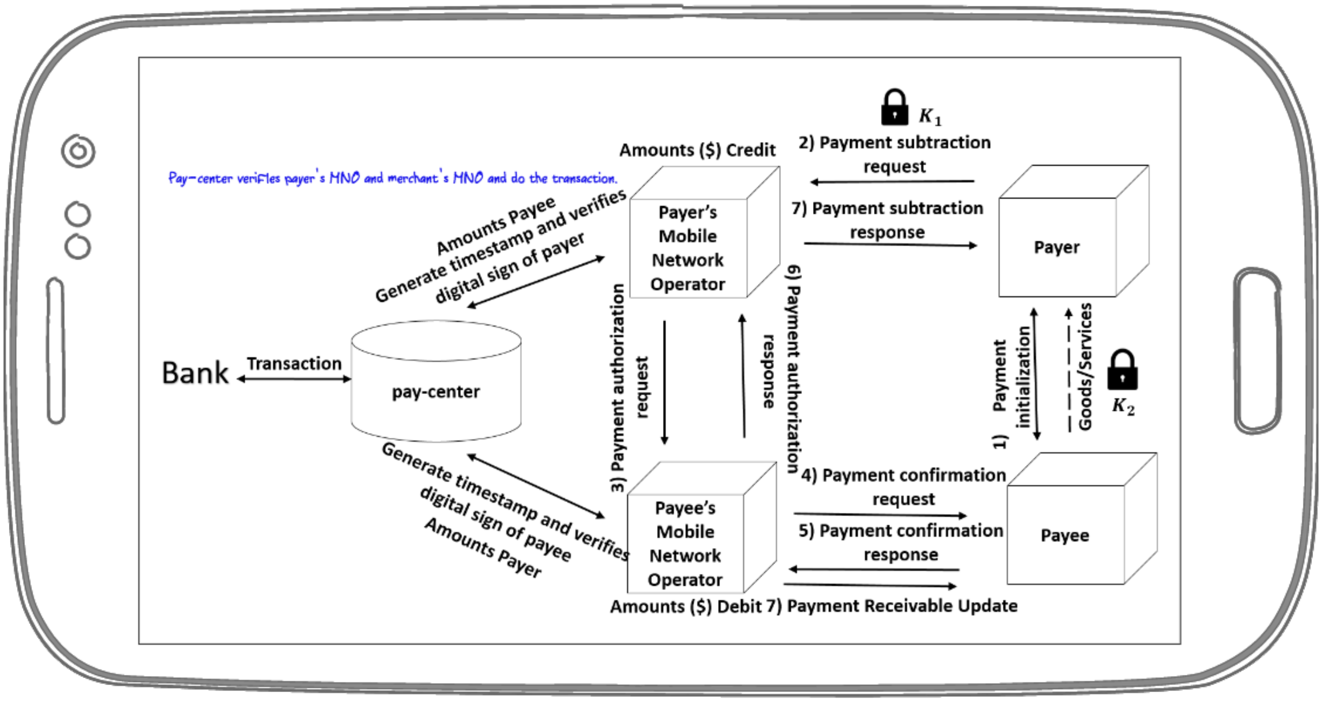
17 - Page 17

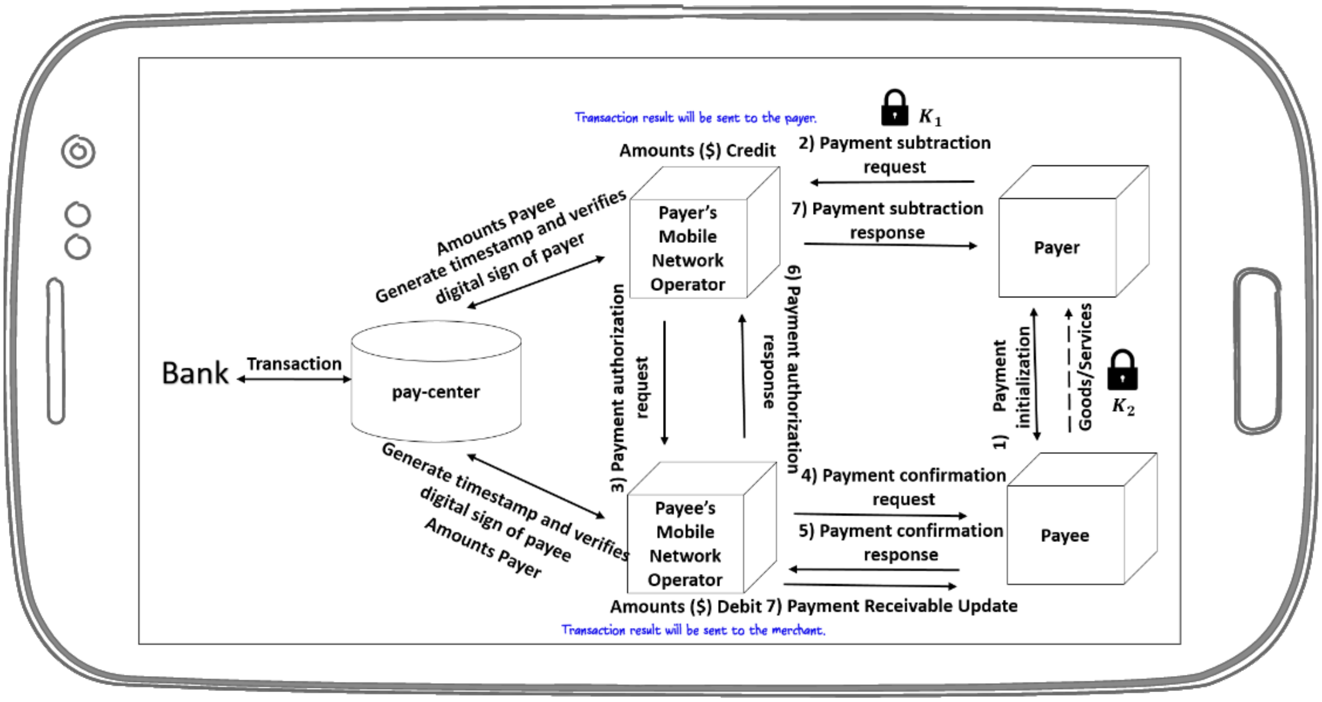

K_1
2) Payment subtraction request
Amounts ($) Credit
Payer's Mobile Network Operator
7) Payment subtraction response
Payer
Amounts Payee
Generate timestamp and verifies digital sign of payer
6) Payment authorization response
3) Payment authorization request
Bank
Transaction
pay-center
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies digital sign of payee
Amounts Payer
4) Payment confirmation request
5) Payment confirmation response
Payee's Mobile Network Operator
Payee
Amounts ($) Debit
7) Payment Receivable Update
Merchant MNO sends the reply to the payer MNO

18 - Page 18

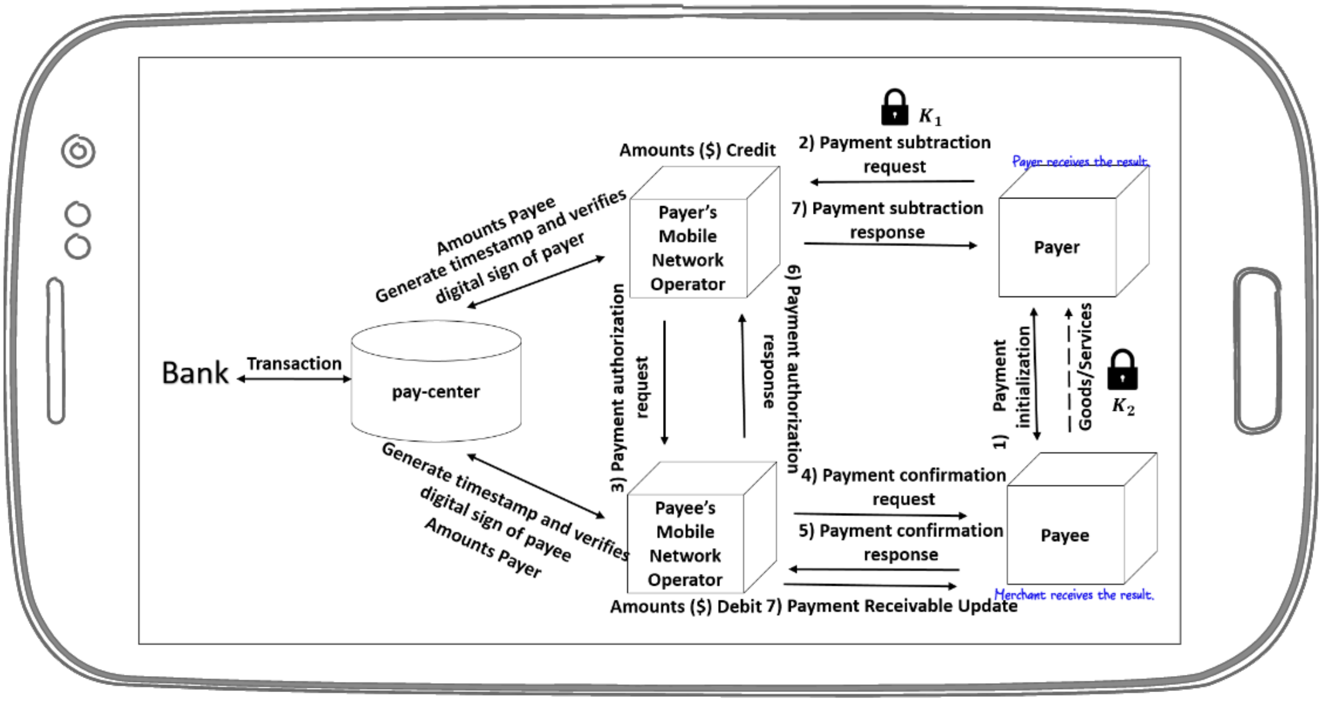

Payer MNO request the payment through the pay-center
K_1
Amounts ($) Credit
2) Payment subtraction request
7) Payment subtraction response
Payer's Mobile Network Operator
Payer
Amounts Payee
Generate timestamp and verifies digital sign of payer
Bank
Transaction
pay-center
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies digital sign of payee
Amounts Payer
Payee's Mobile Network Operator
4) Payment confirmation request
5) Payment confirmation response
Payee
Amounts ($) Debit
7) Payment Receivable Update

19 - Page 19

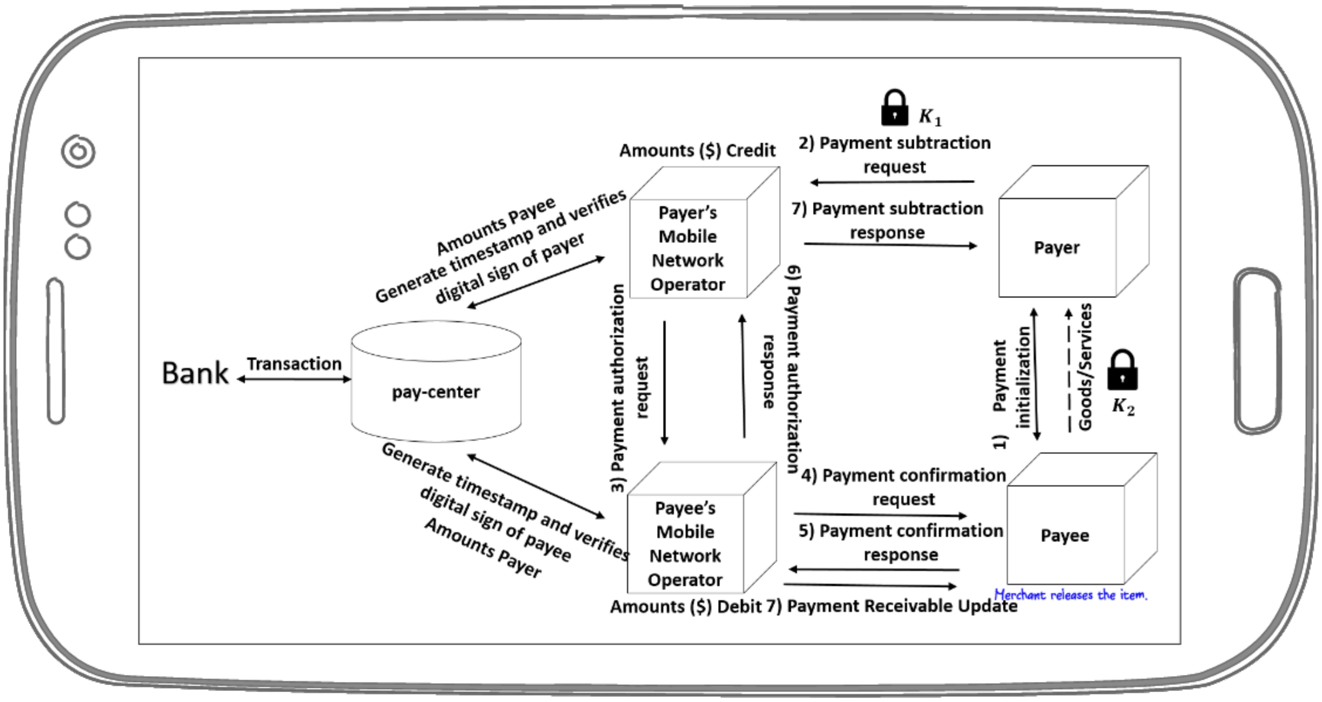

Pay-center verifies payer's MNO and merchant's MNO and do the transaction.
Bank
Transaction
pay-center
Amounts Payee
Generate timestamp and verifies
digital sign of payer
Generate timestamp and verifies
digital sign of payee
Amounts Payer
Amounts ($) Credit
Payer's Mobile Network Operator
Payee's Mobile Network Operator
Amounts ($) Debit
3) Payment authorization request
6) Payment authorization response
K_1
2) Payment subtraction request
7) Payment subtraction response
Payer
Payee
1) Payment initialization
Goods/Services
K_2
4) Payment confirmation request
5) Payment confirmation response
7) Payment Receivable Update


Transaction result will be sent to the payer.
K_1
Amounts ($) Credit
2) Payment subtraction request
7) Payment subtraction response
Payer's Mobile Network Operator
Payer
Amounts Payee
Generate timestamp and verifies digital sign of payer
Bank
Transaction
pay-center
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies digital sign of payee
Amounts Payer
Payee's Mobile Network Operator
4) Payment confirmation request
5) Payment confirmation response
Payee
Amounts ($) Debit
7) Payment Receivable Update
Transaction result will be sent to the merchant.

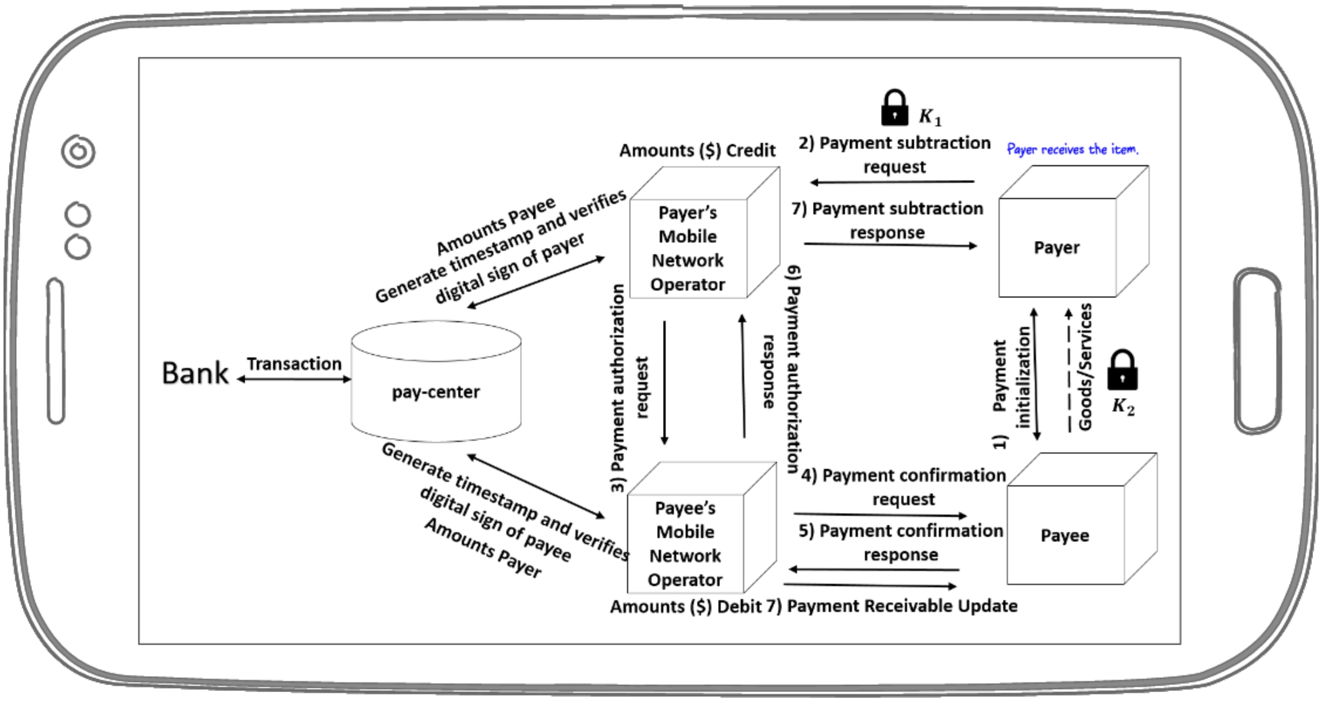
21 - Page 21


Bank
Transaction
pay-center
Amounts Payee
Generate timestamp and verifies
digital sign of payer
Generate timestamp and verifies
digital sign of payee
Amounts Payer
Amounts ($) Credit
Payer's
Mobile
Network
Operator
K_1
2) Payment subtraction
request
7) Payment subtraction
response
Payer receives the result.
Payer
3) Payment authorization
request
6) Payment authorization
response
1) Payment
initialization
Goods/Services
K_2
Payee's
Mobile
Network
Operator
4) Payment confirmation
request
5) Payment confirmation
response
Payee
Merchant receives the result.
Amounts ($) Debit 7) Payment Receivable Update

22 - Page 22


K_1
2) Payment subtraction request
Amounts ($) Credit
7) Payment subtraction response
Payer's Mobile Network Operator
Payer
Amounts Payee
Generate timestamp and verifies digital sign of payer
Bank
Transaction
pay-center
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
Generate timestamp and verifies digital sign of payee
Amounts Payer
Payee's Mobile Network Operator
4) Payment confirmation request
5) Payment confirmation response
Payee
Merchant releases the item.
Amounts ($) Debit
7) Payment Receivable Update

23 - Page 23


Bank
Transaction
pay-center
Amounts Payee
Generate timestamp and verifies
digital sign of payer
Generate timestamp and verifies
digital sign of payee
Amounts Payer
Amounts ($) Credit
Payer's Mobile Network Operator
Payee's Mobile Network Operator
Amounts ($) Debit
K_1
2) Payment subtraction request
7) Payment subtraction response
Payer receives the item.
Payer
3) Payment authorization request
6) Payment authorization response
1) Payment initialization
Goods/Services
K_2
4) Payment confirmation request
5) Payment confirmation response
7) Payment Receivable Update
Payee

Anyone who genuinely and consistently with both hands looks for something will find it.

Jalaluddin Rumi

6 Conclusion

One key agreement protocol and two group key agreement protocols have been suggested in this research. The recommended key agreement protocol is twice as fast as Diffie-Hellman. The first recommended group key agreement protocol is linear group key agreement protocol. This protocol can generate a shared key for a group of n people with a linear cost. In this protocol, one of the group members should play the leader role. The other recommended group key agreement protocol is circular group key agreement protocol. The cost of this protocol is quadratic. The key generation process is circular in this approach.

A new payment protocol is introduced. This protocol is compatible with the mobile platform. The computational cost of generating a shared key between two parties is reduced by using algebra of logarithm instead of the algebra of exponents. Based on the experiments, the recommended protocol is almost twice as fast as the original protocol. Two different random and time-stamp generated numbers are defined to avoid replay attacks. An extra security layer is suggested to prevent card not present fraud. Cloud messaging is recommended to implement this extra security layer in order to solve original 3DS issues.

The computation and communication costs of key agreement protocol can be reduced more. The complexity of defining a shared key for group of n users can be reduced. The performance of the 3DS behavioral model could be improved to detect card not fraud with less issues. So, the extra security layer will be used for transactions that are more likely a threat. The protocol should be extended to support a wider range of devices especially popular devices in Internet of things. Other cloud technologies could be utilized in the payment protocol.

Appendices

A

Android App

In this chapter, the implemented Android app will be reviewed. You can see the architecture of this app in figure 15. This project is composed of three components. The first component is the database named pace_db. This database is to save users and transactions information. There are two important tables in this database. These tables are users and transaction. users table keeps records of the users. transaction table keeps the record of the transactions in the project.

The next element is the PHP APIs. These APIs are the means that enable the users to interact with the database. These APIs provide restful APIs for the Android app. Android app sends/receives JSON objects to/from the APIs. These APIs are developed wit PHP language. There are two important sets of APIs in this section. The first set is Models. These APIs are the ones that read/write information from/to the database. The next important set is controllers. These are the APIs that handles the flow of the Android app and manages the push notifications to/from the mobile devices.

The last component of this project is the Android app. This app is developed in Android. The codes are written by using Android studio (standard Android development IDE). There are different sets of classes in this app. The first set includes the models like user or transaction. The second set of classes is related to the key agreement protocols. There are two important protocols in this set. These protocols are Pace protocol and Diffie-Hellman protocol. There are also some utility classes in the app. There are some classes to generate the shared key for the parties. Also, there are some classes to handle the push notifications. Finally, some of the classes are related to the payment section.

In summary, this project is composed of and Android app, a set of PHP APIs, and a MySQL database. This app enables the users to choose Diffie-Hellman or Pace protocol to generate the shared key. After generating the shared key, the app will show the user total calculation time on the mobile device. As you see in figure 15, the app communicates with the PHP APIs by using the restful APIs. JSON objects will be transferred between the app and the APIs. PHP APIs are the ones that interact with the MySQL database.

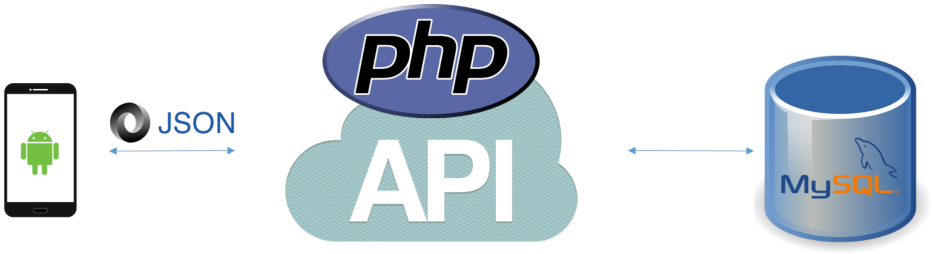


Figure 15: Mobile commerce share of e-commerce

A.1 Database

For this project, MySQL database is used. This project has two important tables. First one is users. users table includes the information of the users (parties). The second table is transaction table. This table keeps the information of the transaction. First, you need to create pace_db in your local MySQL server. Then, you need to build the database by running the code below on your local server. First, you need to create an empty database named pace_db. Then, you need to create the tables by using the script in the link below.

https://bitbucket.org/pacepaymentprotocol/pace-payment-api/src/029e96f724d57b0c866c33afbc63e625cb19e348/DB.txt?at=master&fileviewer=file-view-default

A.2 PHP APIs

Then, you need to host the PHP APIs on a server. These APIs will send/receive the requests/responses. These APIs are using JSON objects. these APIs are handling the communications between the Android app and the MySQL database. Most of the useful APIs that are available, are inside the app folder. Models folder is inside this folder. Models folder includes all the APIs related to the models in the app. The other important folder is Http folder. This folder contains the Controllers folder. All of our controllers are inside this folder. Also, APIs that handle the push notifications are inside this folder. To see the source of these APIs, visit the link below.

https://bitbucket.org/pacepaymentprotocol/pace-payment-api/src

A.3 The Android App

This Android app is for the users (customers and merchants). This app has Diffie-Hellman key agreement protocol and The recommended key agreement protocol. This app will inform the user about the total time of the calculations for establishing a secure channel. This app is working based on Android platform. This app is utilizing the push notification (cloud messaging) for the communications between the parties.

There are different classes in this project. There are some folders inside the project. These folders are models, protocol, and utils. models folder includes all the java classes related to models of the entities like a user. The protocol folder includes all the cryptography protocols includes Diffie-Hellman and Pace protocol. utils folder has all the utility classes like Config and Preferences.

There are also some classes outside these folders. Home activity is the home of our application. Login activity is the activity that handles the login and registration. The main activity is the starting activity. Classes with name Firebase are the classes that handle push notifications. Classes with name Secure Channel are the classes that establish the secure channel between the parties. The classes with name Payment Activity are the ones that handle the payment transaction between the payer and the payee. To see the source of this Android app, visit the link below.

https://bitbucket.org/pacepaymentprotocol/pace-payment/src

B

Java Project

In this chapter, MPCP java project will be reviewed. You can clone this java project from the repository in below.

https://github.com/m-vahidalizadeh/mpcp_project.git

This project has been develped by java language in Intellij IDEA. This project has three classes. These classes are KeyAgreementProtocolTest, GroupKeyAgreementProtocolTest, and PhasesOfMPCP. KeyAgreementProtocolTest is developed to generated shared session key by using Diffie-Hellman and Pace protocol. You need to initialize the parameters and run the main method in the class. You can find the code of this class in the link below.

https://github.com/m-vahidalizadeh/mpcp_project/blob/master/src/KeyAgreementProtocolTest.java

The next class is GroupKeyAgreementProtocolTest. As the name shows, this class is written to simulate the generation of a shared key for a group of parties. This class compares the performance of generating a shared session key by Diffie-Hellman with Pace protocol. You can see the code of this class in the link below.

https://github.com/m-vahidalizadeh/mpcp_project/blob/master/src/GroupKeyAgreementProtocolTest.java

The last class in this project is PhasesOfMPCP. This class is developed to generate the seven phases of MPCP based on chapter 4 of this research. You need to enter payerś selected number and hit enter. Then, you need to enter payeeś selected number and hit enter. Then, the first phase will be generated for you via the logs. Each time that you hit enter, another pahse will be printed in the logs. You can see the code of this class in the link below.

https://github.com/m-vahidalizadeh/mpcp_project/blob/master/src/PhasesOfMPCP.java

References

[1] Total volume of chinese users' mobile payments exceeds japan's gdp, say japanese media reports. URL http://www.yicaiglobal.com.

[2] http://www.purplemath.com/modules/logrules.htm. URL http://www.purplemath.com/modules/logrules.htm.

[3] Mobile marketing statistics compilation 1 mar 2017. URL http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/?new=1.

[4] Set secure electronic transaction specification. MasterCard, Visa, 1997.

[5] Forrester: Mobile and tablet commerce forecast 2015 to 2020., 2016. URL https://www.forrester.com.

[6] Phu Dung Le Boon Tiong Sunny Toh, Supakoq Kungpisdan. Ksl protocol: Design and implementation. *Conference on Cybernetics and Intelligent Systems*, 2004.

[7] Chian Techapanupreeda Supakorn Kungpisdan Chalee Thammarat, Roongroj Chokngamwong. A secure lightweight protocol for nfc communications with mutual authentication based on limited-use of session keys. *ICOIN*, 2015.

[8] Xiao Pan Chunhui Piao, Xiaoyan Li. Research on the user privacy protection method in mobile commerce. *IEEE 11th International Conference on e-Business Engineering*, 2014.

[9] Hicham Madroumi Houssam El Ismaili, Hanane Houmani. A secure electronic transaction payment protocol design and implementation.

(IJACSA) International Journal of Advanced Computer Science and Applications, 2014.

[10] Jose Sierra Caimara Jesu's Tellez Isaac. Anonymous payment in a client centric model for digital ecosystems. *Inaugural IEEE International Conference on Digital Ecosystems and Technologies*, 2007.

[11] Mohsen Guizani Jie Li, Huang Lu. Acpn: A novel authentication framework with conditional privacy-preservation and non-repudiation for vanets. *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, 2015.

[12] E. Konstantinou, E. Klaoudatou, and P. Kamparmpakis. Performance evaluation of id-based group key agreement protocols. In *2011 Sixth International Conference on Availability, Reliability and Security*, pages 377–384, Aug 2011. doi: 10.1109/ARES.2011.63.

[13] Supakorn Kungpisdan. Accountability in centralized payment environments. *ISCIT*, 2009.

[14] Mohamad Ibrahim Ladan. E-commerce security issues. *International Conference on Future Internet of Things and Cloud*, 2014.

[15] M. Geranmaye M. Vahidalizadehdizaj. New mobile payment protocol: Mobile pay center protocol 5 (mpcp5) by using new key agreement protocol: Vg1. In *2011 3rd International Conference on Computer Modeling and Simulation (ICCMS 2011)*, pages 246–252, 2011.

[16] M. Geranmaye M. Vahidalizadehdizaj. New mobile payment protocol: Mobile pay center protocol 6 (mpcp6) by using new key agreement protocol: Vgc3. In *2011 3rd International Conference on Computer Modeling and Simulation (ICCMS 2011)*, pages 253–259, 2011.

[17] Ralf Hauser Amir Herzberg Hugo Krawczyk Michael Steiner Gene Tsudik Els Van Herreweghen Michael Waidner Mihir Bellare, Juan A. Garay. Design, implementation and deployment of the ikp secure electronic payment system. *IEEE Journal of Selected Areas in Communications*, 2000.

[18] Garayy Ralf Hauserz Amir Herzbergy Hugo Krawczyky Michael Steinerz Gene Tsudikz Michael Waidnerz Mihir Bellarey, Juan A. ikp- a family of secure electronic payment protocols. *IBM Zurich Research Laboratory*, 1995.

[19] Supakorn Kungpisdan Pemika Limpit taya, Maykin Warasart. Design and analysis of a secure agent-based mobile bill payment protocol for bulk transactions. *Ninth International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2012.

[20] Mostafa Hashem Sherif. *Protocols for Secure Electronic Commerce, Second Edition (Advanced & Emerging Communications Technologies)*. CRC PRESS, 2016.

[21] Mark Sherman. An introduction to mobile payments: Market drivers, applications, and inhibitors. *MOBILESoft*, 2014.

[22] Bala Srinivasan Supakorn Kungpisdan, Phu Dung Le. A limited-used key generation scheme for internet transactions. *Springer Verlag Berlin Heidelberg*, 2004.

[23] Phu Dung Le Supakorn Kungpisdan, Bala Srinivasan. Lightweight mobile credit-card payment protocol. *Springer-Verlag Berlin Heidelberg*, 2003.

[24] Phu Dung Le Supakorn Kungpisdan, Bala Srinivasan. A practical framework for mobile set payment. *IADIS International Conference e-Society*, 2003.

[25] Phu Dung Le Supakorn Kungpisdan, Bala Srinivasan. A secure account-based mobile payment protocol. *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04)*, 2004.

[26] Phu Dung Le Supakorn Kungpisdan, Bala Srinivasan. A practical framework for mobile set payment. *IADIS International Conference e-Society*, 2013.

[27] Tanapat Thai-udom Supakorn Kungpisdan. Securing micropayment transactions over session initiation protocol. *ISCIT*, 2009.

[28] Supakorn Kungpisdan Surakarn Duangphasuk, Maykin Warasare. Design and accountability analysis of a secure sms-based mobile payment protocol. *The 8th Electrical Engineering/ Electronics, Computer, Telecommunications and Information Technology (ECTI)*, 2011.

[29] Paul Syverson. On key distribution protocols for repeated authentication. *ACM SIGOPS Operating Systems*, 1993.

[30] Jonathan Likoh Rozaini Roslan Tan Soo Fun, Leau Yu Beng. A lightweight and private mobile payment protocol by using mobile network operator. *Proceedings of the International Conference on Computer and Communication Engineering*, 2008.

[31] Mohd Norhisham Razali Tan Soo Fun, Leau Yu Beng. Review of mobile macro-payment schemes. *Journal of Advances in Computer Networks*, 2013.

[32] Rozaini Roslan Habeeb Saleh Habeeb Tan Soo Fun, Leau Yu Beng. Privacy in new mobile payment protocol. *World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 2008.

[33] M. Vahidalizadehdizaj. New mobile payment protocol: Mobile pay center protocol 4 (mpcp4) by using new key agreement protocol: Vac2. In *2011 3rd International Conference on Electronics Computer Technology*, volume 2, pages 67–73, April 2011.

[34] M. Vahidalizadehdizaj and Avery Leider. Golden linear group key agreement protocol. In *2017 14th International Conference on Information Technology : New Generations (ITNG)*, April 2017.

[35] M. Vahidalizadehdizaj and Avery Leider. Mobile pay center protocol 3d by using cloud messaging. In *2017 14th International Conference on Information Technology : New Generations (ITNG)*, April 2017.

[36] M. Vahidalizadehdizaj and L. Tao. A new mobile payment protocol (gmpcp) by using a new key agreement protocol (gc). In *2015 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 169–172, May 2015.

[37] M. Vahidalizadehdizaj, Reza Askari Moghaddam, and Samad Momenebellah. New mobile payment protocol: Mobile pay center protocol (mpcp). In *Proceedings of the 9th WSEAS International Conference on Advances in e-Activities, Information Security and Privacy*, ISPACT'10, pages 41–46, Stevens Point, Wisconsin, USA, 2010. World Scientific and Engineering Academy and Society (WSEAS). ISBN 978-960-474-258-5. URL http://dl.acm.org/citation.cfm?id=1948838.1948843.

[38] M. Vahidalizadehdizaj, R. A. Moghaddam, and S. Momenebellah. New mobile payment protocol: Mobile pay center protocol (mpcp). In *2011 3rd*

International Conference on Electronics Computer Technology, volume 2, pages 74–78, April 2011.

[39] M. Vahidalizadehdizaj, R. A. Moghaddam, and S. Momenebellah. New mobile payment protocol: Mobile pay center protocol 2 (mpcp2) by using new key agreement protocol: Vam. In *Proceedings of 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 12–18, Aug 2011.

[40] M. Vahidalizadehdizaj, Lixin Tao, and J. Jadav. A new mobile payment protocol (gmpcp) by using a new group key agreement protocol (vtgka). In *2015 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–7, July 2015.