情報保護學會論文誌
第11券 第6號, 2001. 12

# 남은 금액을 재사용할 수 있는 오프라인 전자수표시스템

김 상 진*, 오 희 국*

# A New Offline Check System with Spendable Refunds

Sangjin Kim*, Heekuck Oh*

요 약

오프라인 시스템은 은행의 참여 없이 지불을 처리하기 때문에 그 과정에서 발생한 거스름이 화폐의 기능을 갖도록 만드는 것은 어렵다. 대부분의 오프라인 수표시스템은 거스름의 재사용을 고려하지 않고 수표를 두 부분으로 구성하여 하나는 지불에 사용하고 다른 하나는 은행에서 잔액을 환불받을 때 사용하도록 되어있다. 그러나 이 방식은 남은 잔액을 다시 쓸 수 없다는 불편함이 있고, 지불액과 환불액의 보수관계 때문에 수표의 익명성을 저해하는 요인이 있다. 이 논문에서는 수표의 잔액을 다시 쓸 수 있는 오프라인 시스템을 제안한다. 이 시스템은 같은 수표가 사용된 지불행위를 서로 연관시킬 수 없도록 하였으며, 이중사용과 초과사용 등의 부정행위가 발생하면 그 책임자를 밝혀낼 수 있도록 고안하였다. 또한 인출에서 환불까지 소요되는 전체 연산비용은 기존의 수표시스템에서 여러 개의 수표를 인출하여 사용하는 경우보다 적게 들도록 하였다.

ABSTRACT

In an offline system, the bank does not participate in payments. As a result, it is difficult to make the refund spendable. Due to this, current offline systems do not provide spendable refunds. In these systems, a check consists of two parts: a spendable part and a refund part. A client uses the spendable part during the payment phase, and uses the refund part to get the refund for the remainder of the check. Therefore, a client cannot reuse the remaining and must always refund it. Moreover, the relationship between the spent amount and the refund amount can be used to guess which check the client used when the client refunds the remaining. To remedy these problems, we propose a new offline system which allow clients to reuse the remaining values of the check. This system provides unlinkability of the payments made by using a single check. It also provides mechanisms to detect and identify clients who perform misconduct such as double spending and over spending. The required overall computational cost to withdraw, spend and refund a check in our system is lower than using several checks in other offline systems.

**keyword** : *electronic cash, offline check system, reusable refund*

## Ⅰ. 서 론

전자화폐(electronic cash)는 쓰임새에 따라 분류하는 방법이 다양하다. 이러한 분류 가운데 전자화폐를 전자동전(electronic coin)방식[1]과 전자수표(electronic check)방식[2~8]으로 분류하는 것이 있다. 동전방식에서 각 동전은 고정된 액면가를 지니며, 고객은 지불대금에 맞도록 필요한 개수의 동전을 이용하여 지불한다. 따라서 거스름이 필요 없는 지불수단이다. 그러나 지불에 필요한 동전의 수가 많을 경우에는 교환되는 정보와 연산의 양이 증가하기 때문에 효율성이 떨어진다. 또한 고객이 동전을 충분

* 한양대학교 컴퓨터공학과({sangjin, hkoh}@cse.hanyang.ac.kr)

히 가지고 있더라도 지불대금에 정확하게 맞추지 못할 경우에는 지불할 수 없는 불편함을 지니고 있다. 이와는 달리 수표방식에서는 시스템이 정해놓은 지정된 금액의 수표 또는 고객이 원하는 금액의 수표를 인출 받아 사용한다. 이 방식에서는 수표의 액면가보다 적은 금액에 대해서는 항상 지불이 가능하므로 동전방식보다 편리하게 지불할 수 있다. 또한 보통 하나의 수표를 이용하여 지불하게 되므로 액수에 상관없이 교환되는 정보의 양이나 연산의 양이 일정하다는 장점이 있다.

수표방식이 편리한 지불 수단임에도 불구하고 활성화되지 못한 것에는 두 가지 이유가 있다. 첫째, 거스름에 대한 처리가 복잡하고 제한적이다. 수표의 액면가와 지불대금은 언제나 일치할 수는 없다. 이런 경우가 발생하면 고객은 지불한 수표에 대한 거스름을 돌려 받아야 한다. 이와 같은 거스름은 보통 은행에 전달되어 환불받게 된다. 거스름은 지불대금에 따라 다양한 금액이 될 수 있기 때문에 액면가 표현방법이 동전방식에 비해 복잡해진다. 또한 거스름을 통해 먼저 사용된 고객의 수표를 알 수 없도록 거스름과 수표 사이에는 어떠한 연관관계도 없어야 한다. 거스름의 또 다른 문제점은 재사용성이다. 만약 거스름을 지불에 다시 사용할 수 있다면 수표시스템은 보다 편리한 지불 수단이 될 것이다. 그러나 거스름이 다시 사용되려면 수표와 동등한 조건이 요구되며 그와 같은 거스름 메커니즘을 제공하기란 쉽지 않다. 현재까지 제안된 대부분의 수표방식은 시스템이 정해놓은 고정된 금액의 수표만을 인출할 수 있으며, 액면가 표현방법의 한계 때문에 지불과정에서 발생한 거스름을 은행을 통해 환불받거나 일부 제한된 금액에 대해서만 다시 사용할 수 있다.[2,3,5~8] 최근에 본 연구진은 이 문제에 대한 해결책으로 수표의 액수 제한을 없애고, 온라인 상에서 은행이 새로운 수표를 거스름으로 발행해 주는 시스템을 제안한 바 있다.[4]

둘째, 전자화폐는 궁극적으로 오프라인 방식이 되어야 하지만 지불과정에서 수표 발행기관인 은행이 참여하지 않으므로 온라인 방식처럼 그 과정에서 발생한 거스름이 화폐의 기능을 갖도록 만드는 것은 어렵다. 현재까지 제안된 오프라인 시스템[5~8]은 지불과정에서 화폐 기능을 가진 거스름을 생성하지 않고 수표를 두 부분으로 구성하여 사용하는 방식을 택하고 있다. 두 부분 중 하나는 지불에 사용되고 다른 하나는 은행에 전달되어 지불 후에 발생한 거스름 액수만큼 환불받게 된다. 수표의 액면가가 고정되어 있으므로 지불에 사용된 부분과 은행에 전달된 부분은 금액면에서 서로 보수 관계를 지니게 되며, 이것이 고객의 익명성을 저해하는 요인이 된다. 뿐만 아니라 수표를 한 번밖에 사용할 수 없기 때문에 나머지 잔액은 항상 다시 입금되어야 하는 불편한 형태의 지불방식이 된다. 이것은 온라인 방식처럼 지불과정에서 거스름을 발행하지 않고 분할 가능한 화폐[9]처럼 화폐를 분할하여 사용하는 형태이기 때문이다.

이 논문에서는 한 번 사용하고 남은 금액을 다시 사용할 수 있는 오프라인 수표시스템을 제안한다. 이 시스템은 같은 수표가 사용된 지불행위를 서로 연관시킬 수 없도록 하였으며, 이중사용과 초과사용 등의 부정 행위가 발생하면 그 책임자를 찾을 수 있도록 고안하였다. 익명의 수표가 범죄에 악용되는 것에 방지하기 위해 신뢰기관을 통한 화폐 추적과 인출자 추적 기능을 제공한다. 또한 인출에서 환불까지 소요되는 전체 연산비용은 기존의 시스템에서 여러 개의 수표를 인출하여 사용하는 경우보다 적다. 따라서 기존보다 적은 연산 비용을 사용하면서도 지불의 편리성과 고객의 익명성을 동시에 향상시켰다. 그러나 액면가 표현방법의 한계 때문에 수표의 남은 금액에 대해 자유롭게 지불할 수 있게 하려면 추가적인 연산이 들어가야 하는 단점이 있다.

이 논문의 구성은 다음과 같다. 2장에서 이 논문의 이해를 위해 필요한 수학적 배경을 간략히 소개하고, 3장에서는 기존의 오프라인 수표시스템의 특성과 문제점을 분석한다. 4장에서는 이 논문에서 제안하는 새 시스템을 상세히 서술한다. 5장에서는 새 시스템의 안전성을 분석하고, 새 시스템에 대한 몇 가지 변형을 제시한다. 끝으로 6장에서 결론과 향후 연구 방향에 대해 서술한다.

## II. 수학적 배경

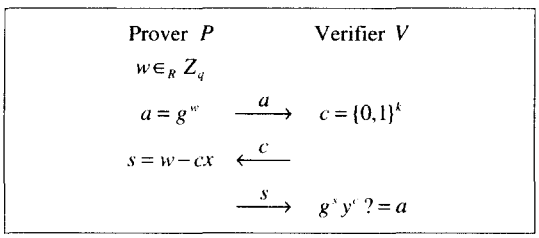
이 논문에 있는 모든 수학 연산은 군(group)의 위수(order)가 매우 큰 소수 $q$인 $G_q$군에서 이루어진다. 이 군은 먼저 매우 큰 소수 $p$를 선택하고 $p-1$의 소인수 중 하나인 $q$를 선택하여 구성된다. 따라서 $G_q$군은 $Z_p^*$의 부분군이 되며, 이 군에서 1을 제외한 모든 수는 군의 생성자(generator)가 된다. 시스템의 안전성은 이 군에서 이산대수(discrete

logarithm)를 계산하는 것은 계산적으로 용이하지 않다는 것에 기반한다. 이 논문에서 이루어지는 모든 연산의 법(modulo)은 $p$ 또는 $q$가 된다. 지수(exponent) 요소와 관련된 연산은 법 $q$에서 이루어지고, 나머지 모든 연산은 법 $p$에서 이루어진다. 문맥에서 연산의 법을 유추할 수 있으므로 이 논문에서는 생략한다.

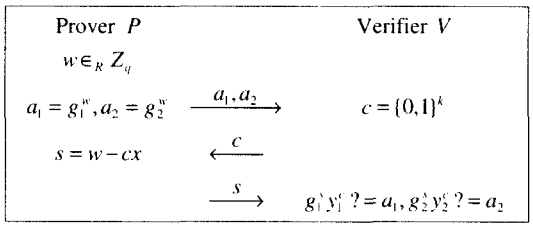
### 2.1 이산대수 관련 영지식 증명

기저 $g$에 대한 $y$의 이산대수를 알고 있음을 영지식(zero knowledge)으로 증명하는 프로토콜은 [그림 1]과 같다.[10] 이 프로토콜에서 $y=g^x$이고, 확인자(verifier)는 $g$와 $y$를 알고 있다. [그림 1]에 기술된 프로토콜은 도전과 응답(challenge-and-response)을 통해 이루어지지만 $c$를 $c=H(g\|y\|a)$로 증명자(prover)가 직접 계산하여 상호작용이 없는 프로토콜로 변경할 수 있다. 상호작용이 없는 프로토콜 버전을 ZKProof( $\log_g y$)로 표현한다. 여기서 '||'은 비트 결합을 나타내며, $H$는 $\{0,1\}^*$를 입력받아 $Z_q$의 임의의 원소로 매핑 해주는 충돌회피(collision-resistant) 해쉬함수이다. 이 프로토콜에서 증명자가 부정할 수 있는 확률은 확인자가 제시할 $c$를 올바르게 추측할 수 있는 확률 $1/2^k$과 같다. 이 증명은 인출과정에서 고객을 인증하기 위해 사용된다.

$\log_{g_1} y_1 = \log_{g_2} y_2$임을 영지식으로 증명하는 프로토콜은 [그림 2]와 같다.[11] [그림 2]에서 $y_1=g_1^x$이고, $y_2=g_2^x$이다. $c$를 $c=H(g_1\|y_1\|g_2\|y_2\|a_1\|a_2)$로 증명자가 계산하여 사용하면 이 프로토콜도 상호작용이 없는 프로토콜로 변경할 수 있다. 상호작용이 없는 프로토콜 버전을 ZKProof( $\log_{g_1} h_1 = \log_{g_2} h_2$)로 표기한다. 이 프로토콜에서 증명자가 부정할 수 있는 확률도 $1/2^k$이다. 이 증명은 익명성 제어 기능을 제공하기 위해 사용된다.




[그림 1] 기저 $g$에 대한 $y$의 이산대수를 영지식으로 증명하는 프로토콜




[그림 2] $\log_{g_1} y_1 = \log_{g_2} y_2$임을 영지식으로 증명하는 프로토콜

### 2.2 표현 문제

표현 문제(representation problem)란 이산대수 문제를 응용한 것으로 여러 개의 생성자로 만든 어떤 값이 있을 때, 이 값을 만들기 위해 사용한 생성자의 색인(index)값을 알아내는 것이 계산적으로 어렵다는 것에 바탕을 두고 있다. 표현 문제와 관련된 정의와 정리 중 이 논문의 이해를 위해 필요한 내용만 소개하면 다음과 같다.[11]

**[정의 1]**

길이가 $l(\geq 2)$인 생성자 튜플은 $(g_1,\ldots,g_l)$을 말하며, 이 때 $g_i \in G_q-\{1\}$는 $G_q$의 생성자이며 $i\neq j$이면 $g_i \neq g_j$이다. 생성자 튜플 $(g_1,\ldots,g_l)$에 대한 $y\in G_q$의 표현은 $\prod_{i=1}^{l} g_i^{x_i} = y$인 색인 튜플 $(x_1,\ldots,x_l)$을 말하며, 이 때 $\forall x_i \in Z_q$이다.

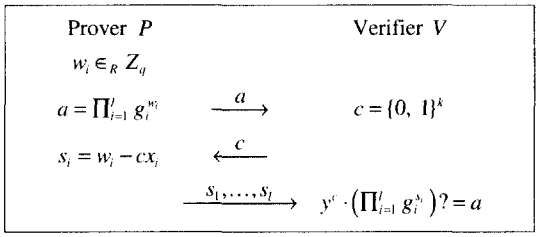
**[정리 1]**

$G_q$에서 이산대수를 계산하는 것이 용이하지 않다고 가정하면 모든 $y\in G_q-\{1\}$에 대해 임의로 선택한 생성자 튜플 $(g_1,\ldots,g_l)$을 입력하였을 때, 어느 정도 성공할 수 있는 확률을 가지고 $y$의 표현을 출력할 수 있는 다항시간 알고리즘은 존재하지 않는다.

**[따름정리 1]**

$G_q$에서 이산대수를 계산하는 것이 용이하지 않다고 가정하면 임의로 선택한 생성자 튜플 $(g_1,\ldots,g_l)$을 입력하였을 때, 어느 정도 성공할 수 있는 확률을 가지고 어떤 수 $y\in G_q-\{1\}$와 $y$의 서로 다른 두 가지 표현을 출력할 수 있는 다항시간 알고리즘은 존재하지 않는다.

정리 1과 따름정리 1에 대한 증명은 이산대수 가정을 이용하여 쉽게 증명할 수 있다.[12] 이 논문에서는 표현문제를 이용하여 수표를 구성하였고, 수표를 나타내는 값의 표현을 알고 있는 고객만 지불할




〔그림 3〕 표현을 영지식으로 증명하는 프로토콜

수 있도록 고안하였다.

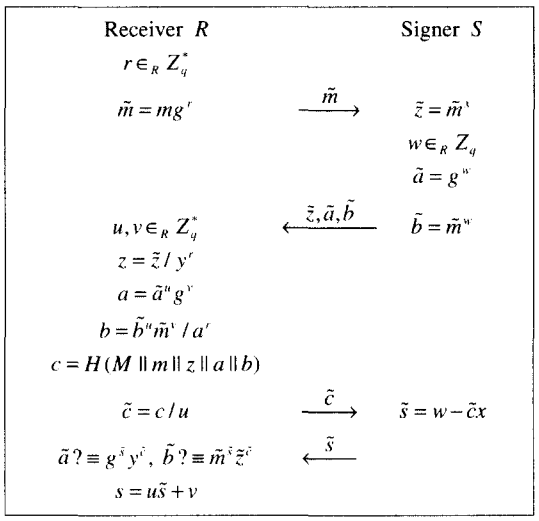
어떤 값에 대한 표현을 알고 있음을 영지식으로 증명하는 프로토콜은 〔그림 3〕과 같다. 여기서 증명자는 $(g_1, \ldots, g_l)$에 대한 $y$의 표현 $(x_1, \ldots, x_l)$를 알고 있음을 증명하고 있다. $c$를 $c = H(g_1 \| \ldots \| g_k \| y \| a)$로 증명자가 직접 계산하여 이 프로토콜을 상호작용이 없는 프로토콜로 변경할 수 있다. 이 프로토콜도 증명자가 부정할 수 있는 확률은 $1/2^k$이다. 이 프로토콜의 상호작용이 없는 버전을 ZKProof$(g_1, \ldots, g_l, y)$로 표기한다.

## 2.2 제한적 은닉서명

제한적 은닉서명(restrictive blind signature)이란 용어는 Chaum과 Pedersen이 제안한 서명 기법[11]을 Brands가 응용하면서 처음으로 사용되었다.[1] 제한적 은닉서명 기법은 기존의 cut-and-choose 기법을 사용하지 않고 서명 받는 사람이 부정을 못하게 하는 은닉서명 기법이다. 이 논문에서 사용하는 제한적 은닉서명은 Brands가 제안한 서명 기법을 de Solages와 Traore가 변형한 서명 기법이다.[8] 이 서명 기법은 〔그림 4〕에 기술되어 있다. 여기서 서명자의 서명키는 $x \in Z_q$이며, 확인키는 $y = g^x$이다. 수신자는 메시지 $m$에 대해 서명을 받기 위해 $m$을 은닉한 $\tilde{m}$을 서명자에게 전달하여 프로토콜을 시작한다. 이 프로토콜이 정상적으로 종료되면 수신자는 메시지 $m$에 대한 서명 $\mathrm{Sig}(m) = (z, c, s)$을 얻는다. 이 때 서명값 $z = m^x$, $c$, $s$는 다음 식을 만족한다.

$$c = H(M \| m \| z \| g^s y^c \| m^s z^c) \quad (1)$$

서명 확인은 위 식을 이용한다. 이 때 $M$은 서명할 때 수신자가 서명과 연관시키기 위해 서명에 포함시키는 값이다. 이것은 값을 명시(commit)하는 것과 같으며, 필요가 없으면 사용하지 않아도 된다. 이




〔그림 4〕 제한적 은닉서명 BlindSig$(M, \tilde{m})$

서명에 관한 몇 가지 정리는 다음과 같다. 각 정리에 대한 증명은 이미 있으므로,[1,7,8,11,12] 이 논문에서는 증명에 대한 개략적인 설명만 한다.

**[정리 2] (정확성)**

수신자가 증명을 인정하면 $\mathrm{Sig}(m)$은 $m$에 대한 올바른 서명이다.

서명은 식 (1)을 이용하여 확인하므로 양자 모두 서명 프로토콜을 충실하게 수행하였다면 $a = g^s y^c$이고 $b = m^s z^c$임을 보이면 된다.[8,11]

**[정리 3] (은닉성)**

수신자가 프로토콜을 충실하게 수행하면 서명자는 $m$과 $\mathrm{Sig}(m)$에 대한 어떤 정보도 얻을 수 없다.

수신자와 서명자가 모두 충실하게 서명을 수행하였으면 수신자는 $m$, $z$, $c$, $s$를 얻으며, 서명자는 $\tilde{m}$, $\tilde{z}$, $\tilde{a}$, $\tilde{b}$, $\tilde{c}$, $\tilde{s}$를 얻게 된다. 이 두 값을 연결시키기 위해 알아야 하는 값은 은닉요소 역할을 하는 $r$, $u$, $v$이다. 이 값들은 임의로 선택되고 이산대수 가정에 의해 직접 계산할 수 없으므로 서명자는 얻을 수 없다. 따라서 두 집합의 값을 연결시키는 $r$, $u$, $v$는 오직 한 종류밖에 없음을 보임으로써 다른 방법으로 두 집합의 값을 연관시킬 수 없음을 증명할 수 있다.[8,11]

**[정리 4] (위조불가능성)**

BlindSig를 여러 번 수행(병행 또는 순차적으로)한다고 하여 원래 얻을 수 있는 서명 쌍을 외에는

추가적인 서명 쌍을 얻을 수 없다. 뿐만 아니라 프로토콜의 수행을 통해 서명자의 서명키 $x$에 대한 어떤 정보도 노출되지 않는다.

정리 4는 전자화폐에서는 가장 중요한 정리이다. 이것이 보장된다는 것은 화폐를 위조할 수 없음을 증명하는 것이 되기 때문이다. Schnorr는 이산대수 기반 은닉서명에 대한 새로운 일반 병행 공격 방법을 제시하였다.[12] 이 공격은 다항시간 해결법이 아직 알려져 있지 않은 ROS 문제[13]에 의존한다. 즉, ROS 문제를 해결할 수 있으면 Schnorr 은닉서명에 대한 $(l, l+1)$ 위조 존재(existential forgery) 공격이 가능하다. $(l, l+1)$ 위조란 어떤 정수 $l$에 대해 공격자가 서명자와 $l$번 서명 프로토콜을 수행하여 $l+1$개의 서명을 얻는 공격을 말한다. 마찬가지로 이 프로토콜에 대한 $(l, l+1)$ 위조 공격도 ROS 문제를 해결할 수 있으면 가능하다. 따라서 이 프로토콜도 Schnorr가 제시한 공격 방법으로 $(l, l+1)$ 위조 존재 공격을 하는 것은 계산적으로 어렵다.

**[정리 5] (은닉 제한)**

수신자가 $\tilde{m}$을 BlindSig에 입력으로 사용하면 생성자 $g$외에 다른 생성자를 이용하여 서명받을 메시지 $m$을 결정할 수 없다.

수신자는 다른 생성자 $g_1$를 이용하여 $\tilde{m}$에서 은닉요소를 제거하기 위해서는 서명자로부터 받은 $\tilde{z}$와 $\tilde{b}$에서 각각 제거할 수 있거나 기저 $g$에 대한 $g_1$의 이산대수를 알아야 한다. 두 경우 모두 이산대수 가정에 의해 계산적으로 용이하지 않다. 따라서 서명해줄 메시지에 특정 고객 신원정보가 포함되는 것과 같은 보장이 필요하면 $\tilde{m}$에 대한 표현 증명을 하도록 하는 등 다른 조치를 취하여야 한다.

## Ⅲ. 기존 오프라인 수표시스템

Chaum 등은 최초로 오프라인 방식의 수표시스템을 발표하였다.[5] 이 시스템에서 수표는 도전 항목, 액면가 항목, 환불 항목, 세 가지 요소로 구성된다. 도전 항목은 이중사용을 방지하기 위해 사용되고, 액면가 항목은 수표의 액면가를 나타내기 위해 사용된다. 환불 항목은 수표를 사용하고 남은 금액을 환불받기 위해 사용된다. 이 항목들은 고객의 익명성을 보장하기 위해 은닉되어 전달되며, 은행은 각 항목들이 제대로 구성되어 있는지 확인하기 위해 비용이 많이 드는 cut-and-choose 기법을 사용한다. 뿐만 아니라 인출할 수 있는 수표의 액면가가 고정되어 있으며, 남은 금액을 다시 사용할 수 없고 반드시 환불받아야 한다.

이 시스템에서 인출자는 cut-and-choose 과정을 수행하기 위해 실제 수표를 구성하는 항의 개수보다 두 배 많은 개수의 항을 전달하고, 은행은 그 중 확인하지 않은 절반의 항을 이용하여 수표를 구성한다. 따라서 한 개의 잘못된 항이 발견되지 않고 통과될 확률은 높다. 도전항목을 한 개 잘못 구성하여 수표를 인출받더라도 수표를 이중사용할 수 있는 확률은 매우 적다. 그러나 액면가 항목 중 하나를 잘못 구성하여 인출받으면 고객은 실제 환불받아야 하는 금액보다 많은 금액을 환불받을 수 있다. Hirschfeld는 이 문제를 발견하고 그것을 보완한 시스템을 발표하였다.[6] 그러나 Hirschfeld 시스템은 여전히 cut-and-choose를 사용하므로 Chaum 등의 시스템의 근본 문제인 효율성을 개선하지는 못하였다.

Brands는 처음으로 cut-and-choose 기법을 사용하지 않는 오프라인 수표시스템을 발표하였다.[7] 이 시스템은 표현 문제를 이용하여 수표의 액면가를 표현하며, cut-and-choose 기법을 사용하지 않아도 고객의 부정을 예방할 수 있는 제한적 은닉서명 기법을 이용한다. Brands 시스템을 설명하기 앞서 표현 문제를 이용한 수표시스템의 기본 개념은 다음과 같다. 이 개념은 이 논문에서 제안하는 시스템의 기본이기도 하다.

은행은 길이가 $l(\geq 2)$인 생성자 튜플 $(g_1, \ldots, g_l)$를 선택하여 공개한다. 이 때 각 생성자 $g_i$는 $2^{i-1}\times 100$원의 액면가를 나타내기 위해 사용된다. 따라서 길이가 $l$인 생성자 튜플을 사용하여 수표의 액면가를 표현하면 인출하는 수표의 액면가는 항상 $(2^l-1)\times 100$원이 된다. 고객은 길이가 $l$인 색인 튜플 $(d_1, \ldots, d_l)$를 임의로 생성하여 $m=\prod_{i=1}^{l} g_i^{d_i}$을 계산한다. 이 $m$이 수표가 되며, 은행으로부터 $m$에 대한 서명 $\mathrm{Sig}(m)$을 받아 사용하게 된다. $J$를 지불대금에 해당하는 색인 위치들의 집합이라고 하면 고객은 $m$, $\mathrm{Sig}(m)$, $d_j (j \in J)$을 상점에게 전달하여 지불한다. 예를 들어 500원을 지불하고자 하면 100원을 나타내는 $d_1$과 400원을 나타내는 $d_3$를 전달한다. 지불과정에서 고객은 전달한 $d_j$들이 $m$을 만들 때 사용한 색인 값임을 증명하여야 한다. 이 증명은 표현 문제를 이용하여 이루어진다. 사용하고 남은 금액은 $m$과 $d_j$ $(j \in \{1, \ldots, l\}-J)$를 은행에 전달하여 환불받는다.

물론 지금까지 소개한 기본 개념에는 익명성, 이중사용 등에 대한 고려가 없지만 여기에 그런 문제를 고려하여 설계하면 효율적인 오프라인 수표시스템을 만들 수 있다.

Brands의 오프라인 시스템은 표현 문제를 이용한 최초의 수표시스템으로서 사용하는 수표의 모습은 다음과 같다.

$$C=\overbrace{\left(\prod_{i=1}^{2} g_i^{u_i}\right) g_3^{v}}^{\text{identity part}} \cdot \overbrace{\left(\prod_{i=1}^{k} g_{1i}^{a_i} g_{2i}^{b_i}\right) g_4^{a_{k+1}}}^{\text{denomination part}}$$

여기서 $g_i$, $g_{1i}$, $g_{2i}$는 $G_q$의 생성자이다. 이 중 $g_1$과 $g_2$는 고객의 신원정보를 나타내기 위해 사용되고, $g_{1i}$와 $g_{2i}$는 수표의 액면가를 나타내기 위해 사용되며, $g_3$과 $g_4$는 포함된 고객 신원정보와 액면가 정보를 식별하지 못하도록 하기 위해 사용된다. 고객은 이렇게 구성한 값 $C$에 대해 제한적 은닉서명을 받아 사용하게 된다. de Solages와 Traore가 제안한 시스템[8]은 Brands가 제안한 시스템과 거의 유사한 시스템이지만 익명성 제어 기능을 추가하였으며, 요구되는 연산의 양을 줄여서 효율성도 높였다. 그러나 서론에서 소개한 바와 같이 이 장에서 설명한 4개의 오프라인 시스템에서 수표는 두 부분으로 구성되어 있으며, 두 부분은 가격측면에서 서로 보수관계를 지니고 있다. 뿐만 아니라 수표를 한번 밖에 사용할 수 없으므로 항상 남은 잔액을 입금하여 환불받아야 한다. 이러한 제한 때문에 사용하기가 불편함은 물론 고객의 익명성을 해칠 수도 있다.

## Ⅳ. 새 오프라인 수표시스템

이 장에서는 기존의 오프라인 수표시스템이 가지고 있는 제한을 부분적으로 해결하여 고객이 좀 더 편리하게 사용할 수 있도록 고안한 새로운 시스템을 제안한다. 이 시스템은 수표의 남은 잔액을 다시 사용할 수 있도록 하기 위해 수표를 인출할 때 두 개의 값에 은행으로부터 은닉서명을 받는다. 그 가운데 하나는 수표를 처음 지불할 때 사용되고, 다른 하나는 이미 사용한 값에 따라 남은 금액을 다른 것에 지불할 때 사용된다. 새 시스템은 표현 문제를 이용하여 수표의 액면가를 나타내며, 제한적 은닉서명을 사용한다.

### 4.1 시스템 설정

은행은 계정을 관리하며, 고객과 상점은 다음 절에 설명된 방법으로 은행에 계정을 개설하여야 지불에 참여할 수 있다. 신뢰기관은 익명의 수표가 범죄에 악용되는 것을 방지하기 위한 화폐 추적과 인출자 추적 기능을 제공하는 기관이다. 은행과 신뢰기관은 시스템을 설정하기 위해 우선 $q \mid p-1$인 두 개의 매우 큰 소수 $p$, $q$를 선택하여 사용할 $G_q$군을 설정한다. 그 다음에 은행은 $l+1$개의 $G_q$군 생성자 $g_1, \ldots, g_{l+1}$를 임의로 선택한다. $g_1$부터 $g_l$은 액면가를 표현하기 위한 생성자이며, $g_{l+1}$은 액면가 표현을 숨기기 위해 사용된다. 따라서 이 시스템에서 수표의 액면가는 $(2^l-1)\times 100$원이다. 이들 생성자 외에 세 개의 $G_q$군 생성자 $g_b$, $g_u$, $g_s$를 임의로 선택한다. $g_b$는 은행의 서명키와 확인키를 위한 생성자이며, $g_u$와 $g_s$는 각각 고객과 상점의 신원을 나타내기 위해 사용된다. 은행은 서명키 $x_b \in Z_q$를 임의로 선택하고 대응되는 확인키 $x_b = g_b^{x_b}$를 계산한다. 은행은 $p$, $q$, $g_1, \ldots, g_{l+1}$, $g_b$, $g_u$, $g_s$, $y_b$를 공개한다. 은행은 이중사용과 초과사용을 방지하기 위한 입금 데이터베이스와 색인목록(index list)을 유지하며, 부정한 방법으로 환불을 요청할 수 없도록 인출 데이터베이스를 유지한다. 데이터베이스와 목록은 추적을 위해서도 필요하다. 신뢰기관은 $G_q$군 생성자 $g_t$를 임의로 선택하고, 화폐 추적을 위한 개인키 $x_{ct} \in Z_q$와 인출자 추적을 위한 개인키 $x_{ot} \in Z_q$를 임의로 선택한다. 그 다음에 대응되는 각 공개키 $y_{ct} = g_t^{x_{ct}}$와 $y_{ot} = y_{ct}^{x_{ot}}$를 계산한다. 신뢰기관은 $g_t$, $y_{ct}$, $y_{ot}$를 공개한다.

### 4.2 계정 개설

고객은 자신의 비밀신원정보 $x_u \in Z_q$를 임의로 선택하여 자신의 공개신원정보 $y_u = g_u^{x_u}$를 계산하고 $y_u$를 은행에 전달한다. 고객은 $\log_{g_b} y_u$를 알고 있음을 [그림 1]의 기술된 프로토콜을 이용하여 은행에 증명한다. 은행은 $y_u$를 해당 고객의 식별자로 기록해 놓는다. 이 증명을 통해 은행이 $y_u$를 확인하지 않으면 고객은 Chan 등이 제시한 공격을 하여 수표를 이중사용할 수 있다.[14] 상점도 고객과 유사하게 $x_s \in Z_q$를 임의로 선택하여 $y_s = g_s^{x_s}$를 계산하고 $y_s$를 은행에 전달하여 계정을 개설한다. 상점도 고객과 같은 방법으로 $\log_{g_s} y_s$를 알고 있음을 증명한다.

### 4.3 인출 프로토콜

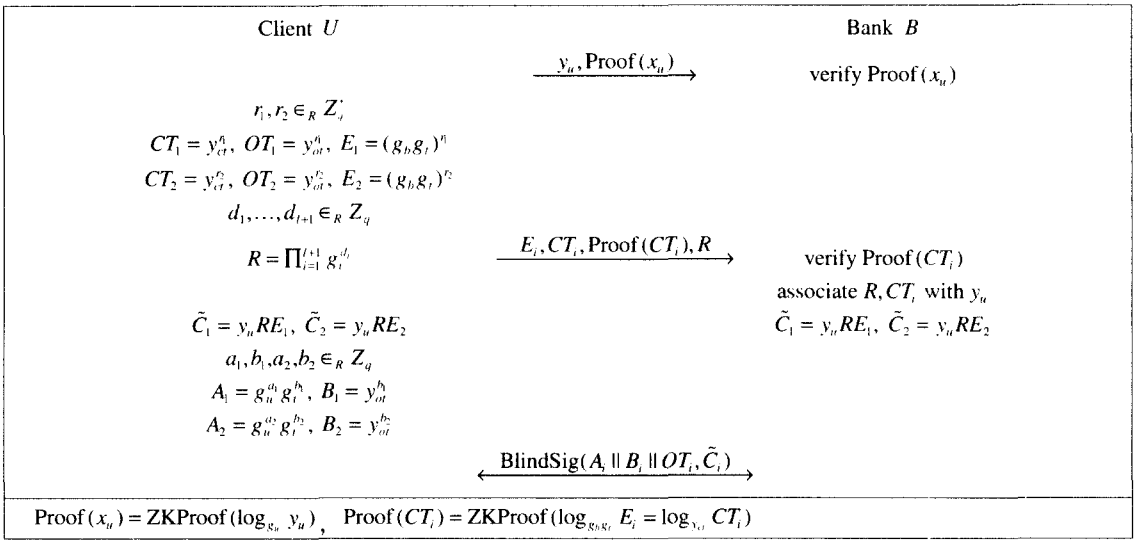
새 시스템의 인출 프로토콜은 [그림 5]와 같다. 고객은 은행에 등록되어 있는 자신의 식별자 $y_u$를 은행에 전달하고 $\log_{g_b} y_u$를 알고 있음을 영지식으로 증명한다. 이 때 시간정보를 증명에 포함하여 공격자가 이 증명을 다시 사용할 수 없도록 한다. 자신의 신원을 입증하였으면 고객은 사용할 두 개의 은닉요소 $r_1$과 $r_2$를 선택한다. 이 요소로 화폐 추적을 위한 정보 $CT_i = y_{ct}^{r_i}$를 계산한다. 이 값을 이용하여 화폐 추적이 가능함을 증명하기 위해 $E_i = (g_b g_t)^{r_i}$를 계산하고, 이산대수 등가증명 ZKProof( $\log_{g_b g_t} E_i = \log_{y_{ct}} CT_i$)를 구성한다. 또한 수표의 액면가 표현인 $R$을 만든다. $R$은 임의로 선택한 $d_1$부터 $d_{l+1}$까지 $l+1$개의 색인을 이용하여 구성한다. $d_{l+1}$을 제외한 나머지 색인은 지불 또는 환불과정에서 모두 공개된다. 따라서 마지막 색인 $d_{l+1}$은 $R$을 구성하는 다른 모든 색인이 공개되더라도 이를 이용하여 $R$을 식별할 수 없도록 하기 위한 조치이다. 이 $R$은 나중에 남은 금액을 환불받을 때에도 사용된다.

고객은 수표의 액면가 표현 $R$, 화폐 추적을 위한 정보 $CT_i$, 수표에 포함할 추적정보 $E_i$, $CT_i$를 통해 화폐가 추적 가능함을 보이기 위한 Proof($CT_i$)를 은행에게 전달한다. 은행은 증명을 확인하여 $CT_i$를 통해 나중에 수표에 대한 화폐 추적을 할 수 있음을 확인한다. 확인이 끝나면 $CT_i$와 $R$을 고객과 인출 데이터베이스에 연관시켜 놓는다. 고객은 인출자 추적을 위한 정보 $OT_i$와 이 값을 통해 인출자 추적이 가능하다는 것을 증명할 때 사용할 $B_i$, 그리고 지불과정에서 상점의 도전에 대한 응답을 계산할 때 사용할 $A_i$을 계산한다. 인출자 추적에 대한 확인은 상점이 지불과정에서 하게 되며, $A_i$을 통해 응답을 계산하도록 제한하는 것은 이중사용이 발생했을 때 고객의 비밀신원정보를 계산할 수 있도록 하기 위함이다.

고객과 은행은 각자 인출할 수표의 은닉된 형태 $\tilde{C}_i = y_u RE_i$를 계산한다. 여기서 $y_u$는 고객의 공개 신원정보이고, $R$은 수표의 액면가 정보이며, $E_i$는 추적을 위한 정보이다. 은행은 이렇게 스스로 은닉된 형태를 만들어 제한적 은닉서명을 수행하므로 나중에 고객이 얻게 되는 수표의 형태가 올바르다는 것을 확신할 수 있으며, 각 $\tilde{C}_i$에 같은 고객 정보와 같은 액면가 정보가 포함되어 있음을 확신할 수 있다. 고객과 은행은 각 $\tilde{C}_i$에 대해 BlindSig($A_i \| B_i \| OT_i, \tilde{C}_i$)을 수행한다. 이렇게 하여 프로토콜이 종료되면 고객은 각 $C_i = y_u R g_t^{r_i}$에 대한 은행의 서명 Sig($C_i$) = ($z_i, c_i, s_i$)를 얻게 되며, $z_i = C_i^{x_b}$, $c_i$, $s_i$는 다음을 만족한다.

$$c_i = H(A_i \| B_i \| OT_i \| C_i \| z_i \| g_b^{s_i} y_b^{c_i} \| C_i^{s_i} z_i^{c_i})$$

고객은 이렇게 얻은 서명 중 하나를 먼저 사용하고




[그림 5] 인출 프로토콜

남은 금액에 대해서는 다른 하나를 사용하게 된다. 또한 두 서명을 모두 사용하고 남은 금액에 대해서는 은행으로부터 환불받을 수 있다. 물론 하나의 서명만 사용하고 나머지 금액에 대해 환불받을 수도 있다.

### 4.4 지불 프로토콜

고객은 [그림 6]에 기술된 프로토콜을 이용하여 지불한다. 고객은 인출을 통해 얻은 두 개의 서명 중 하나를 선택하여 먼저 사용하고, 나머지 서명은 첫 번째 서명을 사용할 때 지불한 대금에 따라 남은 금액에 대해 사용한다. 그러나 남은 금액에 대해 자유롭게 지불할 수 없고, 일부 금액에 대해서만 지불할 수 있는 문제점을 지니고 있다. 예를 들어 길이가 4인 생성자 튜플을 이용하여 액면가를 표현한다고 하자. 그러면 인출한 수표의 액면가는 1500원이 되며, 이 때의 색인 튜플을 $(d_1, d_2, d_3, d_4)$라 하자. 첫 번째 지불에서 600원을 지불하였다면, 이는 고객이 첫 번째 지불에서 $d_3$과 $d_2$를 공개하였다는 것을 의미하며, 두 번째 지불에서는 원래 수표의 액면가와 이미 지불한 금액의 차액 900원을 지불할 수 있다. 그러나 $d_4$와 $d_1$을 이용하여 지불해야 하므로 두 번째 지불에서는 900원, 800원, 또는 100원밖에 지불할 수 없다. 이 문제점에 대해서는 다음 장에서 해결방안을 제시한다.

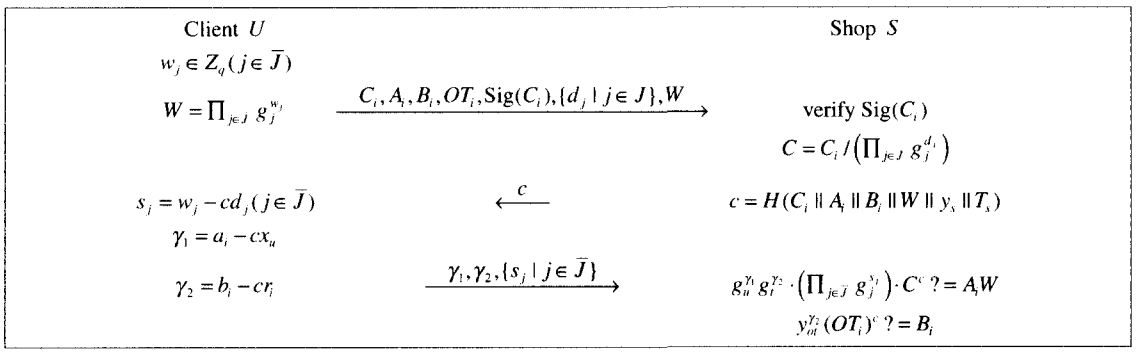
새 시스템의 지불 프로토콜은 크게 두 단계로 구성되어 있다. 첫 번째 단계는 수표에 대한 은행의 서명을 확인하는 단계이고, 두 번째 단계는 지불대금에 해당하는 색인을 고객이 알고 있는지 확인하는 단계이다. 이중사용, 초과사용 등이 발생하였을 때 책임자를 밝혀내기 위해서는 수표를 인출한 고객만이 그 수표를 이용하여 지불할 수 있어야 한다. 제안된 시스템은 표현문제를 이용한 기존 오프라인 화폐시스템과 마찬가지로 화폐를 나타내는 값의 표현을 알고 있는 고객만이 지불할 수 있도록 하였다. 고객이 $C_i$를 이용하여 지불하는 과정을 자세히 설명하면 다음과 같다. 앞으로 $L$은 집합 $\{1,\ldots,l+1\}$를 나타내며, 집합 $J$은 $L$의 부분집합으로서 지불대금을 나타내기 위해 사용한 색인위치들의 집합을 나타낸다. 또한 $\bar{J}$는 $L-J$집합을 나타낸다.

**[단계 1]**

고객은 $C_i$, $A_i$, $B_i$, $OT_i$, $\mathrm{Sig}(C_i)$, 지불대금을 나타내기 위한 색인 $d_j$, 그리고 $C_i$에 포함된 나머지 색인을 알고 있음을 증명할 때 사용할 $W$를 생성하여 상점에게 전달한다. 상점은 서명을 확인하고 수신한 $d_j$를 이용하여 [그림 6]처럼 $C$를 구성한다. 계속해서 $C_i$, $A_i$, $B_i$, $W$, 상점 식별자 $y_s$, 지불시간 $T_s$를 이용하여 만든 도전값 $c$를 고객에게 전달한다. 상점이 전달할 $c$값을 고객이 미리 예측하여 자신에게 유리하도록(예를 들어 이중사용하여도 신원이 들어나지 않도록 또는 인출자 추적이 가능하지 않도록) $A_i$와 $B_i$을 인출 과정에서 조작할 수 없도록 $c$를 만들 때 $A_i$와 $B_i$를 반드시 포함시켜야 한다. 같은 이유에서 $W$도 포함시켜야 한다. 또한 $y_s$는 이 수표를 입금하여 돈을 청구할 수 있는 상점을 제한하기 위해 필요하다.

**[단계 2]**

고객은 도전값 $c$에 대한 응답으로 $j \in \bar{J}$에 대해 $s_j = w_j - cd_j$를 계산하고, $(g_u, g_t)$에 대한 $A_i$의 표현 $(a_i, b_i)$을 이용하여 $\gamma_1$과 $\gamma_2$를 계산하여 상점에게 전달한다. 상점은 수신한 응답값을 확인함으로써




[그림 6] 지불 프로토콜

단계 1에서 수신한 색인이 올바른 색인임을 확인하게 됨은 물론 전달된 색인을 제외한 나머지 색인도 고객이 알고 있음을 확인하게 된다. 또한 $\gamma_2$를 이용하여 $OT_i$를 확인하여 이 수표에 대한 인출자 추적이 가능함을 확인한다. 고객에게 $A_i$의 표현을 이용하여 $\gamma_1$과 $\gamma_2$를 계산하도록 한 것은 $C_i$를 지불할 때 계산되는 응답값을 제한하여 이중사용 하였을 때 고객의 신원을 밝히기 위함이다.

### 4.5 입금 프로토콜

수표를 인출할 때 은행으로부터 받은 두 개의 서명값은 보통 각각 서로 다른 지불에 사용된다. 은행은 고객이 각 값을 이중사용 하였는지 여부와 두 개의 서명을 이용하여 지불한 총액이 수표의 액면가를 초과했는지 검사하여야 한다.

상점은 $C_i$, $A_i$, $B_i$, $OT_i$, $\mathrm{Sig}(C_i)$, 지불대금을 나타내기 위한 색인 $d_j$, $W$, $T_s$, 그리고 고객이 전달한 응답값 $s_j$, $\gamma_1$, $\gamma_2$를 은행에 전달하여 지불대금을 청구한다. 은행은 먼저 도전값 $c$를 스스로 만들고 상점이 했던 방식대로 수신한 트랜스크립트의 이상 여부를 확인한다. 은행이 도전값을 스스로 만들어 확인하는 것은 이 수표에 대한 돈을 청구할 수 있는 상점을 확인하기 위함이다.

정당한 수표임이 확인되면 입금된 $C_i$가 입금 데이터베이스에 있는지 검사한다. 만약 같은 $C_i$가 있으면 먼저 입금될 때 전달된 트랜스크립트와 현재의 트랜스크립트를 비교한다. 이 비교를 하기 위해 트랜스크립트 전체를 보관할 필요는 없고, 필요한 최소 정보만 보관하면 된다. 만약 서로 같은 트랜스크립트로 밝혀지면 상점이 이중청구하는 것이 되고, 반대로 서로 다르면 고객이 이중사용한 것이다. 만약 입금된 $C_i$가 입금 데이터베이스에 없는 새로운 것이면 초과사용을 하였는지 검사하여야 한다. 이를 위해 은행은 색인목록에 입금 또는 환불된 $d_j$를 위치별로 보관한다. 색인목록에 있는 $d_j$와 현재 입금된 $d_j$를 비교하여 같은 값이 존재하면 액면가를 초과해서 지불한 것이 된다.

어떤 고객 $U$가 아래와 같은 수표를 인출하였다고 가정하고, 이중사용과 초과사용이 발생하였을 때 고객의 신원을 알아내는 과정을 설명하면 다음과 같다. 이 때 시스템은 길이가 4인 생성자 튜플을 사용하여 액면가를 표현한다고 가정한다. 즉, 인출할 수 있는 수표의 액면가가 고정되어 있으므로 모든 수표의 액면가는 1500원이 된다.

〔표 1〕 $C_1$과 $C_2$를 지불할 때 교환되는 값

| $C_1$을 이용한 지불 | $C_2$를 이용한 지불 |
|---|---|
| $C_1, A_1, d_1, d_3$ | $C_2, A_2, d_2, d_4$ |
| $s_2 = w_2 - cd_2$ | $s_1' = w_1' - c'a_1$ |
| $s_4 = w_4 - cd_4$ | $s_3' = w_3' - c'a_3$ |
| $\gamma_1 = a_1 - cx_u$ | $\gamma_1' = a_2 - c'x_u$ |
| $\gamma_2 = b_1 - cr_1$ | $\gamma_2' = b_2 - c'r_2$ |

$$C_1 = g_u^{x_u} g_1^{d_1} g_2^{d_2} g_3^{d_3} g_4^{d_4} g_5^{d_5} g_t^{r_1} \quad A_1 = g_u^{a_1} g_t^{b_1} \quad \mathrm{Sig}(C_1)$$
$$C_2 = g_u^{x_u} g_1^{d_1} g_2^{d_2} g_3^{d_3} g_4^{d_4} g_5^{d_5} g_t^{r_2} \quad A_1 = g_u^{a_2} g_t^{b_2} \quad \mathrm{Sig}(C_2)$$

만약 고객 $U$가 $C_1$을 이용하여 500원을 지불하고, $C_2$를 이용하여 1000원 지불하면 〔표 1〕과 같은 값을 상점에게 전달하여야 한다. 이 값들은 표현문제를 이용하여 확인하므로 고객은 잘못된 값을 전달할 수 없다. 만약에 고객 $U$가 $C_1$을 이중사용하면 각 지불마다 다른 도전값에 대해 응답을 하여야 한다. $C_1$을 이용한 두 번째 지불에서 사용된 도전값이 $c''$ 이라 하자. 은행은 나중에 상점이 입금요청을 하면 다음 두 식을 입금된 각 지불 트랜스크립트로부터 얻을 수 있다.

$$\gamma_1 = a_1 - cx_u$$
$$\gamma_1'' = a_1 - c''x_u$$

이 연립방정식을 풀면 $x_u$를 얻을 수 있으며, 은행은 $y_u = g_u^{x_u}$를 계산하여 이중사용한 고객을 밝혀낸다.

만약에 $C_1$과 $C_2$를 사용하여 지불한 총액이 수표의 액면가를 초과했다면 두 지불에서 중복되는 색인이 최소한 하나 존재하게 된다. 이런 중복된 색인이 있으면 은행은 초과사용한 것으로 판단하고 각 수표에 대한 인출자 추적을 신뢰기관에게 요청한다.

### 4.6 익명성 제어

익명의 화폐는 실물화폐처럼 돈세탁, 협박, 불법구매와 같은 범죄에 악용될 수 있다. 따라서 이중사용되지 않은 화폐도 필요하면 익명성을 철회할 수 있어야 한다. 보통 익명성 철회는 신뢰기관을 이용하며 화폐 추적과 인출자 추적 두 가지 형태로 제공된다. 화폐 추적은 인출과정에서 얻은 정보로부터 그 과정에서 인출된 화폐를 구성할 수 있도록 하여

그 화폐가 나중에 입금되면 식별할 수 있게 해준다. 인출자 추적은 입금된 화폐에서 고객의 신원 정보를 계산할 수 있게 해준다.

**[화폐 추적]**

은행은 인출과정에서 확인한 $CT_i$를 신뢰기관에게 전달하여 화폐 추적을 요청한다. 신뢰기관은 법원의 승인이 있으면 화폐 추적을 위한 개인키 $x_{ct}$를 이용하여 $CT_i^{x_{ct}} = g_t^{r_i}$를 계산하고, 이 값을 은행에게 돌려준다. 은행은 받은 값과 액면가 정보 $R$, 그리고 고객의 신원 정보 $y_u$를 이용하여 다음과 같이 인출된 수표를 구성할 수 있다.

$$C_i = y_u \, R g_t^{r_i}$$

**[인출자 추적]**

은행은 입금된 수표의 $OT_i$를 신뢰기관에게 전달하여 인출자 추적을 요청한다. 신뢰기관은 법원의 승인이 있으면 인출자 추적을 위한 개인키 $x_{ct}$를 이용하여 $CT_i^{x_{ct}} = y_{ct}^{r_i} = CT_i$를 계산하고, 이 값을 은행에게 돌려준다. 은행은 $CT_i$를 인출 데이터베이스에서 검색하여 이 수표의 인출자를 찾는다.

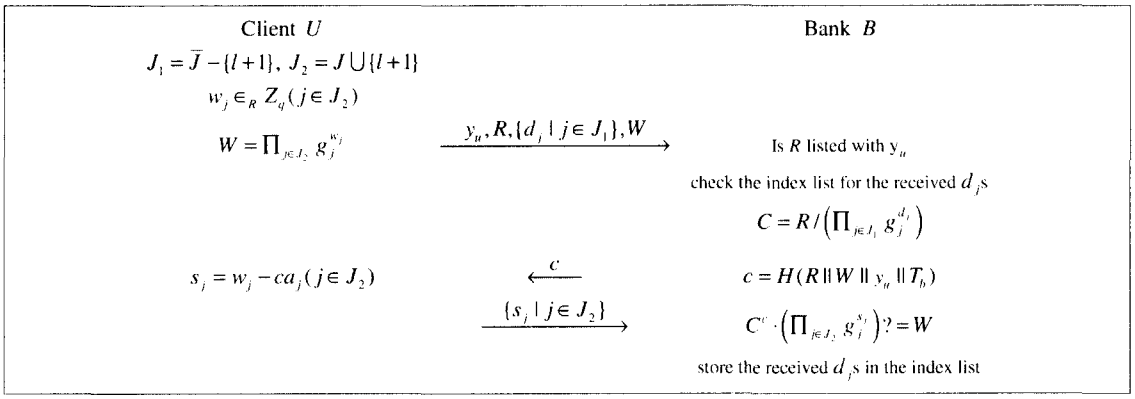
### 4.7 환불 프로토콜

고객은 수표를 사용하고 남은 잔액을 은행으로부터 환불받을 수 있다. 이 때 사용하는 프로토콜을 환불 프로토콜이라 하며 [그림 7]에 기술되어 있다. 고객은 지불에 사용한 색인에 대해서는 임의의 값 $w_j$를 선택하여 [그림 7]에 있는 것처럼 $W$를 계산한다. 고객은 자신의 식별자 $y_u$, 환불받고자 하는 수표의 액면가 표현 $R$, $d_{l+1}$를 제외한 지불에 사용하지 않은 $R$의 색인 $d_j$들, 그리고 계산한 $W$를 은행에 전달하여 환불을 요청한다. 은행은 인출 데이터베이스에서 $R$이 요청한 고객에 등록되어 있는 값인지를 확인한다. 이것을 확인하는 이유는 고객이 임의의 값을 만들어 청구하는 것을 막기 위함이다. 은행은 또한 $R$을 이용하여 이전에 환불받은 적은 없는지 확인한다. 다음에 색인목록에서 $d_j$가 이미 사용된 값은 아닌지 확인하여 환불해줄 금액을 확인한다. 확인되면 [그림 7]처럼 $C$를 계산하고 도전값 $c$를 생성하여 고객에게 전달한다. 고객은 전달한 색인이 $R$을 만들 때 사용한 올바른 색인임을 은행에게 증명한다. 올바른 증명을 수신하면 은행은 수신된 $d_j$들을 환불 목록에 등록한다. 이 때 고객의 신원과 $d_j$들을 연관시켜 놓아야 한다. 이것은 고객이 환불을 먼저 요청한 후에 수표를 지불하여 이득을 얻지 못하도록 하기 위함이다.

서로 다른 수표의 액면가 표현 중에서 특정 위치에 있는 색인 값이 우연히 일치할 수도 있지만 문제가 될 정도의 확률을 가지려면 매우 많은 수의 수표를 인출해야 한다. 뿐만 아니라 유효기간이 만료된 수표의 색인은 다시 사용할 수 있어서 현실적으로 문제를 일으킬 가능성은 매우 작다. 자세한 분석은 다음 장에서 다룬다.

## V. 시스템 분석

이 장에서는 제안된 시스템의 안전성을 분석하고, 기존 시스템과 비교하여 장단점을 분석한다. 또한




[그림 7] 환불 프로토콜

제안된 시스템의 문제점을 보완하기 위한 몇 가지 변형에 대해서도 논한다.

### 5.1 안전성

**[위조불가능성]**

은행을 제외한 어느 누구도 은행의 서명키를 알 수 없으므로 서명을 직접적으로 위조할 수 없다. 제한적 은닉서명은 정리 4에 의해 적응적 선택(adaptive chosen) 메시지 공격 하에서도 $(l, l+1)$ 위조가 계산적으로 어려우므로 이 시스템에서 수표를 위조한다는 것은 계산적으로 용이하지 않다. 정리 5에 서명의 입력값 $\widetilde{C}_i$가 결정된 후에는 생성자 $g_b$만을 이용하여 $C_i$의 서명을 얻을 수 있다. 또한 기저 $g_b$에 대한 다른 생성자의 이산 대수를 계산하는 것이 어려우므로 다른 생성자로 표현되는 고객의 신원정보나 추적정보를 고객이 자신에게 유리하도록 은닉서명 과정에서 변경할 수 없다.

**[익명성]**

은행은 인출과정에서 $R$, $\widetilde{C}_i$, Proof$(CT_i)$, $E_i$, $CT_i$를 얻으며, 고객은 $C_i$와 Sig$(C_i)$를 얻게 된다. 정리 3에 의해 은행은 서명과정에서 얻은 정보로부터 메시지나 서명 결과에 대해 어떤 정보도 얻을 수 없다. 뿐만 아니라 은행은 은닉서명 전에 수신한 Proof$(CT_i)$, $E_i$, $CT_i$로부터 나중에 $C_i$을 식별할 수 있는 어떤 정보도 얻을 수 없다. 이것은 이산대수 가정에 의해 $\log_{y_u} CT_i$를 계산할 수 없기 때문이다. 또한 은닉서명을 받을 때 포함한 $A_i$, $B_i$, $OT_i$를 보지 못하므로 고객의 익명성은 보장된다.

**[연결불가능성]**

고객은 수표를 인출할 때 받은 $C_1$과 $C_2$를 다른 지불에 사용할 수 있다. 이렇게 사용된 $C_1$과 $C_2$가 은행에 입금되어도 은행은 이것을 서로 연관시킬 수 없다. 그것은 두 지불의 트랜스크립트에서 두 값을 연결시킬 수 있는 정보가 없기 때문이다. 금액정보나 공개된 액면가 색인을 이용하여도 연결할 수 없다. 그 이유는 환불까지 진행되어도 $C_1$과 $C_2$에 있는 액면가 정보 $R$의 표현 중 $d_{l+1}$은 공개되지 않기 때문이다. 또한 두 값에 같은 $y_u$(또는 $R$)가 포함되어 있음을 증명하기 위해서는 $C_i$에서 $y_u$(또는 $R$)를 나눈 결과값의 표현을 알고 있어야 한다. 그러나 정리 1에 의해 $C_i$를 만든 고객을 제외하고는 어느 누구도 그것의 표현을 다항시간 내에 알아낼 수 없다. 따라서 이 방법으로 $C_1$과 $C_2$를 연관시키기는 것은 계산적으로 용이하지 않다.

**[이중사용]**

고객은 $C_i$에 대한 서로 다른 여러 가지 표현을 알고 있으면 자신의 신원을 노출하지 않고 $C_i$를 여러 번 사용할 수 있다. 그러나 따름정리 1에 의해 어떤 값에 대한 서로 다른 두 가지 표현을 알아내는 것은 계산적으로 용이하지 않다. 또한 고객은 각 $C_i$마다 미리 명시한 $A_i$를 이용하여 도전에 대한 응답을 해야 하기 때문에 은행은 이중사용한 고객의 신원을 밝혀낼 수 있다. 한편 은행은 $C_i$의 표현을 모르기 때문에 고객에게 이중사용에 대한 누명을 씌울 수 없다. 그렇게 하려면 수신한 트랜스크립트와 다른 유효한 트랜스크립트를 만들어야 하지만 이는 $C_i$의 표현을 알아야만 가능하다. 그러나 정리 1에 의해 고객을 제외한 어느 누구도 알 수 없으므로 누명을 씌울 수 없다.

**[초과사용]**

초과사용은 $C_1$과 $C_2$를 사용하였을 때 중복되는 색인이 최소한 하나 이상 존재하게 된다는 사실을 이용하여 고안하였다. 초과사용된 경우에는 인출자 추적을 통해 부정을 한 고객을 밝혀낸다. 그러나 이중사용처럼 직접 두 지불의 트랜스크립트를 이용하여 고객의 공개신원정보를 계산할 수 없기 때문에 신뢰기관을 통해서만 책임자를 밝혀낼 수 있다는 문제가 있다. 그러나 이런 메커니즘이 있다는 것만으로 많은 고객은 초과사용을 하지 않을 것이다. 초과사용도 은행은 $C_i$의 표현을 모르기 때문에 고객에게 누명을 씌울 수 없다.

**[중복 색인]**

서로 다른 수표의 액면가 표현 중에서 특정 위치의 값이 우연하게 일치할 수도 있다. 만약 일치하면 아무런 부정을 하지 않았음에도 불구하고 초과사용 또는 이중으로 환불요청을 한 것처럼 된다. $n$개의 수표를 인출하고 길이가 $l$인 생성자 튜플을 이용하여 액면가를 표현한다고 하자. 이 때 서로 다른 수표의 액면가 표현 중에서 특정 위치의 값이 우연하게 일치할 확률은 $q$개의 공이 들어 있는 항아리에서 복원

추출하여 $n \times l$행렬을 채울 때, $l$개의 열 중에 중복되는 값이 있는 열이 있을 확률과 같다. 열에 중복되는 값이 하나도 없을 확률은 다음과 같다.

$$\left(1-\frac{1}{q}\right)\left(1-\frac{2}{q}\right)\cdots\left(1-\frac{n-1}{q}\right) \quad (2)$$

그런데 $e^{-x}=1-x+x^2/2!-x^3/3!+\cdots$가 성립하므로 $x$가 매우 작은 수이면 $1-x\approx e^{-x}$로 근사화할 수 있다. 따라서 식 (2)는 다음으로 근사화할 수 있다.

$$\prod_{i=1}^{n-1}\left(1-\frac{i}{q}\right)\approx\prod_{i=1}^{n-1}e^{\frac{-i}{q}}=e^{\frac{-n(n-1)}{2q}}$$

그러므로 중복되는 값이 있는 열이 하나도 없을 확률은 다음과 같다.

$$1-e^{\frac{-n(n-1)l}{2q}}$$

이 확률이 $\varepsilon$라 하면 다음이 성립한다.

$$\begin{aligned} e^{\frac{-n(n-1)l}{2q}} &= 1-\varepsilon \\ -n(n-1)l/2q &= \ln(1-\varepsilon) \\ n^2-n &= \frac{2q\ln(1/(1-\varepsilon))}{l} \end{aligned}$$

위 식에서 $n$항을 무시하고, $\varepsilon=.5$이고 $l=10$이면 다음이 성립한다.

$$n\approx\sqrt{\frac{2\ln(1/(1-\varepsilon))}{l}}\cdot\sqrt{q}=0.37\sqrt{q}$$

$q$는 보통 $2^{160}$이상이므로 $n$이 $2^{77}$이상 되어야 중복되는 것이 있을 확률이 50%가 넘는다. 즉, 이 정도의 수표를 인출하여야 중복된 것이 있을 수 있다는 것이다. 이 논리는 해쉬함수에서 충돌 가능성을 분석하는 것과 같다. 따라서 특정 위치의 색인값이 우연히 일치할 가능성은 매우 희박하다.

### 5.2 두 번 사용과 두 개 사용의 차이

제안한 시스템은 한 번 사용된 수표의 잔액에 대해 다시 사용할 수 있도록 개선한 것이다. 새 시스템이 기존 수표시스템에서 두 개의 수표를 인출하는 것과 비교하였을 때 어떤 장단점을 지니고 있는지 비교하면 다음과 같다. 먼저 연산량 측면에서 비교하면 새 시스템에서는 수표를 인출할 때 두 개 값에 은닉서명을 수행하므로 기존에 두 개의 수표를 인출하는 것과 유사한 비용이 소요된다. 지불 과정에서 새 시스템은 기존 시스템과 같은 수준의 연산량이 소요되며, 환불 프로토콜의 수행을 한 번 줄여주는 효과가 있으므로 전체적으로 요구되는 연산은 기존보다 적다고 할 수 있다. 또한 수표의 활용 측면에서 보면 새 시스템은 한 번 쓰고 남은 수표를 다시 한 번 쓸 수 있으므로 수표의 유용성이 향상되었다. 더욱이 새 시스템에서는 기존 시스템과는 다르게 지불한 금액과 환불받는 금액간에 직접적인 보수관계가 성립하지 않으므로 서로 연관시키는 것을 어렵게 만들어 익명성을 높였다.

### 5.3 시스템 변형

새 시스템의 이런 장점에도 불구하고 수표를 두 번째 사용할 때에는 지불대금에 대한 제약이 있다. 이것은 Chaum의 온라인 수표시스템에서 거스름의 재사용이 제한적인 것과 동일하다. 이것을 해결하기 위한 세 가지 방법을 생각할 수 있다.

**[방법 1]**

이 방법은 액면가를 나타내기 위해 사용하는 생성자 튜플의 길이를 기존 보다 두 배로 늘리고 수표의 액면가도 두 배로 증가하여 사용하는 방법이다. 이 때 상위 절반의 색인은 $C_1$을 지불할 때 사용하고, 나머지 색인은 $C_2$를 지불할 때 사용하도록 한다. 이 방법은 지불대금에 대한 제약 문제를 해결할 수 있으나 기존 시스템에서 두 개의 수표를 인출하여 사용하는 경우와 비교하면 환불 과정만 한번으로 축소된다는 것 외에는 좋아지는 점이 없다.

**[방법 2]**

이 방법은 액면가를 나타내는 각 생성자가 최소 금액 100원을 나타내도록 변경하는 방법이다. 즉, 액면가를 나타내는 생성자 튜플의 길이가 $l$이면 수표의 액면가는 $l\times100$원이 된다. 이렇게 하면 색인이 길이가 길어진다는 것을 제외하고는 모든 문제가 해결된다.

**[방법 3]**

이 방법은 액면가를 나타내는 생성자 튜플의 길이

를 기존 보다 세 배로 늘려 사용하는 대신 방법 1과 달리 액면가는 기존과 동일하게 사용하는 방법이다. 예제를 통해 이 방식에 대해 설명하면 다음과 같다. 먼저 $l=4$라 하자. 그러면 $d_{12}$부터 $d_1$까지 액면가 표현을 위해 사용하게 된다. 이것을 세 부분으로 나누어 $d_{12}$부터 $d_9$까지는 환불받기 위해 사용하고, $d_8$부터 $d_5$까지는 두 번째 지불에서 사용하며, $d_4$부터 $d_1$까지는 첫 번째 지불에서 사용한다. 두 번째 지불은 부정을 방지하기 위해 첫 번째 지불보다 많은 정보를 전달하여야 한다. 예를 들어 첫 번째 지불에서는 700원을 지불한다고 하자. 그러면 $d_3$, $d_2$, $d_1$을 이용하여 지불을 하게 된다. 두 번째 지불에서는 첫 번째 지불에서 사용하고 남은 금액에 대해 자유롭게 지불할 수 있다. 두 번째에서 600원을 지불한다면 $d_7$과 $d_6$뿐만 아니라 첫 번째 지불에서 700원을 사용하였음을 나타내고 거스름을 200원밖에 받을 수 없도록 하기 위해 $d_{12}$, $d_{11}$, $d_9$, $d_3$, $d_2$, $d_1$을 공개한다. 상점은 반드시 이들을 확인하여 수표의 액면가를 초과해서 지불되고 있지 않음을 확인하여야 한다. 이 방법에서 부정한 책임자를 찾는 방법은 기존과 동일하다. 이 방법도 제약 문제를 해결하지만 첫 번째 지불과 두 번째 지불이 상호 연결된다는 것과 색인의 길이가 증가한다는 단점이 있다.

새 시스템은 수표를 인출할 때 두 개의 값에 서명을 받아 사용한다. 이것을 확장하여 $n$개의 값에 서명을 받아 $n$회까지 다시 사용할 수 있도록 확장할 수도 있다. 그러나 $n$이 증가하면 할수록 인출비용이 증가하는 문제점이 있을 뿐만 아니라 사용하지 않는 값들이 많으면 불필요한 오버헤드가 많아진다.

## Ⅵ. 결 론

오프라인 시스템은 은행의 참여 없이 지불을 처리하기 때문에 그 과정에서 발생한 거스름이 화폐의 기능을 갖도록 만드는 것은 어렵다. 따라서 기존 시스템은 거스름의 재사용을 고려하지 않고 수표를 두 부분으로 구성하여 사용하고 있다. 수표의 두 부분 중 한 부분은 지불에 사용되고, 다른 한 부분은 지불 후 발생된 거스름 금액만큼 환불받기 위해 사용된다. 그런데 인출하는 수표의 액면가가 고정되어 있으므로 지불에 사용된 부분과 환불받기 위해 사용된 부분은 금액가치 측면에서 서로 보수 관계가 성립된다. 이 관계는 고객의 익명성을 저해하는 요인이 된다. 뿐만 아니라 수표를 한 번밖에 사용할 수 없기 때문에 나머지 잔액은 항상 다시 입금되어야 하는 불편한 형태의 지불방식이 된다. 이 논문에서는 이런 문제점을 개선하고자 수표의 잔액을 다시 쓸 수 있는 오프라인 시스템을 제안하였다.

이 논문에서 제안한 시스템은 표현 문제를 이용하여 수표의 액면가를 표현하며, 제한적 은닉서명을 사용하여 수표를 인출한다. 이렇게 인출된 수표는 쓰고 남은 금액에 대해 다시 지불할 수 있으며, 같은 수표가 사용된 지불행위를 서로 연관시킬 수 없도록 하였다. 뿐만 아니라 이중사용과 초과사용 등의 부정행위가 발생하면 그 책임자를 찾을 수 있도록 고안하였다. 이를 위해 표현 문제와 이산 대수 문제를 기반으로 하는 영지식 증명을 이용하고 있다. 또한 익명의 수표가 범죄에 악용되는 것을 방지하기 위한 익명성 제어 기능도 제공한다. 인출에서 환불까지 소요되는 전체 연산비용은 기존 수표시스템에서 여러 개의 수표를 인출하여 사용하는 경우보다 적은 수준의 비용이 요구된다. 따라서 한번 쓰고 나머지는 반납해야 하는 기존의 수표와는 달리 보다 적은 비용으로 수표의 잔액을 다시 쓸 수 있어서 편리함과 활용도를 높이고 동시에 고객의 익명성을 향상시켰다. 그러나 액면가 표현방법의 한계 때문에 수표의 남은 금액에 대해 자유롭게 지불할 수 있게 하려면 추가적인 연산이 들어가야 하는 단점이 있다. 앞으로 적은 비용으로 남은 금액에 대해 자유롭게 사용할 수 있는 효율적인 액면가 표현방법에 대한 연구가 필요하다.

## 참 고 문 헌


[1] S. Brands, "Untraceable Off-line Cash in Wallets with Observers," *Advances in Cryptology, Crypto 1993*, LNCS 773, pp. 302~318, Aug. 1993.

[2] D. Chaum, "Online Cash Checks," *Advances in Cryptology, Eurocrypt 1989*, LNCS 434, pp. 288~293, Apr. 1989.

[3] R. H. Deng, Y. Han, A. B. Jeng, T. Ngair, "A New On-Line Cash Check Scheme," *Proc. of the 4th ACM Conf. on Computer and Communications Security*, pp. 111~116, Apr. 1997.

[4] S. Kim, H. Oh, "Making Electronic Refunds Reusable," *Proc. of the 2nd Int. Workshop*

*on Information Security Applications, WISA 2001*, pp. 125~138, Sept. 2001.
[5] D. Chaum, B. den Boer, E. van Heyst, S. Mjoelsnes, A. Steenbeek, "Efficient Offline Electronic Checks," *Advances in Cryptology, Eurocrypt 1989*, LNCS 434, pp. 294~301, Apr. 1989.
[6] R. Hirschfeld, "Making Electronic Refunds Safer," *Advances in Cryptology, Crypto 1992*, LNCS 740, pp. 106~112, Aug. 1992.
[7] S. Brands, "An Efficient Off-Line Electronic Cash System based on the Representation Problem," CWI(Centrum voor Wiskunde en Informatica) Technical Report, CS-R9323, Mar. 1993.
[8] A. de Solages, J. Traore, "An Efficient Fair Off-line Electronic Cash System with Extensions to Checks and Wallets with Observers," *Proc. of the 2nd Int. Conf. on Financial Cryptography, FC 1998*, LNCS 1465, pp. 275~295, Feb. 1998.
[9] T. Okamoto, K. Ohta, "Universal Electronic Cash," *Advances in Cryptology, Crypto 1991*, LNCS 576, pp. 324~337, Aug. 1991.
[10] D. Chaum, J. Evertse, J. van de Graaf, R. Peralta, "Demonstrating Possession of a Discrete Logarithm without Revealing it," *Advances in Cryptology, Crypto 1986*, LNCS 263, pp. 200~212, Aug. 1986.
[11] D. Chaum, T. P. Pedersen, "Wallet Databases with Observers," *Advances in Cryptology, Crypto 1992*, LNCS 740, pp. 89~105, Aug. 1992.
[12] C. P. Schnorr, "Security of Blind Discrete Log Signatures Against Interactive Attacks," *Proc. of the 3rd Int. Conf. on Information and Communications Security, ICICS 2001*, LNCS 2229, pp. 1~13, Nov. 2001.
[13] J. Hastad, "Some Optimal Inapproximability Results," *Proc. of the 29th ACM Symp. on Theory of Computing*, pp. 1~10, May 1997.
[14] A. H. Chan, Y. Frankel, P. D. MacKenzie, Y. Tsiounis, "Mis-representation of Identities in E-cash Schemes and how to Prevent it," *Advances in Cryptology, Asiacrypt 1996*, LNCS 1163, pp. 276~285, Nov. 1996.

## 〈著者紹介〉

**김 상 진 (Sangjin Kim) 정회원**
1995년 2월 : 한양대학교 전자계산학과 졸업
1997년 2월 : 한양대학교 전자계산학과 석사
1997년 3월~현재 : 한양대학교 컴퓨터공학과 박사과정
〈관심분야〉 정보보호, 전자화폐, 전자수표
URL:http://distcomp.hanyang.ac.kr/~sangjin/

**오 희 국 (Heekuck Oh) 종신회원**
1983년 : 한양대학교 전자공학과 졸업
1989년 : 아이오아주립대학 전자계산학과 석사
1992년 : 아이오아주립대학 전자계산학과 박사
1993년~1994년 : 한국전자통신연구원 선임연구원
1995년~현재 : 한양대학교 전자컴퓨터공학부 조교수
〈관심분야〉 정보보호, 전자상거래, 이동컴퓨팅
URL:http://cse.hanyang.ac.kr/~hkoh/