익명성을 제공하는 효율적인 소액지불 프로토콜에 관한 연구

김해만[†] · 이임영[††]

요 약

현재 많은 관심을 받고 있는 전자상거래는 앞으로 더욱 활성화되어질 전망이다. 전자상거래는 디지털 데이터의 구입과 같은 소액 거래가 쉽게 이루어질 수 있다. 소액 거래는 비록 거래 비용은 적지만 그 응용 분야가 넓기 때문에 중요성을 가진다. 이러한 소액 거래를 위해서는 처리비용이 적어야만 한다. 소액 거래를 위한 많은 프로토콜이 연구되고 있는데, 본 고에서는 지금까지 개발된 대표적인 소액지불 프로토콜을 살펴보고 기존의 소액지불 프로토콜에서 대부분 지원하지 않았던 익명성을 부여한 새로운 소액지불 프로토콜을 제안한다.

A Study on Efficient Micro Payment System

Kim Hae-Man[†] and Lee Im-Yeong[††]

ABSTRACT

We are very interested in electronic commerce today, and it will is more developed in the future. In electronic commerce, we can easily purchase a cheap goods like digital data using a micropayment system. Though small value is used in micropayment system, it is important because it has many application. The transaction cost of a micropayment system must be small. Many protocols for the micropayment system are studied, but it's not support user's anonymity. In this paper, we examines the existing micropayment system and proposes a new micropayment protocol that have a anonymity.

1. 서 론

컴퓨터 보급의 확산과 네트워크를 이용한 컴퓨터 통신의 발달로 인터넷 사용자가 전세계적으로 급증함에 따라 이를 상업적으로 이용한 전자상거래가 많은 주목을 받고 있다.

전자상거래는 원래 미국 국방 예산의 비용을 절감하기 위한 효율적인 운영에 대한 방안으로 구상된 CALS 및 EDI로부터 발전한 개념으로서, 정부 및 기업의 모든 거래에 필요한 전자적 수단을 통칭한다. 하지만 최근에는 전자상거래의 개념이 통신망을 이용한 거래 행위를 가리키게 되었다.

전자상거래가 등장하게된 배경을 살펴보면 우선 인터넷 이용자수가 증가했다는 것을 들 수 있다. 적은 투자만으로도 수많은 사람들을 고객으로 확보할 수 있다는 것은 상업적인 면에서 커다란 매력이 아닐 수 없다. 다음으로는 안전한 전자상거래를 위한 보안기술의 발전을 들 수 있다. 전자상거래는 공개된 네트워크를 통해서 이루어지는데 만약 보안상의 문제가 있다면 전자상거래는 널리 이용되지 않을 것이다. 따라서 현재 이런 보안상의 문제를 해결하기 위한 많은 해결책이 연구, 제안되어지고 있다. 또한 판매자가 자신의 상품을 충분히 홍보할 수 있는 쇼핑몰의 구축이 용이하다는 점도 전자상거래가 발전하게 된 중요한 요소라 할 수 있다. 전자상거래가 이루어지기 위해서는 판매자는 상품을 충분히 홍보할 수

본 연구는 한국과학재단 특정기초연구과제 (과제번호 : 97-01-00-06-01-3) 연구비 지원에 의해 수행되었음

[†] 순천향대학교 전산학과 대학원
[††] 순천향대학교 컴퓨터학부 조교수

있어야 하고, 구매자는 원하는 상품에 대한 정보를 얻을 수 있어야 하는데 이러한 것을 인터넷의 홈페이지를 통하여 쉽게 구축하고 찾아 볼 수 있게 된 것이다.

전자상거래는 지불 방식에 따라 지불브로커 시스템과 전자화폐 시스템으로 나눌 수 있다. 지불브로커 시스템은 독립적인 신용구조를 가지지 않고 신용카드나 은행이 계좌를 이용해 네트워크상에서 지불을 하는 방식이다. 현재 널리 사용되고 있는 신용카드를 이용할 수 있기 때문에 손쉽게 적용할 수 있다는 장점이 있다. 전자화폐 시스템은 독립적인 신용구조를 가지고 있어서 물품 구입시 은행이나 카드 발행사로부터의 거래 승인이 필요없다. 전자화폐 시스템은 플라스틱 카드 위에 부착된 IC칩을 이용해 오프라인 대금결재에 활용하는 IC카드형 전자화폐와 화폐가치를 디지털 정보의 형태로 발행하여 네트워크를 통한 온라인 대금결재를 가능하도록 한 네트워크형 전자화폐로 나눌 수 있다.

또한 전자화폐를 지불 금액의 정도에 따라 고액지불 시스템과 소액지불 시스템으로 구분할 수 있다. 고액지불 시스템은 고액의 금액을 안전하게 지불하기 위한 시스템으로 높은 보안성과 안전성이 요구되고 따라서 많은 처리비용이 필요하게 된다. 소액지불 시스템은 1달러 미만의 소액 금액도 거래가 가능하도록 하는 시스템이다. 이를 위해서는 보안성과 안전성의 강도는 다소 낮아지더라도 적은 처리비용이 필요하게 된다.

본 고에서는 기존에 제안된 소액지불 시스템에 대해서 분석하고 기존의 소액지불 프로토콜에서 제공되지 않았던 익명성을 보장함으로써 보다 안전하고 효율적인 소액지불 시스템을 제시하고자 한다.

2. 소액지불 시스템

소액 거래는 비록 거래 단위가 작지만 실생활에서 아주 커다란 부분을 차지하고 있다. 소액 거래를 위한 지불 시스템을 구축한다면 많은 새로운 비즈니스를 제공할 수 있게 될 것이다. 즉, 신용카드와 같이 거래 비용이 많이 드는 지불 시스템에서는 다룰 수 없었던, 인터넷상에서 잡지, 신문, 만화, 음악, 비디오 등의 거래가 가능하게 될 것이다. 또한 소비자는 보다 효율적인 구매가 가능하게 된다. 예를 들면, 어떤 부분적인 정보를 위해서 책 전체를 구입할 필요가 없게 된다는 것이다.

소액지불 시스템을 이루기 위해서는 거래를 위한 처리비용을 최대한 낮춰야 하는데, 이를 위한 고려사항을 살펴보면 다음과 같다.

- 계산량의 감소 : 안전한 전자상거래를 위해서는 암호 기술[8]이 필요하게 된다. 그 중에서 RSA와 같은 경우 안전성도 높고 서명에 사용할 수 있지만 처리 속도가 느리다는 단점이 있다. 일반적으로 RSA 서명 검증은 해쉬함수보다 약 100배 정도 느리고, RSA 서명 생성은 해쉬함수보다 약 10,000배 정도 느리다고 알려져 있다. 따라서 소액지불 시스템을 위해서는 해쉬함수와 같이 처리속도가 빠른 암호기술을 사용하여 처리비용을 줄이도록 해야 할 것이다.
- 통신량의 감소 : on-line 시스템의 경우, 매 거래마다 브로커에 접속을 통하여 인증을 하게 된다. 따라서 통신량이 증가하므로 처리비용도 증가하게 된다. 그러므로 인증이 가능한 전자화폐 등을 이용하여 off-line으로 직접 거래가 가능하도록 해서 통신량을 최대한 줄여야 할 것이다.
- 신용 risk의 감소 : 신용카드거래와 같은 후불 방식에서는 고객의 신용 risk가 생기게 되고, 이는 수수료를 높이는 원인이 된다. 따라서 선불 방식과 같은 방법으로 고객의 신용 risk를 없애면 처리비용을 줄일 수 있을 것이다.
- DB의 감소 : 데이터의 인증이나 법률적인 논쟁 등의 경우를 위해 일정한 데이터를 DB로 관리해야 하는데, 관리해야할 데이터가 많을 수록 처리비용은 증가할 것이다. 따라서 관리해야 할 데이터를 줄이는 것도 소액지불 시스템을 위한 중요한 요소가 된다.

소액 거래를 지원하기 위한 많은 프로토콜이 제안되고 있는데, 대표적인 것으로 Millicent[1,2,5], MPTP[3], Mini-Pay[4,5], Payword[6], MicroMint[6], Cyber-Coin[2,5,7] 등이 있으며, 그 중 몇가지에 대하여 분석하고자 한다.

2.1 Millicent

1997년 3월 11일, Digital Equipment Corporation에서는 인터넷상에서 진정한 의미의 소액 상거래를 이루기 위해 Millicent 계획을 발표하였다.

2.1.1 기본적인 메커니즘

가. 참여 주체

- Customer : Broker로부터 scrip을 구입한 후 vendor와 거래한다.
- Vendor : Customer로부터 scrip을 받고 서비스와 거스름 scrip을 제공한다.
- Broker : Customer와 broker의 계정을 관리하고 scrip과 실제 돈의 거래를 다룬다.

나. Scrip

- 특정한 vendor에게 사용이 가능한 전자 현금을 의미하며, broker와 vendor가 발행을 할 수 있다.
- Scrip의 구조

Vendor	Value	ID#	Cust_ID#	Expires	Props	→ Scrip Body
H(Scrip Body ll master_scrip_secret)						→ Certificate

  - Vendor : scrip을 발행한 상거래 서버의 ID.
  - Value : scrip의 화폐 가치.
  - ID# : scrip의 이중사용을 막기 위한 scrip의 유일한 번호.
  - Cust_ID# : scrip을 사용하는 사용자의 ID.
  - Expires : scrip의 유효기간.
  - Pros : 기타 데이터.
  - Certificate : scrip에 대한 변조 여부의 확인을 위한 인증서

다. 보안 메커니즘

- master_scrip_secret 사용
  - scrip의 유효성을 확인하기 위한 인증서인 certificate를 생성하기 위해 사용되어진다.
  - certificate 생성방법은 DB에 있는 master_scrip_secret들 중 하나를 scrip_body부분의 ID#를 이용하여 선택한 후 scrip_body부분과 함께 해쉬함으로써 생성된다.
- master_customer_secret 사용
  - customer_secret를 생성하는데 사용되어진다.
  - customer_secret의 생성방법은 DB에 있는 master_customer_secret들 중 하나를 scrip_body부분의 Cust_ID#를 이용하여 선택한 후 Cust_ID#와 함께 해쉬함으로써 생성된다.

2.1.2 Millicent 프로토콜

Millicent 프로토콜의 과정은 다음과 같다.

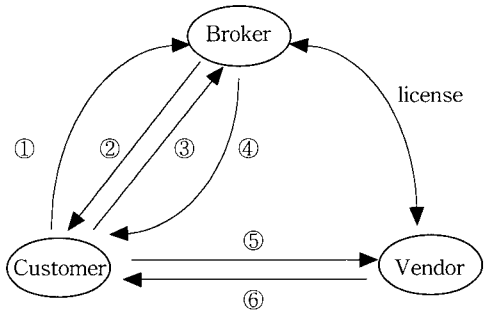


그림 1. Millicent 프로토콜 흐름도

(1) ①, ② 단계(customer_secret, broker_scrip구입)

안전한 채널상에서 broker_scrip과 사용자의 비밀키인 customer_secret을 broker로부터 받는다. Customer_secret은 계속하여 재사용되기 때문에 사용자는 이 키를 안전하게 보관하여야 한다.

(2) ③, ④ 단계(vendor_scrip 구입)

사용자는 ①, ② 단계에서 구입한 broker_scrip으로 거래하고자 하는 vendor의 scrip을 구입한다. 사용자가 제시한 broker_scrip과 broker가 발행한 vendor_scrip의 차액은 다시 새로운 거스름 broker_scrip을 발행함으로써 해결한다. Broker는 certificate를 이용해서 사용자로부터 받은 broker_scrip의 유효성을 확인하고, ID#필드의 일련 번호를 통해서 이중사용을 확인한다.

(3) ⑤, ⑥ 단계 (상품 구입)

Customer는 vendor에게 request문을 보낸다. Request문은 (request || scrip || H(request || scrip || customer_secret))로 구성이 되어 있다. 제 3자는 customer_secret를 알 수 없으므로 부정한 request문을 만드는 것이 불가능하다. Request문에 대한 유효성 검사를 한 후에 vendor는 사용자가 선택한 상품의 배송과 함께 거스름 scrip을 만들어 사용자에게 보낸다.

2.1.3 프로토콜 분석

Millicent 프로토콜은 비밀 정보를 이용하여 해쉬

와 인증을 함으로써, scrip이나 request문을 제 3자가 위조하거나 변경할 수 없다. 또한 vendor에서 ID#를 점검함으로써 이중사용을 방지할 수 있다. 그리고 scrip의 구입이 선불 방식이므로 customer의 신용이 필요 없다는 장점이 있다.

반면에 Millicent는 익명성이 보장되지 않고, 거래시 vendor의 거스름돈이 필요하게 된다. 또한 vendor가 scrip 생성을 위한 모든 정보를 알고 있기 때문에 부정한 scrip 생성이 가능하다.

2.2 MPTP(Micro Payment Transfer Protocol)

MPTP는 1995년에 나온 W3C Working Draft로써 현재 새로운 버전은 나오지 않고 있지만 새로운 token 생성 방식의 사용과 프로토콜에 대하여 많은 관심을 받고 있다.

2.2.1 기본적인 메커니즘

가. 참여 주체

MPTP도 Millicent와 마찬가지로 참여하는 주체는 customer, vendor, broker로 구성되어 있다.

나. 지불 메커니즘

지불 명령은 크게 지불 authority와 지불 token으로 나누어지는데 지불 authority는 지불에 필요한 paychain_root 값, 식별자 등 지불에 필요한 정보를 디지털 서명한다. 지불 token은 연쇄 해쉬 함수를 이용하여 금액을 결정한다.

여기에서 연쇄 해쉬 함수란 token을 생성하고 인증하기 위해 사용되는데, 우선 customer는 랜덤한 w_n 값을 선택한 후 w_i = h(w_i+1)를 계산함으로써 일련의 지불 토큰(paychain) w_0, w_1, · · ·, w_n을 계산한다. 이 때, 금액은 해쉬 횟수에 의해서 결정된다. 여기서 paychain을 생성하는 최초의 root값을 paychain_root라고 한다. h는 MD5와 같은 one_way 해쉬 함수를 나타낸다.

2.2.2 MPTP 프로토콜

지불 처리는 Session Establishment 단계와 Payment Transfer 단계로 나눌 수 있다.

가. Session Establishment 단계

Session Establishment 단계는 지불 처리를 하기 위해 미리 지불 처리에 필요한 정보(authority, 계정 확인서)를 broker, customer, vendor가 서로 주고받는 단계이다. 이 단계는 거래를 하기 전에 한번만 수행을 하고 그 후에는 이 정보를 기반으로 계속적인 거래를 할 수 있다.

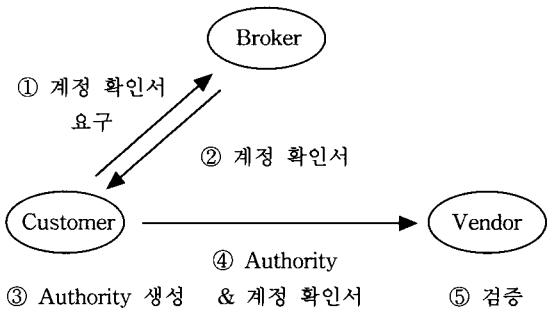


그림 2. MPTP 프로토콜 흐름도 : Session Establishment 단계

(1) ①, ② 단계

Customer는 broker에게 계정 확인서를 요구하고 계정 확인서를 받는 단계이다.

만약 broker가 공개키 서명을 사용하여 계정 확인서를 생성하면, vendor는 broker의 서명을 확인함으로써 계정 확인서를 인증한다. 만약 broker가 대칭키 서명을 사용할 경우 vendor는 broker에게 계정 조회를 의뢰한다.

(2) ③ 단계

Customer가 authority를 생성하는 단계이다. Authority는 authority를 인증할 수 있는 정보와 paychain을 생성할 수 있는 정보를 포함한다.

(3) ④ 단계

Customer는 authority에 서명을 해서 계정 확인서와 함께 vendor에게 전송한다.

(4) ⑤ 단계

Vendor는 authority와 계정 확인서의 서명을 확인하고 유효기간 및 이중사용 등을 검증한다.

나. Payment Transfer

지불 전송하는 단계로써 세션 설정이 이루어진 후 계속해서 지불 처리를 할 수 있다.

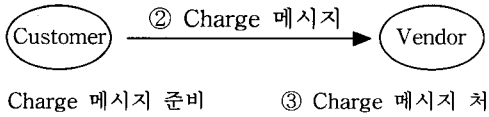


그림 3. MPTP 프로토콜 흐름도 : Payment Transfer 단계

(1) ① 단계

Customer가 charge 메시지를 준비하는 단계로써, charge 메시지는 금액을 결정할 수 있는 정보인 authority_id와 payword 등의 정보를 포함한다.

(2) ② 단계

Charge 메시지를 전송한다.

(3) ③ 단계

Charge 메시지를 점검하는 단계이다. Authority_id를 이용해서 해당 paychain_root를 선택한 후, 이것을 사용하여 연쇄 해쉬 함수를 수행함으로써 payword를 확인한다.

2.2.3 프로토콜 분석

제 3자는 동전을 생성할 수 있는 정보(paychain_root)를 모르기 때문에 부정한 동전을 생성할 수 없고, authority 식별자의 점검으로 이중사용을 방지할 수 있다. 또한 customer가 거래 금액에 맞는 동전을 생성하므로 broker의 부하가 감소하고, vendor의 거스름돈이 필요 없다는 장점이 있다.

반면에 MPTP는 익명성이 보장되지 않고, customer의 신용 risk가 생기고, vendor의 부정한 동전 생성이 가능하다는 단점이 있다.

2.3 Mini-Pay

Mini-Pay는 IBM에서 소액지불을 위해 개발한 프로토콜이다.

2.3.1 기본적인 메커니즘

가. 참여 주체

- Buyer : 상품을 구입하는 사람으로 MiniPay Wallet이 제공되어진다.
- Seller : 서비스 제공자이다.
- IAP(Internet Access Provider) : buyer의 billing system이다.
- ISP(Internet Service Provider) 또는 bank : seller의 billing system이다.

나. 메커니즘

- 하나의 기관이 ISP와 IAP의 기능을 모두 수행할 수도 있다.
- IAP와 ISP의 연결은 직접 연결하거나 중간에 하나 또는 두개의 은행을 통해서 할 수도 있다.

2.3.2 Mini-Pay 프로토콜

Mini-Pay의 프로토콜의 과정은 다음과 같다.

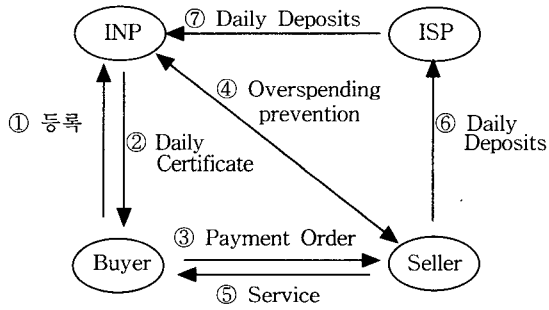


그림 4. Mini-Pay 프로토콜 흐름도

(1) ① 단계 (등록)

계정을 설정하고 이 계정에 사용될 공개키를 확인하는 단계이다. Buyer는 IAP로부터 비밀값 code_B를 받고 공개키(PUB_B)와 개인키(PRIV_B)를 생성한 후, IAP에게 PUB_B, S_B(Reg_req, H(code_B, salt1, PUB_B, acct_B, time)), salt1을 전송하여 등록을 요구하고, IAP는 buyer에게 S_IAP(Reg_res, OK/fail_code, acct_B, PUB_B, time, fees)를 전송함으로써 응답을 한다.

(2) ② 단계 (Daily Certificate)

이 단계는 매일 시작할 때 한번 실행을 하는 단계로써 전날 잔액을 결산하고 IAP는 offline_limit를 포함하는 daily certificate인 Daily_B를 제공한다. Offline_limit는 IAP에게 on-line 확인없이 날마다 buyer가 off-line으로 거래할 수 있는 최대 구매 금액을 나타낸다.

Daily_B의 형식은 다음과 같다.

Daily_B=S_IAP(Daily_res,OK/fail_code,acct_B,
PUB_b, time, offline_limit,
H(total_lim, salt, real_bal))

(3) ③ 단계 (Payment Order)

Buyer는 seller에게 payment order인 Pay_Order 생성하여 Daily_B와 함께 전송하는 단계이다. Pay_Order의 형식은 다음과 같다.

Pay_Order = S_B(Order, amount, day_total, acct_B, time, URL, acct_S)

Seller는 Pay_Order의 서명을 확인하고 공개키를 획득한다. Buyer의 총 사용금액이 offline_limit를 초과하지 않은 경우에는 곧바로 (5)단계를 수행하고 초과한 경우에는 (4)단계를 수행한다.

(4) ④ 단계 (Overspending prevention)

Seller가 IAP에게 Pay_Order를 보내면, IAP는 seller에게 추가적인 금액의 사용을 승인하거나 거절하는 메시지를 전송한다. 이 단계는 상대적으로 드물 것이다.

(5) ⑤ 단계 (Service)

Seller는 요구되어진 정보나 서비스를 buyer에게 제공한다.

(6) ⑥ 단계 (Daily Deposits)

Seller는 payment order를 모은 후 time과 함께 서명한 후 ISP에게 전송을 하고 결제를 요구하면 ISP는 seller에게 결제한 결과를 전송한다.

(7) ⑦ 단계 (Daily Deposits)

(6)단계와 유사한 동작으로써, ISP는 payment order를 모은 후 time과 함께 서명한 후 해당 IAP에게 전송하고 결제를 한다.

2.3.3 프로토콜 분석

서명을 이용함으로써 안전한 메시지 인증은 가능하지만 계산량이 다소 많아질 것이다. 오버헤드를 줄이기 위해서 여러 데이터를 한꺼번에 처리하도록 하였고, Daily Certificate의 offline_limit를 통하여 off-line 거래가 가능하도록 함으로써 통신량을 줄일 수 있도록 하였다. 하지만 buyer가 offline-limit 금액으로 여러 seller와 거래한다면 buyer는 자신이 사용할 수 있는 금액의 범위를 넘어서 사용할 수 있는 문제가 있다. 또한 후불 방식으로 IAP가 신용 risk를 부담하게 된다.

3. 새로운 소액 지불 시스템

본 제안 방식에서는 기존에 제안된 프로토콜의 장·단점을 분석하여 보다 효율적인 프로토콜이 되도록 하였다.[9] 기존의 방식에서는 대부분 익명성을 제공하지 않고 있는데, 익명성은 개인의 프라이버시를 위해 요구되는 기본적인 조건이라 할 수 있다. 본 제안 방식에서는 신뢰할 수 있는 기관인 CA를 통하여 간단한 방식으로 익명성을 제공하도록 하였다.[10,11] 또한 불법적인 사용자에 대해서는 CA와 broker의 협조를 통해서 추적도 가능한 방식이다.

3.1 기본적인 메커니즘

가. 참여 주체

- Customer : broker에게 입금을 한 후 vendor와 거래한다.
- Vendor : customer에게 서비스를 제공한다.
- Broker : customer와 vendor 사이의 실제 돈의 흐름을 관리하고 거래를 인증한다.
- CA : 신뢰된 기관으로써 customer의 신원을 확인하고 PID(pseudo_ID)를 제공한다.

나. Paychain의 생성 방법

기본적으로 paychain을 구성하는 방식은 MPTP의 방식과 유사한 연쇄 해쉬를 통하여 얻어진다. 다음의 식과 같은 연쇄 해쉬를 통하여 일련의 payword인 $C_1, C_2, \ldots, C_n$ 을 생성하게 된다.

$$C_{i+1} = h(C_i \,||\, \text{paychain\_root} \,||\, SN)$$

여기서 최초의 값인 C_0은 비밀값인 paychain_root 값을 의미한다. 그리고 SN은 일련 번호로써 C_i값에 가변성을 제공하게 된다.

다. Paychain_root 값 생성 방법

기존의 MPTP 방식에서는 customer가 해당 vendor에 대한 paychain_root 값을 생성하고 이를 Session Establishment 단계를 통하여 설정을 하게 되는데 이는 다수의 customer가 다수의 vendor와 거래를 할 때 비효율적이다. 따라서 이를 효율적으로 처리하기 위해 본 제안 방식에서는 Diffie-Hellman의 키 교환 방식[8]을 이용하여 paychain_root 값을 결정한다.

이 방식의 절차는 다음과 같다.

(1) customer와 vendor들은 각각의 비밀값 c_i와 v_i를 생성한다. 이 때 g^{c_i}와 g^{v_i}는 공개값이 된다.

(2) 각각의 customer와 vendor 사이에 교환된 비밀키가 paychain_root 값이 된다.

$$paychain\_root = g^{c_i v_i}$$

또한, 이러한 Diffie-Hellman의 키 교환 방식으로 각각의 비밀키 BCu_key(broker와 customer), BCA_key(broker와 CA), BV_key(broker와 vendor) 등을 분배할 수 있다.

라. 금액 결정 방법 (금액 생성 범위 인증서)

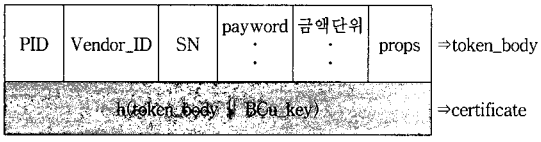
본 제안 방식에서는 선불 방식으로 처리하기 위해 customer는 먼저 broker에게 지불을 하고 해당 금액에 맞는 금액 생성 범위 인증서를 받는다.

여기서 금액 생성 범위 인증서는 지불된 금액에 해당하는 금액의 생성 범위를 인증하게 되며, 형식은 다음과 같다.

금액 생성 범위 인증서 =
M_max || PID || E || SN || h(M_max || PID || E || SN || BV_key)

여기서 M_max는 사용 가능한 최대 금액, PID는 customer의 Pseudo_ID, E는 이 인증서의 유효 기간 등을 각각 나타낸다. 또한 SN은 Serial Number로써 이 인증서의 부정한 사용을 막기 위해 사용되어지고, BV_key는 broker와 vendor의 비밀키로써 이 비밀키를 모르는 제 3자는 이 인증서를 조작할 수 없도록 하는 역할을 한다.

마. Token의 구조



· token_body
- PID : customer의 Pseudo_ID
- Vendor_ID : token을 사용할 vendor의 ID
- SN : token의 일련 번호
- payword : paychain_root를 해당 금액만큼 해쉬한 값
- 금액 단위 : payword가 의미하는 금액 단위 결정
- props : 기타 데이터

· certificate
: token_body와 BCu_key(broker와 customer의 비밀키)를 해쉬하여 생성

3.2 지불 프로토콜

제안 방식의 프로토콜의 과정은 다음과 같다.

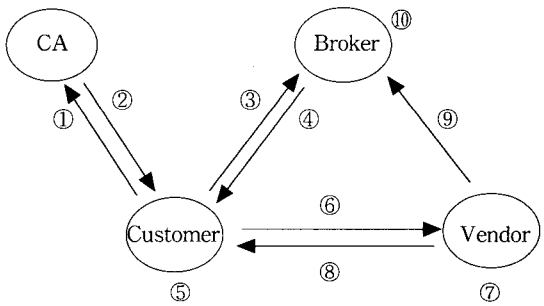


그림 5. 제안 방식 프로토콜 흐름도

(1) ① 단계 (PID 요구)

익명성을 유지하기 위한 PID를 요구하는 단계이다. Customer는 CA에게 자신의 실제 신원정보를 전송하고 PID를 요구한다.

(2) ② 단계 (PID 전송)

Customer에 신원을 확인하고 PID와 PID 인증서를 제공한다. PID 인증서 형식은 다음과 같다.

PID_certificate = h(PID || BCA_key)

여기서 BCA_key는 broker와 CA가 공유한 비밀값으로 제 3자가 조작할 수 없도록 한다.

(3) ③ 단계 (금액 생성 범위 인증서 요구)

Customer는 자신이 사용할 적당한 금액을 선불로 입금하고, PID와 PID 인증서를 전송함으로써 금액 생성 범위 인증서를 요구한다.

(4) ④ 단계 (금액 생성 범위 인증서와 공개값 인증서 전송)

Broker는 customer로부터 받은 PID와 BCA_key

로 해쉬함으로써 PID 인증서를 검증하고 지불된 금액에 맞는 금액 생성 범위 인증서를 생성하여 전송한다. 또한 공개값 인증서를 생성하여 전송한다. 공개값 인증서 형식은 다음과 같다.

Pub_certificate = h(PID || g^c || BV_key)

여기서 공개값 g^c에 사용되는 비밀값 c는 broker와 customer의 비밀값인 BCu_key를 의미한다.

(5) ⑤ 단계 (token 생성)

Customer는 vendor에 접속해서 원하는 상품을 선택한 후 해당 금액에 맞는 token을 생성한다.

Paycain_root값은 해당 vendor의 공개값 g^v와 자신이 생성한 비밀값 c로 생성된 g^{cv}이다.

(6) ⑥ 단계 (request문 전송)

Customer는 상품 정보, token, 금액 생성 범위 인증서, 자신의 공개값 g^c를 이용하여 생성한 request문을 전송한다. request문의 형식은 다음과 같다.

request = 상품 정보 || token || 금액 생성 범위 인증서 || g^c || Pub_certificate
|| h(상품 정보 || token || 금액 생성 범위 인증서 || g^c || Pub_certificate || paychain_root)

여기서 비밀값인 paycahin_root값을 이용하여 해쉬함으로써 이 값을 모르는 제 3자는 request문을 조작할 수 없다.

(7) ⑦ 단계 (검증)

Vendor는 우선 Pub_cerificate를 통하여 PID와 g^c 값을 검사한 후, g^c값과 자신의 비밀값 v를 이용하여 paychain_root값인 g^{cv}를 계산한 다음 paychain_root 값을 이용하여 해쉬함으로써 request문을 검증하고, token의 금액을 검사하고, 자신의 DB를 이용하여 금액 생성 범위의 초과 여부를 검사한다. 모든 검사가 올바르게 이루어졌다면, vendor는 DB의 사용 금액 정보를 갱신한다.

(8) ⑧ 단계 (상품 전송)

⑦단계에서 인증이 올바르다면 vendor는 customer에게 해당 상품을 전송한다.

(9) ⑨ 단계 (token 반환)

Vendor는 모아진 token을 자신의 계좌에 예치하기 위해 broker에게 반환한다

(10) ⑩ 단계 (token 확인 후 인출)

Broker는 customer와의 비밀값인 BCu_key를 이용하여 해쉬함으로써 token의 certificate를 검사한 후 해당 금액을 조사하여 올바르다면, 해당 금액만큼 vendor의 계정에 예치한다. 모든 과정이 올바르게 이루어졌다면, 여기서도 ⑦ 단계와 유사하게 broker는 자신의 DB 정보를 갱신한다.

3.3 프로토콜 분석

제안 방식의 특성을 살펴보면 다음과 같다.

가. 익명성 및 부분적인 추적성 제공

신뢰할 수 있는 기관인 CA가 PID를 발급함으로써 간단하게 익명성을 제공할 수 있도록 하였다. CA는 customer의 실제 신원 정보와 PID에 관한 정보만 알고 거래 내역을 알 수 없기 때문에, CA에 대한 customer의 익명성이 보장된다. 또한 customer는 거래를 할 때 CA로부터 받은 PID를 이용하기 때문에, broker와 vendor는 customer의 PID와 거래 내역만을 알 수 있다. 따라서 PID에 대한 실제 신원 정보를 모르는 broker와 vendor에 대해서도 customer의 익명성은 보장된다. 즉, 각각은 실제 customer에 대한 거래 내역을 알 수가 없다. 그러나 불법적인 사용과 같이 추적이 필요한 경우에는 CA와 broker의 협조를 통하여 신원을 조회할 수 있는 부분적인 추적성을 가진다. Customer는 PID에 대한 인증서를 broker에게 전송해야 하기 때문에 자신의 신원을 속일 수 없다.

나. 안전성 제공

Token이나 request문 등 전송되는 메시지에 당사자들만이 알고 있는 비밀값을 이용하여 해쉬함으로써 이 값을 모르는 제 3자가 메시지를 조작하거나 위조할 수 없도록 하였다.

다. Customer의 부정 방지

Customer가 상품 금액과 다른 값을 사용하거나 overspending 하는 경우, vendor는 payword와 금액

생성 범위를 검사함으로써 곧바로 알 수 있다.

라. Vendor의 부정 방지

Vendor는 임의로 token을 생성하여 broker를 속일 수 없다. 왜냐 하면 BCu_key값을 모르는 vendor는 token의 certificate 부분을 올바르게 생성할 수 없기 때문이다. 또한 vendor가 모아둔 token을 중복해서 broker에게 사용할 경우, broker는 token의 SN (Serial Number)를 통하여 이중 사용 여부를 알 수 있을 것이다.

마. 처리 비용(통신량 및 계산량)

①, ② 단계는 거래를 시작하기 전에 한번만 수행하면 되고, 선택적으로 수행할 수도 있다. 즉 customer가 특별히 익명성을 요구하지 않을 경우에는 수행하지 않아도 된다. ③, ④ 단계는 입금한 금액을 다 사용했을 경우에만 수행하면 된다. 즉, 입금한 금액을 다 사용할 때까지는 customer와 vendor만으로 실제 거래가 이루어질 수 있다는 것을 의미한다.

사용된 암호 알고리즘을 살펴보면 모든 과정에서 속도가 빠른 해쉬 알고리즘만을 사용함으로써 처리비용을 줄일 수 있도록 하였다.

앞장에서 기술한 기존의 프로토콜과 제안 방식의 처리비용을 통신량, 계산량, 신용 risk 측면에서 비교해보면, 통신량 측면에서는 모든 방식이 off-line형이기 때문에 통신량은 큰 차이가 없다. 계산량 측면에서 보면 Millicent가 거래시 customer는 1번(request 문의 certificate 생성시), vendor는 6번(request문 확인, scrip 확인, 거스름 scrip 생성을 위해 각각 2번의 해쉬 요구)의 해쉬만을 이용하면 되기 때문에 가장 빠르고, 제안 방식과 MPTP는 해쉬 횟수에 의해서 금액이 결정되기 때문에 사용되는 금액에 따라 해쉬 횟수가 변하게 된다. 제안 방식에서는 해쉬 횟수를 줄이기 위해서 token의 필드에 '금액단위'라는 항목을 부여하였다. 예를 들어 한번 해쉬한 값이 1원을 나타낼 경우, 111원을 생성하기 위해서 MPTP는 111번 해쉬를 해야하지만 제안 방식에서는 3번(각 금액 단위별로 1번씩)만 하면 된다. 따라서, 사용 금액이 가변적일 경우에는 제안 방식이 훨씬 효율적이다. Mini-Pay는 기술된 방식 중에서 유일하게 공개키 암호를 사용함으로써 가장 느리다. 신용 risk측면에서 보면 Millicent와 제안 방식은 선불 방식으로 신용 risk가 없는 반면 MPTP와 Mini-Pay는 후불 방식으로 신용 risk가 생김으로써 처리비용이 높아지게 된다. 종합적으로 처리비용을 비교해보면 Millicent, 제안 방식, MPTP, Mini-Pay와 같은 순서로 처리비용이 적다.

바. token의 사용 제약

본 방식에서 vendor에서 customer의 부정을 발견할 수 있도록 하기 위하여, 기존의 Millicent나 MPTP와 같이, 특정한 token은 특정 vendor에게 사용되도록 하였다. 즉, 새로운 vendor와 거래하기 위해서는 broker를 통해 새로운 vendor에 맞는 금액 생성 범위 인증서를 받아서 거래를 해야 한다는 것을 의미한다.

4. 결 론

앞으로 전자상거래는 더욱 활성화될 전망이고 그 중에서 소액 거래는 전자상거래의 특성상 그 응용범위가 넓기 때문에 더욱 중요하게 부각될 것이다. 소액 거래를 위해서는 처리 비용을 최대한 줄여야 하는데 가장 중요한 요소가 통신량과 계산량이다. 통신량을 줄이기 위해서는 off-line형 시스템이 적합하고 계산량을 줄이기 위해서는 속도가 느린 RSA와 같은 공개키 암호 알고리즘 대신에 속도가 빠른 해쉬함수를 이용하는 것이 좋다. 본 고에서는 off-line형과 해쉬함수만을 이용한 프로토콜을 제안함으로써 소액 지불에 적합하도록 하였다. 또한 기존의 제안된 소액 지불 프로토콜의 장·단점을 분석해서, 각 프로토콜의 장점을 살려 보다 효율적인 프로토콜이 되도록 하였고 기존에 소액지불에서는 제공되지 않았던 익명성을 간단한 방법으로 제공할 수 있는 방식을 제안하였다.

기존의 방식과 제안 방식을 비교해보면 표 1과 같다.

표 1에 나타난 것과 같이 본 제안 방식은 기존의 방식과 비교하여 효율적이라 할 수 있다. 앞으로 처리 비용을 낮추면서도 보다 안전하고 효율적인 프로토콜을 개발하여 실용화한다면, 실생활에서의 편리함과 다양한 비즈니스 제공 등 많은 이익을 제공할 것이다.

참 고 문 헌

[1] Steve Glassman, Mark Manasse, Mart Abadi,

표 1. 각 소액지불 프로토콜의 비교

	Millicent	MPTP	Mini-Pay	제안 방식
시스템 분류	**off-line형**	**off-line형**	**off-line형**, on-line형	**off-line형**
지불 시점	**선불 방식**	후불 방식	후불 방식	**선불 방식**
신용 risk	×	○	○	×
안전성	○	○	○	○
이중사용방지 (overspending)	○	○	(×)	○
vendor의 부정방지	×	×	○	○
거스름돈	○	×	×	×
익명성	×	×	×	○

Paul Gauthier and Patrick Sobalvarro, "The Millicent Protocol for Inexpensive Electronic Commerce", http://HTTP.CS.Berkeley.EDU/~gauthier/millicent/millicent.html

[2] Peter Wayner, "Digital Cash", AP professional, pp 217-227, 1997

[3] Phillip M. Hallam-Baker, "Micro Payment Transfer Protocol(MPTP) Version 0.1", W3C Working Draft, http://www.w3.mag.keio.ac.jp/TR/WD-mptp-951122, 1995

[4] Amir Herzberg and Hilik Yochai, "Mini-Pay : Charging per Click on the Web" http://www6.nttlabs.com/HyperNews/get/PAPER99.html

[5] Petri Aukia and Jean-Baptiste Lehmann, "Mechanisms in electronic commerce using micropayments", http://studwww.eurecom.fr/~lehmann/study/

[6] Ronald L. Rivest and Adi Shamir, "PayWord and MicroMint: Two simple micropayment schemes", Technical report, MIT LCS, 1996. 5

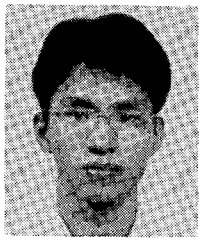
[7] CyberCash Inc., http://www.cybercash.com/

[8] 최용락, 소유영, 이재광, 이임영, 통신망 정보 보호, 도서출판 그린, 1996

[9] 김해만, 채승철, 이임영, "새로운 소액지불에 관한 연구", 한국멀티미디어학회 춘계학술발표논문집, pp 181-186, 1998. 6. 5

[10] 김해만, 이임영, "분할성과 부분적인 추적이 가능한 효율적인 전자 지불 시스템에 관한 연구", '98 한국 멀티미디어학회 추계학술 발표 논문집, pp 49-54, 1998. 12. 12

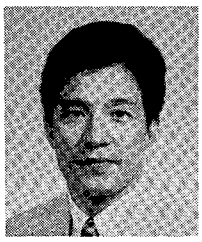
[11] 김해만, 이임영, "3익명성을 부여한 소액지불 프로토콜에 관한 연구", CISC '98, pp 123-135, 1998. 12. 18



김 해 만

1998년 2월 순천향대학교 전산학과 졸업

1998년 3월~현재 순천향대학교 전산학과 대학원



이 임 영

1977년~1981년 홍익대학교 전자공학과 졸업

1984년~1986년 일본 오오사카 대학교 통신 공학 석사

1986년~1989년 일본 오오사카 대학교 통신 공학 박사

1989년~1994년 한국전자통신 연구원 선임연구원

1994년~현재 순천향대학교 컴퓨터공학부 조교수