# A handy multi-coupon system

Sébastien Canard[1], Aline Gouget[2], and Emeline Hufschmitt[1]

[1] France Telecom, R&D Division
42 rue des Coutures, BP 6243, 14066 Caen Cedex, France
{sebastien.canard,emeline.hufschmitt}@orange-ft.com
[2] Gemalto
34 rue Guynemer, 92447 Issy-les-Moulineaux, France
{Aline.GOUGET}@gemalto.com

**Abstract.** A coupon is an electronic data that represents the right to access a service provided by a service provider (e.g. gift certificates or movie tickets). Recently, a privacy-protecting multi-coupon system that allows a user to withdraw a predefined number of single coupons from the service provider has been proposed by Chen et al. at *Financial Crypto 2005*. In this system, every coupon has the same value which is predetermined by the system. The main drawbacks of Chen et al. proposal are that the redemption protocol of their system is inefficient, and that no formal security model is proposed. In this paper, we consequently propose a formal security model for coupon systems and design a practical multi-coupon system with new features: the quantity of single coupons in a multi-coupon is not defined by the system and the value of each coupon is chosen in a predefined set of values.
**Keywords.** Electronic coupons, e-cash, security model, proof of knowledge.

## 1 Introduction

The issues of *electronic money* [13, 16, 20, 17, 18, 7, 9] and *electronic coupons* [21] are closely related since both are electronic data for payment. The former involves a *Bank* $\mathcal{B}$, a *User* $\mathcal{U}$ and a *Merchant* $\mathcal{M}$; $\mathcal{B}$ delivers electronic coins to $\mathcal{U}$, later $\mathcal{B}$ accepts these coins from $\mathcal{M}$ and in exchange credits the banking account of $\mathcal{M}$, $\mathcal{U}$ withdraws electronic coins from $\mathcal{B}$ and spends them to get goods or services delivered by $\mathcal{M}$, and $\mathcal{M}$ deposits the coins at the bank $\mathcal{B}$. The latter involves a *Service Provider* $\mathcal{SP}$ playing both the roles of $\mathcal{B}$ and $\mathcal{M}$, and a *User* $\mathcal{U}$ that withdraws electronic coupons from the $\mathcal{SP}$ and later redeems these coupons to get an access to specific services offered by the $\mathcal{SP}$.

The usually required security properties of electronic coin systems and those of electronic coupons systems are closely related. For instance, the privacy of the users must be protected, i.e. , it must be impossible to link a withdrawal protocol with a user identity as well as to link two spending/redemption protocols, and it must be impossible to link a spending/redemption protocol to a withdrawal protocol (except for the owner of the coin/coupon).

As it is easy to duplicate electronic data, an electronic payment system requires a mechanism that prevents a user from spending the same electronic data (e.g. a coin or a coupon) twice. The problem of detecting the double-redemption of a coupon is simpler than the problem of detecting the double-spending of a coin. Indeed, in a coupon system, every coupon is redeemed to the service provider that has previously delivered the coupon to a user; the service provider can then easily check the redeemed coupons database in order to detect a double-redemption. In an electronic coin system, the merchant cannot detect a double-spending during a payment protocol since the coins delivered by the bank can be spent at several merchants. Then, the detection of a double-spending is done by the bank.

For a practical use, it is important to consider the efficiency of each protocol of the electronic coin/coupon scheme. For instance, the withdrawal of $m$ coins/coupons should be more efficient than $m$ executions of the withdrawal protocol of one coin/coupon; an efficient solution has been recently proposed [9]. In the same way, the spending/redemption of $m$ coins/coupons should be more efficient than $m$ executions of the spending/redemption protocol; this is still an open problem. Another practical property that should be considered is the size of the electronic wallet/multi-coupon.

In real life, coupons are widely used by vendors. For instance gift certificates are useful means to attract the attention of potential customers. Due to the diversification of the activities of more and more shops, it becomes common that a vendor gives to customers a money-off coupon book with coupons of different values or dedicated to different parts of the goods shop. Then, an electronic coupon system must not only be secure and efficient, but it should also offer such features of real life multi-coupon systems.

## 1.1 Related works

The coupon system proposed by Chen *et al.* [21] allows to create multi-coupons where a multi-coupon is a set of $m$ coupons ($m$ is a predetermined value of the system) and every coupon has the same value $V$. This system does not require the existence of a trusted third party. The usual security properties required in the context of electronic payment are fulfilled by this coupon scheme, i.e. the unforgeability (of a multi-coupon or of a coupon), the unlinkability (of a withdrawal protocol with a redemption protocol, or between several redemption protocols), and the detection of the double-redemption of a coupon. In [21], a multi-coupon is composed of *non-detachable* coupons (i.e. if a user wants to transfer coupons to another user, she must give all her coupons or nothing). This property can be suitable when coupons are used as drug prescriptions from a doctor. However, this property seems to be inconvenient in many other applications such as movie tickets or reduction tickets, for which a user must be allowed to detach a single coupon from her multi-coupon. The redemption protocol proposed in [21] is not efficient. Indeed, it is based on a proof of OR statement that is proportional to the number of withdrawn coupons and consequently unpractical.

Camenisch *et al.* [9] have recently proposed an efficient *compact e-cash* system[3] that allows a user to withdraw a wallet with $2^\ell$ coins such that the space required to store these coins, and the complexity of the withdrawal protocol are proportional to $\ell$ rather than to $2^\ell$. This scheme fulfills the anonymity and unlinkability properties usually required for electronic cash schemes. The compact e-cash scheme combines Camenisch-Lysyanskaya's signature [8], Dodis-Yampolskiy's verifiable random function (VRF) [24] and an innovative system of serial numbers and security tags. As for the coupon system of Chen *et al.*, the number of coins withdrawn during a withdrawal protocol and the coin values are predetermined by the system. The main drawback of the compact e-cash system is that it does not address the problem of *divisibility*: the property that payments of any amount up to the monetary amount of a withdrawn coin can be made. This functionality is considered by the *divisible* e-cash systems.

In [30, 29], the authors proposed *unlikable divisible* e-cash systems, i.e. schemes allowing a user to withdraw a single coin and next to spend this coin in several times by dividing the value of the coin. The usual properties of anonymity and unlinkability are fulfilled by these unlinkable divisible e-cash schemes. Contrary to the schemes mentioned above, the unlinkable divisible e-cash scheme requires a trusted third party. The scheme of Nakanishi and Sugiyama is less efficient than the compact e-cash scheme since it uses double decker proofs of knowledge that are expensive.

Note that all schemes mentioned above suffer from the fact that it is not possible to choose the number of coins/coupons and to choose the value of each coin/coupon.

## 1.2 Our contribution

We first propose a security model suitable for electronic multi-coupon systems that include the usual security properties, i.e. the unforgeability and the unlinkability but also the propery for a user to split her multi-coupon. In the coupon system of Chen et al., a user can give either her

[3] In [9], an extension of this system provides traceable coins without any trusted third party but this property is not relevant in our context.

whole multi-coupon or nothing. The protection against splitting of a multi-coupon can be suitable when coupons are used such as drug prescriptions from a doctor. However, this protection seems to be unsuitable in many other real life applications such as movie tickets or reduction tickets, for which a user must be allowed to detach a coupon from her multi-coupon and transfer it to another user. Then, we propose a model suitable for electronic multi-coupon systems that allows the transfer of coupons.

We then propose a new multi-coupon scheme that is more efficient than the proposal of Chen et al. [21] and in addition offers the following new features: the number of coupons of a multi-coupon can be chosen during a withdrawal protocol, i.e. the size of the multi-coupons is not predetermined by the system. Moreover, for each coupon of a multi-coupon, its value can be chosen among a set of values during the withdrawal protocol; the set of possible values is predetermined by the system. The owner of a multi-coupon can redeem each coupon of her multi-coupon to the appropriate service provider. Furthermore, the owner of a multi-coupon can give a part of her multi-coupon to another user, which means that, a first user can transfer a set of coupons to a second user and then the first user looses the possibility to redeem the coupons she gave and the second user can redeem only the coupons she received. Our redemption protocol is based on a proof of the OR statement that is only proportional to the logarithm of the maximum number of withdrawn coupons, which is far more efficient than the one of Chen et al. [21].

Very recently, some of the ideas present in this paper have been independently proposed by Nguyen [31].

### 1.3 Organization of the paper

This paper is organized as follows. Section 2 describes the security model and requirements for a multi-coupon system. In Section 3, we list and describe the cryptographic tools we need. Section 4 is the main one: it contains the new multi-coupon system. In Section 5, we study the security of our scheme and Section 6 compares it to Nguyen's coupon system. Section 7 concludes this paper.

## 2 Security Model

An electronic coupon system involves a service provider and several users. The Service Provider is denoted by $\mathcal{SP}$ and a user by $\mathcal{U}$. The set of authorized values for coupons is $\mathcal{V} = \{V_1, \dots, V_n\}$. A coupon $C$ is formed by an identifier $I_C$ and a value $V_i \in \mathcal{V}$. A multi-coupon is formed by a multi-coupon identifier $I$ and the set $\mathcal{S} = \{(J_i, V_i); i \in [1, n]\}$ where $J_i$ is the number of coupons of value $V_i$. We set $\mathcal{J}_i = \{0, \dots, J_i - 1\}$.

### 2.1 Algorithms

– `ParamKeyGen`: a probabilistic algorithm taking as input the security parameter $k$. This algorithm outputs some secret parameters *sParams* and some public parameters *pParams* including the authorized values of the coupons $\mathcal{V} = \{V_1, \dots, V_n\}$.

– `SPKeyGen`: a probabilistic algorithm executed by $\mathcal{SP}$ taking as inputs the security parameter $k$ and the parameters of the system *sParams* and *pParams*. This algorithm outputs the key pair $(sk_{\mathcal{SP}}, pk_{\mathcal{SP}})$ of $\mathcal{SP}$.

– `Withdraw`: an interactive protocol between the service provider $\mathcal{SP}$ taking as inputs $(sk_{\mathcal{SP}}, pk_{\mathcal{SP}})$ and *pParams*, and a user $\mathcal{U}$ taking as inputs $pk_{\mathcal{SP}}$ and *pParams*. For every $i \in [1, n]$, the user chooses the number $J_i$ of coupons of value $V_i$ she wants to withdraw. At the end of the protocol, the user's output is the multi-coupon, i.e. an identifier $I$ and the set $\mathcal{S} = \{(J_i, V_i); i \in [1, n]\}$, or an error message. The Service Provider's output is its view $\mathcal{V}_{\mathcal{SP}}^{\texttt{Withdraw}}$ of the protocol.

– `Redeem`: an interactive protocol between a user $\mathcal{U}$, taking as inputs a multi-coupon, i.e. an identifier $I$ and the set $\mathcal{S} = \{(J_i, V_i); i \in [1, n]\}$, the public key $pk_{\mathcal{SP}}$ and *pParams*, and the service provider $\mathcal{SP}$, taking as inputs the public key $pk_{\mathcal{SP}}$ and *pParams*. The user $\mathcal{U}$ chooses the value $V_j$ of the coupon she wants to redeem. At the end of the protocol, the Service Provider $\mathcal{SP}$ obtains from the User $\mathcal{U}$ a coupon $C$ of value $V_j$ with a proof of validity and outputs its view $\mathcal{V}_{\mathcal{SP}}^{\texttt{Redeem}}$ of the protocol. $\mathcal{U}$ outputs an updated multi-coupon, i.e. the identifier $I$ and the set $\{(J'_i, V_i); i \in [1, n]\}$ where $J'_j = J_j - 1$ and $J'_i = J_i$, $i \in [1, n]$ and $i \neq j$, or an error message.

– `Transfer`: an interactive protocol between a user $\mathcal{U}_1$, taking as inputs a multi-coupon, i.e. an identifier $I$ and the set $\mathcal{S} = \{(J_i, V_i); i \in [1, n]\}$, the public key $pk_{\mathcal{SP}}$ and *pParams*, and a second user $\mathcal{U}_2$ taking as inputs $pk_{\mathcal{SP}}$ and *pParams*. For every $i \in [1; n]$, the user $\mathcal{U}_1$ chooses the number $J'_i$, $J'_i \leq J_i$, of coupons of value $V_i$ she wants to transfer to $\mathcal{U}_2$. At the end of the protocol, the user $\mathcal{U}_2$ outputs a new multi-coupon, i.e. an identifier $I'$ and the set $\{(J'_i, V_i); i \in [1, n]\}$, and the user $\mathcal{U}_1$ outputs an updated multi-coupon, i.e. the identifier $I$ and the set $\{(J_i - J'_i, V_i); i \in [1, n]\}$, or an error message.

## 2.2 A formal model

In this section, we propose a formal model for secure multi-coupon systems. A valid coupon is a coupon obtained from a valid `Withdraw` or `Transfer` protocol and it has not been previously redeemed.

– **Correctness**: if an honest user $\mathcal{U}$ runs `Withdraw` with an honest Service Provider $\mathcal{SP}$, then neither will output an error message; if an honest user $\mathcal{U}$ runs `Redeem` with an honest service provider $\mathcal{SP}$, then $\mathcal{SP}$ accepts the coupon if it is valid; if an honest user $\mathcal{U}_1$ runs `Transfer` with an honest user $\mathcal{U}_2$, then $\mathcal{U}_2$ gets a valid coupon (possibly by assuming that $\mathcal{SP}$ is honest).

– **Unforgeability**: from the Service Provider's point of view, what matters is that no coalition of users can ever spend more coupons than they withdrew. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine. At the begining of the game, $\mathcal{A}$ is given the public key $pk_{\mathcal{SP}}$ and the public parameters *pParams* of the system. Furthermore, at any time during the game:
  1. $\mathcal{A}$ can execute in a concurrent manner `Withdraw` protocols with honest service providers,
  2. $\mathcal{A}$ can execute `Redeem` protocols with honest service providers,
  3. $\mathcal{A}$ can execute `Transfer` protocols with honest users playing the role of $\mathcal{U}_1$ or $\mathcal{U}_2$.

  At some point of the game, the adversary $\mathcal{A}$ can legitimately extract, from these protocols, a list $\mathcal{L}$ of valid coupons $C$ with identifiers $I$'s. At the end of the game, $\mathcal{A}$ outputs a coupon $C \notin \mathcal{L}$ and a `Redeem` protocol (or a `Transfer` protocol) is played by $\mathcal{A}$ with an honest service provider $\mathcal{SP}$ (resp. an honest user $\mathcal{U}$ playing the role of $\mathcal{U}_2$).
  We require that for every adversary playing the previous game, the probability that the honest Service Provider $\mathcal{SP}$ (resp. the honest user $\mathcal{U}$ playing the role of $\mathcal{U}_2$) accepts the `Redeem` protocol (resp. the `Transfer` protocol) is negligible.

– **Unlinkability**: from the privacy point of view, what matters to users is that the service provider, even cooperating with any collection of malicious users, cannot learn anything about the user's spendings other than what is available from side information from the environment. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine. At the begining of the game, $\mathcal{A}$ is given the key pair $(pk_{\mathcal{SP}}, sk_{\mathcal{SP}})$ of the Service Provider $\mathcal{SP}$ and the public parameters *pParams* of the system. Furthermore, at any time during the game:
  1. $\mathcal{A}$ can execute in a concurrent manner `Withdraw` protocols with honest users,
  2. $\mathcal{A}$ can execute `Redeem` protocols with honest users,
  3. $\mathcal{A}$ can execute `Transfer` protocols with honest users playing the role of $\mathcal{U}_1$ or $\mathcal{U}_2$.

  At some point of the game, the adversary $\mathcal{A}$ outputs two views $\mathcal{V}_{\mathcal{A}}^{\texttt{Withdraw}_1}$ and $\mathcal{V}_{\mathcal{A}}^{\texttt{Withdraw}_2}$ of previously executed `Withdraw` protocols. Then, for the two challenged withdrawn multi-coupon, the adversary outputs a value $V_i$ and the rank $j \in \mathcal{J}_i$ of a coupon that has not been

already redeemed. We require that these two coupons must not be redeemed by the adversary. A further step of the game consists in choosing secretly and randomly a bit $b$. Then, a `Redeem` protocol (or a `Transfer` protocol) is played by $\mathcal{A}$ with the owner of the multi-coupon outputted from $\texttt{Withdraw}_b$. Finally, $\mathcal{A}$ outputs a bit $b'$.
We require that for every adversary playing the previous game, the success probability that $b = b'$ differs from $1/2$ by a fraction that is at most negligible.

## 2.3 Comparison between our security model and Chen et al.'s security model

Let us now show that our formulation is strong enough to capture all informal security requirements introduced in [21].

**Unforgeability.** Chen *et al.* defined the unforgeability as the infeasibility to create new multi-coupons, to increase the number of unspent coupons, or to reset the number of spent coupons. In addition, Chen *et al.* defined a property called *redemption limitation* that consists in limiting the number of times by at most $m$ that a service provider accepts an $m$-redeemable coupon $M$. The property of *redemption limitation* means that the user is not able to increase the quantity of coupons contained in her multi-coupon, that is, the user is not able to create a new coupon in her multi-coupon. In our security model, the property of *unforgeability* includes the property of *redemption limitation.*

**Double-redemption detection.** The property of *double-redemption detection* is defined in the security model of Chen *et al.* However, in the context of coupon systems, this property is useless. Indeed, before accepting a coupon, a service provider checks that the coupon is fresh, i.e. the coupon has not been redeemed before. Then, a double-redemption is impossible. We consequently include the impossibility to use twice the same coupon in the correctness of the system.

**Unlinkability and minimum disclosure.** The property of unlinkability is similar of those given in [21]. Here, the unlinkability must be ensured between a withdrawal protocol and a redemption protocol, between a withdrawal protocol and a transfer protocol, between a redemption protocol and a transfer protocol, between two redemption protocols and between two transfer protocols.

The property of minimum disclosure defined by Chen *et al.* is that the number of unspent coupons cannot be inferred from any redemption protocol run. Chen *et al.* separate the property of minimum disclosure from the property of unlinkability. However, since the minimum disclosure property is included in the unlinkability property, we do not keep the separation of these two properties.

**Coupon transfer property / protection against splitting.** The main difference between the issues of our coupon system and Chen *et al.*'s is the property of transferability or untransferability.

It is trivially not possible to prevent a user to give all her multi-coupon to another user. Beyond that, a first possibility, which was chosen by Chen *et al.*, consists in preventing a user to give a part of her multi-coupon to another user without giving her whole multi-coupon, i.e. protect a multi-coupon system against splitting. The protection against splitting is defined in [21] as follows: a coalition of customers $\mathcal{U}_i$ should not be able to split an $m$-redeemable multi-coupon $M$ into (disjoint) $s_i$-redeemable shares $M_i$ with $\sum_i s_i \leq m$ such that $M_i$ can only be redeemed by customer $\mathcal{U}_i$ and none of the other customers $\mathcal{U}_j$, $j \neq i$, or a subset of them is able to redeem $M_i$ or a part of it.

Chen *et al.* defined a *weak protection against splitting* property, assuming that users trust each other not to spend (part of) the multi-coupon they have not. With this assumption, user $\mathcal{U}_1$ (resp. is $\mathcal{U}_2$) is sure that user $\mathcal{U}_2$ (resp. $\mathcal{U}_1$) will not use one of the coupon of the multi-coupon $\mathcal{C}'$ (resp. $\hat{\mathcal{C}}$).

A second possibility, that we adopt in this paper, is to permit the splitting of a multi-coupon by adding a new algorithm called `Transfer` as defined above. A user $\mathcal{U}_1$ with the coupons $\mathcal{C} = \{C_0, \ldots, C_{m-1}\}$ can transfer to a user $\mathcal{U}_2$ part of $\mathcal{C}$. At the end of the protocol, $\mathcal{U}_1$ obtains the coupons $\mathcal{C}'$ and $\mathcal{U}_2$ obtains the coupons $\hat{\mathcal{C}}$ such that $\hat{\mathcal{C}} \cup \mathcal{C}' = \mathcal{C}$ and $\hat{\mathcal{C}} \cap \mathcal{C}' = \emptyset$.
In this paper, we consequently add an optional secure `Transfer` algorithm that implies an honest service provider during the `Transfer` algorithm which is reponsible for the creation of two new multi-coupons $\mathcal{C}'$ and $\hat{\mathcal{C}}$ from $\mathcal{C}$.

## 3 Useful tools

In this section, we first introduce the notations and the complexity assumptions that we will use all along the paper. We next present some cryptographic tools: proofs of knowledge, a type of signature schemes introduced by Camenisch and Lysyanskaya and the Dodis-Yampolskiy pseudorandom function.

### 3.1 Notation

Throughout the paper, the symbol $\|$ denotes the concatenation of two strings. The notation "$x \in_R E$" means that $x$ is chosen uniformly at random from the set $E$. For an integer $p$, $\mathbb{Z}_p$ denotes the residue class ring modulo $p$ and $\mathbb{Z}_p^*$ the multiplicative group of invertible elements in $\mathbb{Z}_p$. $\mathcal{G}$ denotes a cyclic group. $PK(\alpha : f(\alpha, \ldots))$ will denote a proof of knowledge of a value $\alpha$ that verifies the predicate $f$. $PedCom(x_1, \ldots, x_l)$ is the Pedersen commitment on values $x_1, \ldots, x_l$. Other notations and definitions will be set as needed.

### 3.2 Complexity assumptions

The security of our coupon system is based on the following assumptions.

**Flexible RSA assumption [2, 26]:** given an RSA modulus $n$ of special form $pq$, where $p = 2p'+1$ and $q = 2q' + 1$ are safe primes, and a random element $g \in \mathbb{Z}_n^*$, it is hard to output $h \in \mathbb{Z}_n^*$ and an integer $e > 1$ such that $h^e \equiv g \mod n$.

**$y$-Strong Diffie-Hellman assumption ($y$-SDH) [5]:** given a random generator $g \in \mathcal{G}$ where $\mathcal{G}$ has prime order $p$, and the values $(g, g^x, \ldots, g^{x^y})$, it is hard to compute a pair $(c, s)$ such that $s^{x+c} = g$.

**$y$-Decisional Diffie-Hellman Inversion assumption ($y$-DDHI) [4]:** given a random generator $g \in \mathcal{G}$ where $\mathcal{G}$ has prime order $p$ and the values $(g, g^x, \ldots, g^{x^y})$ for a random $x \in \mathbb{Z}_p$, and a value $R \in \mathcal{G}$, it is hard to decide if $R = g^{1/x}$ or not.

### 3.3 Proofs of knowledge

The zero-knowledge proofs of knowledge above are constructed over a cyclic group $\mathcal{G} =< g >$ either of prime order $p$ or of unknown order[4](but where the bit-length of the order is $l_{\mathcal{G}}$). The base of each construction is either the Schnorr authentication scheme [37] or the GPS authentication scheme [27, 34]. These are interactive proofs of knowledge where the prover sends a commitment and then responds to a challenge from the verifier.

[4] Fujisaki and Okamoto [26] showed that, under the Flexible RSA Assumption, the standard proofs of knowledge that work for a group of known order are also proofs of knowledge in this setting.

**Proof of knowledge of a representation.** A proof of knowledge of a representation $(x_1, \ldots, x_k)$ of $C$ in base $(g_1, \ldots, g_k)$, that is, such that $C = \prod_{i=1}^{k} g_i^{x_i}$, is done using an extension of the proof of knowledge of a discrete logarithm modulo a prime [37] or a composite [27, 34]. This interactive proof is denoted by

$$PK(\alpha_1, \ldots, \alpha_k / C = \prod_{i=1}^{k} g_i^{\alpha_i}).$$

**Proof of equality of two known representations.** It is possible to prove the equality of representations of elements from two possibly different groups $\mathcal{G}_1$ and $\mathcal{G}_2$. Let $(g_1, \ldots, g_k) \in \mathcal{G}_1$ and $(h_1, \ldots, h_k) \in \mathcal{G}_2$, let $C_1 \in \mathcal{G}_1$ and $C_2 \in \mathcal{G}_2$ and let $(x_1, \ldots, x_k)$ be the representation of $C_1$ in base $(g_1, \ldots, g_k)$ and of $C_2$ in base $(h_1, \ldots, h_k)$. This proof can be done by using [19, 11]. This interactive proof is denoted by

$$PK(\alpha_1, \ldots, \alpha_k / C_1 = \prod_{i=1}^{k} g_i^{\alpha_i} \wedge C_2 = \prod_{i=1}^{k} h_i^{\alpha_i}).$$

**Proof of the OR statement** Let $g_1, \ldots, g_l$ be some generators of a group $G$. The proof of the OR statement consists in proving, for some commitments $C_1, \ldots, C_k$ where $C_i = \prod_{j \in J_i} g_j^{x_{ij}}$ for all $i \in [1, k]$ where $J_i \subseteq [1, l]$, the knowledge of at least one tuple $\{x_{ij}; j \in J_i\}$. Such a proof can be found in [22, 36]. This interactive proof is denoted by

$$PK(\{\alpha_{ij}; j \in J_i\} / \bigvee_{i=1}^{k} C_i = \prod_{j \in J_i} g_j^{\alpha_{ij}}).$$

**Proof that a committed value lies in an interval.** A proof that a committed value lies in an interval, that is the proof of knowledge of $(x, r)$ such that $C = g^x h^r$ and $0 \leq x < a$ with $C, g, h \in \mathcal{G}$ and $a = 2^l$ a defined integer, can be done using [6, 11, 14]. This interactive proof is denoted by

$$PK(\alpha, \beta / C = g^\alpha h^\beta \wedge 0 \leq \alpha < a).$$

In this paper, we will be in the case where the committed value is small. It is then possible to use another technique, secure under the discrete logarithm assumption, that consists in using its binary representation [3]. We thus obtain the procedure described as follows.
We consider the binary decomposition of $x$: $x = x_0 + x_1 2 + \ldots + x_{l-1} 2^{l-1}$.

1. The prover randomly chooses $r, r_0, \ldots, r_{l-1} \in_R \mathbb{Z}_p$ and computes

$$\begin{aligned} C &= g^x h^r \\ C_0 &= g^{x_0} h^{r_0} \\ C_1 &= g^{x_1} h^{r_1} \\ &\ldots \\ C_{l-1} &= g^{x_{l-1}} h^{r_{l-1}} \\ \tilde{C} &= \prod_{i=0}^{l-1} C_i^{2^i} \end{aligned}$$

   Note that the element $\tilde{C}$ can be computed by both the prover and the verifier. Moreover, note that we have $\tilde{C} = g^{\tilde{x}} h^{\tilde{r}}$ and consequently we have $C\tilde{C}^{-1} = g^{x-\tilde{x}} h^{r-\tilde{r}}$.
2. The prover and the verifier then make the following interactive proof of knowledge

$$\begin{gathered} PK(\alpha, \beta, \gamma_0, \ldots, \gamma_{l-1}, \delta / \\ C_0 = h^{\gamma_0} \vee C_0/g = h^{\gamma_0} \wedge \ldots \wedge \\ C_{l-1} = h^{\gamma_{l-1}} \vee C_{l-1}/g = h^{\gamma_{l-1}} \wedge \\ C = g^\alpha h^\beta \wedge C\tilde{C}^{-1} = h^\delta) \end{gathered}$$

By proving that she knows the discrete logarithm of $C\tilde{C}^{-1}$ in base $h$ which is $r - \tilde{r}$, the prover has proved that $x = \tilde{x}$.

**Proof that a committed value is less than another committed value.** A proof that a committed value is less than another committed value consists in proving that $0 \leq x < y$ where $x$ and $y$ are committed with $C = g^x h^r$ and $D = g^y h^w$. This interactive proof is denoted by

$$PK(\alpha, \beta, \gamma, \delta / C = g^\alpha h^\beta \wedge D = g^\gamma h^\delta \wedge 0 \leq \alpha < \gamma).$$

In our case, $x$ and $y$ are $l$-bit integers with $l$ relatively small (see below), that is $x = x_0 + x_1 2 + \ldots + x_{l-1} 2^{l-1}$ and $y = y_0 + y_1 2 + \ldots + y_{l-1} 2^{l-1}$. The proof can consequently be done as follows.

- Using all the possibilities of inequality.
  1. The prover randomly chooses $r, r_0, \ldots, r_{l-1} \in_R \mathbb{Z}_p$, $w, w_0, \ldots, w_{l-1} \in_R \mathbb{Z}_p$ and computes

$$\begin{aligned} C &= g^x h^r & D &= g^y h^w \\ C_0 &= g^{x_0} h^{r_0} & D_0 &= g^{y_0} h^{w_0} \\ C_1 &= g^{x_1} h^{r_1} & D_1 &= g^{y_1} h^{w_1} \\ & \ldots \\ C_{l-1} &= g^{x_{l-1}} h^{r_{l-1}} & D_{l-1} &= g^{y_{l-1}} h^{w_{l-1}} \\ \tilde{C} &= \prod_{i=0}^{l-1} C_i^{2^i} & \tilde{D} &= \prod_{i=0}^{l-1} D_i^{2^i} \end{aligned}$$

     Note that the elements $\tilde{C}$ and $\tilde{D}$ can be both computed by the prover and the verifier. Moreover, note that we have $\tilde{C} = g^{\tilde{x}} h^{\tilde{r}}$ and $\tilde{D} = g^{\tilde{y}} h^{\tilde{w}}$ and consequently wehave $C\tilde{C}^{-1} = g^{x-\tilde{x}} h^{r-\tilde{r}}$ and $D\tilde{D}^{-1} = g^{y-\tilde{y}} h^{w-\tilde{w}}$.
  2. Then, the prover and the verifier make the following interactive proof of knowledge

$$\begin{gathered} PK\Big(\alpha, \beta, \gamma_0, \ldots, \gamma_{l-1}, \delta, \epsilon, \zeta, \eta_0, \ldots, \eta_{l-1}, \rho_0, \ldots, \rho_{l-1}, \theta / \\ (C_0 = h^{\gamma_0} \vee C_0/g = h^{\gamma_0}) \wedge \ldots \wedge (C_{l-1} = h^{\gamma_{l-1}} \vee C_{l-1}/g = h^{\gamma_{l-1}}) \wedge \\ (D_0 = h^{\eta_0} \vee D_0/g = h^{\eta_0}) \wedge \ldots \wedge (D_{l-1} = h^{\eta_{l-1}} \vee D_{l-1}/g = h^{\eta_{l-1}}) \wedge \\ C = g^\alpha h^\beta \wedge C\tilde{C}^{-1} = h^\delta \wedge D = g^\epsilon h^\zeta \wedge D\tilde{D}^{-1} = h^\theta \wedge \\ \big((C_{l-1}/D_{l-1} = h^{\rho_{l-1}} \wedge C_{l-2} = h^{\gamma_{l-2}} \wedge D_{l-2}/g = h^{\eta_{l-2}}) \vee \\ (C_{l-1}/D_{l-1} = h^{\rho_{l-1}} \wedge C_{l-2}/D_{l-2} = h^{\rho_{l-2}} \wedge C_{l-3} = h^{\gamma_{l-3}} \wedge D_{l-3}/g = h^{\eta_{l-3}}) \\ \vee \ldots \vee \\ (C_{l-1}/D_{l-1} = h^{\rho_{l-1}} \wedge \ldots \wedge C_1/D_1 = h^{\rho_1} \wedge C_0 = h^{\gamma_0} \wedge D_0/g = h^{\eta_0})\big)\Big). \end{gathered}$$

     This proof contains a number of proof of the OR statement that is in $\mathcal{O}(l)$.
- Using the fact that $y - x - 1 \geq 0$. This proof necessitates that $2^l < p/2$.
  1. The prover randomly chooses $r, r_0, \ldots, r_{l-1} \in_R \mathbb{Z}_p$, $w, w_0, \ldots, w_{l-1} \in_R \mathbb{Z}_p$. We note $u = y - x - 1$ and $u = u_0 + u_1 2 + \ldots + u_{l-1} 2^{l-1}$. The prover then computes

$$\begin{aligned} C &= g^x h^r & D &= g^y h^w \\ C_0 &= g^{x_0} h^{r_0} & D_0 &= g^{u_0} h^{w_0} \\ C_1 &= g^{x_1} h^{r_1} & D_1 &= g^{u_1} h^{w_1} \\ & \ldots \\ C_{l-1} &= g^{x_{l-1}} h^{r_{l-1}} & D_{l-1} &= g^{u_{l-1}} h^{w_{l-1}} \\ \widetilde{C} &= \prod_{i=0}^{l-1} C_i^{2^i} & \widetilde{D} &= \prod_{i=0}^{l-1} D_i^{2^i} \\ \overline{D} &= D/(gC) \end{aligned}$$

Note that the elements $\tilde{C}$, $\tilde{D}$ and $\bar{D}$ can be computed by the prover and the verifier. Moreover, note that we have $\overline{D} = g^{y-x-1}h^{w-r} = g^{u}h^{w-r}$. By noting $\widetilde{C} = g^{\tilde{x}}h^{\tilde{r}}$ and $\widetilde{D} = g^{\tilde{u}}h^{\tilde{w}}$, we consequently obtain that $C\widetilde{C}^{-1} = g^{x-\tilde{x}}h^{r-\tilde{r}}$ and that $\overline{D}\widetilde{D}^{-1} = g^{u-\tilde{u}}h^{w-r-\tilde{w}}$.

2. Then, the prover and the verifier make the following interactive proof of knowledge

$$\begin{gathered} PK\Big(\alpha, \beta, \gamma_0, \ldots, \gamma_{l-1}, \delta, \epsilon, \zeta, \eta_0, \ldots, \eta_{l-1}, \theta, \rho, \iota / \\ (C_0 = h^{\gamma_0} \vee C_0/g = h^{\gamma_0}) \wedge \ldots \wedge (C_{l-1} = h^{\gamma_{l-1}} \vee C_{l-1}/g = h^{\gamma_{l-1}}) \wedge \\ (D_0 = h^{\eta_0} \vee D_0/g = h^{\eta_0}) \wedge \ldots \wedge (D_{l-1} = h^{\eta_{l-1}} \vee D_{l-1}/g = h^{\eta_{l-1}}) \wedge \\ C = g^{\alpha}h^{\beta} \wedge C\widetilde{C}^{-1} = h^{\delta} \wedge D = g^{\epsilon}h^{\zeta} \wedge \overline{D} = g^{\rho}h^{\iota} \wedge \overline{D}\widetilde{D}^{-1} = h^{\theta}\Big). \end{gathered}$$

This proof is more efficient than the previous but needs either that $2^l < p/2$ (which is note very restrictive in many cases[5]) or the Flexible RSA assumption, that is working in a group of unknown order.

One may use Boudot's proof [6] but this implies necessarily the use of a group of unknown order, and consequently larger parameters (e.g. exponent of size 1024 bits instead of 160 bits in our case). Thus, even if Boudot's proof is proportional to $\mathcal{O}(1)$ w.r.t. the size of $x$ and $y$, instead of $\mathcal{O}(l)$ for us, the value of $l$ will be smaller enough in practice to make Boudot's proof less efficient.

## 3.4 Camenisch-Lysyanskaya type signature schemes with Pedersen commitment

The Pedersen commitment scheme [33] permits a user to commit to some values $x_1, \ldots x_l \in \mathbb{Z}_p$ without revealing them, using some public elements of a cyclic group $\mathcal{G}$ of prime order $p$ with generators $(g_1, \ldots, g_l)$. To do that, the user computes the commitment $C = \prod_{i=1}^{l} g_i^{x_i}$. Such commitment is secure under the Discrete Logarithm assumption.

Camenisch et Lysyanskaya [10] have proposed various signature schemes, called CL signature schemes for short, based on Pedersen's scheme to which they add some specific protocols:

- an efficient protocol between a user and a signer that permits the user to obtain from the signer a signature $\sigma$ of some commitment $C$ on values $(x_1, \ldots, x_l)$ unknown from the signer. The latter computes $\texttt{CLSign}(C)$ and the user obtains $\sigma = \texttt{Sign}(x_1, \ldots, x_l)$.
- an efficient proof of knowledge of a signature of some committed values. The proof is divided into two parts: the computation of a witness, denoted *witness*$(\sigma)$, and the following proof of knowledge
$$PK(\alpha_1, \ldots, \alpha_l, \beta / \beta = \texttt{Sign}(\alpha_1, \ldots, \alpha_l)).$$

Such signature schemes verifies the standard definitions of digital signature schemes in terms of security, that is, verifies the following properties:

- **Correctness**: if a message $m$ is in the message space for a given public key $pk$, and $sk$ is the corresponding secret key, then the output of the signature algorithm on $m$ using $sk$ will always be accepted by the verification algorithm.
- **Unforgeability**: even if an adversary has oracle access to the signing algorithm which provides signatures on messages of the adversary's choice, the adversary cannot create a valid signature on a message not explicitly queried.

These constructions are close to group signature schemes and it seems possible to easily proceed from one to another. This is the case of the two following examples, one base on the ACJT signature scheme [1] and the other base of the BBS one [5].

[5] This restriction does not permit an attacker to use its knowledge of the order $p$ of $g$ to use the representation between 0 and $p$ of a negative integer.

**The ACJT signature scheme.** The first one is derived from the group signature of [1] and is secure under the Flexible RSA assumption and works as follows.

- *Public parameters.* The signer chooses a safe $RSA$ modulus $n = pq$, where $p = 2p' + 1$, $q = 2q' + 1$ and $(l + 2)$ random elements of $QR(n)$: $a_0, \ldots, a_{l+1}$. The public key is $PK = (n, a_0, \ldots, a_{l+1})$ and the secret key is the factorization of $n$.
- *Signing algorithm.* To sign a block of messages $m_1, \ldots, m_l$, the signer chooses a random prime number $e$ and a random prime number $s$. Then, she computes the value $A$ such that
$$A^e = a_0 \prod_{i=1}^{l} a_i^{m_i} a_{l+1}^s \pmod n$$
The signature of $m = (m_1, \ldots, m_l)$ is a tuple $(A, s, e)$.
- *Verification algorithm.* This signature can be verified by anyone by checking that $A^e = a_0 \prod_{i=1}^{l} a_i^{m_i} a_{l+1}^s \pmod n$.
- *Proof of knowledge of a signature.* A proof of knowledge of a signature $(A, s, e)$ of a message $m_1, \ldots, m_l$ is denoted by
$$PK(\alpha_1, \ldots, \alpha_l, \beta, \gamma, \delta/(\beta, \gamma, \delta) = \mathtt{Sign}(\alpha_1, \ldots, \alpha_l))$$
$$PK(\alpha_1, \ldots, \alpha_l, \beta, \gamma, \delta/\beta^\delta = a_0 \prod_{i=1}^{l} a_i^{\alpha_i} a_{l+1}^{\gamma})$$
This proof is divided into two parts. The first one consists in computing $witness(A, s, e)$, that is $T_1 = Ag^w$, $T_2 = g^w h^{r_w}$, $T_3 = g^e h^{r_e}$, $T_4 = g^s h^{r_s}$, $T_5 = g^{ew} h^{r_{ew}}$ and $T_6 = T_1^e h^r$ and the second one is the following proof of knowledge.
$$\begin{gathered} PK(\alpha_1, \ldots, \alpha_l, \beta, \gamma, \delta, \epsilon, \zeta, \eta, \theta, \iota, \kappa/ \\ T_6 = T_1^\beta h^\kappa \wedge T_3 = g^\beta h^\eta \wedge \\ T_6/a_0 = \prod_{i=1}^{l} a_i^{\alpha_i} a_{l+1}^{\gamma} g^\zeta h^\kappa \wedge \\ T_2 = g^\delta h^\epsilon \wedge T_4 = g^\gamma h^\theta \wedge \\ T_5 = g^\zeta h^\iota \wedge T_5 = T_2^\beta h^\mu) \end{gathered}$$

**The BBS signature scheme.** The second example is derived from the group signature of [5], secure under the $y$-SDH assumption and works as follows.

- *Public parameter.* The signer chooses a group $\mathcal{G}$ of prime order $p$, a secret $\gamma \in \mathbb{Z}_p$ and random generators $(g_1, g_2, h_1, \ldots, h_l)$ in $\mathcal{G}$. Finally, the signer compute $w = g_2^s$.
- *Signing algorithm.* To sign a block of messages $m_1, \ldots, m_l$, the signer chooses a random prime number $x$ and computes $A$ such that $A = (g_1 \prod_{i=1}^{l} h_i^{-m_i})^{\frac{1}{x+s}}$. The signature on $m = (m_1, \ldots, m_l)$ is a couple $(A, x)$. Such a signature verifies the following equation:
$$e(A, wg_2^x) \prod_{i=1}^{l} e(h_i, g_2)^{m_i} = e(g_1, g_2).$$
- *Verification algorithm.* This signature can be verified by checking that
$$A^{x+s} \prod_{i=1}^{l} h_i^{m_i} = g_1.$$
- *Proof of knowledge of a signature.* A proof of knowledge of a signature $(A, x)$ on a message $m_1, \ldots, m_l$ is denoted by
$$PK(\alpha_1, \ldots, \alpha_l, \beta, \gamma/(\beta, \gamma) = \mathtt{Sign}(\alpha_1, \ldots, \alpha_l))$$
$$PK(\alpha_1, \ldots, \alpha_l, \beta, \gamma/\beta^{\gamma+s} \prod_{i=1}^{l} h_i^{\alpha_i} = g_1)$$
$$PK(\alpha_1, \ldots, \alpha_l, \beta, \gamma/e(\beta, wg_2^\gamma) \prod_{i=1}^{l} e(h_i, g_2)^{\alpha_i} = e(g_1, g_2))$$

This proof is divided into two parts. The first one consists in computing $witness(A, x)$, that is $T = Ah_1^r$. From this equation and the relation $A^{x+s}\prod_{i=1}^{l} h_i^{m_i} = g_1$, we obtain

$$e(T, g_2)^x e(h_1, g_2)^{m_1 - rx} \prod_{i=2}^{l} e(h_i, g_2)^{m_i} e(h_1, w)^{-r} = e(g_1, g_2)e(T, w)^{-1}$$

The second part of the proof consequently consists in the following proof of knowledge

$$\begin{gathered} PK(\alpha_1, \ldots, \alpha_l, \beta, \gamma / \\ e(T, g_2)^{\beta} e(h_1, g_2)^{\alpha_1} \textstyle\prod_{i=2}^{l} e(h_i, g_2)^{\alpha_i} e(h_1, w)^{-\gamma} = e(g_1, g_2)e(T, w)^{-1}) \end{gathered}$$

### 3.5 The Dodis-Yampolskiy pseudorandom function

A cryptographically secure pseudorandom function (PRF) is an efficient algorithm that when given a seed and an argument returns a new string that is undistinguishable from a truly random function. Such function takes as input some public parameters, a seed $s$ and a value $x$ and outputs a pseudorandom value (plus a proof of validity). In our paper, we will use the Dodis-Yampolskiy pseudorandom function [24] which is secure under the $y$-DDHI assumption. In particular, Dodis and Yampolskiy show that their construction verifies the following property.

- **Pseudorandomness**: let an avdersary be a p.p.t. Turing Machine that have access to all public keys of the PRF and that can send some values $x$ to a PRF oracle that sends back the output of the pseudorandom function on this value. The game is then the following. The adversary outputs a random integer $x_0$ for which it has not asked the oracle. After that, a bit $b$ is secretly and randomly chosen. If $b = 0$, the adversary receives in return the output of the PRF on $x_0$. If $b = 1$, the adversary receives in return a random value. The advsersary can then ask again the PRF oracle with inputs different from $x_0$. Finally, the adversary outputs a bit $b'$. We require that the success probability of the adversary in predicting $b$ differs from 1/2 by a fraction that is at most negligible.

The construction of Dodis and Yampolskiy works as follows. Let $\mathcal{G}$ be a group of order $p$, $g$ a generator of $\mathcal{G}$ and $s$ a seed in $\mathbb{Z}_p$. The Dodis-Yampolskiy pseudorandom function $f$ takes as input a $x \in \mathbb{Z}_p$ and outputs $f_{g,s}(x) = g^{\frac{1}{s+x+1}}$.

## 4 Description of the *handy* multi-coupon system

In this section, we present our new construction of a multi-coupon system based on the compact e-cash scheme [9] of Camenisch et al. We first give the general principle of our improvement and then describe all algorithms.

### 4.1 General principle

A user can withdraw a number of coupons of her choice. Futhermore, a user can also choose the value of each coupon from a set of values $\mathcal{V} = \{V_1, \ldots, V_n\}$ predetermined by the service provider. For each possible value $V_i$, the user decides, with the service provider, the number $J_i$ of coupons of value $V_i$ that she withdraws. In our construction, due to the used proof of knowledge, the possible number of coupons she can withdrawn must be less than a fixed value, $2^l$. This is not really restrictive in practice. The numbers $J_1, \ldots, J_n$ are chosen by the user[6], known and signed by the service provider during the withdrawal protocol, but unrevealed during the redemption protocol. Each value $V_i$ is linked to a random value $\tilde{g}_i$ in $\mathcal{G}$ that is used to trace a designated coin. During a redemption protocol of a coupon of value $V_i$, a user chooses a *fresh* integer in the set $\mathcal{J}_i = \{0, \ldots, J_i - 1\}$ in such a way that for each redemption protocol of a coupon of value $V_i$, the

[6] The values $J_1, \ldots, J_n$ can also be chosen by the service provider if required by the application.

user must choose an integer distinct from the ones revealed during previous redemption protocols of coupons of the same value $V_i$. Consequently, we can associate the monetary value of the coupon, the set $\mathcal{J}_i = \{0, \ldots, J_i - 1\}$ and the generator $\tilde{g}_i$ in $\mathcal{G}$.

*Remark 1.* Another solution (not addressed in this paper) is to choose the value $j$ in the set $\mathcal{J} = \{0, \ldots, J_m - 1\}$ in such a way that $\mathcal{J}_1 = \{0, \ldots, J_1 - 1\}$ corresponds to the value $V_1$, $\mathcal{J}_2 = \{J_1, \ldots, J_2 - 1\}$ corresponds to the value $V_2$, etc. and $\mathcal{J}_n = \{J_{n-1}, \ldots, J_n - 1\}$ corresponds to the value $V_n$. All values $J_1, \ldots, J_n$ are chosen by the user, known and signed by the bank but unrevealed during the redemption protocol. This solution is nevertheless less efficient.

### 4.2 Setup

Let $k$ be the security parameter. We consider a group $\mathcal{G}$ of order $p$. $\tilde{g}_1, \ldots, \tilde{g}_n, g, h, h_0, \ldots, h_{n+1}$ are randomly chosen in $\mathcal{G}$. All these data compose the public parameters *pParams* of the system. The service provider $\mathcal{SP}$ computes the key pair $(sk_{\mathcal{SP}}, pk_{\mathcal{SP}})$ of a Camenisch-Lysyanskaya signature scheme that will permit it to sign multi-coupons, using the `CLSign` algorithm (see Section 3.4 for details). The number $2^l$ of coupons a user $\mathcal{U}$ can withdraw for each value $V_i$ must be less than $p/2$, due to the use of the proof that a commited value is less than another commited value described in Section 3.3.

### 4.3 Withdrawal protocol

During a withdrawal protocol, a user $\mathcal{U}$ takes as inputs *pParams* and $pk_{\mathcal{SP}}$ and interacts with a service provider $\mathcal{SP}$, that takes as inputs *pParams* and $(sk_{\mathcal{SP}}, pk_{\mathcal{SP}})$, as follows.

1. $\mathcal{U}$ and $\mathcal{SP}$ both participate to the randomness of the secret $s$. First, $\mathcal{U}$ selects a random value $s' \in \mathbb{Z}_p$, sends to $\mathcal{SP}$ a commitment $C' = PedCom(s', r)$ and the numbers $J_1, \ldots, J_n$ corresponding to the number of coupons of values $V_1, \ldots, V_n$ she wants to withdraw. $\mathcal{SP}$ sends a random $r' \in \mathbb{Z}_p$ and $\mathcal{U}$ can compute the secret $s$ as $s = s' + r'$.
2. $\mathcal{U}$ and $\mathcal{SP}$ run the CL protocol's for obtaining $\mathcal{SP}$'s signature on committed values contained in the commitment $C = PedCom(s, J_1, \ldots, J_n, r)$. As a result, $\mathcal{U}$ obtains $\sigma = \mathtt{Sign}(s, J_1, \ldots, J_n, r)$.
3. $\mathcal{U}$ saves the multi-coupon, i.e. the identifier $I = (s, r, \sigma)$ and the set $\mathcal{S} = \{(J_i, V_i); i \in [1, n]\}$.

The withdrawal protocol is presented in Figure 1.

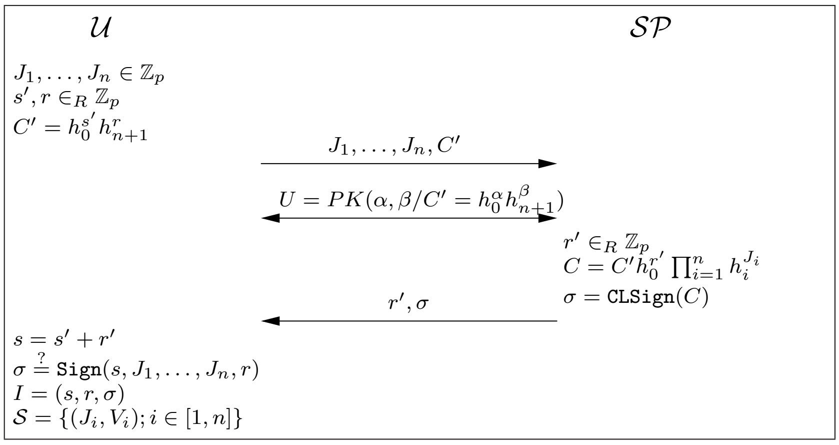

$\mathcal{U}$
$\mathcal{SP}$
$J_1, \ldots, J_n \in \mathbb{Z}_p$
$s', r \in_R \mathbb{Z}_p$
$C' = h_0^{s'} h_{n+1}^r$
$J_1, \ldots, J_n, C'$
$U = PK(\alpha, \beta / C' = h_0^{\alpha} h_{n+1}^{\beta})$
$r' \in_R \mathbb{Z}_p$
$C = C' h_0^{r'} \prod_{i=1}^n h_i^{J_i}$
$\sigma = \mathtt{CLSign}(C)$
$r', \sigma$
$s = s' + r'$
$\sigma \overset{?}{=} \mathtt{Sign}(s, J_1, \ldots, J_n, r)$
$I = (s, r, \sigma)$
$\mathcal{S} = \{(J_i, V_i); i \in [1, n]\}$

**Fig. 1.** Withdrawal protocol

## 4.4 Redemption protocol

When a user wants to redeem a coupon from her multi-coupon $(I, \mathcal{S})$, she first has to choose the value $V_i$ of the coupon she wants to redeem. Then, the user chooses the rank $j$ of the coupon she wants to redeem in the set of all possible coupons of value $V_i$, that is between 0 and $J_i - 1$.
As explained in Figure 2, a redemption protocol consists in the following.

1. Computing the coupon's identifier as the Dodis-Yampolskiy pseudorandom function with seed $s$ and generator $\tilde{g}_i$ associated to the monetary value $V_i$, on the input $j$: $S = \tilde{g}_i^{\frac{1}{s+j+1}}$.
2. A proof of validity of this coupon, that is an interactive proof of knowledge[7] of a $\mathcal{SP}$ signature on the secrets $(s, J_1, \ldots, J_n, r)$, plus a proof that the selected coupon belongs to the set $\mathcal{J}_i = \{0, \ldots, J_i - 1\}$.

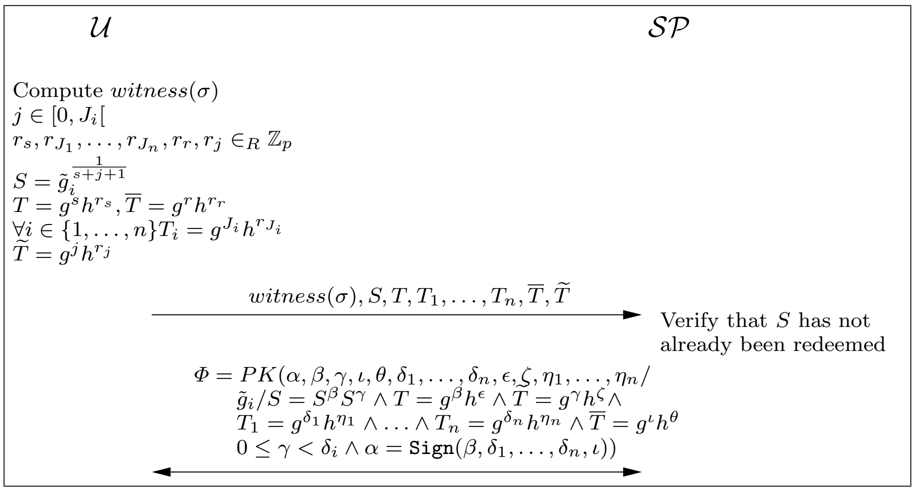

$\mathcal{U}$
$\mathcal{SP}$
Compute $witness(\sigma)$
$j \in [0, J_i[$
$r_s, r_{J_1}, \ldots, r_{J_n}, r_r, r_j \in_R \mathbb{Z}_p$
$S = \tilde{g}_i^{\frac{1}{s+j+1}}$
$T = g^s h^{r_s}, \overline{T} = g^r h^{r_r}$
$\forall i \in \{1, \ldots, n\} T_i = g^{J_i} h^{r_{J_i}}$
$\widetilde{T} = g^j h^{r_j}$
$witness(\sigma), S, T, T_1, \ldots, T_n, \overline{T}, \widetilde{T}$
Verify that $S$ has not already been redeemed
$\Phi = PK(\alpha, \beta, \gamma, \iota, \theta, \delta_1, \ldots, \delta_n, \epsilon, \zeta, \eta_1, \ldots, \eta_n /$
$\tilde{g}_i / S = S^\beta S^\gamma \wedge T = g^\beta h^\epsilon \wedge \widetilde{T} = g^\gamma h^\zeta \wedge$
$T_1 = g^{\delta_1} h^{\eta_1} \wedge \ldots \wedge T_n = g^{\delta_n} h^{\eta_n} \wedge \overline{T} = g^\iota h^\theta$
$0 \leq \gamma < \delta_i \wedge \alpha = \mathtt{Sign}(\beta, \delta_1, \ldots, \delta_n, \iota))$

**Fig. 2.** Redemption protocol

Note that the proof of knowledge $\Phi$ (see Figure 2) includes a challenge $c$ sent by the service provider $\mathcal{SP}$.

*Remark 2.* $S = \tilde{g}_i^{\frac{1}{s+j+1}}$ can also be written $\tilde{g}_i / S = S^s S^j$, which explains the proof of knowledge.

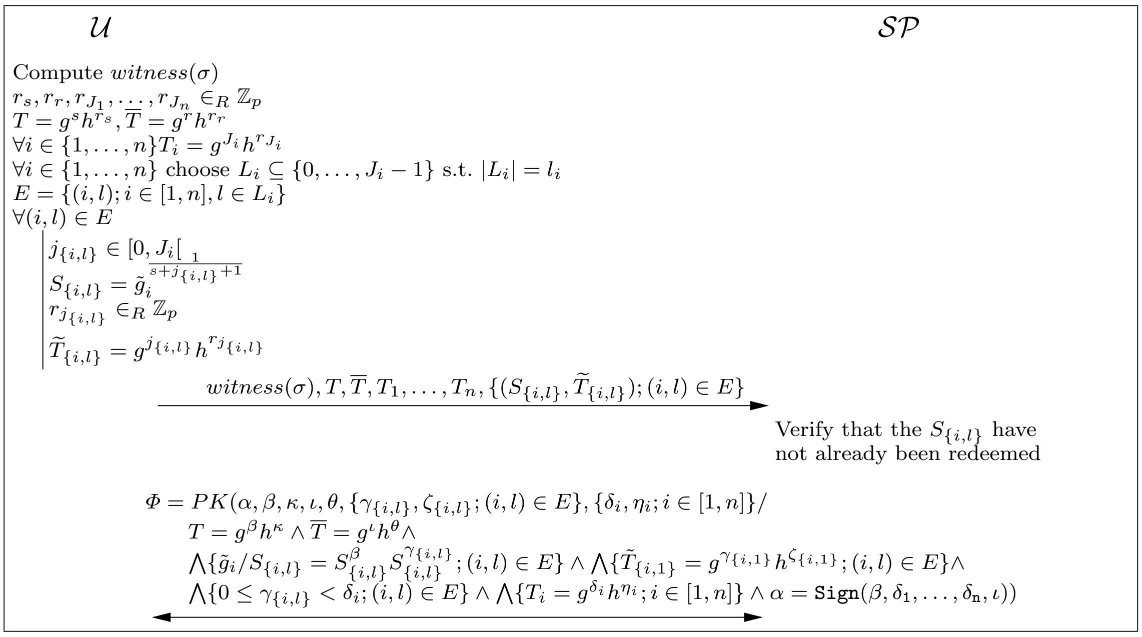
## 4.5 Multi-redemption protocol

The multi-redemption protocol consists in redeeming several coupons of a multi-coupon in a single interactive protocol with $\mathcal{SP}$. The global protocol is more efficient than simply executing the redemption protocol in Figure 2 for each redeemed coupon. In fact, the proof of knowledge of the $\mathcal{SP}$ signature $\sigma = \mathtt{Sign}(s, J_1, \ldots, J_n, r)$ only needs to be done once whereas the computation involving the rank of each redeemed coupon needs to be done for each coupon. The resulting protocol is described in Figure 3.

## 4.6 Transfer protocol

As explained in Section 2.3, it can be interesting to design the possibility for one user $\mathcal{U}_1$ to transfer some coupons of a multi-coupon to another user $\mathcal{U}_2$. A straightforward solution is described in

[7] This proof consequently does not necessitates the Fiat Shamir heuristic and a hash function. Thus, our construction is on the standard model.



$\mathcal{U}$ $\qquad\qquad$ $\mathcal{SP}$

Compute $witness(\sigma)$

$r_s, r_r, r_{J_1}, \ldots, r_{J_n} \in_R \mathbb{Z}_p$

$T = g^s h^{r_s}, \overline{T} = g^r h^{r_r}$

$\forall i \in \{1, \ldots, n\} T_i = g^{J_i} h^{r_{J_i}}$

$\forall i \in \{1, \ldots, n\}$ choose $L_i \subseteq \{0, \ldots, J_i - 1\}$ s.t. $|L_i| = l_i$

$E = \{(i,l); i \in [1,n], l \in L_i\}$

$\forall (i,l) \in E$

$j_{\{i,l\}} \in [0, J_i[$

$S_{\{i,l\}} = \tilde{g}_i^{\frac{1}{s+j_{\{i,l\}}+1}}$

$r_{j_{\{i,l\}}} \in_R \mathbb{Z}_p$

$\widetilde{T}_{\{i,l\}} = g^{j_{\{i,l\}}} h^{r_{j_{\{i,l\}}}}$

$$\xrightarrow{witness(\sigma), T, \overline{T}, T_1, \ldots, T_n, \{(S_{\{i,l\}}, \widetilde{T}_{\{i,l\}}); (i,l) \in E\}}$$

Verify that the $S_{\{i,l\}}$ have not already been redeemed

$$\begin{aligned}\Phi = PK(&\alpha, \beta, \kappa, \iota, \theta, \{\gamma_{\{i,l\}}, \zeta_{\{i,l\}}; (i,l) \in E\}, \{\delta_i, \eta_i; i \in [1,n]\}/\\ &T = g^\beta h^\kappa \wedge \overline{T} = g^\iota h^\theta \wedge\\ &\bigwedge\{\tilde{g}_i / S_{\{i,l\}} = S_{\{i,l\}}^{\beta} S_{\{i,l\}}^{\gamma_{\{i,l\}}}; (i,l) \in E\} \wedge \bigwedge\{\tilde{T}_{\{i,1\}} = g^{\gamma_{\{i,1\}}} h^{\zeta_{\{i,1\}}}; (i,l) \in E\} \wedge\\ &\bigwedge\{0 \le \gamma_{\{i,l\}} < \delta_i; (i,l) \in E\} \wedge \bigwedge\{T_i = g^{\delta_i} h^{\eta_i}; i \in [1,n]\} \wedge \alpha = \mathtt{Sign}(\beta, \delta_1, \ldots, \delta_{\mathtt{n}}, \iota))\end{aligned}$$

$$\longleftrightarrow$$

**Fig. 3.** Multi-redemption protocol

Figure 4 and include the participation of the Service Provider $\mathcal{SP}$. The first step consists for $\mathcal{U}_1$ in choosing the coupons she wants to transfer and to redeem them by interactive with $\mathcal{SP}$. The second step is a withdrawal protocol between the user $\mathcal{U}_2$ and $\mathcal{SP}$ with the quantity and the right values of transfered coupons. At the end of this global protocol, $\mathcal{U}_1$ obtains an updating multi-coupons since she has withdraw some of her coupon. $\mathcal{U}_2$ obtains a new multi-coupon, as after a withdrawal protocol.

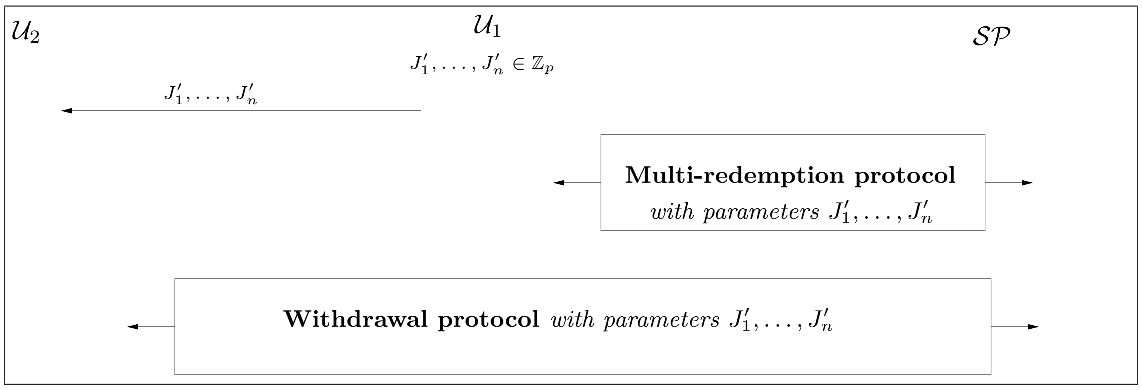

$\mathcal{U}_2$
$\mathcal{U}_1$
$\mathcal{SP}$
$J'_1, \ldots, J'_n \in \mathbb{Z}_p$
$J'_1, \ldots, J'_n$
**Multi-redemption protocol** *with parameters* $J'_1, \ldots, J'_n$
**Withdrawal protocol** *with parameters* $J'_1, \ldots, J'_n$

**Fig. 4.** Transfer protocol

### 4.7 Revocation and expiration date of a multi-coupon

The revocability of a multi-coupon is not a property considered in [21]. However, this property can be added to our scheme. The revocation means that the coupon must not be accepted by the Service Provider if it decides that this multi-coupon is no longer valid. To revoke a multi-coupon, the service provider $\mathcal{SP}$ has to calculate a new key pair $(sk_{\mathcal{SP}}, pk_{\mathcal{SP}})$ and the users have to update $pk_{\mathcal{SP}}$ and their multi-coupon. It consists in revoking the signature made during the corresponding withdrawal protocol. The revocation scheme of our multi-coupon system thus relies on the revocation mechanism of the group signature underlying the CL signature scheme. When using a BBS signature scheme we can use the revocation scheme described in [5]. For an ACJT signature scheme, the revocation can be done as in [8].

We can also add an expiration date to the multi-coupon in case the Service Provider wants to limitate its use. To do so, we simply modify the withdrawal and redemption protocols. During the Withdraw protocol the Service Provider adds to the signature a value which represents the expiration date. Then, during the Redeem protocol, the user proves to the Service Provider that the date contained in her signature is more than the current date.

## 5 Security Arguments

Let us now give the security theorem that our proposal is secure under the definition given above.

**Theorem 1.** *In the standard model, under the y-DDHI assumption and the security assumptions of the used CL signature scheme (Flexible RSA if ACJT and y-SDH if BBS), the multi-coupon system described in Section 4 is secure w.r.t. the security model described in Section 2.*

*Proof.* The proof of the theorem consists in proving that the construction of Section 4 verifies all security properties listed in Section 2.

- **Correctness**: by construction.
- **Unforgeability**: in this part of the proof, according to the construction of the `Transfer` protocol described in Section 4.6, an adversary playing a `Transfer` protocol with an honest user playing the role of $\mathcal{U}_2$ corresponds to the execution of a `Redeem` protocol with an honest service provider. Similarly, an adversary playing a `Transfer` protocol with an honest user playing the role of $\mathcal{U}_1$ corresponds to the execution of a `Withdraw` protocol with an honest service provider.
  The game is as follow. An adversary $\mathcal{A}$ can legitimately extract a list of coupons $\mathcal{L}$. Then, $\mathcal{A}$ outputs a coupon $C \notin \mathcal{L}$ and a `Redeem` protocol is played by $\mathcal{A}$ with an honest service provider $\mathcal{SP}$.
  We require that for every adversary playing the previous game, the probability that the honest service provider $\mathcal{SP}$ accepts the `Redeem` protocol is negligible.
  We show that if there exists such an adversary $\mathcal{A}$ that succeeds in the previous game with non-negligible probability, then we can construct a machine $\mathcal{M}$ with access to $\mathcal{A}$ that can break the CL signature scheme described in Section 3.4. We suppose that the machine $\mathcal{M}$ has access to a CL signature oracle $\mathcal{S}_{CL}$ that takes as input a commitment and that outputs a signature on commited values.
  The keys of the CL signature scheme are chosen and the public key is given to $\mathcal{M}$. $\mathcal{M}$ chooses a group $\mathcal{G}$ or order $p$ and random elements $\tilde{g}_1, \ldots, \tilde{g}_n, g, h, h_0, \ldots, h_{n+1} \in \mathcal{G}$. These values and the CL signature public key are then fed to $\mathcal{A}$. When $\mathcal{A}$ executes a `Withdraw` protocol, the machine $\mathcal{M}$ plays the role of the service provider as follows.
  1. $\mathcal{M}$ first receives the $J_i$'s and the commitment $C'$. It then interacts with $\mathcal{A}$ during the proof of knowledge $U$ and extracts $s'$ and $r$ by rewinding of $\mathcal{A}$.
  2. $\mathcal{M}$ chooses $r \in_R \mathbb{Z}_p$, computes $C$ as explained in Section 4.3 and then asks $\mathcal{S}_{CL}$ a signature on $(s = s' + r, J_1, \ldots, J_n, r)$. $\mathcal{S}_{CL}$ is given the commitment $C$ and outputs the signature $\sigma$.
  3. $\mathcal{M}$ then sends $\sigma$ to $\mathcal{A}$.

  After several executions of `Withdraw` protocols, $\mathcal{A}$ obtains a list of coupons $\mathcal{L}$. Each coupon is associated to a multi-coupon, and then to a CL signature obtains from the signature oracle $\mathcal{S}_{CL}$. Then $\mathcal{A}$ outputs a coupon that does not belong to the list $\mathcal{L}$ and plays a `Redeem` protocol that is accepted with non-negligible probability. For this purpose, $\mathcal{A}$ computes all necessary

data (see Figure 2 for details) and the proof of knowledge $\Phi$ by interacting with $\mathcal{M}$. The latter executes the `Redeem` protocol normaly.
$\mathcal{M}$ uses the proof of knowledge $\Phi$ to extract, in particular, the values $(s, J_1, \ldots, J_n, r, \sigma)$ satisfying the relations embedded into the valid proof of knowledge. Consequently, $\sigma$ is a signature (forgery) in the CL's scheme on the message $(s, J_1, \ldots; J_n, r)$. Since, $\mathcal{A}$ succeeds, this means that $\sigma$ is different from the signatures obtain during the execution of the `Withdraw` protocol and submitted to $\mathcal{S}_{CL}$. As the CL signature scheme is proven secure against adaptive chosen message attacks under the Flexible RSA assumption or the $y$-SDH, it follows that $\mathcal{A}$ cannot succeed with non-negligible probability.
Because our proof requires rewinding to extract $s'$ and $r$ from an adversary $\mathcal{A}$, our theorem is only valid against sequential attacks. Indeed, in a concurrent setting where the attacker is allowed to interact with the service provider in an arbitrarily interleaving manner, our machine may be forced to rewind an exponential number of times. This drawback could be overcome by using for instance well-known techniques [23] which would require from the user to encrypt $s'$ and $r$ in a verifiable manner [12].

– **Unlinkability** (sketch of proof): the game is as follows. Let $\mathcal{A}$ an adversary that has access to the key pair $(sk_{\mathcal{SP}}, pk_{\mathcal{SP}})$. $\mathcal{A}$ outputs two views $\mathcal{V}_{\mathcal{A}}^{\texttt{Withdraw}_1}$ and $\mathcal{V}_{\mathcal{A}}^{\texttt{Withdraw}_2}$ of previously executed `Withdraw` protocols. After that, a bit $b$ is then secretly and randomly chosen. Then, a `Redeem` protocol is played by $\mathcal{A}$ with the owner of the multi-coupon outputted from $\texttt{Withdraw}_b$. Finally, $\mathcal{A}$ outputs a bit $b'$.
We require that for every adversary playing the previous game, the success probability of $\mathcal{A}$ in predicting $b$ differs from 1/2 by a fraction that is at most negligible.
The proof is then divided into two parts:

1. During the first part:
   - $\mathcal{A}$ can execute `Withdraw` protocols playing $\mathcal{SP}$ with honest users. For each `Withdraw` protocol, the user $\mathcal{U}$ and the adversary $\mathcal{A}$ plays a CL signature protocol where $\mathcal{U}$ chooses the values $J_1, \ldots, J_n \in \mathbb{Z}_p$ and $s' \in_R \mathbb{Z}_p$, and $\mathcal{A}$ chooses the value $r' \in_R \mathbb{Z}_p$. For each `Withdraw` protocol, the view of the protocol $\mathcal{V}_{\texttt{Withdraw}}$ is stocked in the list $\mathcal{V}$.
   - $\mathcal{A}$ can execute `Redeem` protocols playing $\mathcal{SP}$ with honest users. For each `Redeem` protocol, the user $\mathcal{U}$ redeems one coupon of her multi-coupon by following the redemption protocol described in Section 4.4.
   - $\mathcal{A}$ can execute `Transfer` protocols playing both $\mathcal{U}_2$ and $\mathcal{SP}$ with honest users. According to the construction of the `Transfer` protocol described in Section 4.6, that comes to say that $\mathcal{A}$ can execute `Redeem` protocols playing $\mathcal{SP}$ with an honest user $\mathcal{U}$.
   - $\mathcal{A}$ can execute `Transfer` protocols playing both $\mathcal{U}_1$ and $\mathcal{SP}$ with honest users. According to the construction of the `Transfer` protocol described in Section 4.6, that comes to say that $\mathcal{A}$ can execute `Withdraw` protocols playing $\mathcal{SP}$ with an honest user $\mathcal{U}$.
2. The work of the adversary is consequently, from the service provider side, to distinguish between two `Redeem` protocols, even having the view of the withdrawal protocols from which the redempted coupons has been withdrawn.

   Suppose that $\mathcal{A}$ succeeds at the game described above with a probability which differ from 1/2 by a significant fraction. There are two possibilities.

   - $\mathcal{A}$ must have linked with a probability that differs from 1/2 by a significant fraction two `Redeem` protocols, the challenged one and the other obtained with an honest user $\mathcal{U}$. In this case, this implies one of the following.
     (a) During a `Redeem` protocol, the user $\mathcal{U}$ proves the knowledge of a CL signature (with Pedersen commitments) without revealing information neither about the values $s, J_1, \ldots, J_n$ that has been previously signed by $\mathcal{A}$, nor about the corresponding signature, due to the security of the used proofs of knowledge (see Section 3.3). Thus, if the adversary wins, it implies that it has learn information about the secret information in the proof of knwoledge. This case happens with negligible

probability under the assumption that the proof is zero-knowledge. We thus need the $y$-DDHI[8] and the Flexible RSA assumptions[9].

(b) Due to the security of Dodis-Yampolskiy, the coupon $S = \tilde{g}_i^{\frac{1}{s+j+1}}$ taking on input a value $s$ and a value $j$ is computationally indistinguishable from another Dodis-Yampolskiy function's output with the same $s$ but another $j$, or with the same $j$ but another $s$ or with two different values $s$ and $j$. Thus, the adversary succeeds with negligible probability under the $y$-DDHI assumption [24].

- $\mathcal{A}$ must have linked with a probability that differs from 1/2 by a significant fraction a `Redeem` protocol and a `Withdraw` protocol obtained with the user $\mathcal{U}$. During a `Withdraw` protocol, $\mathcal{A}$ does not learn anything meaningful about the secret $s$ that is signed due to the CL signatures, the Pedersen commitment and the proofs of knowledge. In fact this value can be information theoretically-hidden from $\mathcal{A}$ by requesting a signature on $(s, J_1, \ldots, J_n, r)$ for a random $r$ [10, 1, 5]. Consequently, after a `Withdraw` protocol the adversary knows only the CL signature and the values $J_i$'s. The adversary then cannot succeed for the following reasons.

(a) The adversary knows which signature is embedded in the proof of knowledge underlying the `Redeem` protocol. This case happens with negligible probability under the assumption that the proof is zero-knowledge. This requires the $y$-DDHI and the Flexible RSA assumptions, for the reasons given above.

(b) The adversary can open a Pedersen commitment on the $J_i$'s. This case happens with negligible probability under the $y$-DDHI assumption[10] [33].

## 6 Recent work on coupon systems

Recently, Nguyen [31] has independently proposed a multi-coupon system and a formal security model. Our model is quite close to Nguyen's, except that we include a transfer protocol, which is not compatible with his property of unsplittability.

As we do in this paper, Nguyen adapted the compact e-cash system [9] to the electronic coupon context. In his adaptation, Nguyen focused on the efficiency of the redemption protocol and consequently had a protocol with constant cost for communication and computation. However the size of the multi-coupon increases proportionally to the number of coupons, whereas in our scheme, the multi-coupon has a small constant size.

Apart from the adaptation of the compact e-cash system, Nguyen also permitted the revocation of a multi-coupon, as we do. He also suggested a solution, different from ours, to permit the user to choose the number of coupons she wants to withdraw. It will be interesting in the future to study the efficiency of these two solutions w.r.t. the size of the multi-coupon, the number of withdrawn coupons and the application (efficiency of withdrawal protocol vs. efficiency of redemption protocol).

Finally, we also add the possibility to have coupons of different values, which is not studied by Nguyen.

## 7 Conclusion

In this paper, we first introduced a strong and formal model suitable for electronic multi-coupon systems. We then proved the existence of a system, meeting our requirements, based on standard

[8] In fact, due to Section 3.3, we need the Discrete Logarithm assumption but if we assume the $y$-DDHI assumption, then it is necessary that the Discrete Logarithm assumption holds.

[9] In fact, the Flexible RSA assumption is only needed if the chosen CL signature scheme is based on ACJT since the proofs that a commited value lies in a (commited) interval are done using the binary representation of the secret(s) (see Section 3.3) and the BBS based CL signature scheme concerns a group of known prime order.

[10] In fact, the Discrete Logarithm assumption.

complexity assumptions, in the standard model. We introduced in the context of electronic coupon schemes the transfer of coupons which seems to be suitable for most of the applications of the real life. Furthermore, our scheme allows a user to choose the number of coupons she wants to withdraw, and the value of each coupon of a multi-coupon is chosen by the user among a set of pre-defined values; as far as we know, our electronic coupon scheme is the first scheme that propose these features. Moreover, the latter improvements can also be used in an electronic cash system such as the compact e-cash of Camenisch *et al.*

It will be useful in the future to design a transfer protocol which does not involve the service provider, as is it closer to reality and consequently more practical. Moreover, the multi-redeem protocol may be run more efficiently, possibly by permitting the computation of coupon identifiers iteratively for each redeemed coupon.

**Acknowledgment:** We would like to thank Jacques Traoré and Marc Girault for their help and anonymous referees for their useful comments. This work is partially supported by the French Ministry of Research RNRT Project "CRYPTO++".

## References

1. Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *ASIACRYPT'00*, pages 255–270, 2000.
2. N. Barić and B. Pfitzmann. Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. In W. Fumy, editor, *Advances in Cryptology - Eurocrypt '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–484. Springer-Verlag, 1997.
3. Mihir Bellare and Shafi Goldwasser. Verifiable partial key escrow. In *ACM Conference on Computer and Communications Security*, pages 78–91, 1997.
4. D. Boneh and X. Boyen. Short Signatures without Random Oracles. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology - Eurocrypt '04*, volume 3027 of *Lecture Notes in Computer Science.* Springer-Verlag, 2004.
5. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Franklin [25], pages 41–55.
6. F. Boudot. Efficient Proofs that a Committed Number Lies in an Interval. In Preneel [35], pages 431–444.
7. Stefan Brands. Untraceable off-line electronic cash in wallets with observers. In D. R. Stinson, editor, *Advances in Cryptology - Crypto '93*, volume 773 of *Lecture Notes in Computer Science*, pages 302–318. Springer-Verlag, 1993.
8. J. Camenisch and A. Lysyanskaya. A Signature Scheme with Efficient Protocols. *Third Conference on Security in Communication Networks - SCN '02*, 2576:268–289, 2002.
9. Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In Ronald Cramer, editor, *Advances in Cryptology - Eurocrypt '05*, volume 3494 of *Lecture Notes in Computer Science*, pages 302–321. Springer-Verlag, 2005.
10. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Franklin [25], pages 56–72.
11. Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number is the product of two safe primes. In J. Stern, editor, *Advances in Cryptology - Eurocrypt '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 107–122. Springer-Verlag, 1999.
12. Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. In Dan Boneh, editor, *Advances in Cryptology - Crypto '03*, volume 2729 of *Lecture Notes in Computer Science*, pages 126–144. Springer, 2003.
13. S. Canard and J. Traoré. On fair e-cash systems based on group signature schemes. *ACISP*, pages 237–2480, 2003.
14. Agnes Hui Chan, Yair Frankel, and Yiannis Tsiounis. Easy come - easy go divisible cash. In Nyberg [32], pages 561–575.
15. D. Chaum, editor. *Advances in Cryptology, Crypto '83*, Lecture Notes in Computer Science. Plenum Publishing, 1984.
16. D. Chaum. Blind Signatures for Untraceable Payments. In *Advances in Cryptology - Crypto '83* [15], page 153.

17. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In S. Goldwasser, editor, *Advances in Cryptology - Crypto '88*, volume 403 of *Lecture Notes in Computer Science*, pages 319–327. Springer-Verlag, 1988.
18. D. Chaum and T. Pedersen. Transferred cash grows in size. In R. A. Rueppel, editor, *Advances in Cryptology - Eurocrypt '92*, volume 658 of *Lecture Notes in Computer Science*, pages 390–407. Springer-Verlag, 1993.
19. D. Chaum and T. Pedersen. Wallet Databases with Observers. In E. F. Brickell, editor, *Advances in Cryptology - Crypto '92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer-Verlag, 1993.
20. David Chaum. Blind signatures systems. In Chaum [15], page 153.
21. Liqun Chen, Matthias Enzmann, Ahmad-Reza Sadeghi, Markus Schneider II, and Michael Steiner. A privacy-protecting coupon system. In *Financial Cryptography*, Lecture Notes in Computer Science, pages 93–108. Springer-Verlag, 2005.
22. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y. Desmedt, editor, *Advances in Cryptology - Crypto '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer-Verlag, 1994.
23. Ivan Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In Preneel [35], pages 418–430.
24. Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In Serge Vaudenay, editor, *Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 416–431. Springer, 2005.
25. M. K. Franklin, editor. *Advances in Cryptology - Crypto 04*, volume 3152 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.
26. E. Fujisaki and T. Okamoto. Statistical Zero-Knowledge Protocols to Prove Modular Polynomial Relations. In B. S. Kaliski Jr., editor, *Advances in Cryptology - Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer-Verlag, 1997.
27. M. Girault. Self-Certified Public Keys. In D. W. Davies, editor, *Advances in Cryptology - Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer-Verlag, 1991.
28. Helena Handschuh, Yiannis Tsiounis, and Moti Yung. Decision oracles are equivalent to matching oracles. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography, Second International Workshop on Practice and Theory in Public Key Cryptography, PKC '99, Kamakura, Japan, March 1-3, 1999, Proceedings*, volume 1560 of *Lecture Notes in Computer Science*, pages 276–289. Springer, 1999.
29. Toru Nakanishi, Mitsuaki Shiota, and Yuji Sugiyama. An efficient online electronic cash with unlinkable exact payments. In *ISC'04*, pages 367–378, 2004.
30. Toru Nakanishi and Yuji Sugiyama. Unlinkable divisible electronic cash. In *ISW*, pages 121–134, 2000.
31. L. Nguyen. Privacy-protecting coupon system revisited. In *Financial Cryptography '06*, Lecture Notes in Computer Science, 2006.
32. K. Nyberg, editor. *Advances in Cryptology - Eurocrypt '98*, volume 1403 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998.
33. T. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Advances in Cryptology - Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer-Verlag, 1992.
34. G. Poupard and J. Stern. Security Analysis of a Practical "on the fly" Authentication and Signature Generation. In Nyberg [32], pages 422–436.
35. B. Preneel, editor. *Advances in Cryptology - Eurocrypt '00*, volume 1807 of *Lecture Notes in Computer Science*. Springer-Verlag, 2000.
36. A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung. On Monotone Formula Closure of SZK. *FOCS 1994*, pages 454–465, 1994.
37. C. P. Schnorr. Efficient Identification and Signatures for Smart Cards. In G. Brassard, editor, *Advances in Cryptology - Crypto '89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer-Verlag, 1990.