Anonymity in Transferable E-cash*

Sébastien Canard[1] and Aline Gouget[2]

[1] Orange Labs R&D, 42 rue des Coutures, BP6243, F-14066 Caen Cedex, France.
[2] Gemalto, 6, rue de la Verrerie, F-92190 Meudon, France.

Abstract. Regular cash systems provide both the *anonymity* of users and the *transferability* of coins. In this paper, we study the anonymity properties of transferable e-cash. We define two natural additional levels of anonymity directly related to transferability and not reached by existing schemes that we call *full anonymity (FA)* and *perfect anonymity (PA)*. We show that the FA property can be reached by providing a generic construction and that the PA's cannot. Next, we define two restricted perfect anonymity properties and we prove that it is possible to design a transferable e-cash scheme where a bounded adversary not playing the bank cannot recognize a coin he has already owned.

1 Introduction

Electronic cash systems aim at emulating regular cash. Users withdraw coins from a bank, and next pay merchants using them. Then, merchants deposit coins to the bank. Even if the property of *transferability* (i.e. received cash can be spend later without involving the bank) is seen as a fundamental property of regular cash, it is usually disregarded in the electronic setting. This lack of interest for transferable e-cash may be explained by the result given in [6] showing that it is impossible to transfer a coin without increasing its size. However, this apparent drawback is not always unacceptable for applications depending on the available amount of storage data. The main advantage of the transferability of e-cash would be the decrease of communications between the bank and all users.

The anonymity in (non transferable) e-cash systems is well-studied in the literature. When introducing the transferability property into e-cash schemes, new notions of anonymity appear that have not already been described in the literature. These new anonymity notions related to transferable e-cash are studied in this paper.

As far as we know, the first transferable e-cash schemes that provides a weak level of anonymity has been proposed in [10, 11]. The anonymity

* This work has been financially supported by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT and by the French Agence Nationale de la Recherche and the TES Cluster under the PACE project.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

level is said *weak* since the spenders identities are protected but it is possible to link several spends of the same user.

Another method for transferring e-cash has been presented in [14, 6]. The anonymity level of this scheme is said *strong* since the spender identities are protected and it is not possible to link several spends of the same user. Very recently, two transferable e-cash schemes have been proposed in [5]. Both schemes improve the efficiency of [14, 6] by reducing the number of communications between the bank and users. One scheme offers a computational *strong* anonymity while the other one offers an unconditional *strong* anonymity. However, none of these schemes offers a "perfect" anonymity of spends since it is always possible for an adversary to recognize a coin that he has previously seen being spent.

There is a gap between the highest level of anonymity achieved by the transferable e-cash schemes of the state-of-the-art and the impossibility result given in [6] showing that transferable e-cash cannot fulfils an *unconditional* "perfect" anonymity since an unbounded payer can always recognize his own money if he sees it later in a payment.

In this paper, we contribute to reduce this gap, in one hand by showing the possibility for an e-cash system to fulfil higher levels of anonymity, and on the other hand by proving that a computational payer can always recognize his own money if he sees it later in a payment, meaning that transferable e-cash cannot provide a *computational* "perfect" anonymity.

In Section 2, we give formal definitions for transferable e-cash. In Section 3, we focus on the security properties related to anonymity and we introduce two new properties: the *Full Anonymity (FA)* meaning that the adversary $\mathcal{A}$ is not able to recognize a coin he has already observed during a spending between honest users, and the *Perfect Anonymity (PA)* meaning that $\mathcal{A}$ is not able to decide whether or not he has already owned a coin he is receiving. In Section 4, we show that a transferable e-cash scheme can fulfil the FA property by providing a generic construction, and that no scheme can fulfil the PA property by improving the impossibility result given in [6]. In Section 5, we give evidence that it is possible to design a PA scheme if we assume that the (not unbounded) adversary is not the bank. Finally, we conclude in Section 6.

2 Transferable E-cash

In this section, we define the algorithms of transferable e-cash, the variables and oracles used by the adversaries, and the classical security

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

properties that are not related to anonymity; anonymity properties are defined in Section 3. Note that our model can easily be extended to wallets by using the compact e-cash techniques [2].

2.1 Algorithms

A transferable e-cash system involves two types of player: a bank $\mathcal{B}$ and a user $\mathcal{U}$. A coin is represented by an identifier Id and some values π needed to prove its validity.

– $\texttt{ParamGen}(k)$ is a probabilistic algorithm that outputs the parameters of the system Par (including the security parameter k).

– $\texttt{BKeyGen}(Par)$ (resp. $\texttt{UKeyGen}(Par)$) is a probabilistic algorithm executed by $\mathcal{B}$ (resp. $\mathcal{U}$) that outputs the key pair $(sk_{\mathcal{B}}, pk_{\mathcal{B}})$ (resp. $(sk_{\mathcal{U}}, pk_{\mathcal{U}})$).

– $\texttt{Withdraw}(\mathcal{B}(sk_{\mathcal{B}}, pk_{\mathcal{B}}, pk_{\mathcal{U}}, Par),\ \mathcal{U}(sk_{\mathcal{U}}, pk_{\mathcal{U}}, pk_{\mathcal{B}}, Par))$ is an interactive protocol where $\mathcal{U}$ withdraws from $\mathcal{B}$ one coin. At the end, $\mathcal{U}$ either gets a coin $C = (Id, \pi)$ and outputs OK, or outputs $\perp$. The output of $\mathcal{B}$ is either its view $\mathcal{V}_{\mathcal{B}}^{\mathtt{W}}$ of the protocol (including $pk_{\mathcal{U}}$), or $\perp$.

– $\texttt{Spend}$ $(\mathcal{U}_1(Id, \pi, pk_{\mathcal{U}_2}, Par), \mathcal{U}_2(sk_{\mathcal{U}_2}, pk_{\mathcal{B}}, Par))$ is an interactive protocol where $\mathcal{U}_1$ gives a coin to $\mathcal{U}_2$. At the end, either $\mathcal{U}_2$ outputs a coin $C = (Id_C, \pi_C)$ or $\perp$, and either $\mathcal{U}_1$ saves that C is a spent coin and outputs OK, or $\mathcal{U}_1$ outputs $\perp$.

– $\texttt{Deposit}$ $(\mathcal{U}(Id, \pi, sk_{\mathcal{U}}, pk_{\mathcal{U}}, pk_{\mathcal{B}}, Par), \mathcal{B}(sk_{\mathcal{B}}, pk_{\mathcal{B}}, pk_{\mathcal{U}}, \mathcal{L}, Par))$ is an interactive protocol where $\mathcal{U}$ deposits a coin (Id, π) at the bank $\mathcal{B}$. If (Id, π) is not consistent/fresh, then $\mathcal{B}$ outputs $\perp_1$. Else, if Id already belongs to the list of spent coins $\mathcal{L}$, then there is an entry (Id, π') and $\mathcal{B}$ outputs $(\perp_2, Id, \pi, \pi')$. Else, $\mathcal{B}$ adds (Id, π) to its list $\mathcal{L}$, credits $\mathcal{U}$'s account, and returns $\mathcal{L}$. $\mathcal{U}$'s output is OK or $\perp$.

– $\texttt{Identify}$ (Id, π, π', Par) is a deterministic algorithm executed by $\mathcal{B}$ that outputs a public key $pk_{\mathcal{U}}$ and a proof Π_G. If the users who had submitted π and π' are not malicious, then Π_G is evidence that $pk_{\mathcal{U}}$ is the registered public key of a user that double-spent a coin.

– $\texttt{VerifyGuilt}(pk_{\mathcal{U}}, \Pi_G, Par)$ is a deterministic algorithm that can be executed by any actor. It outputs 1 if Π_G is correct and 0 otherwise.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

2.2 Global Variables

The set of user's public (resp. secret) keys is denoted by $\mathcal{PK} = \{(i, pk_i) : i \in \mathbb{N}\}$ (resp. $\mathcal{SK} = \{(i, sk_i) : i \in \mathbb{N}\}$; $sk_i = \perp$ if user i is corrupted). The set of views of supplied coin by oracles is denoted by $\mathcal{SC}$ and the set of all coins owned by the oracles is denoted by $\mathcal{OC}$. The set of deposited electronic cash (corresponding to $\mathcal{L}$) is denoted by $\mathcal{DC}$.

2.3 Oracles

By convention, the name of an oracle corresponds to the action done by this oracle.

Creation and corruption of users. $\texttt{Create}(i)$ executes $\texttt{UKeyGen}(Par) = (sk_i, pk_i)$, it defines $\mathcal{PK}[i] = pk_i$ and $\mathcal{SK}[i] = sk_i$ and it outputs pk_i. The oracle $\texttt{Corrupt}(i, pk_i)$ defines $\mathcal{PK}[i] = pk_i$ and $\mathcal{SK}[i] = \perp$ and it outputs OK. $\texttt{Corrupt}(i)$ outputs the value $\mathcal{SK}[i] = sk_i$ and it updates $\mathcal{SK}[i] = \perp$.

Withdrawal protocol. $\texttt{Suppl}()$ plays the bank side and it updates $\mathcal{SC}$ by adding $\mathcal{V}_{\mathcal{B}}^{\mathtt{W}}$ with bit 1 to flag it as a corrupted coin. The oracle $\texttt{Withd}(U)$ plays the user $\mathcal{U}$ side and it updates $\mathcal{OC}$ by adding the value (U, Id, π). The oracle $\texttt{Withd\&Suppl}(U)$ plays both sides and it updates $\mathcal{OC}$ as for $\texttt{Withd}(U)$ and $\mathcal{SC}$ by adding $\mathcal{V}_{\mathcal{B}}^{\mathtt{W}}$ with flag 0. It outputs the communications between $\mathcal{B}$ and $\mathcal{U}$.

Spending protocol. The oracle $\texttt{Rcv}(U_2)$ plays the role of user $\mathcal{U}_2$ with secret keys of user U_2 and it updates the set $\mathcal{OC}$ by adding a new entry (U_2, Id, π). The oracle $\texttt{Spd}(U_1)$ plays the role of user $\mathcal{U}_1$ by spending a coin in $\mathcal{OC}$ owned by user U_1. It uses and updates the entry (U_1, Id, π) of $\mathcal{OC}$ as the `Spend` protocol describes it.
The oracle $\texttt{Spd\&Rcv}(U_1, U_2)$ plays the role of both $\mathcal{U}_1$ and $\mathcal{U}_2$ and it executes the spending of a coin owned by user U_1 to user U_2. It updates $\mathcal{OC}$ by adding the value (U_2, Id, π) and by flagging the coin as spent by U_1. It outputs all the communications of the spending.

Deposit protocol. $\texttt{CreditAccount}()$ plays the role of the bank and it updates the set $\mathcal{DC}$. If the executed `Deposit` protocol outputs $(\perp_2, Id, \pi, \pi')$, then the oracle `CreditAccount` runs the algorithm `Identify` and outputs the result of this algorithm on inputs (Id, π, π'). The oracle $\texttt{Depo}(U)$ plays the role of the user U during a `Deposit` protocol.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

2.4 Classical Security Properties not related to anonymity

Unforgeability. No collection of users can ever spend more coins than they withdrew.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the set of all user's public keys $\mathcal{PK}$, the bank's public key $pk_{\mathcal{B}}$ and Par. $\mathcal{A}$ can play as many times as he wants with the oracles: `Create`, `Corrupt`, `Suppl`, `Withd&Suppl`, `Spd`, `Spd&Rcv`, `Rcv` and `CreditAccount`.

Let q_W denote the number of successful queries to the oracle `Suppl`. Let q_R denote the number of successful queries to the oracle `Spd`. Let q_S be the number of successful queries to the oracle `Rcv`. The adversary $\mathcal{A}$ wins the game if, at any time during the game, we have $q_W + q_R < q_S$.

Identification of double-spenders. No collection of users can double-spend a coin twice without revealing one of their identities.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the $\mathcal{PK}$ global variable, the bank's public key $pk_{\mathcal{B}}$ and Par. $\mathcal{A}$ can play as many times as he wants with the oracles: `Create`, `Corrupt`, `Suppl`, `Withd&Suppl`, `Spd`, `Spd&Rcv`, `Rcv` and `CreditAccount`.

$\mathcal{A}$ wins the game if, at any time of the game, the oracle `CreditAccount` outputs $(\perp_2, Id, \pi, \pi')$ and the output of the oracle `Identify` on inputs (Id, π, π') is not a public key related to a secret key $\perp$ in $\mathcal{SK}$.

Exculpability. The bank, even when cooperating with any collection of malicious users cannot falsely accuse users from having double-spent a coin.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the $\mathcal{PK}$ global variable, the bank's key pair $(pk_{\mathcal{B}}, sk_{\mathcal{B}})$ and Par. $\mathcal{A}$ can play as many times as he wants with the oracles: `Create`, `Corrupt`, `Withd`, `Spd`, `Spd&Rcv`, `Rcv` and `Depo`. At any time of the game, $\mathcal{A}$ outputs two spends (Id_1, π_1) and (Id_2, π_2). $\mathcal{A}$ wins the game if the outputs of the algorithm `Identify` on inputs (Id_1, π_1, π_2) is a public key pk such that the related secret key in $\mathcal{SK}$ is not $\perp$ together with a valid proof Π_G, and the output of the algorithm `VerifyGuilt` on inputs (pk, Π_G) is 1.

3 Anonymity Properties in Transferable E-cash

In this section, we focus on security properties related to anonymity, remembering usual ones and introducing our two new ones.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

3.1 Overview

The *Weak Anonymity (WA)* and the *Strong Anonymity (SA)* properties are classical properties. Informally speaking,

- an e-cash scheme fulfils the WA property if an adversary cannot link a spending to a withdrawal. However, the adversary may know if two spends are done by the same user or not.
- An e-cash scheme fulfils the SA property if it fulfils the WA property and if an adversary is not able to decide if two spends are done by the same user or not. However, the adversary may recognize a coin that he has already observed during previous spends.

We introduce two new anonymity properties directly related to the transferability property in e-cash that we call *Full Anonymity (FA)* and *Perfect Anonymity (PA)*. Informally speaking,

- an e-cash scheme fulfils the FA property if it fulfils the SA property and if an adversary is not able to recognize a coin that he has already observed during a spending between two honest users. However, the adversary may be able to recognize a coin he has already owned.
- An e-cash scheme fulfils the PA property if it fulfils the FA property and if an adversary is not able to decide whether or not he has already owned a coin he is receiving.

3.2 Description of the Game

Before defining the game, we need to recall that a transferred cash necessary grows in size [6]. This property may be exploited by the adversary $\mathcal{A}$ to win the game and break the anonymity property. Indeed, $\mathcal{A}$ may choose two users that do not own coins of the same size so as to distinguish which one is used at the end of the game.

Consequently, in the following game, we impose that the two challenged users i_0 and i_1 own coins of the same size and the coin used during the final call to the `Spd` oracle should be one of these coins.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the set of all user's public keys $\mathcal{PK}$.

1. $\mathcal{A}$ is given $(sk_{\mathcal{B}}, pk_{\mathcal{B}})$, Par and $\mathcal{A}$ can play with the oracles: `Create`, `Corrupt`, `Withd`, `Spd`, `Rcv`, `Spd&Rcv` and `Depo`.
2. At any time, $\mathcal{A}$ chooses two public keys $pk_{i_0}, pk_{i_1} \in \mathcal{PK}$, such that:

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

(a) $\mathcal{SK}[i_0] \neq \perp$ and $\mathcal{SK}[i_1] \neq \perp$;
(b) users i_0 and i_1 own coins of the same size;
(c) users i_0 and i_1 have been used only by a set of *authorized oracles* that depends on the power of the adversary $\mathcal{A}$ (see below).

3. A bit b is secretly and randomly chosen and $\mathcal{A}$ plays with $\texttt{Spd}(i_b)$.
4. $\mathcal{A}$ outputs a bit b'.

3.3 Security Properties Related to Anonymity

We define four adversaries related to the four anonymity properties (WA, SA, FA and PA) that can be used to play the game described at Section 3.2. Indeed, the adversary is allowed to observe the transactions involving the coin that will be spend at step 3 with some specific restrictions depending on the anonymity property.

Definition 1 (Adversaries). *We denote by i_0 and i_1 the two users chosen by the adversary at Step 2 of the game described in Section 3.2.*

- *The adversary $\mathcal{A}_{\texttt{WA}}$ is allowed to manipulate i_0 and i_1 only with the oracles:* `Create`, `Withd` *and* `Depo`.
- *The adversary $\mathcal{A}_{\texttt{SA}}$ is allowed to manipulate i_0 and i_1 only with the oracles:* `Create`, `Withd`, `Spd`, `Spd&Rcv` *with the additional restriction that i_0 and i_1 do not play the role of $\mathcal{U}_2$ and* `Depo`.
- *The adversary $\mathcal{A}_{\texttt{FA}}$ is allowed to manipulate i_0 and i_1 only with the oracles:* `Create`, `Withd`, `Spd`, `Spd&Rcv` *and* `Depo`.
- *The adversary $\mathcal{A}_{\texttt{PA}}$ is allowed to manipulate i_0 and i_1 with all the oracles except the* `Corrupt` *oracle.*

Definition 2 (Anonymity properties). *A transferable e-cash system fulfils the property $\mathcal{P} \in \{\texttt{WA}, \texttt{SA}, \texttt{FA}, \texttt{PA}\}$ if for an adversary $\mathcal{A}_{\mathcal{P}}$ playing the game described at Section 3.2, the probability that $b = b'$ differs from $1/2$ by a fraction that is at most negligible.*

Remark 1. By construction, the anonymity properties are (exclusively) related one to the other as follows: $PA \Rightarrow FA \Rightarrow SA \Rightarrow WA$.

4 Study of Anonymity Properties

The schemes proposed in [14, 6, 5] fulfil both the WA and the SA properties but, as far as we know, the FA property (and consequently the PA property) is not achieved by any state of the art scheme. In this section, we first show that the FA property can be reached by providing a generic construction. Next, we prove that the PA property cannot be achieved.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

4.1 Achieving the Full Anonymity Property

The difference between the SA game and the FA game is that the coin received at the challenge step by the adversary may have already been observed by $\mathcal{A}_{\mathtt{FA}}$ during a call to the `Spd&Rcv` oracle, whereas this case cannot happen during the SA game. The generic construction we propose is built upon an SA scheme. The key idea of our construction is to modify an SA scheme to get an FA scheme by protecting the communications of the spending protocol using the establishment of a unilateral authenticated secure channel between the receiver and the spender in order to prevent an active adversary to recognize a coin that he has previously seen being spent.

We assume that $\mathcal{S}$ is a transferable e-cash scheme that fulfils the SA property. From $\mathcal{S}$, we construct a transferable e-cash scheme $\mathcal{S}'$ that fulfils the FA property. We re-define only the spending protocol of $\mathcal{S}$ and we additionally use two building blocks: a secure symmetric encryption scheme $\mathcal{E} = (\mathtt{Enc}, \mathtt{Dec})$, and a unilateral authenticated key agreement protocol `KE` between two users (including a key-confirmation step) secure against an active adversary. In particular, `KE` is resistant to man-in-the-middle attacks.

A user $\mathcal{U}_1$ spends a coin to user $\mathcal{U}_2$ by first playing the `KE` protocol. At the end of the `KE` protocol, $\mathcal{U}_1$ and $\mathcal{U}_2$ share a unilateral authenticated session key K. Next, $\mathcal{U}_1$ and $\mathcal{U}_2$ play the `Spend` protocol of $\mathcal{S}$ by encrypting all the communications using the algorithms `Enc` and `Dec` with the common session key K.

Theorem 1. *Under the assumptions that $\mathcal{S}$ fulfils the SA property, and* `EK` *and $\mathcal{E}$ are secure, the system $\mathcal{S}'$ fulfils the FA property.*

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{FA}}$ breaks the FA property by determining between users i_0 and i_1, which one is i_b. By definition $\mathcal{A}_{\mathtt{FA}}$ is not allowed to manipulate i_0 and i_1 with the oracle `Rcv` and thus $\mathcal{A}_{\mathtt{FA}}$ owns the coin of the challenge only at step 3 of the game. $\mathcal{A}_{\mathtt{FA}}$ should have seen it during the withdrawal of this coin (using the oracle $\mathtt{With}(U)$) and possibly during spends between honest users (using the oracle `Spd&Rcv`).

By assumption ($\mathcal{S}$ fulfils the SA property), $\mathcal{A}_{\mathtt{FA}}$ cannot get the serial number of a coin involved in a withdrawal protocol. By construction of $\mathcal{S}'$, all communications related to the spending of a coin are encrypted with a unilateral authenticated ephemeral session key, which includes the communications related to a call to the oracle `Spd&Rcv`. Thus, $\mathcal{A}_{\mathtt{FA}}$ has no information about the identifier of the coin embedded into the spending

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

($\mathcal{A}_{\mathtt{FA}}$ may know the entry number of the coin in $\mathcal{OC}$ but not the serial number), except if $\mathcal{A}_{\mathtt{FA}}$ has succeeded in breaking either the security of KE to obtain the session key K or the security of $\mathcal{E}$ to decrypt the communications without knowing the decryption key. □

4.2 Impossibility of the Perfect Anonymity Property

It is known that a payer with unlimited computing power can always recognize his own money if he sees it later being spent [6], and thus the PA property cannot be achieved by an unlimited powerful adversary. In this section, we show that a bounded adversary $\mathcal{A}_{\mathtt{PA}}$, acting as the bank, can also win the anonymity game, meaning that the PA property cannot be achieved by transferable e-cash.

Attack against the PA Property. During the PA game, $\mathcal{A}_{\mathtt{PA}}$ creates users and corrupts some of them. At any time, $\mathcal{A}_{\mathtt{PA}}$ owns a set of valid coins $\{C_0, \cdots, C_l\}$ that he got from his interactions with the oracle Spd.

$\mathcal{A}_{\mathtt{PA}}$ chooses two honest users i_0 and i_1 such that they have no coins. Next $\mathcal{A}_{\mathtt{PA}}$ chooses two coins C_0 and C_1, spends coin C_0 to user i_0 and coin C_1 to user i_1 using the oracle Rcv. Then, $\mathcal{A}_{\mathtt{PA}}$ outputs i_0 and i_1, according to the PA game. The challenger next chooses at random a bit b and $\mathcal{A}_{\mathtt{PA}}$ plays a Spend protocol with the oracle Spd on input the user i_b. Acting as the bank, $\mathcal{A}_{\mathtt{PA}}$ then simply executes the Deposit algorithm for the coins C_b and C_0[1]. If a double spending is detected, then $\mathcal{A}_{\mathtt{PA}}$ outputs $b' = 0$ and he outputs $b' = 1$ otherwise. Thus, $\mathcal{A}_{\mathtt{PA}}$ can always succeed in guessing b.

5 Variants of the Perfect Anonymity Property

The attack described in Section 4.2 shows that the PA property cannot be achieved. In this section, we describe the two most natural ways to modify the PA game in order to prevent this attack. We define two distinct properties called PA_1* and PA_2* and show that both properties can be achieved. Next, we prove that there is no inclusion relation between the FA property and PA_1* (resp. PA_2*).

5.1 Additional Anonymity properties

The first possibility to make impossible the attack described in Section 4.2 is to prevent the adversary from receiving the coin C_b at Step 3 of the

[1] Even if this is a fraud, $\mathcal{A}_{\mathtt{PA}}$ can deposit the coin C_0 he has already spent.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

game described in Section 3.2. Then the coin C_b may have been manipulated by the adversary before Step 3 but the adversary only observes the spending of coin C_b between two honest users at Step 3.

Definition 3 (Adversary PA$_1$*). *The adversary $\mathcal{A}_{\mathtt{PA}_1^*}$ can manipulate the challenged users with all the oracles except the* `Corrupt` *oracle.*

Game of the PA$_1$* property. We modify the game described in Section 3.2 as follows. The steps 1 and 2 are unchanged. In step 3, the call to $\mathtt{Spd}(i_b)$ is replaced by a call to $\mathtt{Spd\&Rcv}(i_b, i)$ where i is a randomly chosen honest user.

The second possibility to avoid the attack against the PA property is to prevent the attacker to execute the deposit. We thus separate the power of the bank into two entities with distinct keys: $\mathcal{B}_W$ (resp. $\mathcal{B}_D$) is responsible of the withdraw (resp. the deposit) part. In the new property, the adversary is not allowed to control $\mathcal{B}_D$. Moreover, the `Deposit` and `Identify` protocols should be protected by a secret key of the bank $\mathcal{B}_D$. We should prevent the adversary from being able to simulate the deposit phase, as a user can do when this phase is based on public algorithms.

Definition 4 (Adversary PA$_2$*). *The adversary $\mathcal{A}_{\mathtt{PA}_2^*}$ can manipulate the challenged users with all the oracles except the oracles* `CreditAccount` *and* `Corrupt`*.*

Game of the PA$_2$* property. We modify the game described in Section 3.2 as follows. Only Step 1 is modified: `Withd` is replaced by `Suppl`, and `Depo` is replaced by `CreditAccount`.

Remark 2. Note that the discussion on public [13] or secret [2] `Deposit` and `Identify` is controversial in non-transferable e-cash definitions.

5.2 Studying the PA$_1$* Property

We first show that the PA$_1$* property can be achieved by proving that the scheme $\mathcal{S}'$ described in Section 4.1 is PA$_1$*.

Theorem 2. *The scheme $\mathcal{S}'$ described in Section 4.1 is PA$_1$*.*

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{PA}_1^*}$ breaks the PA$_1$* property of $\mathcal{S}'$ by determining between users i_0 and i_1 which one is i_b. Before Step 3 of the game, $\mathcal{A}_{\mathtt{PA}_1^*}$ has observed the withdrawal of the coin of the challenge but

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

cannot get the related serial number from it ($\mathcal{S}$ fulfils the SA property). Moreover, $\mathcal{A}_{\mathtt{PA}_1^*}$ may have owned many times the coin of the challenge using the oracles `Spd`, `Rcv` and `Spd&Rcv`. But, by construction of $\mathcal{S}'$, all communications related to the spending of a coin are encrypted with a unilateral authenticated ephemeral session key. Thus, all communications of the final call to the `Spd&Rcv` oracle are encrypted, which means that $\mathcal{A}_{\mathtt{PA}_1^*}$ cannot recognize the serial number of the spent coin embedded into the spending, except if $\mathcal{A}_{\mathtt{PA}_1^*}$ has succeeded in breaking either the security of `KE` or the security of $\mathcal{E}$ by decrypting the communications without knowing the decryption key. □

We then show that the previously defined anonymity properties and PA_1^* are independent (i.e. one property does not imply the other).

Proposition 1. *WA (resp. SA, resp. FA) and PA$_1$* are independent properties.*

Proof. WA $\not\Rightarrow$ PA$_1$*. The scheme proposed by Okamoto and Ohta [10, 11] fulfils the WA property but not the PA$_1$* one since the serial number of a coin is not protected and it is not modified from one spend to another.

PA$_1$* $\not\Rightarrow$ WA. If we apply the general construction of Section 4.1 onto a transferable e-cash scheme that does not fulfil the WA property, we can easily show that the new scheme fulfils PA$_1$* but not WA.

SA $\not\Rightarrow$ PA$_1$*. The schemes proposed in [14, 6, 5] fulfil the SA property but not the PA$_1$* one since the serial number of a coin is not protected and it does not change from one spend to another.

PA$_1$* $\not\Rightarrow$ SA. From SA$\Rightarrow$ WA and PA$_1$* $\not\Rightarrow$ WA, we have PA$_1$* $\not\Rightarrow$ SA.

FA $\not\Rightarrow$ PA$_1$*. See Appendix A.

PA$_1$* $\not\Rightarrow$ FA. This is due to FA$\Rightarrow$ SA and PA$_1$* $\not\Rightarrow$ SA. □

5.3 General description of a PA$_2$* scheme

In this section, we want to prove that PA$_2$* can be reached by a transferable e-cash scheme and the efficiency of the constructed scheme is out of the scope of this paper. The construction of a PA$_2$* transferable e-cash scheme is less straightforward than the PA$_1$*'s one. Indeed, we need to use an additional tool called *metaproof system* that has been introduced in [12] by de Santis and Yung.

Metaproofs. Roughly speaking, the metaproof tool corresponds to a NIZK (Non-Interactive Zero-Knowledge) proof of the existence of a NIZK proof to a statement. More precisely, they provide a metaproof system

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

for 3SAT and prove that their metaproof system is a bounded NIZK proof system [1]. The metaproof system gives an *indirect* proof covered by additional encryption mechanism such that the metaprover possesses a zero-knowledge witness and does not necessary know the witness of the proof itself. Eventually, the metaproof can be applied recursively.

Overview of our PA$_2$* transferable e-cash scheme. A spent coin is classically represented by at least a serial number S, a security tag T (that permits the identification of double-spenders) and a proof that S and T are correct. Then, a transferable spent coin should consists in at least a serial number S, a set of security tags $\mathcal{T} = \{T_1, \cdots, T_l\}$ and a proof of validity.

We first notice that, in a PA$_2$* e-cash scheme, a coin should be transferred without revealing any information on previous spends, even for the user that is receiving the coin. Moreover, the bank needs to retrieve the serial number and all security tags describing the history of the coin, which can be done by encrypting these values using the bank's public key.

Since a user should not be able to recognize a coin previously owned, the receiver should be able to verify the validity of the coin without being able to retrieve neither the serial number nor any security tag. Moreover, since the spender can next become a receiver of this coin, the spent coin should be modified at each spend so that she cannot recognize it. We thus need a cryptographic tool permitting someone to send the serial number, security tags and proofs without revealing nor knowing them but proving that they are valid, which can be done using metaproofs [12].

More precisely, if a user withdraws a coin, she spends it by computing the serial number S of the coin and the security tag T_1, plus a proof of validity V_1 that S and T_1 are well-formed. If the receiver wants to spend this coin, she computes a security tag T_2, she encrypts S and T_1 and she proves that T_2 is well-formed and that she knows the encryption of the serial number S, the encryption of the first security tag T_1 and a proof of validity V_1 without revealing the encrypted values nor V_1, using in particular a metaproof.

Description of our PA$_2$* Scheme. We largely use the proposal of transferable e-cash scheme from Canard, Gouget and Traoré [5] to describe our PA$_2$* scheme, with the restriction that a user withdraws one coin at a time, and not a wallet. In the following, we only give a high level description of our scheme and we refer to [5] and [12] for details.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Setup. Let $\mathcal{G}$ be a group of prime order p and g, g_0 be two random generators in $\mathcal{G}$. These data constitute the public parameters Par. Let $\mathcal{H}$ be a cryptographic hash function. In the `BKeyGen` algorithm, $\mathcal{B}$ computes two key pairs $(sk_{\mathcal{B},1}, pk_{\mathcal{B},1})$ and $(sk_{\mathcal{B},2}, pk_{\mathcal{B},2})$ of a CL signature scheme [3] that permit it to sign coins and enroll users, respectively. During the `UKeyGen` algorithm, each user $\mathcal{U}_i$ obtains a CL (verifiable) signature $C_i = \mathtt{Sign}(u_i, w_i)$ associated to his public key $pk_{\mathcal{U}_i} = g_0^{u_i}$ and a random data w_i. Let $\mathtt{Enc}_{\mathcal{B}}$ be a secure verifiable probabilistic encryption scheme (e.g. El Gamal) to encrypt messages for the bank.

Withdrawal protocol. Following [2, 5], a coin C withdrawn at the bank is a CL signature σ under the bank's public key $pk_{\mathcal{B},1}$ on the set of values (s, u_i, t, x) where u_i is the user secret key, s, t and x are random values; $C = (s, (u_i, t, x, \sigma))$. The value s implicitly defines the *serial number* of the coin and the value t implicitly defines the corresponding *security tag* (using the Dodis-Yampolskiy Pseudo Random Function [7]).

Spending of a withdrawn coin. A user $\mathcal{U}_i$, owning a coin $C = (s, (u_i, t, x, \sigma))$ withdrawn from $\mathcal{B}$, wants to spend a coin to a user $\mathcal{U}_j$.

1. $\mathcal{U}_j$ computes $r_j = g_0^{\frac{1}{u_j + d_j}}$ where d_j represents some data related to the transaction. Next, $\mathcal{U}_j$ sends r_j and d_j to $\mathcal{U}_i$.
2. $\mathcal{U}_i$ computes $S = g^s$, $T_i = pk_{\mathcal{U}_i} g^{\frac{r_j}{t + d_j}}$ and a NIZK proof V_i that
   - $\mathcal{U}_i$ knows a signature σ on s, u_i, t and x,
   - $S = g^s$ and $T_i = pk_{\mathcal{U}_i} g^{\frac{r_j}{t + d_j}} = g_0^{u_i} g^{\frac{r_j}{t + d_j}}$,

   without revealing s, t, u_i, x nor σ.
3. The spent coin is represented by $(S, \pi = (T_i, V_i, r_j, d_j))$.

First transfer of a coin. Let us now consider the user $\mathcal{U}_j$ that has received a coin $(S, \pi = (T_i, V_i, r_j, d_j))$ from user $\mathcal{U}_i$ during the above protocol. If $\mathcal{U}_j$ wants to spend this coin to a user $\mathcal{U}_k$, he has to proceed as follows.

1. $\mathcal{U}_k$ computes $r_k = g_0^{\frac{1}{u_k + d_k}}$ where d_k represents some data related to the transaction. Next, $\mathcal{U}_k$ sends r_k and d_k to $\mathcal{U}_j$.
2. $\mathcal{U}_j$ computes $T_j = pk_{\mathcal{U}_j} g^{\frac{r_k}{u_j + S + d_k}}$, $dS = \mathtt{Enc}_{\mathcal{B}}(S)$, $dT_i = \mathtt{Enc}_{\mathcal{B}}(T_i)$ and a NIZK proof V_j that
   - $\mathcal{U}_j$ knows a signature C_j on u_j and w_j
   - $T_j = pk_{\mathcal{U}_j} g^{\frac{r_k}{u_j + S + d_k}} = g_0^{u_i} g^{\frac{r_k}{u_j + S + d_k}}$ and $r_j = g_0^{\frac{1}{u_j + d_j}}$,

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

- dS and dT_i are correct encryptions of the unrevealed values S and T_i, respectively,
- there exists a NIZK proof V_i proving that S and T_i are well-formed, using in particular r_j and d_j, and linked to a valid signature of the bank (this step corresponds to a metaproof as described in [12]),

without revealing u_j, S, C_j, w_j, r_j, d_j, T_i nor V_i.

3. The spent coin is represented by $(dS, \pi = (T_j, dT_i, V_j, r_k, d_k))$.

Second transfer of a coin. Let us now consider the user $\mathcal{U}_k$ that has received a coin $(dS, \pi = (T_j, dT_i, V_j, r_k, d_k))$ from user $\mathcal{U}_j$ during the above protocol. If $\mathcal{U}_k$ wants to spend it to $\mathcal{U}_l$, he has to proceed as follows.

1. $\mathcal{U}_l$ computes $r_l = g_0^{\frac{1}{u_l + d_l}}$ where d_l represents some data related to the transaction. Next, $\mathcal{U}_l$ sends r_l, d_l to $\mathcal{U}_k$.
2. $\mathcal{U}_k$ computes $T_k = pk_{\mathcal{U}_k} g^{\frac{r_l}{u_k + dS + d_l}}$, $d^2S = \mathtt{Enc}_{\mathcal{B}}(dS)$, $dT_j = \mathtt{Enc}_{\mathcal{B}}(T_j)$, and $d^2T_i = \mathtt{Enc}_{\mathcal{B}}(dT_i)$ and a NIZK proof V_k that
   - $\mathcal{U}_k$ knows a signature C_k on u_k and w_k
   - $T_k = pk_{\mathcal{U}_k} g^{\frac{r_l}{u_k + dS + d_l}} = g_0^{u_k} g^{\frac{r_l}{u_k + dS + d_l}}$ and $r_k = g_0^{\frac{1}{u_k + d_k}}$,
   - d^2S, dT_j and d^2T_i are correct encryption of the unrevealed values dS, T_j and dT_i, respectively,
   - there exists a NIZK proof V_j proving that dS, T_j and dT_i are well-formed, using in particular r_k and d_k, and linked to valid signatures of the bank, the one from the withdrawal phase and the one corresponding to the certificate of $\mathcal{U}_j$ (this step corresponds to a metaproof as described in [12]),

   without revealing u_k, dS, C_j, w_k, T_j, dT_i nor V_j.
3. The spent coin is represented by $(d^2S, \pi = (T_k, dT_j, d^2T_i, V_k, r_l, d_l))$.

The next spendings of this coin work similarly and are not described in this paper.

Deposit and Identify. During a deposit, $\mathcal{U}$ sends the received coin (e.g. of the form $(d^2S, \pi = (T_k, dT_j, d^2T_i, V_k, r_l, d))$) to $\mathcal{B}$. Then $\mathcal{B}$ first checks if this coin is fresh by decrypting d^2S until obtaining the initial S and by testing if S already belongs to $\mathcal{L}$. If this is not the case, then everything is ok. Otherwise, there is a double-spending and $\mathcal{B}$ has two deposited coins. Then, $\mathcal{B}$ compares the first spending of both coins. If they are identical, then $\mathcal{B}$ goes to the second one an so on until two spends at the same level are different (this case always happens). $\mathcal{B}$ finally retrieves the identity of the cheater by first decrypting the related values T and T' and next using the compact e-cash technique to retrieve the cheater public key.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

5.4 Achieving the PA$_2$* Property

Note that our PA$_2$* scheme is in accordance with the result of [6] which says that an unbounded adversary can always recognize his own coin during the game if it sees it later in a payment since such adversary is capable of decrypting values to retrieve the spender's identity.

Theorem 3. *Under the security of the used encryption scheme (e.g. El Gamal), the security of NIZK proofs and the security of the Dodis-Yampolskiy PRF, the proposed scheme fulfils the PA$_2$* property.*

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{PA}_2^*}$ succeeds in breaking the PA$_2$* property of the scheme described at Section 5.3. That means that $\mathcal{A}_{\mathtt{PA}_2^*}$ is able to decide, between two honest users i_0 and i_1 chosen by $\mathcal{A}_{\mathtt{PA}_2^*}$, which user is the spender i_b during a call to the oracle $\mathtt{Spd}(i_b)$. Note that, there is no restriction on the list of *authorized oracles* for such adversary.

The best strategy for $\mathcal{A}_{\mathtt{PA}_2^*}$ is to choose the users i_0 and i_1 such that he has previously manipulated all the coins owned by these users. Then $\mathcal{A}_{\mathtt{PA}_2^*}$ has to recognize the coin $C = (d^i S, \pi = (T_l, dT_k, \cdots, d^i T_j, V_l, r_m, d_m))$ sent by i_b that he has previously owned. Consequently, $\mathcal{A}_{\mathtt{PA}_2^*}$ knows some values that has been used to compute this coin (such as e.g. $d^{i_0} T_{j_0}$). When receiving a coin, $\mathcal{A}_{\mathtt{PA}_2^*}$ cannot learn anything from:

- the encrypted serial number $d^i S$ under the security of the probabilistic encryption scheme (even if he knows the value that is encrypted),
- the encrypted security tags $dT_k, \cdots, d^i T_j$ under the security of the encryption scheme (even if he knows some encrypted values),
- the security tag T_l of the spender under the security of the Dodis-Yampolskiy PRF (see [7, 5] for details),
- the values r_m and d_m that comes from $\mathcal{A}_{\mathtt{PA}_2^*}$ himself,
- the proof V_l by definition of a NIZK proof. In particular, see the result on metaproofs [12] and on usual zero-knowledge proofs of knowledge based on the discrete logarithms [8, 4].

Consequently, even if $\mathcal{A}_{\mathtt{PA}_2^*}$ has already seen the spent coin, he cannot recognize it. Thus, $\mathcal{A}_{\mathtt{PA}_2^*}$ cannot win the PA$_2$* game and our construction is PA$_2$*, which concludes the proof. □

We finally show that there is no relation between previously defined anonymity properties and the PA$_2^*$ one.

Proposition 2. *PA$_2$* and WA (resp. SA, resp. FA, resp. PA$_1$*) are independent properties.*

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Sketch of Proof. WA $\not\Rightarrow$ PA$_2$*. The scheme given in [10, 11] fulfils the WA property but not the PA$_2$* property since the serial number of a coin is not protected and it does not change from one spend to another.

PA$_2$* $\not\Rightarrow$ WA. See appendix B.

PA$_2$* $\not\Rightarrow$ SA. This is due to SA $\Rightarrow$ WA and PA$_2$* $\not\Rightarrow$ WA.

SA $\not\Rightarrow$ PA$_2$*. The schemes proposed in [14, 6, 5] fulfil the SA property but not the PA$_2$* property since the serial number of a coin is not protected and it does not change from one spend to another.

PA$_2$* $\not\Rightarrow$ FA. This comes from FA $\Rightarrow$ SA and PA$_2$* $\not\Rightarrow$ SA.

FA $\not\Rightarrow$ PA$_2$*. The scheme proposed in Section 4.1 fulfils the FA property but not the PA$_2$* property.

PA$_1$* $\not\Rightarrow$ PA$_2$*. The generic construction given in Section 4.1 fulfils PA$_1$* but not PA$_2$* property (for obvious reasons).

PA$_2$* $\not\Rightarrow$ PA$_1$*. The PA$_2$* scheme proposed in Section 5.3 does not fulfil the PA$_1$* property since the adversary being the bank in the PA$_1$* game can decrypt all encrypted data of all spends. □

6 Conclusion

In this paper, we provide the first study-in-depth of anonymity properties in transferable e-cash by introducing the full anonymity (FA) and perfect anonymity (PA). We show that the FA property can be reached by providing a generic construction and we prove that the PA cannot. We then define two restricted PA properties called PA$_1$* and PA$_2$* and we show that both restricted properties can be reached. Finally, we show that FA, PA$_1$* and PA$_2$* are three separate properties. Thus, an anonymous transferable e-cash scheme should ideally fulfils these three properties, which is the case (obviously inefficiently) if we apply the trick of Section 4.1 to the PA$_2$* scheme of Section 5.3. Note that all our results can easily be extended to wallets by using the compact e-cash techniques [2].

Acknowledgments. We are grateful to Marc Girault, Pascal Paillier and Jacques Traoré for their suggestions of improvement, and to anonymous referees for their valuable comments.

References

1. M. Blum, P. Feldman, and S. Micali. Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract). In *STOC'88*, pages 103–112. ACM, 1988.
2. J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact E-Cash. In *Eurocrypt'05*, volume 3494 of *LNCS*, pages 302–321. Springer, 2005.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

3. J. Camenisch and A. Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *Crypto'04*, volume 3152 of *LNCS*, pages 56–72. Springer, 2004.
4. S. Canard, I. Coisel, and J. Traoré. Complex Zero-Knowledge Proofs of Knowledge Are Easy to Use. In *ProvSec'07*, volume 4784 of *LNCS*, pages 122–137. Springer, 2007.
5. S. Canard, A. Gouget, and J. Traoré. Improvement of Efficiency in (Unconditional) Anonymous Transferable E-Cash. In *Financial Cryptography and Data Security'08*, volume 4887 of *LNCS*, pages 571–589. Springer, 2008.
6. D. Chaum and T.P. Pedersen. Transferred Cash Grows in Size. In *Eurocrypt'92*, volume 658 of *LNCS*, pages 390–407. Springer, 1992.
7. Y. Dodis and A. Yampolskiy. A Verifiable Random Function with Short Proofs and Keys. In *PKC'05*, volume 3386 of *LNCS*, pages 416–431. Springer, 2005.
8. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable Signatures. In *Eurocrypt'04*, volume 3027 of *LNCS*. Springer, 2004.
9. Y. Kim, A. Perrig, and G. Tsudik. Communication-Efficient Group Key Agreement. In *IFIP/Sec'01*, volume 193 of *IFIP Conference Proceedings*, pages 229–244. Kluwer, 2001.
10. T. Okamoto and K. Ohta. Disposable Zero-Knowledge Authentications and Their Applications to Untraceable Electronic Cash. In *Crypto'89*, volume 435 of *LNCS*, pages 481–496. Springer, 1990.
11. T. Okamoto and K. Ohta. Universal Electronic Cash. In *Crypto'91*, volume 547 of *LNCS*, pages 324–337. Springer, 1991.
12. A. De Santis and M. Yung. Cryptographic Applications of the Non-Interactive Metaproof and Many-Prover Systems. In *Crypto'90*, volume 537 of *LNCS*, pages 366–377. Springer, 1990.
13. M. Trolin. A Stronger Definition for Anonymous Electronic Cash. In *ePrint Archive*, 2006.
14. H. van Antwerpen. Electronic Cash. Master's thesis, CWI, 1990.

A Proof of Proposition 1: FA $\not\Rightarrow$ PA$_1$*

We first describe a toy scheme $\mathcal{TS}$ and next we prove that $\mathcal{TS}$ fulfils the FA property but it does not fulfil the PA$_1$* property.

Description of the Toy Scheme $\mathcal{TS}$. We assume that $\mathcal{S}$ is a transferable e-cash scheme that fulfils the SA property (e.g. [14, 6, 5]). We need to redefine only the spending protocol of $\mathcal{S}$. We additionally use as building blocks a secure symmetric encryption scheme $\mathcal{E} = (\mathtt{Enc}, \mathtt{Dec})$ and a unilateral authenticated group key agreement (GKA) scheme which uses e.g. the proposal of [9] where g is a public element. Each user has a signature key pair together with a valid certificate. In particular, this permits us to make the GKA scheme resistant to man-in-the-middle attacks. If $\mathcal{U}_1$ wants to spend a withdrawn coin to $\mathcal{U}_2$, he has to proceed as follows.

– $\mathcal{U}_1$ chooses at random a value K_1 and sends g^{K_1} to $\mathcal{U}_2$. User $\mathcal{U}_2$ chooses at random a value K_2, computes g^{K_2}, signs $g^{K_1} \| g^{K_2}$ and sends g^{K_2}

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

and the signature to $\mathcal{U}_1$. Both can securely and secretly compute $K = g^{K_1 K_2}$ and execute a key-confirmation protocol.
- $\mathcal{U}_1$ and $\mathcal{U}_2$ play together the `Spend` protocol of $\mathcal{S}$ by encrypting communications using the encryption algorithm `Enc` with the common secret key K. Both $\mathcal{U}_1$ and $\mathcal{U}_2$ can decrypt communications using the decryption algorithm `Dec` with the common secret key K.

If $\mathcal{U}_2$ wants to spend a received coin to $\mathcal{U}_3$, the protocol is as follows.

- $\mathcal{U}_2$ sends $g^K = g^{g^{K_1 K_2}}$ to $\mathcal{U}_3$. User $\mathcal{U}_3$ chooses at random a value K_3, computes g^{K_3}, signs $g^{K_3} \| g^K$ and sends g^{K_3} and the signature to $\mathcal{U}_2$. Both can compute $K' = g^{K K_3} = g^{g^{K_1 K_2} K_3}$, as in [9], and execute a key-confirmation protocol.
- $\mathcal{U}_2$ and $\mathcal{U}_3$ play together the `Spend` protocol of $\mathcal{S}$ using `Enc` and the session key K', as for the spending of a withdrawn coin

Note that the adversary playing the role of $\mathcal{U}_2$ cannot take any advantage in not sending the correct value $g^K = g^{g^{K_1 K_2}}$ for obvious reasons.

Proposition 3. *Under the assumptions that $\mathcal{S}$ fulfils the SA property, and $\mathcal{E}$ is secure, the $\mathcal{TS}$ system fulfils the FA property.*

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{FA}}$ succeeds in breaking the FA property of $\mathcal{TS}$ and thus decide, between i_0 and i_1, which user is the user i_b from which $\mathcal{A}_{\mathtt{FA}}$ receives the coin of the challenge.

Note that the oracle `Rcv` is not allowed for the manipulation of users i_0 and i_1. Thus, at Step 2 of the Game, $\mathcal{A}_{\mathtt{FA}}$ chooses users i_0 and i_1 such that all the coins owned by users i_0 and i_1 have never been owned by $\mathcal{A}_{\mathtt{FA}}$. Then, $\mathcal{A}_{\mathtt{FA}}$ owns the coin of the challenge for the first time at step 4.

Before Step 3 of the game, $\mathcal{A}_{\mathtt{FA}}$ has observed the withdrawal of the coin of the challenge (using the oracle `With`(U)) and $\mathcal{A}_{\mathtt{FA}}$ may have observed many times a spending between two honest users involving the coin of the challenge, using the oracles `Spd&Rcv`.

By assumption ($\mathcal{S}$ fulfils the SA property), $\mathcal{A}_{\mathtt{FA}}$ cannot get the serial number of a coin involved in a withdrawal protocol. By construction of $\mathcal{TS}$, all communications related to the spending of a coin are encrypted with an anonymous ephemeral session key. Thus, all communications related to a call to the oracle `Spd&Rcv` are encrypted. That means that $\mathcal{A}_{\mathtt{FA}}$ has no information about the identifier of the coin embedded into the spending ($\mathcal{A}_{\mathtt{FA}}$ may know the entry number of the coin in $\mathcal{OC}$ but not the serial number), except if $\mathcal{A}_{\mathtt{FA}}$ has succeeded in breaking either the security of the unilateral authenticated group key agreement or the security

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

of $\mathcal{E}$ to decrypt the communications without knowing the decryption key which is impossible by assumption. □

Proposition 4. *$\mathcal{TS}$ does not fulfil the PA$_1$* property.*

Sketch of Proof. $\mathcal{TS}$ does not fulfil the PA$_1$* property (by construction). Indeed, $\mathcal{A}_{\mathtt{PA}_1^*}$ can always choose one of his coins and give it to i_0 that has no coin. Since $\mathcal{A}_{\mathtt{PA}_1^*}$ has received the coin, he necessarily knows the session key K. During the game, $\mathcal{A}_{\mathtt{PA}_1^*}$ chooses i_0 as defined previously and i_1 at random. Then, the oracle $\mathtt{Spd\&Rcv}(i_b, i)$, where i is a random honest user, is called. The underlying `Spend` protocol uses a session key K' from a key $\tilde{K}$ introduced by i_0 or i_1 and a random key K_3 introduced by the receiver, as in the spending protocol. Then $\mathcal{A}_{\mathtt{PA}_1^*}$ can easily check if the key $\tilde{K}$ corresponds or not to the key K he knows. If this is the case, $\mathcal{A}_{\mathtt{PA}_1^*}$ outputs 0 and he outputs 1 otherwise and wins the game with a probability of success equal to 1, which concludes the proof. □

B Proof of Proposition 2: PA$_2$* $\not\Rightarrow$ WA

In order to prove that PA$_2$* $\not\Rightarrow$ WA, we describe a toy scheme and we prove that it fulfils the PA$_2$* property but not the WA one.

Withdrawal protocol. The user $\mathcal{U}$ gets from the bank $\mathcal{B}$ a signature σ on the serial number s of the coin. Note that the serial number is not hidden to the bank that consequently knows s and σ.

Spending a withdrawn coin. The user $\mathcal{U}_1$ spends the coin (s, σ) to the user $\mathcal{U}_2$ by encrypting s and the signature σ to obtain E and producing a NIZK proof that the encrypted σ is a signature of the encrypted value s.

Spending a received coin. $\mathcal{U}_2$ spends a received coin (E, U) to $\mathcal{U}_3$ by using the metaproof technique [12] to produce a NIZK proof V that there exists a NIZK proof U of validity of the spent coin. This step can be done many times so that the coin can be spent again and again.

This scheme is straightforwardly PA$_2$* but does not achieve the WA property since the bank can decrypt all spends to retrieve s and thus make the link with the initial withdrawal.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.