A PDA Implementation of an Off-line e-Cash Protocol

Efrén Clemente-Cuervo[1], Francisco Rodríguez Henríquez[1], Daniel Ortiz-Arroyo[2] and Levent Ertaul[3]
[1] Computer Science Department
Centro de Investigación y de Estudios Avanzados del IPN
Av. Instituto Politécnico Nacional No. 2508, México D.F.
ecuervo@computacion.cs.cinvestav.mx, francisco@cs.cinvestav.mx
[2]Computer Science and Engineering Department Aalborg
University Niels Bohrs Vej 8, 6700 Esbjerg Denmark
do@cs.aaue.dk
[3] Department of Math & Computer Science
California State University, East Bay.
levent.ertaul@csueastbay.edu

***Abstract*— We present an efficient implementation of a fair e-cash protocol especially designed for mobile wireless environments. Our protocol attempts to offer a reasonable balance between the anonymity feature on one side, and the possibility of revoking that anonymity in case that there exist reasonable doubts about the behavior of a given user. Our system considers two protocols especially designed for tracing purposes: a coin tracing and an owner tracing protocol. Furthermore, we propose the usage of TLS as an extra security layer in order to provide both, confidentiality and authentication. Our system was written in Java and it was implemented using wireless technology and PDA mobile devices.**

Keywords: E-Cash, Cryptography, E-Commerce, Mobile Security.

I. Introduction

The era of electronic commerce has changed the way transactions are being carried out traditionally. Transactions are now performed through private dedicated networks or through the internet. Money is being associated with cards issued by financial institutions, that customers employ to perform transactions online.

The concept of electronic money was introduced by Chaum [1]. Electronic money (digital money, or e-cash) tries to emulate its paper money counter part in terms of functionality. E-cash refers to cash and the associated transactions that are performed with it on a communication network. Special protocols are used to manage secure e-cash transactions. Since its introduction, several new features such as *off-line payment* [2] and *divisibility* [3] have been added to the concept of electronic money. Off-line refers to the fact that transactions can be carried out on-line even if a financial entity involved is off-line. Divisibility is a property of e-cash that enables dividing the value of a coin into smaller parts.

Cryptographic techniques have added security and anonymity to the use of electronic money [4]. In these systems, special features enable authorities to know how money has been used and the route that an electronic transaction has followed [5].

In this contribution we present the design and implementation of a fair off-line e-cash protocol especially tailored for mobile environments. We strive for reducing the client computational effort so that the mobile device performance (where typically client entities will be implemented) get not affected by the burden of hosting that application. We pretend to introduce those modifications without detriment to the security thanks to the usage of the Transport Layer Security (TLS) protocol for performing secure communications through a wireless Internet connection.

The rest of the paper is organized as follows. In next Section, a brief summary of some of the most well known e-cash protocols is given. Then in Section III we describe our proposed scheme in detail, which is a variation of the one proposed in [5], and where several cryptographic tools are utilized to obtain the desired security properties. In section IV, we describe the design and Java implementation of an electronic wallet system. In Section V we discuss the Security of our proposed system by examining possible attack scenarios and the way that our system can help to thwart them. Finally in Section VII concluding remarks are drawn.

II. Comparative Analysis of e-Cash Protocols

In this section we present a summary and a comparative analysis of some of the most well known e-cash protocols.

In 1988, David Chaum proposed a way to make electronic payments anonymously, introducing the concept of e-cash. E-cash was called in this way because of its similarity with paper money that guarantees purchaser anonymity. However some drawbacks of e-cash is that electronic money could be copied and reused. To avoid that a coin could be reused Chaum [1] proposed that the Bank keeps a list of all coins spent. This list is checked to verify that all coins deposited will not be reused. However, to perform this checking, the Bank should

be online, otherwise a store will not be able to guarantee that a payment is valid.

Forcing a Bank to stay on-line when a payment is made, is a strong limitation of this scheme that was corrected by Chaum, Fiat and Naor [2]. Their model was called *off-line cash*. In an off-line cash system, the protocol for deposit/collection occurs at a different time than the one used for payment/purchasing. Blind signatures, implemented with RSA public signing key, are used to guarantee anonymity. The use of the scheme *cut-and-choose* allows the Bank to detecting possible frauds and identify malicious purchasers. The scheme *cut-and-choose* was proposed by Michael O. Rabin[6]. In this protocol the opportunity to commit fraud is determined by the value $k/2$, where k is the security parameter employed by a Bank, which should be greater than 2. The probability of committing fraud is reduced when the value of k is increased by 1 in $2^{k/2}$[7].

Later, Okamoto and Otha[3] proposed a protocol that established the desirable properties that electronic money should have i.e. independence, security, privacy, off-line payment, transferability, divisibility. This protocol uses a binary tree representation for constructing a scheme of mutual agreement whose security relies on the hardness of the factorization and quadratic residues problems. To tackle the problem of divisibility authors in [3] proposed to use a binary tree that allows coin division but forbids falsification or reuse of the same coin. Unfortunately, it was recognized that this protocol may be traceable electronically.

In 1993, the *single-term* protocols were proposed. Among these protocols, the protocol proposed by Ferguson[8] combines RSA digital signatures and random blind signatures. In this system it is proposed to use the scheme of secret sharing, with which it is possible to know who commits fraud. Unfortunately, the security of this protocol was not tested [5], being this protocol superseded by the protocol proposed by Brands[4].

In 1993, S. Brands[4] proposes a new protocol, which was a refinement of the protocols proposed by: Chaum and Pedersen[9], and Cramery and Pedersen[10]. In this protocol, Brands makes use of the homomorphic properties of discrete logarithms. This scheme is based in the concepts of Schnorr digital signatures and the problem of representing groups of prime order.

In 1996, Frankel, Tsiounnis and Yung [11], [12] proposed a new architecture, creating the concept of *Fair Off-line e-Cash*. Their scheme works under the S. Brands e-cash model. In their proposal, a fourth entity called the *Authority* was added, which is used to guarantee the anonymity of a purchaser as long as he/she makes legal transactions. If a purchaser tries to commit fraud, the Bank could request the tracing of a coin or the tracing of the owner of a coin. In the protocols for coin or coin owner tracing only the Bank and the authority interact. With this feature Frankel, Tsiounnis and Yung tried to avoid crimes such as money laundry, blackmailing etc. in an efficient way.

Recently, several other protocols have been proposed [13], [14], [15], [16]. The protocol proposed in [13] is focused on an e-cash system for mobile devices. However, one weakness of this scheme is the requirement for the Bank to publish its coins to the Stores. The publication of these data makes difficult to consider that protocol as an off-line protocol.

III. Protocol Description

In the following, we propose an e-cash protocol based on the one presented in [4], which corresponds to a variant of the FOLC scheme proposed in [11].

A. A Fair Off-line Electronic Scheme

A fair off-line Electronic scheme defines the following four entities,

- The Bank: The financial entity, which provides and warrants the electronic money that is given to the client. The Bank is also responsible for detecting fraudulent transactions.
- The Purchaser: The entity that will use the electronic money for purchasing.
- The Store: This entity has as one of its functions, exchanging products or services for electronic money. It is also responsible for verifying the authenticity of the electronic money involved in a transaction. If a transaction is valid this entity will be enabled to deposit the electronic money received into its banking account.
- The Authority: This entity provides information on coin tracing or the ownership of a coin, when this procedure is legally allowed. The authority may perform this task any time under suspicion that malicious activity is being carried out on the part of a purchaser.

Additionally, this schemes comprises five sub-protocols, namely,

- Protocol for money withdrawal: Defines the interaction between the bank and the purchaser who wants to get a coin.
- Protocol for payment/purchasing: Defines the interaction between the purchaser and the product or service supplier Store.
- Protocol for deposit/collection: Defines the requirements that a Store must meet for being enabled to deposit coins into its baking account.
- Protocol for owner tracing: this protocol is executed by the Bank and Authority entities and is used to trace the identity of the owner of a specific coin. To perform this task, the Bank sends to the Authority a *brief* of what it was received through the deposit protocol. Next, the Authority sends back a string, containing the information with which the Bank is able to obtain the identity of a client by accessing its banking account data base.
- Protocol for Coin Tracing: this protocol is also executed by the Bank and Authority entities and is used to trace a coin created during the execution of the withdrawal protocol. The Bank provides to the Authority a brief of the withdrawal protocol, with which the Authority is capable of knowing where the coin has been spent.

In the rest of this Section we give a mathematical description of all cryptographic operations that the four entities of the protocol just described must perform in order to correctly execute above five subprotocols.

B. Parameters Description

The notation and principal parameters to be used to describe protocol's operation is as follows:

- q : parameter, $2^{159} < q < 2^{160}$
- p : given l such that $0 \leq l \leq 8$, let p be a prime such that $2^{511+64l} < p < 2^{512+64l}$, with the property that q divides $(p-1)/2$, i.e, $q|(p-1)/2$.
- g : the square of a primitive root mod *p.*
- g_1, g_2 : After picking two random numbers $x_1, x_2 \in Zq$, g_1 and g_2 are defined as ${g_1}^{x_1} \bmod p$ and ${g_2}^{x_2} \bmod p$, respectively.
- X_B : private key for the Bank
- h_1, h_2 : Bank's public keys
- X_T : Trustee's private key
- f_2 : Trustee's public key where, $f_2 = g_2^{X_T} \bmod p$
- Id_{Store} : store identifier
- u_1 : a secret random number for the creation of the Purchaser identifier.
- I : Purchaser identifier.
- s : a secret random number for coin creation
- w: a serial number for coin creation.
- H: hash function
- (A, B, z, a, b, r): A coin
- r_1, r_2 : parameters for fraud control
- t : time stamp

C. Initialization Process

First, the Bank performs an initialization procedure, selecting two prime numbers *p* and *q* such that *q*=(*p*-1)/2.

With g being the square of a primitive root $\in G_p$, three random numbers $x_0, x_1, x_2 \in Zq$ are picked for computing,

$g \equiv g^{x_0}$, $g_1 \equiv g^{x_1}$, $g_2 \equiv g^{x_2}$ (mod *p)*

The numbers, g, g_1 and g_2 are published. Lastly a random number X_B is chosen, which will be the Bank's secret key. With this key h, h_1, h_2 and h_3 are created as follows,

$h \equiv g^{X_B} \bmod p$, $h_1 \equiv g_1^{X_B} \bmod p$,

$h_2 \equiv g_2^{X_B} \bmod p$ $h_3 \equiv f_2^{X_B} \bmod p$

Those numbers will be published as the Bank identification. The Purchaser chooses a secret number and calculates what will become his account number as,$I \equiv g_1^u$ (mod *p*)

The number I is sent to the Bank, jointly with the user information (name, address, etc.). The Bank responds to the Purchaser with: $z' \equiv (Ig_2)^{X_B}$ (mod p). Lastly, the Bank registers each of the Stores with an identification number Id_{Store}. Once this initialization process has been accomplished, the entities Bank, Purchaser and Store will be ready to start interacting according to the protocols specified below.

D. Withdrawing Protocol

The goal of this protocol is enabling the Purchaser to obtain a valid coin created by the Bank. A coin will be granted as long as the Bank is sure of knowing the identity of who is requesting the coin. The coin is represented as a six-tuple: {A, B, z, a, b, r}. These values are generated as follows,

1) *Coin Request.* The Purchaser requests a coin from the Bank, by identifying himself with his number I and choosing a random number s, which is used to compute $A'_1$and A'_2. The Bank will verify that the Purchaser is in its banking account data base. Additionally, the Bank must verify that the number s chosen by the Purchaser during the coin generation process is the same as the one created for A'_2.. This verification is accomplished by using the Schnorr authentication protocol [17].

Purchaser		Bank
Coin request	$\Longrightarrow$	
$s, k \in_R Zq$		
$A'_1 = Ig_2 f_2^{s^{-1}}$		
$A'_2 = f_2^s, y = g_2^k$	$\Longrightarrow$	
	$\Longleftarrow$	e=H(I | y | X_B)
$r = es + k$	$\Longrightarrow$	if $g_2^r = A'^e_1 y$

2) *Definition of coin identifier* The Bank chooses a random number w, with which it will identify a coin. Then, it calculates the parameters $a'$and b' and sends those values to the Purchaser.

Purchaser		Bank
	$\Longleftarrow$	$a' = g^w$ and $b' = (A'_1)^w$

3) *Coin Creation.* The Purchaser chooses four random numbers and calculates *A, B, z, a, b*. All these calculations depend on the random numbers selected and the numbers $a'$and b' . Once this is done, it calculates c using a hash function H followed by a division by the value u. This value c' is sent to the Bank.

Purchaser		Bank
$A = (A'_1)^s$		
$z' = h_1^{u_1} h_2 h_3^{s^{-1}}, z = z'^s$		
$x_1, x_2, u, v \in_R Zq$		
$B_1 = g_1^{x_1}, B_2 = g_2^{x_2}$		
$B = [B_1, B_2]$		
$a = (a')^u g^v$		
$b = (b')^{su} A^v$		
$c = H(A\|B\|z\|a\|b)$		
$c' = c/u$	$\Longrightarrow$	

4) *Coin Signature.* This process is accomplished using a Schnorr signature protocol [17]. Thus, the Bank calculates r' using the value w (coin identifier) and the value of X_B (secret key of the Bank). A Purchaser may calculate r with the random values u and v to form the six-tuple *(A, B, z, a, b, r)*, which will identify a coin.

Purchaser		Store
	$\Longleftarrow$	$r' = c' X_B + w$
$r = r'u + v \bmod q$		
Verify:		
$g^r = h^{c'} a'$		
$(Ig_2 f_2)^{r'} = z'^{c'} b'$		

E. *Payment/purchasing protocol*

In this protocol, the Purchaser attempts to use a coin to pay for a product or service received from a Store. The protocol consists of three phases: first the coin is verified as being valid; second the values D1, D2 are verified, these values will be useful for owner tracing; and third, information is obtained so that a Bank could be able to identify any person trying to commit fraud. The protocol consists of the following steps:

1) *Establishing a purchase.* To start this protocol, the Store and Purchaser agree with respect to the price of a product and the amount of coins that are needed for the purchase.

Purchaser		Store
Purchase	$\Longleftrightarrow$	Sell

2) *Sending a coin.* The Purchaser sends a coin to the Store, along with four extra values, (A_1,A_2, D_1, D_2). With the first two the coin authenticity will be verified and the last two will be useful for owner tracing. The parameters, D_1 and D_2 are created in such a way that the Store can prove that the entity that created them is the same as the one that granted the coin. That way the Authority will be able of performing the trace protocol effectively in case that this feature may be required.

Purchaser		Store
$A_1 = I^s, A_2 = g_2^s$		
$m \in_R Zq$		
$D_1 = g_2^m$		
$D_2 = I + f_2^m$		
(A_1, A_2, D_1, D_2)	$\Longrightarrow$	
a coin (A, B, z, a, b, r)	$\Longrightarrow$	

3) *Validating the coin.* Store verifies a coin by performing the calculations described below. With these calculations it is shown that a Store can accept or reject a coin with the certainty that the coin, if it is accepted, is valid and that an invalid coin will always be rejected.

Purchaser		Store
	$\Longrightarrow$	if $A = A_1 A_2$
		if $A \neq 1, D_2 \neq 1$
		if Sign$(A, B) = (z, a, b, r)$

4) *Hash Function for a coin.* In the second phase of the protocol, the Store starts calculating d which is the evaluation of the hash function H_0 from *A, B, M* (Store's identifier) and a time stamp *t*. Besides computing this hashes three more computations are required, obtaining, D', f_2' and f_3' which are needed in order to verify the authenticity of the values D_1 and D_2. Once these values are calculated, they are sent to the Purchaser.

Purchaser		Store
		$d = H(A_1, B_1, A_2, B_2, Id_{Store}, t)$
		$S_0 \in_R Zq$
		$D' = D_1^{S_0} D_2$
		$f_2' = f_2 g_2^{S_0}$, $f_3' = D_1^{S_0} / g_2^{S_0}$
	$\Longleftarrow$	d, D', f_2', f_3'

5) *Control fraud data.* The Purchaser calculates r_1 and r_2 (which are the data that the Bank will use to get the identity of anyone attempting to perform a fraud), calculates V (for owner tracing) and finally it send those values to the Store.

Purchaser		Store
$r_1 = d(u_1 s) + x_1$		
$r_2 = ds + x_2$		
$V = H([(D' - f_2')/f_3']^s)$		
r_1, r_2, V	$\Longrightarrow$	Verify:
		If $g_1^{r_1} = A_1^d B_1$
		If $g_2^{r_2} = A_2^d B_2$
		If $H(A_1 \vert A_2^{S_0})$

F. *Deposit/collection protocol*

This protocol stipulates how the Store cash the coins previously owned by Purchasers, by asking the Bank to deposit them into his/her account. In this protocol, a coin is sent to the Bank in the form of a tuple *(A_1,A_2, D_1, D_2, A, B, z, a, b, r)* . This tuple was built by the Bank during the withdrawal protocol. Store also sends to the Bank the information required for fraud detection, namely, *(r_1,r_2,d, V)* .

Store		Bank
(A_1,A_2, D_1, D_2,		if $A = A_1 A_2$
A, B, z, a, b, r)	$\Longrightarrow$	if $A \neq 1, D_2 \neq 1$
		if Sign$(A, B) = (z, a, b, r)$
(r_1,r_2,d, V)	$\Longrightarrow$	Verify:
		If $g_1^{r_1} = A_1^d B_1$
		If $g_2^{r_2} = A_2^d B_2$
		If $H(A_1 A_2^{S_0})$

G. *Fraud Control*

In order to avoid frauds in the processing of electronic money, a secure e-cash system must have a way to detect them. The most common type of fraud is repeated use of an electronic coin issue known as the *double spending problem*. Our protocol specifies a mechanism that allows knowing the identity of those who attempt to pay out with the same coin twice. To see how this mechanism works, let us assume that Store-1 sends to the Bank the tuple (r_1, r_2, d) and that a Store-2 sends the tuple (r_1', r_2', d'), both invoices were produced by the reuse of the same coin with different vendors. In this case the Bank will obtain the value *u*, with a simple calculation as shown below. Given that:

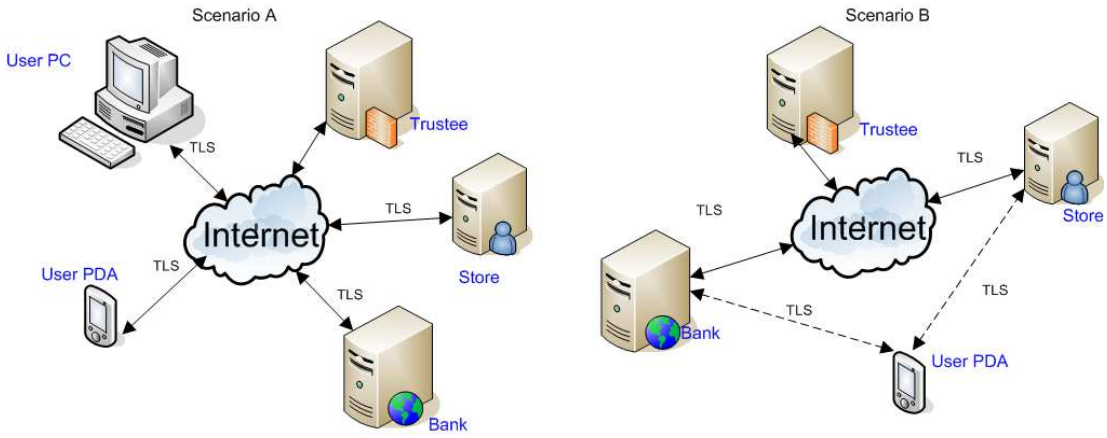

Fig. 1: scenarios used for the movile e-cash system

r_1 - r_1'$\equiv us(d-d')$ and r_2 - r_2'$\equiv s(d-d') \bmod q$; Then, u$\equiv (r_1 - r_1\prime)\ /(r_2 - r_2\prime) \bmod q$.

Therefore the Bank can calculate: $I \equiv g_1^u \pmod p$, thus identifying the malicious Purchaser.

H. Owner Tracing

Tracing the owner of a coin is a simple task, since each coin contains the identity of its owner. We modified Brands' scheme in [4] to use an special encryption system, in such a way that the encryption is also linked to the coin. To trace a coin's owner, the Bank sends the ciphered data (D1,D2) to the Authority, enabling it to decipher the value I=$D_2 - (D_1)^{X_T} \bmod p$, which is the identity of the coin's owner. In this process we have assumed that the Store previously performed the coin verification procedure.

I. Coin Tracing

Coin tracing is performed by changing the way in which the value A is calculated during coin generation. Let us recall that the coin was generated via three generators, namely, g_1, g_2 and f_2. The first two values are used to verify a coin, and the third one to trace the coin. From the withdrawal protocol we know that the value A was generated as, $A=I^s {g_2}^s f_2 \ \ mod\ p$. This parameter is later signed using a Schnorr digital signature. Lastly, to trace coins, the Bank will have to send the value A_2' to the Authority, so that the Authority calculates ${A_2'}^{X_T} = g_2^s = A_2$, and sends back this value to the Bank. This step will allow the Bank to know, which Store has provided such value. In this way, the Bank will know where a coin was spent.

IV. Design and Implementation

The protocol reviewed in this paper, use well defined, and independent entities. The design presented in this section follows the same strategy, keeping the entities and the processes linked to each sub-protocol in a separated manner. The e-cash system was designed in such a way that it could be incorporated into an existing e-commerce system. The system was tested using TLS in the communication layer through the internet. We also tested the system in an ad-hoc wireless communication network, using PDA as client and TLS to guarantee secure communication between entities. Figure 1 shows the two scenarios tested in the e-cash system implemented in this work.

In order to implement both scenarios shown in Figure 1, we wrote a code using Java WEB version, i.e., using JSP's and Servlets technologies, on the Apache's Tomcat servlet container. MySQL DBMS was used to store user data and e-commerce transactions. Clients running on PDA's or desktops were implemented as applets. Whereas the entities Bank, Store and Authority are all WEB servers which additionally of holding a WEB site, they host WEB applications that perform the corresponding protocols. That application is independent of the WEB site implementation, allowing the possibility that this application may be installed in any WEB site. Figure 2 shows the complete system architecture and the main components of each entity.

It should be noticed that all Purchasers owning personal computers may interact via HTTP/HTTPS with any of the protocol authorities. Whenever an entity needs to execute any of the sub-protocols, a user must download via Internet a signed applet with which all required operation will be performed using his/her default WEB browser. This scheme brings a process which is virtually user-transparent. However, those purchasers using a PDA must access to the system via HTTP/HTTPS. Then, after all the necessary data are available they must initialize an application loading the data obtained via WEB. Only then, they will be able to execute each sub-protocol application. This relative inconvenience is due to the restrictions that PDA devices currently show whose WEB browsers do not support signed applets.

V. System Security

A. Double Spending Problem

The double spending problem occurs whenever any user and/or entity attempts to use a coin more than once. There exist three scenarios in which such action could happen. Below we discuss briefly each one of them.

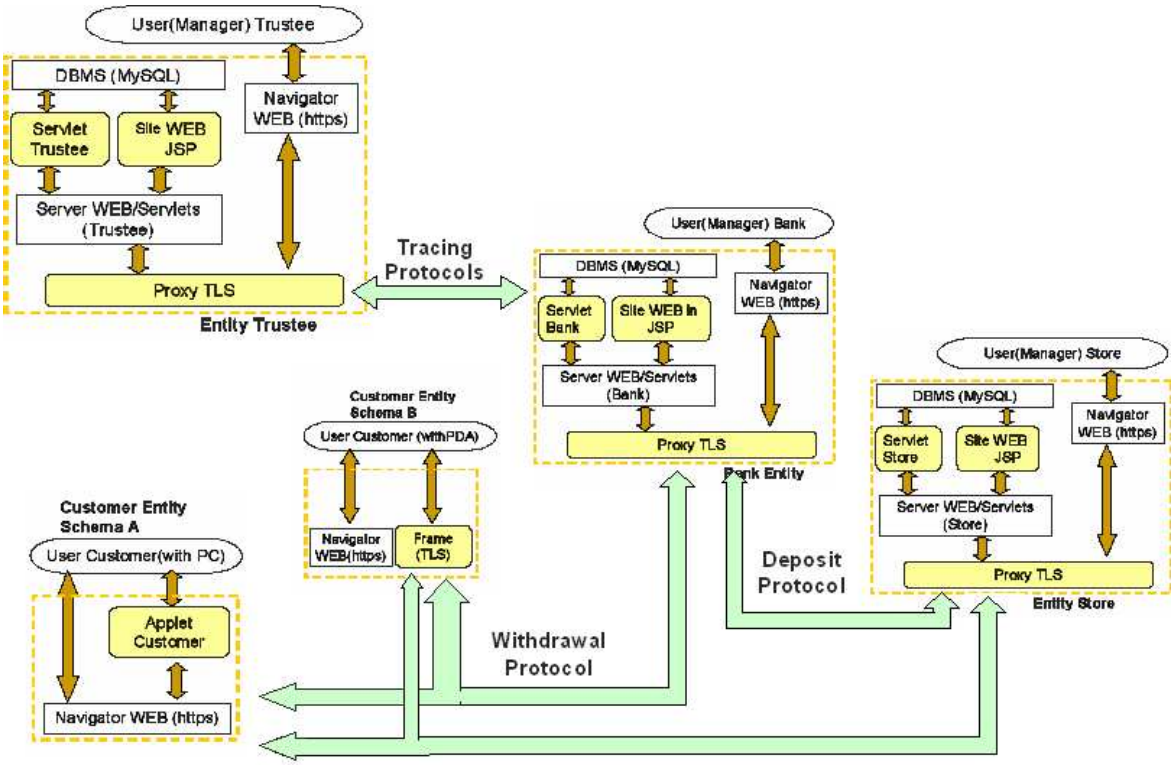

Fig. 2: Design

Malicious Purchaser: The first scenario considered occurs when a Purchaser attempts to spend his/her coin two or more times in different Stores. When this problem shows up, the Stores have not chance to detect the fraudulent transaction. However, when both Stores deposit their coins in the Bank, this entity will be able to detect that received coins are actually the same. Then, having double spent coin's information the Bank can deduce the identity of the person who committed fraud. Therefore the Bank assumes the cost of this fraud since involved Stores do receive their money. However, since the Bank can obtain cheater's identity it can initiate a legal process in order to recover its money. Notice that our protocol can only detect the double spending problem, but not prevent it.

Malicious Store: The second scenario occurs when a Store attempts to deposit two or more times the same coins received by an honest Purchaser. However, this situation means no problem due to the fact that if the Bank accepts two identical coins the verification parameters would be identical. Therefore, assuming that a Store has the coin (A, B, z, a, b, r), then after executing the purchasing protocol the Bank can obtain r_1 y r_2. If it is attempted to spend this same coin twice, the Bank will reject it, in virtue that the Bank will notice the values d, r_1 y r_2 would be the same. On the other hand, the Store cannot create valid r_1 and r_2 parameters since it does not know the values $\mathrm{u}_1, x_{1,}, x_2$ and s, which are only known by the Purchaser.

Store and Purchaser Colluded: A third scenario would occur in case that the Store and the Purchaser collaborate in a joint effort to cheat the Bank. Even in this case it would be always possible for the Bank to find a guilty person (either the Purchaser or the Store). Additionally the possibility that a Purchaser has been impersonated by someone else is eliminated because it is Bank's responsibility to verify customer's identity at the moment that the withdraw protocol is being executed.

B. *Digital Crimes*

A digital crime may be defined as any law violation by means of digital mediums. Ironically, a classic e-cash system strives for providing anonymity mechanisms that avoid the possibility of detecting suspicious users. Therefore, the main purpose of the coin and owner trace protocol is that of allowing the identification of a user who is suspected of having committed a crime.

Kidnapping, Blackmail, Bribes, Money-Laundry: Let us think in situations where kidnapping, blackmail and bribes kind of crimes could occur. For instance in a kidnapping situation, a criminal can demand the ransom to be paid using an e-cash system. Due to the anonymity feature of a classic e-cash system it would be a perfect tool for criminals to accomplish their goals. However, in our system the Authority would be able of hunting down any criminals provided that they attempt to spend coins in the electronic system. Therefore, owner and coin trace protocols are a valuable tool for preventing those crimes.

VI. Experimental Results

A brief performance evaluation in terms of message size and required number of cryptographic operations is presented in this Section. The traffic generated in the withdrawal, payment, deposit and tracing protocols are shown in Table I. Both, wallet and coin sizes are function of n, where n is the security parameter defined in Section III. In our experiments we used $n = 128$ bits and for this case the wallet size and the coin size are 384 and 907 bytes, respectively

TABLE I: Protocol Cryptographic Operations

Protocol	Crypto Operations	Traffic per coin
Withdrawal	15 exponentiations 2 inverses	1.7 KB
Payment	6 exponentiations 1 inverses	2.5 KB
Deposit	0 exponentiations 0 inverses	1.9 KB
Tracing	1 exponentiations 1 inverses	657 B

Finally, the timing needed for coin creation were of 1 and 3 seconds in the server and in the PDA client, respectively. In our experiments the Bank and the Purchaser were hosted in a PC computer Pentium IV with 128-MB RAM and a PDA Sharp-Zaurus5600, respectively.

VII. Conclusions

In this paper, we presented a comparative analysis of some well known e-Cash protocols. Furthermore, inspired in the protocols reported in [4], [11] we proposed a protocol that has proved to be a feasible option for mobile wireless environments. With the modifications implemented, the system is able to trace coins and also trace the owner of such coins whenever suspicious activity is detected. Our system exhibit all the desirable features that a secure e-Cash system should have, namely, independence, security, privacy, off-line payment, transferability, and divisibility. The proposed architecture was coded in Java and fully-implemented in personal computers interacting with PDA devices. Our experiments show that the e-cash protocol presented in this work can be efficiently implemented in those mobile devices without significant detriment of their performance.

References

[1] D. Chaum. *Blind signatures for untraceable payments*. Proc. of CRYPTO'82, pp. 199-203. Springer-Verlag(1982)

[2] D. Chaum, A. Fiat, and M. Naor. *Untraceable electronic cash*. LNCS 403, Proc. Of CRYPTO'88 pp. 319-327. Springer-Verlag(1988).

[3] T. Okamoto and K. Otha. *Universal electronic cash* . LNCS 576, Proc. Of CRYPTO'91, pp. 324-337, Springer-Verlag(1991).

[4] S. Brands. *Untraceable off-line cash in wallet with observer*. LNCS 773, Proc. Of CRYPTO'93, pp. 302-318. Springer-Verlag(1993).

[5] Y. Tiannis. *Efficient Electronic Cash: New Notations and Techniques*. PhD Thesis, Northeastern University Boston, Massachusetts(1997).

[6] Michael O. Rabin. *Digitalized Signatures*. Foundations of Secure Computation, ed. by R.A. DeMillo, D.P. Dobkin, A.K. Jones, R.J. Lipton; Academic Press, N.Y. 1978, 155-166.

[7] M. J. Farsi. Digital Cash. MSc. Thesis. Departament of mathematics and computing science, Goteborg University (1997)

[8] N Ferguson. *Single term off-line coins*. LNCS 765 EUROCRYPT'93 pp. 318-328. Springer-Verlag(1993).

[9] D. Chaum and T. Pedersen. *Wallet databases with observers* LNCS 740. Advances in Cryptology CRYPTO'92, pp 90-106 Springer-Verlag (1992)

[10] R. Cramer and T. Pedersen. *Improved privacy in wallets with observers* LNCS 765. Advances in Cryptology EUROCRYPT'93, pp 329-343 Springer-Verlag (1993)

[11] Y. Frankel, Y. Tsiounis, and M Yung. *Indirect Discourse Proofs: Achieving Efficient Fair Off-line E-Cash.* Asiacrypt '96, Lecture Notes in Computer Science 1163, pp 286-300, November 3-7, South Korea.

[12] Y. Frankel, Y. Tsiounis, and M. Yung. *Fair off-line e-cash made easy*. ASIACRYPT'98, LNCS 1514, Springer-Verlag, pp. 257-270, 1998.

[13] W. Ham. *Design of Secure and efficient E-commerce protocols using cryptographic primitives*. MSc. Thesis, School of engineering Information and Communications University, Daejeon Korea(2002).

[14] Ronggong Song and Larry Korba. *How to Make E-cash with Non-Repudiation and Anonymity*. Proceeding of the International Conference on Information Technology (ITCC 2004), pp 167-172, 2004.

[15] Robert Tracz and Konrad Wrona. *Fair Electronic Cash Withdrawal and Change Return for Wireless Networks*. WMC '01: Proceedings of the 1st international workshop on Mobile commerce, pp 14-19. ACM Press 2001.

[16] Alwyn Goh, Kuan W. Yip and David C. L. Ngo. *Flexibly Configurable and Computation-Efficient Digital Cash with Polynomial-Thresholded Coinage*. LNCS 2828, Springer-Verlag, pp 181-193, 2003.

[17] C.P. Schnorr. *Efficient signature generation by smard cards* . Journal of Cryptology 4(3) pp. 161-174, (1991).