Final Report

Evertson Croes (4241754)
Danny Hendrix (4241746)
Carlo Meijer (4335880)
Roland Verbruggen (3038084)
Akis Dimakogiannis (4384202)

December 5, 2013

This document serves as a high level design of an Electronic Purse card (E-purse).

1 Stakeholders and assets

1.1 Stakeholders

The Stakeholders can be divided in three groups. These are the Cardholders, Card Issuers and the shop keepers (Point of Sale).

1.1.1 Cardholders

A cardholder is the original owner of a card and the person using the e-purse to purchase goods and services from stores and other businesses. The Cardholder is a stakeholder, because he or she will be using the card to make payments. In this document cardholders might be referred to as customers.

1.1.2 Card Issuer

The card issuer is the company that provides the e-purse. By default there is only one card issuer involved in the e-purse. The card issuer will initialize the cards and send them to their customers (cardholders). Tasks of the card issuer include initializing and supplying the card and blocking cards after an attack or theft/loss of a card.

1.1.3 POSs

A POS (Point of Sale) is a business that allows their customers to pay for goods and/or services using the e-purse. POSs are stakeholders, because they will have to accept the E-purse Card as a valid way of payment.

1.1.4 Banks

The bank is not really a stakeholder in our model. This is because we are a trusted third party. We assure that all transactions performed are legitimate. The pin code used to with our system to top up the E-Purse does not have to be the same as the one you use with your bank.

1.2 Assets

There are four assets involved in the e-purse. These are the E-purse cd, the Top-up terminal, the POS (Point of Sale) terminal and the Personalization terminal. Additional, non physical assets involved are money on the bank and virtual money stored on the card.

1.2.1 E-purse card

The e-purse card is being used by cardholders to pay for goods and services. The cardholder will carry his card and is therefore responsible for protecting it against theft or losing it. When this happens, the cardholder should contact the card supplier to block the card. Information that is stored on the card is the card ID, the pincode, current balance, the public key of the central server and the card-specific private key signed by the central server.

1.2.2 Top-up Terminal

The top-up terminal is a physical machine that allows a cardholder to recharge his or her E-purse or to check the current balance on the card. The top-up terminal is placed in public locations and may or may not be observed by security cameras. The top-up terminal is always online.

1.2.3 POS Terminal

The point of sale terminal is a physical device that is used in businesses to allow their customers to pay for their goods or services. Th POS terminal should be protected against threats from both the business and the cardholders point of action. We do not assume the POS terminal to be always online.

1.2.4 Personalization Terminal

The personalization terminal is a setup device that is being used only at the card suppliers office/business. This terminal will initialize the card with the basic necessary information. The device is only being used at the card supplier.

2 Use Cases

2.1 Initialize card

2.1.1 Main success scenario

The card issuer wants to initialise a card for a client. The card issuer inserts the card into the personalisation terminal. The system asks for a password. The card issuer inserts the password. The system shows the personalisation options. The card issuer inserts and confirms the personalisation options. The system initialises the card with the given options.

2.1.2 Alternate scenarios

If the card is not recognized, the system returns an error notifying the card issuer.If the password is incorrect, the system returns an error notifying the card issuer and asks to re-enter the password. There is a limit of 3 tries.

2.2 Top-up a card's balance

2.2.1 Main success scenario

The card holder wants to increase his balance on his card. The card holder inserts the card into the top-up terminal. The terminal asks for a PIN code. The card holder inserts a PIN code. The terminal shows the balance of the card. Next, the terminal asks for the amount to be added to the balance. The card holder inserts and confirms the amount to be added to the balance. The terminal adds the balance with the given amount and gives an indication whether the transaction was successful.

2.2.2 Alternate scenarios

If the card is not recognized, the terminal returns an error notifying the card holder. If the PIN code is incorrect, the terminal returns an error that the PIN code is incorrect and asks to re-enter the PIN code. There is a limit of 3 tries. If the sum of the current balance and the top-up amount is more than the maximum allowed balance, the terminal will give an error and ask to re-enter the top-up amount. If the transaction is not successful, the terminal returns an error.

2.3 Pay with card

2.3.1 Main success scenario

The card holder wants to pay using the card. The cashier inserts a password into the terminal to unlock it. The cashier inserts the amount to be paid into the terminal. The terminal shows the amount to be paid and asks to insert the card. The card user inserts the card and confirms the payment. The terminal makes the transaction.

2.3.2 Alternate scenarios

If the password inserted into the terminal is incorrect, the terminal returns an error and asks to re-enter the password. After 3 failed attempts of entering the password into the terminal the terminal is blocked. If the card is not recognized, the terminal returns an error. The card user should re-insert the card or check if the card is valid/operational. If the user is not in agreement with the account to be paid, the user selects cancel payment on the terminal.If the transaction fails, the terminal returns an error. Restart the use case.

2.4 Block card

There are two reasons why the card issuer would want to block the card

(i) The card user has called and reported the card as stolen or missing.

(ii) There is evidence of tampering with the card by the card holder.

In both cases, the card issuer would need to update the blacklist of cardID's.

2.4.1 Main success scenario

The card issuer wants to block a card. The card issuer enters a password into the system. The system shows a message that the password is correct. The card issuer selects to block a card. The system asks for the cardID to be blocked. The card issuer enters a cardID. The system adds the cardID to the blacklist.

2.4.2 Alternate scenarios

If the password is incorrect, the system returns an error and asks the card issuer to re-enter the password. There is a limit of 3 tries. If the cardID is invalid, not found or already blocked, the system returns an error with the appropriate message. Extra: Once the card is blocked, it is rendered useless. If the card holder wishes to continue use of the e-purse, the card holder needs to ask for a new card. (the card is blacklisted and all terminals have a blacklist)

3 Security requirements

3.1 Confidentiality

The card's PIN is confidential. It is never sent in cleartext over the wire. Also the card posesses a private key, which is confidential, even to the bank.

3.2 Integrity

The current balance on the E-Purse should be integer. Only the top-up terminal should be able to increase the balance and only the POS terminal should be able to lower the balance. No other systems or users should be able to change the balance. The balance is a single value on the card.

3.3 Authentication

3.3.1 When topping up the balance at top-up terminal

The User of the card has to identify himself to the card. This is required since otherwise someone may finds or steal the card and he could top-up the card. Once the card has authenticated with the user with a PIN code, the card has to authenticate the user with the bank account of the user. This is done by a protocol given in Section 6.1.1.

3.3.2 When making purchase transaction at a POS terminal

The user of the card has to authenticate to the POS terminal. (the POS terminal wants to know who is paying and wants to lower the balance of the card) The shopkeeper has to authenticate to the user of the card. (the user of the card wants to be sure that the money goes to the shop keeper of who he is purchasing something. The user does not have to authenticate to the bank client, the money comes from the E-Purse.

3.3.3 When personalizing card

The user has to authenticate to the personalization terminal. The terminal needs to be sure that he is assigning the right personal information to the card of that person or user. The personalization terminal needs to authenticate to the card: someone could place a fake personalization terminal and lure users into initializing their card at the fake terminal and abuse this in some way.

3.4 Non-repudiation

3.4.1 When topping up the balance at the top-up terminal

The user should be able to prove anyone that he topped up his chipcard with amount x and that the amount is withdrawn from his bank account.

3.4.2 When making purchase transaction at a POS terminal

The shopkeeper should not be able to deny having received money from the buyer. the buyer or user of the E-Purse has to be able to prove that the balance of his E-Purse has been lowered with the price of the product that he bought at the shop and that this money has been transferred into the shop keepers bank account.

3.5 Availability

On the first place the card block service has to be available 24/7. This so that when the card is lost the user can prevent damage. The top-up terminal is always required to have a connection to central server. For convenience the top-up terminal should always be available. The POS terminal should be available when the shop is open.

4 Threats and abuse cases

4.1 Attacker transfers money from someones bank account onto his card

There is a threat that an attacker transfers money from someones bank account to his own E-Purse. The attack here is that the attacker is on the line between the E-Purse and the payment terminal and repeats the payment message sent by the client to the E-Purse. If there is a target ID of a E-Purse in the message the attacker could change this card ID into his own card ID.

4.2 Attacker tries to put money on his E-Purse without paying

This is done by tearing the card out the top-up terminal when the money is received and the balance is increased but before the card sends out information to the terminal to lower the bank account balance. This is not possible if the bank account balance is decreased first.

4.3 Abuse stolen smart card

The threat is that when an attacker steals a smartcard he can retrieve (private) information from the card or use the E-Purse and pay with someone else's money. Or in an even worse case, top-up the E-Purse and steal more money from the victim.

4.4 Abuse POS or top-up terminal (skimming)

It is a threat that the owner of a POS terminal or someone with access to a top-up terminal abuses this terminal. The attacker can obtain a clone of the card and the pin code by skimming.

4.5 Personalization Terminal

A threat at the personalization terminal is that an attacker "steals your card". He could try to go to a personalization terminal and download someones else personal data onto his card.

5 Attacker model

5.1 Assumptions

In this assignment we assume that confidentiality and integrity of the card is guaranteed and cannot be observed or altered. We do however assume that if one card is physically broken, an attacker can retrieve all the information stored on it. This includes retrieving the card's private key. In case the private key is leaked, this will allow an attacker to clone the card and allow the card's balance to go below zero. The card must therefore be blocked as soon as the attack is detected. We assume the master key of the central server to be well protected such that it is always available and remains confidential.

5.2 Possible attacks

5.2.1 Man in the Middle

The attacker can position himself in the line between either the top-up terminal and the bank card or the top-up terminal and the server or both. He has the ability to inject, replay and/or modify messages sent over the wire. For example he can replay the message instructing the card to increase its balance, attempting to increase the balance multiple times with a single payment. Another example is a merchant owning a POS terminal, who can replay a payment transaction, attempting the payment to be deposited on his account multiple times.

5.2.2 Tear card attack

We assume that the attacker has a special device with which he can very precisely time the tear of the card. In order to prevent a Tear card attack we need to be sure that when topping up a card we need to be sure that the bank account balance of the user is decreased before the balance is added to the card. In the same way we need to be sure that when a user buys something at the store the balance on the E-Purse is lowered before the money has been sent to the shop owner.

5.2.3 Steal a smartcard

An attacker could abuse a stolen smart card. Because we may assume that the integrity of the card is safe and information cannot be observed from it we will not look in to attacks like trying to read the balance by physically looking at the memory of the card. We will look in the possibility for the attacker to insert a card at a terminal and start brute forcing the PIN to topping up the card, pay with the card or change the PIN in something the attacker wants it to be. This is prevented by limiting the number of attempts that a pin can be tried at a terminal in a safe way. The attacker can pay with a stolen card, there is no PIN code needed for this. This is a pre-calculated loss. The damage is controlled by putting a maximum on the E-purse balance of 500 euro.

5.2.4 Corrupt owner of Point of sale terminal

The attack here is that the owner of the point of sale terminal puts a device in the card reader that clones the card when it is being read. He could also log the pin code that the user enters.

5.2.5 Skim Top-up terminal

The attack is that the skimmer places a device in the top-up terminal so that when a users tops up his E-Purse with more money the device clones the card and logs the users password. The attacker can later use this to steal money from the users bank account, top-up his cloned card and use this to pay with.

6 High level design

6.1 Security protocols

This section describes the most important, non-trivial security protocols used in communication between a card, some terminal and the central server. Using the protocols described will ensure certain security properties hold given the attacker model from section 5. To make you, the reader, familiar with the syntax used we will first privide you with a legend of symbols commonly used in the security protocols in Table 1.

M	: Private master key.
new N	: Nonce (infeasible to guess random number) N is introduced.
pk_C	: Card-specific public key for card C.
$(m)_k$	: Message m symetrically encrypted with key k.
$(m)_{pk_C}$	: Message m asymetrically encrypted with card-specific private key of card C.
$(m)_{sk_M}$	: Message m asymetrically signed with master key M.

Table 1: Legend of symbols and syntax used in security protocols

Before stating the protocols, we will first briefly state their goals and properties given the attacker model from section 5:

- Secrecy of the master key
- Authentication of the card
- Secrecy of the card-specific private key
- Secrecy of the PIN
- Authenticity and integrity of a transaction
- Authenticity and integrity of a transaction receipt

6.1.1 Topping up a card's balance

The security protocol is depicted in Figure 1. It describes the interaction between Card, Terminal and Server for topping up the balance of the card. In addition to the general protocol goals and properties given above, it ensures the following properties hold:

- The card's PIN authentication cannot be bypassed.
- A top-up transaction occuring on a card must first occur on the central server.

For this protocol the properties have been verified using ProVerif.

We designed the protocol in a way such that the top-up terminals don't need to keep secret keys, allowing them to be tampered with to a certain degree without affecting the security guarantees of the protocol. The initial design of the protocol used symetric crypto with card-specific pre-shared keys. However, we decided to switch to asymetric encryption for the entire protocol for two reasons:

- Either the terminal would be required to keep secret keys or send a cryptographic hashed value of the PIN (which may be mixed with a known nonce). Since nonces are known to an attacker by definition (as the terminal does not keep secret keys), the only unknown value that is left to brute-force from such a hashed value is the PIN, which is typically a 4-digit number. Therefore an offline brute-force attack is feasible.
  This can be mitigated by using asymetric encryption and, in addition to the PIN, encrypt a nonce N_T generated by the terminal. This would allow the card to decrypt the result and ignore N_T. However, the brute-force attack now becomes infeasible since N_T must also be guessed.
- Using symetric crypto, the server would use the card-specific pre-shared key to sign the receipt. The signature can be verified only by the server and the card. Switching to asymetric crypto allows anyone to verify the signature.

6.1.2 Making a purchase transaction

The protocol for making a purchase transaction is depicted in Figure 2. It describes the interaction between Card, POS terminal and Server for making a purchase. In addition to the general goals and properties guaranteed by all protocols, the following property is also guaranteed by this protocol:

- A purchase transaction occuring on the central server must have first occured on a card.

For the entire transaction asymetric encryption and signing is used. We chose to do so because we want the POS terminal to be able to work off-line and still be able to verify a transaction to some extent without communicating with the central server. The advantage in this case of using asymetric crypto vs symetric is the fact that an attacker will need to obtain the card-specific private key from a card in order to forge signatures. This is a serious obstacle and requires many skills, resources and dedication, making the attack much less likely.

However, in the unlikely event an attacker does succeed in obtaining the private key and purchases goods using it, the central server will detect that the card's balance goes below

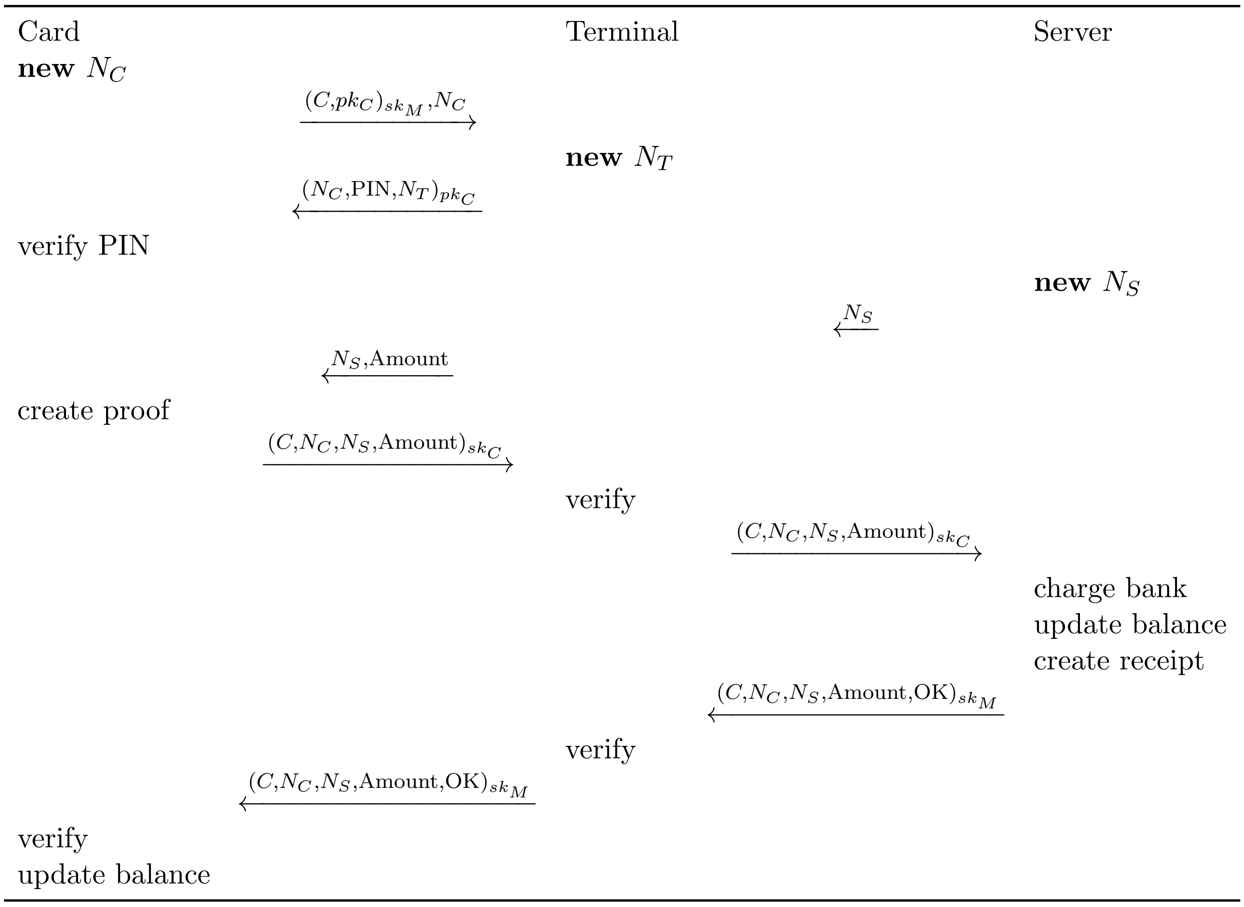
Card
Terminal
Server
new N_C
$(C, pk_C)_{sk_M}, N_C$
new N_T
$(N_C, \text{PIN}, N_T)_{pk_C}$
verify PIN
new N_S
N_S
N_S,Amount
create proof
$(C, N_C, N_S, \text{Amount})_{sk_C}$
verify
$(C, N_C, N_S, \text{Amount})_{sk_C}$
charge bank
update balance
create receipt
$(C, N_C, N_S, \text{Amount}, \text{OK})_{sk_M}$
verify
$(C, N_C, N_S, \text{Amount}, \text{OK})_{sk_M}$
verify
update balance

Figure 1: Security protocol for topping up a card's balance

zero and will immediately block the card. Admittedly, this will be detected after the attacker already left with the goods but the window of opportunity is quite small if merchants synchronize a list of block cards frequently. We believe the business model for such a sophisticated attack with yields that are not very high is not very tempting to say the least.

A replay attack performed by the POS terminal, although not visible from the protocol, can be mitigated by the central server. For example by keeping track of all nonces N_C that have occured in transactions for all cards in a database. However, with large volumes of transactions in mind, such a database tends to grow huge. Fortunately, the nonce N_C can be implemented as a transaction counter without compromising security. Doing so will allow the server not to keep track of all invalid values for N_C less than some $N_{C_{min}}$. Transactions with an N_C value less than $N_{C_{min}}$ will be regarded as a replay attempt. One may argue that some transactions may never be synchronized, effectively causing $N_{C_{min}}$ to stop increasing at a certain point. To mitigate this, a transaction synchronization timeout can be introduced.

Note that in the protocol the POS terminal is never authenticated by the card. As a consequence, this enables anyone with the ability to build a custom POS terminal to deduct an

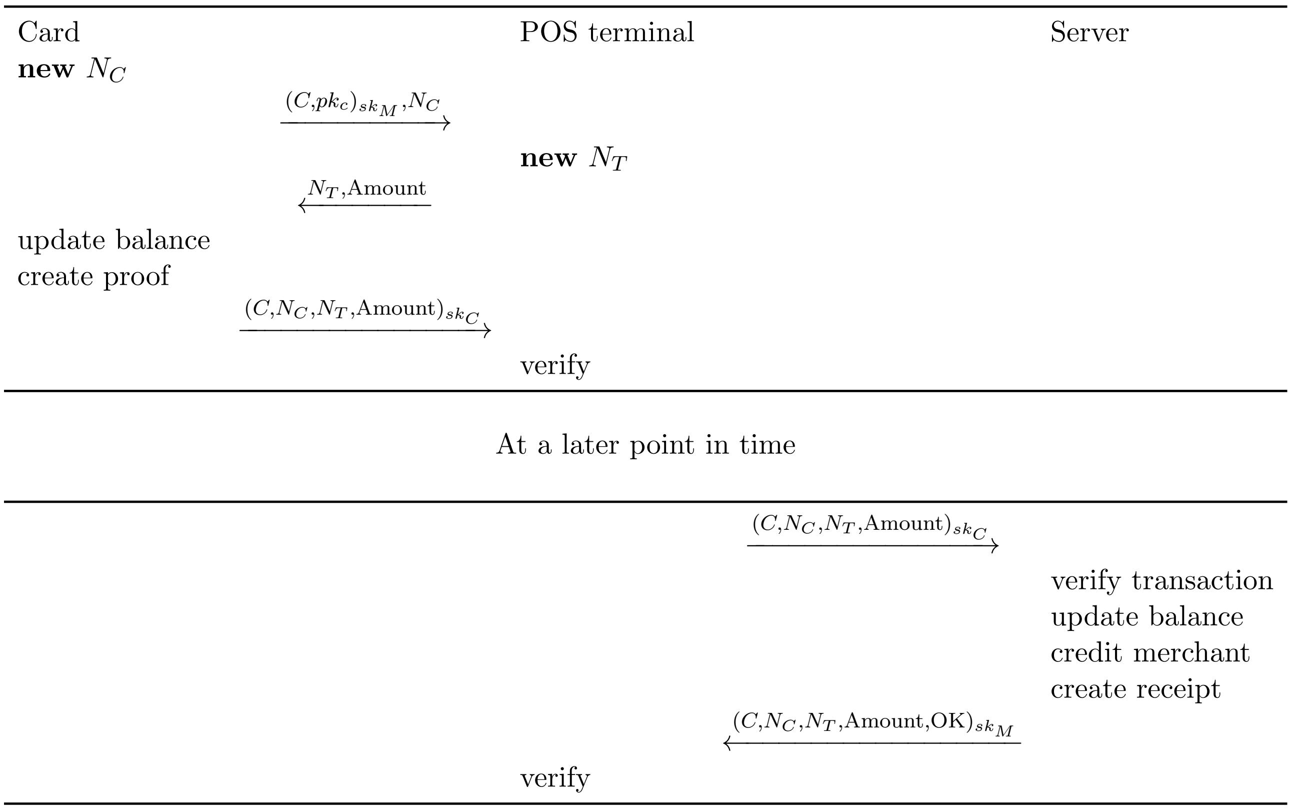

Figure 2: Security protocol for making a purchase transaction

arbitrary amount from the card without user interaction. This is a concious decision. We believe that introducing POS terminal authentication yields little added security since a genuine POS terminal can still be tampered with and one could attempt to extract their private key. We also believe making POS terminals tamper resistant is practically infeasible.

7 Deviations from the original design

The implementation differs slightly from the protocol depicted above in the sense that it does not have a central server. Instead the central server's capabilities are embedded in the terminals. During implementation we decided to assign a low priority to having a central serer and focus primarily on the card-to-terminal communication. However, our design is modular and allows for this to be implemented at a later point in time. Currently we do store receipts of all topup and payment transactions, allowing for them to be verified by the central server in the case one is implemented. Besides this, our implementation is equivalent to the design.