mPay : 초소액 지불시스템
(mPay : A New MicroPayment System)

신 준 범 † 이 광 형 ††
(Jun-Bum Shin) (Hyung Lee-Kwang)

요 약 본 논문에서는 효율성, 안전성 및 이중사용방지 요구 조건을 만족하면서 동시에 시스템 사용 과정에서 생길 수 있는 여러 분쟁 유형에 대해서 효과적으로 대처할 수 있는 초소액 지불시스템을 제안한다. 이를 위하여 기존의 시스템들이 효율성 향상을 위해 많이 사용해 온 S/KEY 방식의 해쉬 체인을 변형한 이중 해쉬체인 구조를 제안한다. 제안 시스템인 mPay-1, mPay-2는 이 구조를 기반으로 하였으며 안전성 및 분쟁해결성 측면에서 좋은 결과를 보인다. mPay-1은 기존의 시스템과 동일한 효율성을 갖으나 보다 안전하다. mPay-1의 확장형인 mPay-2는 추가적으로 여러 분쟁 유형들에 대해 효과적으로 대처할 수 있다.

Abstract In this paper, we propose a MicorPayment system which satisfies the following requirements: efficiency, security, double spending protection, and accountability. We, also, propose a new double hash chain mechanism which is an extension of S/Key hash chain. The proposed systems, mPay-1 and mPay-2, are based on the proposed mechanism and the systems show better results on security and accountability. Analysis shows that mPay-1 is efficient and more secure and that mPay-2, which is an extended version of mPay-1, can handle additional dispute types.

1. 서 론

네트워크를 보다 효율적으로 이용하기 위한 움직임의 결과로 네트워크를 통한 상거래인 전자상거래가 등장하였다. 그러나 현재 네트워크는 보안을 고려하여 설계되지 않았기 때문에 네트워크를 통하여 상품을 구매하는 것은 보안성 측면에서 많은 문제점을 갖는다. 전자지불 시스템은 사용자가 네트워크를 통하여 안전하게 상품을 구매할 수 있도록 도와주는 시스템이다. 전자지불 시스템은 위에서 언급한 보안성 문제를 해결함은 물론, 실제 상거래 과정에서 생길 수 있는 여러 문제점들을 해결할 수 있도록 설계되어야 한다[1]. 그리고 전자지불 시스템 설계 과정에서 고려되어야 할 대표적인 요소들 중 대표적인 것들은 다음과 같다.

- 안전성: 프로토콜은 외부의 공격으로부터 안전해야 한다.
- 이중사용 방지: 같은 지불데이타를 두 번 사용할 수 없어야 한다.
- 분쟁해결성: 지불과정에서 생길 수 있는 분쟁 유형에 대해 효과적으로 대처할 수 있어야 한다.
- 효율성: 지불 처리비용이 저렴해야 한다.
- 사생활 보호: 거래 과정에서 거래자의 개인정보가 노출되는 것을 막을 수 있어야 한다.

현재까지 개발된 전자지불 시스템들은 각기 다른 특성을 갖는다. 이러한 전자지불 시스템 중, 지불 처리비용을 최소화하여 상대적으로 적은 금액의 지불(10원 또는 100원부터 10,000원까지)에 사용할 수 있는 것을 초소액 지불 시스템(MicroPayment System)이라 한다. 초소액 지불 시스템을 사용하여 구매할 수 있는 대표적인 상품은 네트워크를 통하여 전송 가능한 신문, 저널 등의 기사 및 주식 정보, 음악 및 그림 파일, 그리고 자바 애플릿 등과 같은 작은 소프트웨어가 있다[2].

초소액 지불 시스템이 다른 전자지불 시스템과 차별화되는 가장 큰 특징은 효율적 지불 처리를 통한 지불 처리비용의 최소화이다. 그리고 현재까지 제안된 대부분의 초소액 지불 시스템들은 효율성을 극대화 할 수 있도록 설계되었다. 대표적인 것들로는 μ-iKP[3], Net-

† 비 회 원 : 한국과학기술원 전자전산학과
jbshin@monami.kaist.ac.kr
†† 종신회원 : 한국과학기술원 전자전산학과 교수
khlee@fuzzy.kaist.ac.kr
논문접수 : 1999년 5월 3일
심사완료 : 2000년 5월 6일

Card[4], PayWord[5], CAFE(phone call)[6], MPTP [7], NetCheque[8], 그리고 Millicent[9] 등이 있다. 그러나 위의 시스템들은 전자지불 시스템이 가져야 할 중요한 조건 중 하나인 분쟁해결성이 부족하며, 실제적으로 분쟁이 발생할 경우 이의 해결을 위하여 많은 비용이 든다는 단점을 갖는다. 이러한 측면은 전체적으로 볼 때, 개별 거래의 처리비용을 증가시키는 효과를 낳는다. 이러한 문제점을 해결하기 위해서 NetBill[10], Mini-Pay[11]등의 시스템이 제안되었다. 그러나 NetBill의 경우 안전성 및 분쟁해결성 측면에서는 매우 우수하나 효율성 측면에서 다른 시스템에 비해 부족하고, Mini-Pay는 분쟁해결성이 NetBill에 비해 부족하다.

본 논문에서는 이러한 기존 연구들의 단점을 극복하기 위하여 mPay-1, mPay-2, 두 개의 초소액 지불 시스템을 제안한다. 본 논문에서는 이들 시스템의 분석을 위하여 초소액 지불 시스템이 가져야 할 기본 조건들인 효율성, 안전성 및 이중사용방지, 그리고 분쟁해결성과 관련된 기본적인 요구조건들을 제안한다. mPay-1은 제안된 요구조건 중, 효율성, 안전성, 그리고 이중사용 방지 기능을 만족한다. 또한 기본적인 분쟁해결성 요구조건을 만족한다. mPay-1의 확장형인 mPay-2는 추가적으로 본 논문에서 제안한 초소액 지불 시스템 요구조건들을 모두 만족한다. 효율성 및 안전성 측면에서 mPay-1은 현재까지 제안된 시스템 중, S/KEY방식[12]을 채택하여 가장 효율성이 높은 기존의 시스템들인 μ-iKP, NetCard, PayWord, CAFE(phone call), 그리고 MPTP 등과 동일하다. mPay-2는 효율성은 S/KEY방식을 사용한 시스템들보다는 조금 떨어지나 그 외 다른 방식을 사용한 시스템들인 Millicent, Mini-Pay 보다는 우수하다. 또한 안전성 및 분쟁해결성은 지금까지 제안된 시스템 중, 가장 우수한 NetBill과 동일하다.

mPay는 이러한 성질들을 만족하기 위하여 S/KEY방식의 해쉬 체인(hash chain)을 변형한 이중 해쉬 체인 구조를 사용한다. 이중 해쉬 체인 구조는 두 개의 해쉬 체인을 함께 사용하는 시스템이다. 하나의 해쉬 체인은 일반적인 해쉬 함수를 사용하여 이루어지며 다른 하나는 앞서 말한 해쉬 체인 데이타의 암호화 및 무결성 확보를 위해 사용된다. 분쟁해결성 측면에서 mPay는 NetBill등에서 사용한 방법과 같이 분쟁 해결 정책을 확립하고, 분쟁이 일어났을 경우 거래 과정에서 얻어지는 데이타로부터 분쟁을 해결할 수 있도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 초소액 지불 시스템 요구조건을 분석 및 제시하고, 3장에서는 제안 시스템에서 사용되는 기본적인 암호 함수를 소개한다. 4장에서는 mPay 시스템의 기반 메커니즘인 이중 해쉬 체인 구조를 정의 및 분석하고, 이를 기반으로 하여 mPay-1을 설계한다. 그리고 mPay-1이 효율성, 안전성 및 이중 사용 방지 요구조건을 만족함을 보인다. 5장에서는 mPay-1의 확장 모형인 mPay-2를 제안하고, 본 논문에서 제안한 모든 요구조건을 만족함을 보인다. 6장에서는 제안한 방식과 기존의 연구들을 비교하고, 7장에서 결론을 맺는다. 그리고 부록에서는 사생활보호 기능을 갖을 수 있도록 제안시스템을 확장하는 방법을 다룬다.

2. 사전준비

초소액 지불 시스템 관점에서 볼 때, 지불 시스템은 초소액 지불 시스템을 연동할 일반적인 지불 시스템(MacroPayment System)과 초소액 지불 시스템(Micro-Payment System)으로 구분할 수 있다. 이 장에서는 일반적인 지불 시스템의 특성과 일반적인 지불 시스템을 사용하여 초소액 지불 시스템으로 확장할 수 있는 방법을 다룬다. 그리고 본 논문에서 사용하는 기본적인 암호함수들을 소개한다.

2.1 지불 시스템 분류

2.1.1 일반적인 지불 시스템

현재 우리가 사용하고 있는 지불 시스템을 네트워크 상에서 운용할 수 있도록 하기 위한 많은 연구들이 있어왔다. 이들 중, 일반적인 지불 시스템은 현재 우리가 사용하고 있는 지불 시스템에 대해서 네트워크에서 사용할 수 있도록 모방한 형태를 갖는다. 현재까지 개발된 전자지불 시스템의 종류는 [13]을 참조 바란다. Asokan등은 전자지불 시스템을 사용자의 구좌에서 금액이 인출되는 시점을 구매 시점을 기준으로 분류하였다[1]. Asokan등이 제안한 분류 기준은 다음과 같다.

- cash-like 모델: 상품 구매 이전에 사용자가 지불 가능한 금액을 가지고 있어야 하는 사전지불(Pre-Paid) 방식의 지불 시스템이 해당된다. 우리가 사용하는 현금, 전화카드 등이 이 모델에 속하며, 전자지불 시스템의 경우에는 ecash[14], NetCash[15]등이 해당된다.
- check-like 모델: 상품 구매 이전에 사용자가 지불 가능한 금액을 가지고 있지 않아도 되는 지불 시스템이다. 동시지불(Pay-Now) 또는 사후지불(Pay-later) 방식의 지불 시스템이 이에 해당된다. 우리가 사용하는 신용카드, 전화요금 지불 등이 이 모델에 속하며, 전자지불 시스템의 경우에는 SET[16], iKP[17] 등이 해당된다.

2.1.2 초소액 지불 시스템

초소액 지불 시스템은 우리가 사용하는 전화카드 또

는 지하철 패스 등과 같이 일반적인 지불 시스템의 확장 형태로 이루어 진다. 사용자는 초소액 지불 시스템 사용을 위한 쿠폰 묶음을 거래에 참여하지 않는 제 3자인 브로커로부터 구매를 하고, 실제 거래 과정에서는 쿠폰을 사용하여 금액을 지불한다. 이 과정을 요약하면 그림 1과 같다.

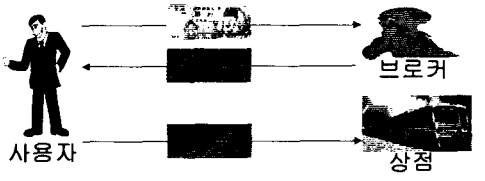

그림 1 초소액 지불 시스템 구성방식

초소액 지불 시스템을 사용할 때, 사용자가 사용가능한 쿠폰 묶음을 가지고 있지 않는 경우에는 쿠폰 묶음을 구입한 후, 실제 시스템을 사용한다고 가정한다. 초소액 지불 시스템 설계와 관련된 이슈로는 브로커로부터 구입한 쿠폰 묶음의 사용에 있어 여러 상점에 지불을 허용할 지 여부이고, 다른 하나는 거래과정에서 브로커의 참여 정도이다.

일반적으로 브로커로부터 구입한 쿠폰 묶음을 하나의 상점에 대해서만 사용할 수 있을 경우, 초소액 지불 시스템의 실효성이 떨어지게 된다. 자세한 언급은 3장을 참조 바란다. 이러한 고찰 하에서 우리는 여러 상점에 대해서 지불이 가능한 경우에 대해서 초소액 지불 시스템을 분류한다. 분류 기준은 거래 과정에서의 브로커의 참여 정도이다.

- 온라인(on-line) 방식: 모든 쿠폰의 사용에 있어 브로커가 실시간으로 참여한다.
- 반-오프라인(semi-off-line) 방식: 상점을 선택할 경우에 대해서만 브로커가 실시간으로 참여하고, 쿠폰 사용의 경우에는 브로커가 실시간으로 참여하지 않는다.
- 완전-오프라인(full-off-line) 방식: 상점 선택 또는 쿠폰 사용 과정 모두에서 브로커가 실시간으로 참여하지 않는다.

우리는 반-오프라인 방식과 완전-오프라인 방식을 합하여 오프라인 방식이라 한다.

2.2 암호 함수

이 장에서는 본 논문에서 사용되는 기본적인 암호 함수들을 기술한다.

2.2.1 전자 서명

전자 서명은 공개키 암호 시스템을 기반으로 한다. 대표적인 전자 서명 프로토콜로는 RSA[18]와 DSS[19]가 있다. 전자 서명 프로토콜은 공개키 암호 시스템을 기반으로 하기 때문에 공개키(public key)와 개인키(private key), 두 개의 키가 사용되고, 특정 공개키로 검증 가능한 서명은 대응되는 개인키를 알아야만 생성할 수 있도록 구성된다. 전자 서명에서 사용되는 공개키는 서명을 확인하고자 하는 사람이 알 수 있다고 가정한다. 공개키가 특정 사용자의 공개키임을 확인하기 위한 방법으로는 PKI(Public-Key Infrastructure)[20] 또는 PGP[21] 방식이 있다. 자세한 내용은 [22]와 [23]을 참조 바란다. 본 논문에서 전자 서명에 사용하는 기호는 다음과 같다.

- 사용자 U의 개인키 : PK_U^{-1}
- 사용자 U의 공개키 : PK_U
- m에 대한 사용자 U의 서명 : $Sign_{PK_U^{-1}}(m)$

2.2.2 암호학적 해쉬 함수 및 키를 갖는 해쉬 함수

암호학적 해쉬 함수(cryptographic hash function)란 해쉬 함수 중, 다음의 성질을 만족하는 것을 가리킨다 (h는 해수 함수이다.).

- 약한 충돌 회피성(weak collision freeness) : 특정 값 a에 대해서 $h(a)$값이 주어졌을 때, $h(b)=h(a)$를 만족하는 a와 다른 b를 찾기 어렵다.
- 단방향 성질(one-wayness) : $h(a)$값을 알 때, a 값을 알기 어렵다.

이러한 성질을 만족하는 대표적인 해쉬 함수로는 SHS[24]와 MD5[25]가 있다.

키를갖는 해쉬함수(keying hash function)는 암호학적 해쉬 함수에 기밀성을 가질 수 있도록 만든 함수이다. 키를갖는 해쉬함수는 암호학적 해쉬 함수를 MAC (message authentication code)에 사용하기 위하여 개발되었다. 키를 갖는 해쉬함수로는 암호학적 해쉬함수의 조합으로 구성되며 대표적인 구성 방식으로는 HMAC가 있다[26]. 키를 갖는 해쉬함수는 다음의 성질을 만족한다.

- 기밀성 : 키 K_{AB}를 모를 경우 특정 값 m에 대해서 $h_{K_{AB}}(m)$값을 계산할 수 없다.
- 약한 충돌 회피성 : 키 K_{AB}를 모를 경우 특정 값 a에 대해서 $h_{K_{AB}}(a)$값이 주어졌을 때, $h_{K_{AB}}(b)=h_{K_{AB}}(a)$를 만족하는 a와 다른 b를 찾기 어렵다.
- 단방향 성질 : $h_{K_{AB}}(a)$값을 알 때, a값 및 K_{AB}을 알기 어렵다.

2.2.3 해쉬 체인

해쉬체인의 기본 구조는 L. Lamport[27]에 의해 제안되었다. 이 구조는 인증 프로토콜인 S/KEY[12]에 사용되며 PayWord를 비롯한 여러 종류의 초소액 지불 시스템에서 사용하였다[3, 4, 5, 6, 7]. 암호학적 해쉬 함수 h를 사용한 해쉬 체인의 구조는 다음과 같다.

- $Chain(n, Seed) = \{w_0, w_1, \cdots, w_n\}$
- $w_i = h(w_{i+1}) = h^{n-i}(Seed)$

초소액 지불 시스템은 다음과 같이 해쉬 체인을 사용할 수 있다. 우선 사용자는 $Seed$를 랜덤하게(random) 생성하고, $Chain(n, Seed)$을 계산한다. 그리고 서버에게 w_0값을 제공한다. 이 과정을 쿠폰 묶음 등록 절차라 한다. 실제적으로 해쉬 체인을 사용하기 위해서 만일 j번째 사용하는 경우라면 j번째의 쿠폰, 즉 w_j값을 서버에게 주면 된다. 서버는 이 값을 받은 다음에 그 이전에 사용자로부터 받은 값인 w_{j-1}을 사용하여 w_j가 식 $w_{j-1} = h(w_j)$를 만족하는지를 검사한다. 이 방식은 매번 접속시마다 사용되는 키 값이 바뀌므로 일회용 패스워드라 불리기도 한다.

이 방식은 암호학적 해쉬 함수의 단방향 성질로부터 w_j를 알아도 그 이후의 hash chain의 값들(w_{j+1},..., $Seed$)을 알 수 없으므로 도청에 안전하다. 안전성 증명은 [6]을 참조 바란다. 해쉬체인을 이용한 방법은 지불처리과정에서 암호학적 해쉬 함수 연산만을 사용하므로 매우 효율적이다. 그러므로 S/KEY가 발표된 이후 제안된 많은 종류의 초소액 지불 시스템들이 이 방식을 사용하였다. 그러나 이 방식은 3장에서 언급할 **Risk 3** 및 **Risk 7~9** 에 취약하다는 단점을 갖는다.

3. 요구 조건 분석

초소액 지불 시스템이 기본적으로 만족하여야 할 요구조건은 효율성, 안전성, 그리고 분쟁해결성을 만족해야 한다. mPay는 설계과정에서 아래의 성질을 충족시킬 수 있도록 하였다.

3.1 효율성

초소액 지불 시스템의 효율성은 거래 처리비용을 최소화 하기 위한 것이다. 본 논문에서는 다음의 요소들를 고려하였다.

- 여러 상점 상품 구매 기능
- 거래 과정에서 브로커 실시간 개입의 최소화
- 암호함수 계산의 고속화
- 효율적 데이타 베이스 처리

초소액 지불 시스템 제안의 가장 큰 목적은 거래 처리비용의 최소화이다. 일반적인 지불 시스템의 경우, 시스템 수수료 등으로 인하여 특정 금액 이상의 상품 구입에만 처리가 가능하다. 그러나 초소액 지불 시스템에서 사용하는 쿠폰이 하나의 상점에만 사용이 가능하다면 쿠폰 묶음이 등록되었을 경우, 등록된 쿠폰은 하나의 상점에서 모두 사용해야만 한다는 단점을 갖는다. 또한 사용하고 남은 금액을 브로커가 환불해 준다고 하더라도 일반적인 지불 시스템 사용 회수의 증가로 인해 거래 처리 비용이 증가하게 된다. 따라서 브로커로부터 구입한 쿠폰 묶음을 여러 상점에 대해 사용할 수 있도록 하는 것이 바람직하다.

- **Req. 1**(여러상점): 브로커로부터 구입한 하나의 쿠폰 묶음을 여러 상점에 사용할 수 있어야 한다.

쿠폰 사용 과정에서 모든 거래에 대해 브로커가 실시간으로 개입할 경우, 서버 부담이 증가하게 되어 서버의 처리 속도가 떨어지게 된다. 서버 부담의 증가는 전체 트랜잭션 처리에 있어 병목현상을 초래할 위험이 크므로 이를 줄일 수 있도록 전체 지불 시스템이 설계되어져야 한다.

- **Req. 2**(오프라인 시스템) : 중앙 서버의 병목 현상을 줄일 수 있도록 자체적으로 거래 데이타의 유효성을 판별할 수 있어야 한다.

초소액 지불 시스템은 안전성 획득을 위하여 여러 암호 함수들을 사용한다. 그러나 암호 함수들은 각각 다른 처리 속도를 갖는다. 암호화 해쉬 함수의 연산 및 비밀키 암호 알고리즘에 의한 암호화의 경우에는 처리 속도가 공개키 암호 연산에 비해 매우 빠르므로 전체 시스템 성능에는 영향을 주지 않는다. 전자 서명 알고리즘의 경우는 보다 복잡한 성능 특성을 갖는다. RSA[18]와 같이 인수분해에 기반한 시스템은 서명 생성 시에 많은 시간이 소요되며 서명 검증시에는 적은 시간이 소요된다. 그러나 DSS[19]와 같이 이산대수에 기반한 시스템은 사전 계산을 적용할 경우, 반대로 서명 생성시에는 적은 시간이 소요되며 서명 검증시에 많은 시간이 소요된다. 따라서 전자지불 시스템 설계시 서버의 서명 생성은 DSS와 같은 시스템을 사용하는 것이 바람직하다. 이 때, 서명 생성을 위한 사전 계산은 서버가 쉬거나 또는 한가할 때 수행한다. 그리고 사용자 서명 생성의 경우에는 RSA와 같은 시스템을 사용하는 것이 권장된다.

- **Req. 3**(암호 함수 사용시 유의 사항) : 공개키 서명 알고리즘의 적절한 사용 및 사용 회수의 최소화.

지불 시스템의 효율성과 관련하여 마지막으로 고려해야 할 요소는 거래 정보를 저장할 데이타베이스(Data Base, DB)의 크기 문제이다. DB 크기가 지속적으로 증가할 경우, 지불 처리비용이 증가하므로 이를 막을 수

있어야 한다.

- **Req. 4**(데이타 베이스 크기) : 중앙 서버 및 상점이 관리해야 할 DB 크기는 사용자 또는 상점의 전체 개수에만 비례하여야 하고, 시간 및 개별 지불 회수와는 관련이 없어야 한다.

3.2 안전성 및 이중사용방지

암호 프로토콜은 시스템의 안전성을 향상시키기 위하여 암호 함수를 사용한다. 그러나, 그 사용의 부적절성으로 인하여 많은 공격 방법들이 제안되었다[23, 28]. 이러한 공격 방법들 중, 일부는 전자지불 시스템을 공격하는 데에서 사용될 수 있다. 전자지불 시스템은 금액의 전달이 주목적이므로 안전성과 관련된 다음의 정의를 내릴 수 있다.

정의 1. *안전성을 만족하는 전자지불시스템은 공격자가 프로토콜에 대한 공격을 통하여 자신이 원하는 상품을 구매할 수 없어야 한다.*

현재까지의 전자지불 시스템은 상품의 전달 과정에서 암호화 과정을 지원하지 않으므로 이와 같은 정의는 전체 시스템의 안전성과 동일하다. mPay는 다음의 위험 요소들에 대하여 정의 1에서 언급한 안전성을 만족함을 보인다.

- **Risk 1** : 사용자가 사용한 데이타를 도청하였다가 다시 사용하는 행위.
- **Risk 2** : 공격자가 상점인 것처럼 가장하여 데이타를 얻는 행위이다.
- **Risk 3** : 사용자가 보낸 지불 데이타를 상점에 전달하지 않고, 가로채어 사용하는 행위.

이 중, **Risk 1**과 **Risk 3**의 차이점은 **Risk 1**은 수동적 공격으로 사용자가 보낸 데이타가 상점에게 전달되나 **Risk 3**은 능동적 공격으로 사용자가 보낸 데이타가 상점에게 보내어지지 않을 수도 있다는 것을 가정한다.

전자지불 시스템이 암호 프로토콜과 다른 가장 큰 특성은 이중사용 방지이다. 암호 프로토콜의 경우 정당한 사용자가 인증을 위하여 같은 값을 두 번 사용하는 것은 문제가 되지 않는다. 그러나 전자지불 시스템을 고려할 경우, 이는 한번의 지불로서 두 번의 구매를 할 수 있다는 것을 의미한다.

- **Risk 4** : 사용자가 같은 쿠폰을 두 번 사용하는 행위

3.3 분쟁해결성

전자 지불 시스템은 대면 접촉을 통한 금액의 지불이 아닌 네트워크를 통한 금액의 지불에 사용되므로 금액을 지불하는자와 받는자 사이에서 분쟁이 발행할 가능성이 높다. 그러나 지금까지 제안된 많은 전자지불 시스템이 이를 고려하지 않았다[29, 30, 31]. mPay는 다음의 분쟁유형들을 고려하였다.

- **Risk 5** : 사용자가 물품을 구매하였을 때, 사용자 계좌의 감소분과 상점 계좌의 증가분이 다름.
- **Risk 6** : 사용자가 상점으로부터 상품을 받고 대금을 지불하지 아니함.
- **Risk 7** : 상점은 사용자로부터 상품 금액을 받고 상품을 제공하지 아니함.
- **Risk 8** : 사용자가 구매한 상품과 받은 상품이 다름.
- **Risk 9** : 사용자의 허위고발 : 사용자가 상점으로부터 상품을 받은 다음, 새로운 상품의 구매를 위하여 받지 않았다고 주장[1].

D. Tygar[29]의 분류에 따르면 **Risk 5**는 금액의 단일성(money atomicity)에 해당하며 **Risk 6** 및 **Risk 7**은 상품의 단일성(good atomicity)에 해당한다. 그리고 **Risk 8**은 배달의 인증성(Certified Delivery)에 해당한다. 본 논문에서는 실제적인 분석을 위하여 D. Tygar의 분류를 조금 더 세분화하였으며 추가적으로 **Risk 9**를 고려하였다.

4. 기본 프로토콜(mPay-1)

mPay-1은 2장에서 기술된 요구 조건 중, 안전성 및 효율성을 만족하는 초소액 지불 시스템이다. 이 장에서는 mPay-1의 기반 메커니즘 이중 해쉬 체인구조를 기술하고, mPay-1의 구성원 및 이들 사이의 관계를 기술한다.

4.1 기반 메커니즘 : 이중해쉬체인

mPay는 해쉬체인의 효율성을 유지하면서 안전성을 높이기 위하여 이중 해쉬체인 구조를 사용한다. 이중 해쉬체인 구조는 두개의 해쉬체인으로 이루어진다. 하나는 일반적인 암호학적 해쉬 함수를 사용한 해쉬 체인이고, 다른 하나는 이 해쉬체인의 데이타를 키를 갖는 해쉬 함수를 사용하여 암호화하기 위한 해쉬 체인이다.

$$Chain1(n, Seed) = \{w_0, w_1, \cdots, w_n\},\quad w_i = h^{n-i}(Seed)$$

$$Chain2(n, Sec) = \{Sec_0, Sec_1, \cdots, Sec_n\},$$

$$Sec_i = h^{n-i}(Sec)$$

이중 해쉬 체인은 challenge-response방식으로 이루어진다. 사용자와 상점이 Sec 및 w_0를 공유하고 있다고 가정할 때, 이 중 j번째 쿠폰, w_j를 사용하여 상품 ID가 ID_{item}인 상품을 구입하는 방법은 다음과 같다.

---

1) NetBill[10]을 제외한 다른 초소액 지불 시스템은 이러한 분쟁이 발생할 경우 정책에 따라 사용자 또는 상점이 일방적인 피해를 입는다. Millicent의 경우 이 상황이 지속적으로 발생할 경우 상점은 운영권을 잃는다[9].

step 1. 상점은 사용자에게 TransID를 준다(TransID는 상점이 관리하며 거래별로 유일한 값이다.).

step 2. 사용자는 $CR_j = h_{Sec_j}(TransID, ID_{item})$값을 계산한다.

step 3. 사용자는 $Pay_j = w_j \oplus CR_j$를 계산하여 상점에게 전달한다.

step 4. 상점은 Sec를 이용하여 $CR_j = h_{Sec_j}(TransID, ID_{item})$를 계산한다.

step 5. 상점은 $w_j = Pay_j \oplus CR_j$값을 계산한다. 만일 이 값이 $w_{j-1} = h(w_j)$을 만족할 경우 지불이 성립된다.

이중 해쉬 체인은 양측이 공유하는 값인 CR_j를 사용하여 j번째 쿠폰 w_j를 XOR 함수를 사용하여 암호화하는 방식을 사용한다. 이 방식은 상품 ID(ID_{item})를 고정하기 위한 challenge-response방식을 사용하였으므로 공격자가 이 값을 가로챈다고 하더라도 이 값을 가지고 자신이 원하는 상품을 선택할 수 없다. 따라서 이중해쉬 체인 구조는 정의 1에 따라 **Risk 3**에 대해 안전하다.

4.2 시스템 구조

mPay-1은 사용자, 상점, 그리고 브로커의 3개 구성원을 갖는다. 사용자와 상점은 실제 상거래가 이루어지는 구성원이다. 브로커(broker)는 이들 사이에서 전체 지불을 관리하는 기관이다. 제안 시스템은 **Req. 1**, 및 **Req. 2** 를 만족하기 위하여 사용자는 상점에 물품을 구매하기 이전에 브로커에게 쿠폰 묶음 생성 인증서를 구매한다. 사용자는 쿠폰 사용 이전에 쿠폰 묶음 생성 인증서를 이용하여 생성한 쿠폰 묶음을 상점에게 등록하는 추가적인 절차를 거친다. 상점은 사용자로부터 받은 쿠폰과 사용자의 쿠폰 묶음 생성 인증서를 이용하여 일정 시점이 지난 후, 브로커에게 쿠폰의 개수에 대응하는 대금을 청구한다. 브로커는 상점이 요구가 있을 경우 요구받은 금액만큼 사용자의 계좌(account)를 감소시키고, 상점의 계좌를 증가시킨다. 각 구성원 사이의 관계를 요약하면 그림 2와 같다.

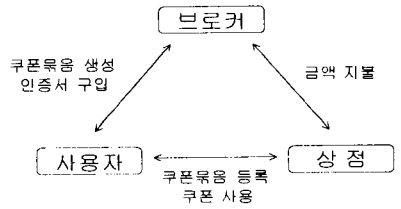

그림 2 제안 시스템 구성도

4.3 사용자 브로커 관계

mPay는 일반적인 지불 시스템과의 연동에 있어 cash-like 모델과 check-like 모델 모두에 적용이 가능하다. 그러나 안전성 향상을 위하여 일반적인 지불 시스템이 전달되는 메시지의 암호화 전송과 전자서명 생성을 지원한다고 가정한다. 이 과정은 Web 표준 암호화 프로토콜인 SSL/TLS[32, 33]에 기반한 지불 시스템의 경우 별도의 전자서명 생성 모듈을 가지고 있을 경우 처리가 가능하며, SET[16] 또는 iKP[17]와 같은 전자지불 전용 프로토콜을 이용할 경우 이미 전자서명 생성 모듈이 프로토콜내에 포함되어 있으므로 약간의 조작으로 처리가 가능하다.

사용자가 브로커로부터 쿠폰묶음 생성 인증서를 받는 절차는 연동되는 일반적인 지불 시스템별로 다르다. 그러나 이들 시스템이 암호화 전송 및 전자서명 생성을 지원한다고 가정할 때, 사용자는 브로커에게 자신이 생성한 난수인 사용자 유사 개인 식별 번호(Pseudo-ID, $PsID_U$)와, 각 상점에 대한 쿠폰묶음 등록에 사용할 사용자의 공개키(PK_U)를 암호화하고 전자서명하여 전송하고, 브로커가 일반적인 지불 시스템에서 상품 구매에 대한 영수증을 지급하는 과정에서 사용자의 쿠폰묶음 생성 인증서(C_U)가 암호화 되어 전송된다고 가정한다.

$$C_U = \{Sign_{PK^{-1}_{Broker}}(CertData),\ CertData\}$$

$CertData$는 브로커 개인 식별 번호(ID_{Broker}), 사용자 유사 개인 식별 번호(Pseudo-ID, $PsID_U$), 사용자 공개키(PK_U), 인증서 유효기간(Ex_{C_U}), 분쟁해결 가능기간(Ex_{DH}), 그리고 구매할 수 있는 최대 금액(MAX)로 구성되어 있다.

$$CertData = \{ID_{Broker},\ PsID_U,\ PK_U,\ Ex_{C_U},\ Ex_{DH},\ MAX\}$$

그리고 쿠폰묶음 생성 인증서(C_U)는 $CertData$에 대한 브로커의 서명이다. 그리고 브로커는 C_U, $PsID_U$, PK_U, Ex_{C_U}, Ex_{DH}, MAX를 DB에 저장한다. 사용자는 쿠폰묶음 생성 인증서를 이용하여 여러 상점과 거래를 할 수 있으나 전체 사용 금액은 MAX를 넘지 않아야 한다.

4.4 사용자 상점 관계

사용자가 쿠폰 묶음을 상점에서 사용하는 절차는 두 단계로 나뉘어 진다. 하나는 쿠폰묶음 생성 인증서를 이용하여 상점에 쿠폰 묶음을 등록하는 절차이고, 다른 하나는 실제적으로 상품의 구매를 위하여 쿠폰을 사용하는 절차이다.

쿠폰 등록 : 사용자는 상점에 쿠폰을 등록하기 위해

해쉬 체인을 생성하고, 해쉬 체인을 상점에 보내어 등록한다. 사용자가 상점에 쿠폰을 등록하는 과정은 다음과 같다.

step 1. 사용자는 두 개의 난수 *Seed*와 *Sec*를 랜덤(random)하게 생성한다.

step 2. 사용자는 이 두 값을 이용하여 *Chain1*, *Chain2*, 두 개의 해쉬 체인을 생성한다.

step 3. 사용자는 *Chain2*를 사용하기 위한 값인 *Sec*는 상점의 공개키를 사용하여 암호화한다.

step 4. 사용자는 *Chain1*을 사용하기 위한 값인 w_0와 다른 관련 값들, 즉 사용자의 인증서(C_U), 상점 개인 식별 번호(ID_V), 그리고 *Sec*의 해쉬 값($h(Sec)$)을 브로커로부터 인증 받은 키로 서명한다. 서명된 데이타는 다음과 같다.

$$Sign_{PK_i^{-1}}(C_U, ID_V, w_0, h(Sec))$$

step 5. 사용자는 다음의 데이타를 상점에 전달한다.

$$C_U, ID_V, w_0, E_{PK_V}(Sec), Sign_{PK_i^{-1}}(C_U, ID_V, w_0, h(Sec))$$

step 6. 상점은 $E_{PK_V}(Sec)$를 복호화 한다.

step 7. 상점은 $Sign_{PK_i^{-1}}(C_U, ID_V, w_0, h(Sec))$값이 맞는지를 검사한다. 그리고 인증서 유효기간(Ex_{C_U}) 과 문쟁해결 가능 기간(Ex_{DH})이 지나지 않았는지를 검사한다.

step 8. step 6. 과 step 7.이 맞을 경우 상점은 C_U로부터 $PsID_U$, *MAX* 값을 얻는다.

step 9. 상점은 $PsID_U$, w_0, *Sec*, 0, *MAX* 값을 DB에 기록한다(상점이 관리해야 할 DB의 크기는 자신에게 등록된 쿠폰묶음의 개수에 비례한다.).

쿠폰 사용 : 사용자는 상품을 선택한 후, 등록된 쿠폰을 사용하여 상품을 구입할 수 있다. 쿠폰의 사용은 이중 해쉬 체인 구조를 사용한 challenge-response방식이다. 이중해쉬 체인에 대한 자세한 내용은 4.1절을 참조 바란다. 사용자가 쿠폰을 사용하는 과정은 다음과 같다. 사용자는 쿠폰 등록후, i개의 쿠폰을 사용했고, 현재 지불해야 할 쿠폰은 k개라 가정한다($j = i + k$).

step 1. 사용자는 상점 서버에 접속하여 상품을 선택한다.

step 2. 사용자는 자신의 유사 ID($PsID_U$) 및 선택한 상품의 ID(*IDitem*)를 상점에 전달한다.

step 3. 상점은 인증서 유효기간(Ex_{C_U}) 과 문쟁해결 가능 기간(Ex_{DH})이 지나지 않았을 경우, 사용자에게 발급할 TransID를 계산한다. 이 값은 사용자의 유사 ID, 상품 ID, 그리고 현재 시각인 타임 스탬프(time stamp, *T*) 값을 *Sec*를 키로 하여 Keyed MAC 함수(h)를 계산한 값이다.

$$TransID = h_{Sec}(PsID_U, IDitem, T)$$

step 4. 상점은 TransID 발급에 사용된 타임 스탬프(time stamp, *T*)와 TransID를 사용자에게 전달한다.

step 5. 사용자는 타임 스탬프가 유효한지를 검사한다.

step 6. 사용자는 TransID가 올바른 값인지를 검사한다.

step 7. 사용자는 TransID를 사용하여 CR_j를 생성한다.

$$CR_j = h_{Sec_j}(TransID, ID_{item})$$

step 8. 사용자는 j번째 지불을 위한 값, w_j를 생성하고, 이 값을 CR_j와 XOR함수를 사용하여 암호화 한 값 ($Pay_j = w_j \oplus CR_j$)을 계산한다.

step 9. 사용자는 상점에게 Pay_j를 전달한다.

step 10. 상점은 이 값을 받은 다음 *Sec*값을 사용하여 CR_j를 생성한다. 그리고 이 값을 Pay_j와 XOR하여 w_j를 복호화 한다.

step 11. 상점은 DB에 기록된 w_i값 및 index(i)를 얻고, w_j값이 상품의 금액만큼의 가치를 갖는지를 검사한다. 상품의 금액이 k일 경우 $w_i = h^k(w_j)$를 만족해야 한다.

step 12. 상점은 사용자에게 상품을 전달한다.

step 13. 상점은 DB에 기록되어 있는 값, w_i 와 index(i)를 각각 w_j 및 index(j) 로 변경한다(이 경우, 쿠폰 사용과 관련하여 상점이 관리해야 할 DB의 크기는 증가하지 않는다).

4.5 상점 브로커 관계

상점은 유효기간이 지난 쿠폰 묶음에 대해서는 브로커에게 해당 금액을 모두 지급 받고, DB에서 삭제한다. 그리고 유효기간이 지나지 않았다고 하더라도 브로커로부터 금액을 지불 받고 싶을 경우, 상점은 사용자로부터 받은 쿠폰을 이용하여 브로커에게 대금 지불을 요구한다. 이 과정은 서버가 한가한 특정 시간에 수행된다. 상점이 사용자 U로부터 j개의 쿠폰을 받았을 경우 상점은 브로커에게 다음의 자료를 전송한다.

$$C_U,\ ID_V,\ Sign_{PK_i^{-1}}(C_U, ID_V, w_0, h(Sec)),\ w_i,\ w_j$$

w_i는 상점이 이 해쉬체인에 브로커에게 지불을 요구한 마지막 값이고, w_j는 새로이 지불을 요구할 값이다. 하나의 해쉬 체인에 대해서 처음 지불을 요구할 경우 $w_i = w_0$이다. 브로커는 이 값을 받은 다음 C_U의 유효성을 검증하고, $Sign_{PK_i^{-1}}(C_U, ID_V, h(CR), w_i,$

w_j, T)로부터 상점의 ID(ID_V) 및 상점이 이 해쉬체인에 대하여 이전에 받은 금액을 계산한다. 브로커는 처음 지불이 요구되는 쿠폰 묶음에 대해서는 C_U, ID_V, w_j에 대한 새로운 DB를 생성한다. 그리고 이전에 지불이 요구된 적이 있는 쿠폰 묶음에 대해서는 데이타베이스에 기록되어 있는 w_i 값을 w_j값으로 바꾼다. 그리고 상점에게 해당 금액을 지불한다. 이 경우, 브로커는 사용자의 쿠폰묶음 생성 인증서인 C_U를 이용해서 생성한 쿠폰묶음 들에 대해서 전체 사용 금액이 C_U에 명시된 상품 구입 가능 최대값인 MAX를 초과하였는지를 검사하고, 만일 초과하였을 경우, 해당 사용자를 부정 사용자로 고발한다.

브로커는 유효기간이 지난 쿠폰 묶음 인증서는 DB에서 삭제한다.

4.6 mPay-1 분석

이 절에서는 mPay-1이 2장에서 기술된 요구 조건 중, **Req. 1~4**를 만족하고, **Risk 1~6**에 대해서 효과적으로 대처할 수 있음을 보인다.

- **Req. 1** : mPay-1은 브로커로부터 구입한 하나의 쿠폰 묶음 사용 인증서를 이용하여 여러 상점과의 거래가 가능하다.

- **Req. 2** : mPay-1은 완전-오프라인 시스템이다.

- **Req. 3** : mPay-1은 지불 과정에서 공개키 암호 알고리즘을 사용하지 않는다.

- **Req. 4** : mPay-1의 DB 크기는 브로커와 상점으로 나누어 생각할 수 있다. 브로커가 저장하여야 할 DB의 크기는 자신이 발급한 쿠폰묶음 사용 인증서와, 그 인증서를 이용하여 생성된 쿠폰 묶음의 개수에 비례한다(쿠폰 사용 회수와는 무관함). 그러나 브로커는 유효기간 이내의 인증서와 그에 해당하는 쿠폰 묶음에 대한 데이타만을 보관하면 된다. 따라서 전체 시스템으로 볼 때, 개별 사용자가 구입한 쿠폰 사용 인증서의 개수 및 등록된 쿠폰 묶음의 개수는 일정 시간이 지나게 되면 일정한 값을 갖게 되므로 브로커가 저장해야 할 DB의 크기는 사용자 또는 상점의 수가 증가하지 않을 경우 일정하다. 상점이 저장해야 할 DB의 크기는 자신에게 등록된 쿠폰 묶음의 개수에만 비례하고, 쿠폰 사용 회수와는 무관하다. 따라서 상점의 경우 역시 효율적인 DB 관리가 가능하다.

- **Risk 1** : mPay-1은 사용자가 쿠폰을 사용할 때마다 지불하는 쿠폰(Pay)이 다르다. 따라서 공격자가 사용자가 사용했던 쿠폰을 안다고 하더라도 그 쿠폰을 자신의 상품 주문을 위하여 사용할 수 없다. 따라서 Risk.1 은 성립하지 않는다.

- **Risk 2** : 사용자가 쿠폰 묶음을 등록할 때, 사용자는 상점의 고유번호가 명시된 전자 서명을 사용한다. 상점이 브로커에게 대금을 청구할 때, 브로커는 사용자의 전자 서명을 기반으로 어느 상점에게 금액을 지급할 것인지를 판단하므로 쿠폰 등록과정에서 사용자가 지명한 상점이 아닌 다른 자는 브로커로부터 금액을 받을 수 없다. 쿠폰 사용과정에서는 공격자가 상점인 것처럼 가장해서 사용자로부터 지불 데이타를 받을 수 있다. 그러나 공격자가 자신이 원하는 상품을 주문하기 위해서는 $Pay_j = w_j \oplus CR_j = w_j \oplus h_{Sec_j}(TransID, ID_{item})$값을 계산해야 한다. 그러나 공격자는 Sec_j를 모르므로 $h_{Sec_j}(TransID, ID_{item})$값을 계산할 수 없다. 따라서 공격자는 자신이 원하는 상품에 대한 Pay_j값을 계산할 수 없다. 따라서 정의 1에 의해서 mPay-1은 Risk 2에 대해 안전하다.

- **Risk 3** : mPay-1에서 지불 과정에서 사용하는 메커니즘인 이중 해쉬 체인 구조는 상품 ID(ID_{item})를 고정하기 위한 challenge-response방식이다. 4.1절에 따르면 공격자는 이 값을 가로챈다고 하더라도 이 값을 가지고 자신이 원하는 상품을 선택할 수 없다. 따라서 mPay-1은 정의 1에 따라 **Risk 3**에 대해 안전하다.

- **Risk 4** : mPay-1은 쿠폰 사용시 사용자와 상점 모두가 쿠폰 인덱스를 검사한다. 따라서 사용자가 같은 쿠폰을 두 번 사용할 경우 상점은 그 쿠폰을 인정하지 않는다. 따라서 mPay-1은 **Risk 4**를 허용하지 않는다.

- **Risk 5.** : 브로커는 사용자와 상점의 계좌 변동을 상점이 제시한 값, 즉 사용자가 사용한 마지막 해쉬 체인의 쿠폰(w_j)으로부터 판단한다. 해쉬 체인의 성질로부터 상점이 w_j를 제시하였다면 사용자는 최소한 j개의 쿠폰을 사용한 것임을 알 수 있다. 따라서 분쟁 발생시 브로커는 w_j값에 의거하여 분쟁을 해결한다.

- **Risk 6** : mPay-1은 사용자가 금액을 지불한 후, 상품을 제공한다. 따라서 사용자의 서비스 거부는 성립하지 않는다.

mPay-1은 **Risk 7~9**에 대해서는 취약하다. 그러나, **Risk 8**은 상점이 정직하다는 가정 하에서 해결할 수 있다.

- **Risk 8** : mPay-1에서 상점은 사용자, 상품, 그리고 거래 시점에 따라 유일한 TransID를 생성하고, 이 값에 기반하여 사용자로부터 받은 값 Pay_j으로부터 실제 쿠폰 데이타 w_j를 얻는다. Pay_j는 TransID 및 상품의 종류에 따라 변하는 값이므로 **Risk 3**에서 언급한

공격 방식에 대하여 안전하다. 따라서 상점이 아는 사용자가 주문한 상품 내용과 사용자가 실제로 주문한 상품의 내용은 동일하다. 따라서 상점이 정직할 경우 사용자에게 제공하는 상품은 사용자가 선택한 상품과 동일하다.

5. 확장 프로토콜(mPay-2)

전자 지불 시스템에서 가장 중요한 기능중의 하나는 분쟁해결성이다. 이 장에서는 mPay-1의 확장 시스템인 mPay-2를 제안하고, mPay-2가 **Risk 7~9**를 해결할 수 있음을 보인다. 그리고 mPay-2를 효율적으로 구현할 수 있는 방안을 제시한다.

5.1 프로토콜

전자 지불 시스템 구성원들 사이에 분쟁이 발생할 경우 자신의 행동이 옳음을 증명할 수 있는 자료(영수증, receipt)가 필요하다. mPay-2는 이 조건을 만족하기 위하여 공개키 전자 서명을 사용한다. mPay-2는 사용자가 상점의 서명 확인키, PK_V를 가지고 있다고 가정한다. 이 키는 쿠폰 묶음을 등록하는 과정 또는 별도의 다른 경로를 통해서 받을 수 있다.

사용자 브로커 관계 : mPay-1과 동일하다.

사용자 상점 관계 : 쿠폰 묶음의 등록 과정은 mPay-1과 동일하다. 그러나 분쟁해결성을 갖기 위해 mPay-1의 쿠폰 사용 과정을 부분적으로 변경한다.

step 1~3. mPay-1과 동일

step 3-1. 상점은 사용자가 지금까지 사용한 쿠폰의 마지막 값(w_j)과, 사용자가 주문한 상품의 금액(k), TrandID, *IDitem*, 그리고 타임스템프(T)에 대한 서명 ($Sign_{PK_V}(TransID, IDitem, w_j, k, T)$)을 생성한다.

step 4. 상점은 TransID, 타임스템프(T), 그리고 step 3-1에서 생성한 서명을 사용자에게 전달한다.

step 5~6. mPay-1과 동일

step 6-1. 사용자는 상점의 서명 확인키를 이용하여 $Sign_{PK_V}(TransID, IDitem, w_j, k, T)$값을 검사한다.

step 7~13. mPay-1과 동일

mPay-2는 분쟁해결성을 강화한 프로토콜이다. 그러나 위에서 변경한 프로토콜은 모든 분쟁 해결 기능이 브로커를 통해서 얻어진다는 단점을 갖는다. 분쟁의 유형 중에서 사용자 또는 상점이 의도적으로 상대방을 속이는 경우에는 브로커를 통한 해결이 필수적이다. 그러나 네트워크 오류로 인하여 이러한 분쟁이 발생할 수 있고, 모든 종류의 분쟁을 브로커를 통해 해결하는 것은 브로커에게 많은 부담을 줄 수 있으므로 이러한 문제에 대한 해결책이 요구된다.

mPay-2는 사용자는 금액을 전달했으나 네트워크 오류등으로 인하여 상품이 전달되지 않는 경우를 해결하기 위하여 다음의 보조 프로토콜을 사용한다.

step 1. 사용자는 자신의 공개키를 이용하여 $PsID_U$, *IDitem*, $Sign_{PK_U}(TransID, IDitem, w_i, k, T)$, 현재 시간($T_{current}$)에 대한 서명($Sign_{PK_U}(PsID_U, IDitem, T_{current}, Sign_{PK_V}(TransID, IDitem, w_i, k, T))$)을 생성한다.

step 2. 사용자는 $PsID_U$, *IDitem*, $Sign_{PK_U}(TransID, IDitem, w_i, k, T)$, $T_{current}$, 그리고 $Sign_{PK_U}(PsID_U, IDitem, T_{current}, Sign_{PK_V}(TransID, IDitem, w_i, k, T))$를 상점에 전달한다.

step 3. 상점은 DB에서 w_j값을 얻는다.

step 4. 상점은 *IDitem*에서 상품의 금액(k)을 확인한 후, $w_i = h^k(w_j)$를 계산한다.

step 5. 상점은 *TransID*를 계산하고, 사용자의 서명이 맞는지를 검사한다.

step 6. 상점은 위의 모든 결과가 맞을 경우 상품을 전달한다.

이 프로토콜은 상품 전달상의 오류를 브로커의 도움없이 1차적으로 해결할 수 있다는 장점을 갖는다.

상점 브로커 관계 : mPay-1과 동일하다.

5.2 mPay-2 분석

mPay-2는 mPay-1의 확장형이므로 **Risk 1~5**를 효과적으로 해결할 수 있다. mPay-1은 상점이 정직하다는 가정하에서 **Risk 8**을 해결할 수 있었으나 mPay-2는 이러한 가정 이 없어도 해결할 수 있다. 그리고 추가적으로 **Risk 7, 9**를 해결할 수 있다. 효율성 요구조건은 5.3절을 참조 바란다.

- **Risk 7** : 사용자는 현재 거래가 있기 전 $l-1$개의 쿠폰을 사용하였다고 가정한다. 따라서 상점은 w_{l-1}값을 알고 있다. 상점의 서비스 거부는 상점이 사용자로부터 받은 Pay_l을 이용하여 w_l값은 알았으나 상점은 이 금액에 해당하는 상품을 제공하지 않은 경우이다. 이와 같은 경우 사용자는 $Sign_{PK_U}(PsID_U, TransID, IDitem, w_i, k, T)$를 이용하여 상점을 브로커에게 고발할 수 있다. 브로커는 이 값을 받은 후, 상점의 서명이 올바른지 여부를 확인한다. 맞을 경우에는 상점에게 w_l값을 보내고 *IDitem*에 해당하는 상품을 사용자에게 보낼 것을 요구한다.

- **Risk 8** : 사용자는 상점으로부터 받은 서명을 이

용하여 상점이 보내는 상품이 자신이 선택한 상품과 동일한 것인지를 확인할 수 있다. 만일 실제로 받은 상품과 주문한 상품이 다른 경우에는 **Risk 7**과 같은 방법으로 대응한다. 이 경우는 상점이 상품을 잘못 보낸 것이므로 상품의 중복 전달이 생긴다고 하더라도 이는 전적으로 상점의 책임이다.

- **Risk 9** : 사용자의 허위 고발 : 사용자는 지불 후, 상품을 받았으나 상점에 해를 끼칠 의도로 상점을 허위 고발할 수 있다. mPay-2는 사용자의 상점에 대한 고발 과정은 사용자가 **Risk 7**에서와 같이 상점에게서 받은 서명값을 브로커에게 보내는 방식을 사용한다. 따라서 사용자가 상점에게서 상품을 받았음에도 불구하고 허위로 고발을 할 경우에 대해서도 사용자는 상점으로부터 원래 받은 상품 이외의 상품은 받을 수 없다. 또한 초소액 지불 시스템을 통하여 구입하는 상품은 디지털 데이타 형태이므로 사용자에게 보낸 상품을 다시 보내도 상점 측에는 아무런 피해가 없다.

5.3 mPay-2 효율성 증진 방안

mPay-2는 효율성 요구 조건 중, **Req 1, 2, 4**를 만족한다. 그러나 쿠폰 사용시 항상 전자서명이 요구되므로 **Req. 3**은 만족하지 못한다. 이를 위해서 실제 서비스 운용에 있어 다음의 정책을 택할 수 있다.

우선 w_j의 (l>1) 번째 사용의 경우는 배제할 수 있다. 이 경우는 모든 거래는 완결되었으나 사용자가 의도적으로 서비스를 방해하기 위하여 상품을 받지 않았다고 할 경우를 대비하기 위한 것이다. 그러나 이러한 사용자 수는 전체 사용자 수에 비해서 매우 적다고 가정할 수 있으므로 전체 시스템의 성능 저하에는 큰 영향을 끼치지 않는다. 따라서 상점 서버의 병목 현상을 막기 위해 $Sign_{PK_V^{-1}}(TransID, IDitem, w_i, k)$의 생성을 효과적으로 처리하는 것이 요구된다.

서버가 서명할 때, 사용하는 알고리즘은 여러 가지가 있을 수 있다. 이 때, 구현하는 방향에 따라서 다음의 방향을 따른다. 각 서명 방식의 효율성 관련 측면은 2.1절을 참조 바란다.

RSA 유형의 서명을 사용할 경우 : RSA 유형의 전자서명은 서명 생성과정에서 사전 계산을 할 수 없다는 단점이 있다. 이를 해결하기 위한 방법 중, 대표적인 것은 SASC(Server Aided Secret Computation)을 사용하는 것이다. SASC를 사용할 경우, 서버 혼자 서명을 생성하는 것에 비해 10배 이상의 효율성 향상을 가질 수 있다[34].

DSS 유형의 서명을 사용할 경우 : DSS 유형의 전자서명은 사전 계산을 통하여 서명 생성시간을 줄일 수 있다는 장점이 있다. 그러나 전자서명 알고리즘의 특성상 사전 계산된 데이타가 한번 사용되었다면 다음에는 사용할 수 없으므로 실제 구현시에는 사전 계산이 특별한 효과를 얻지 못할 수도 있다. 이를 해결하기 위한 방법으로 다음의 두 가지 방법을 사용할 수 있다.

방법 1 : 모듈러 멱승 연산을 효과적으로 수행할 수 있는 방법을 사용한다. 모듈러 멱승 연산의 고속 수행을 위한 방법은 여러 가지가 연구되어 있다. DSS와 같이 이산대수에 기반한 알고리즘은 여러 계산에 있어 공통적으로 사용 가능한 사전계산값을 이용할 수 있다. 이러한 방법을 사용할 경우, 일반적으로 계산하는 방식에 비해 3배 이상의 효율성 향상을 얻을 수 있다[35, 23].

방법 2 : 사전계산값이 들어가 있는 큐(queue)를 사용한다. 큐는 고정된 길이를 가지며 생성된 값이 고정 길이를 모두 채우게되면 큐에 새로운 데이타를 집어넣는 것을 중지한다. 이는 서버가 한가할 때에 사전계산을 수행하기 위한 것이다. 최근 시스템의 경우 메모리 값은 매우 싸므로 큐를 크게 잡을 경우, 사전 계산 효과를 충분히 얻을 수 있다[2]. 큐에 넣을 사전계산값의 계산은 서버 구현시 다중 스레드를 이용하여 처리할 수 있다.

6. 관련 연구

이 장에서는 관련 연구들을 소개하고, 제안된 시스템과 비교한다. 관련 연구들은 사용하는 핵심 메커니즘을 기준으로 소개한다.

초소액 지불 시스템의 설계를 위해 사용하는 핵심 메커니즘은 크게 다음의 전자서명, 비밀키 암호함수, 그리고 해쉬 체인에 기반한 방식으로 구분할 수 있다.

- 전자서명에 기반한 방식: 지불 시스템 설계에서 분쟁해결성을 갖기 위해 전자서명의 부인방지 기능을 사용하기 위해 많이 사용되는 방식이다. 초소액 지불 시스템의 경우, NetBill과 Mini-Pay등에서 사용되었으나 나머지 두가지 방식에 비해 효율성이 떨어진다는 단점을 가지므로 일반적으로는 많이 사용되지 않는다.

- 비밀키 암호 함수에 기반한 방식: Kerberos[36]와 같은 전통적인 암호 프로토콜 설계에 많이 사용되는 방식을 적용한 것이다. 이 방식의 경우, 초소액 지불 시스템 설계에서 Millicent 등에 사용되었다. 이 방식은 전자서명에 기반한 방식에 비해 효율성 측면에서 뛰어나다는 장점을 갖으나 분쟁 해결성 측면에서 취약하다.

2) 1024 bit의 키를 사용할 때, 100M byte의 램을 추가적으로 사용할 경우 10,000 개 이상의 사전 계산 값을 기억할 수 있다.

- 해쉬 체인을 사용하는 방식: 해쉬 체인은 자체적으로 부인 방지 기능을 갖으며, 또한 매우 효율적인 연산이 가능하므로 초소책 지불 시스템의 경우, μ-iKP, NetCard, PayWord, CAFE(phone call), 그리고 MPTP 등에 사용되었다. 그리고 제안 시스템의 경우, 해쉬체인을 부분 변형한 이중 해쉬 체인을 사용하였다.

위에서 언급한 내용 이외에 3장에서 제시한 요구조건에 대한 관련 연구와 제안 시스템의 세부적인 특성은 표 1과 같다. 표 1에서 μ-iKP는 원 논문에서 제시된 두가지 방법을 구분하여 고려하였다[3].

표 1 초소액 지불 시스템의 특성 비교.(* mPay-1은 상점이 정직하다는 가정하에서 해결 가능함. ** mPay-2는 공개키 암호 시스템을 사용하는 시스템의 효율적 구현방안 제시)

	Milli-cent	μ-iKP (on-line)	μ-iKP (off-line)	NetBill	mPay-1	mPay-2
Req.1	O	O	X	O	O	O
Req.2	X	O	O	X	O	O
Req.3	O	O	O	X	O	X(**)
Req.4	O	O	O	O	O	O
Risk 1	O	O	O	O	O	O
Risk 2	O	O	O	O	O	O
Risk 3	O	X	X	O	O	O
Risk 4	O	O	O	O	O	O
Risk 5	X	O	O	O	O	O
Risk 6	X	O	O	O	O	O
Risk 7	X	X	X	O	X	O
Risk 8	X	X	X	O	X(*)	O
Risk 9	X	X	X	O	X	O

7. 결 론

본 논문에서는 mPay-1, mPay-2 두 개의 초소액 지불 시스템을 제안하였다. 초소액 지불 시스템이 만족해야 할 조건으로는 여러 가지가 있으나 대표적인 것은 효율성, 안전성 및 이중사용방지, 그리고 분쟁해결성이다. 본 논문에서는 초소액 지불 시스템의 안전성을 향상시키기 위하여 이중 해쉬 체인 구조를 제안하였으며, 이 구조를 mPay-1, mPay-2에 적용하였다. mPay-1 및 mPay-2는 기존의 S/KEY 방식을 사용한 초소액 지불 시스템과 동일한 효율성을 가지나 보다 안전하다. 그리고, mPay-2는 추가적으로 3장에서 제안한 모든 위협요소들에 대해 효과적으로 대처할 수 있다.

향후 연구 과제로는 보다 완성도 높은 초소액 지불 시스템 설계를 위하여 보다 엄밀한 요구 조건의 분석 및 이 조건에 맞는 초소액 지불 시스템의 구현이 요구된다. 또한 다른 종류의전자 지불 시스템과의 연동성에 대한 연구가 필요하다.

참 고 문 헌

[1] N. Asokan, P. Janson, M. Steiner and M. Waidner, State of the Art in Electronic Payment Systems, *IEEE Computer Magazine*, September, v. 30, n. 9, pp. 28-35, 1997

[2] A. Herzberg, Safeguarding Digital Library Contents, *D-Lib Magazine*, January, 1998, ISDN 1082-9873

[3] Micro-Payments based on iKP, Ralf Hauser, Michael Steiner and Michael Waidner, *IBM Research, 12 February 1996, Research Report 2791 (# 89269)*, presented at SECURICOM96.

[4] R. Anderson, C. Manifavas and C. Sutherland, NetCard - A Practical Electronic Cash System, *Proc. Security Protocol Workshop*, LNCS. 1189, pp. 49-57, 1997

[5] R.L. Rivest and A. Shamir, PayWord and MicroMint--Two Simple Micropayment Schemes, *presented at RSA Security conference*, 1996.

[6] T. Pederson, Electronic Payments for Small Amounts, *Proc. Security Protocol Workshop*, LNCS. 1189, pp. 59-68, 1997

[7] Phillip M. Hallam-Baker, Micro Payment Transfer Protocol(MPTP), *W3C Working Draft WD-mptp-951122 (22-Nov-95)*. at http://www.w3.org/TR/WD-mptp

[8] Cliff Newman and G. Medvinsky, Requirements for Network Payment: the Netcheque Perspective, *Proc. of the IEEE Compcon'95*, March 1995.

[9] S. Glassman, M. Manasse, M. Abadi, P. Gauthier, and P. Sobalvarro, The Millicent Protocol for Inexpensive Electronic Commerce. *In World Wide Web Journal, Fourth International World Wide Web Conference Proceedings*, pages 603-618. O'Reilly, December 1995.

[10] Benjamin Cox, J. D. Tygar and Marvin Sirbu, NetBill Security and Transaction Protocol, *in Proceedings of the First USENIX Workshop on Electronic Commerce*, 1995.

[11] Amir Herzberg and Hilik Yochai, Mini-Pay : Charging per Click on the Web, IBM Research. at http://www.hrl.il.ibm.com/mpay/docs/papers/mpay

[12] N. M. Haller. The S/KEY one-time password system, In *ISOC*, 1994

[13] You are interested in: Electronic commerce, payment systems, and security, at *http://www.*

semper.org/sirene/outsideworld/ecommerce.html
[14] Digicash homepage at *http://www.digicash.com*
[15] G. Medvinsky and C. Newman, NetCash: A design for practical electronic currency on the internet, *Proceedings of the First Annual Conference on Computer and Communications Security*, 1993
[16] Secure Electronic Transactions, MasterCard and VISA, at http://www.mastercard.com/set/.
[17] M. Bellare, J.A. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, M. Waidner, iKP - A Family of Secure Electronic Payment Protocols, *1st USENIX Workshop on Electronic Commerce*, 1995
[18] R. L. Rivest, A. Shamir and L. M. Adleman, A Method for Obtaining Digital Signature and Public Key Cryptosystems, *Communications of the ACM*, v. 21, n. 2, pp. 120-126, Feb. 1978
[19] National Institute for Standard and Technology(NIST), Digital Signature Standard (DSS), FIPS 186, Nov 1994,
[20] Public Key Infrastructure(X.509), at http://www.ietf.org/html.charters/pkix-charter.html
[21] PGP(Pretty Good Privacy) http://thegate.gamers.org/~tony/pgp.html
[22] B. Schneier, Applied Cryptography(Second Edition), John Wiley & Sons, 1996
[23] A. J. Menezes, P. C. Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997
[24] National Institute for Standard and Technology (NIST), Secure Hash Standard (SHS), FIPS 180-1, Apr. 1995
[25] R. L. Rivest, The MD5 message-digest algorithm. Internet Request for Comments, RFC 1321, Apr. 1992
[26] M. Bellare, R. Canetti, and H. Krawczyk, Keying Hash Functions for Message Authentication, *Advances in Cryptology - CRYPTO' 96*, LNCS 1109, 1996
[27] L. Lamport, Password Authentication with Insecure Communication, *CACM*, v. 24, n. 11. pp. 770-772, 1981
[28] M. Burrows, M. Abadi, and R. Needham, A logic of Authentication, *DEC SRC report #39*, Digital Equipment Cooperation, Palo Alto, CA. Feb. 1989. Revised Feb. 1990
[29] D. Tygar, Atomicity in Electronic Commerce. In Internet Besieged. Addison-Wesley and ACM Press. October 1997, pages 389-406.
[30] R. Kailar, Accountability in Electronic Commerce Protocols, *IEEE. Transactions on Software Engineering*, v. 22, n. 5, pp. 313-328, 1996
[31] N. Asokan, E. Van Herreweghen, and M. Steiner. Towards a framework for handling disputes in payment systems. *Research Report RZ 2996, IBM Research*, March 1998.
[32] A. O. Freier, P. Karlton, and P. C. Kocher, The SSL Protocol Ver. 3.0, Netscape Communication Corporation. March 1996
[33] T. Dierks and C. Allen, The tls protocol version 1.0, RFC 2246, January 1999.
[34] Seong-Min Hong, Jun-Bum Shin, H.Lee-Kwang and Hyunsoo Yoon, "A new approach to server-aided secret computation", *International Conference on Information Security and cryptology*, Seoul, 1998
[35] E. F. Brickell, D. M. Gordon, K. S. Mccurley, and D. B. Wilson, "Fast exponentiation with precomputation", *Advances in Cryptography - EUROCRYPT'92*(LNCS 658), pp. 200-207, 1993
[36] J. G. Steiner, B. C. Newman, and J. I. Schiller.Kerberos: An authentication service for open network systems, Proceedings of the Winter 1988 Usenix Conference, 1988, pages 191 - 201
[37] T. Okamoto and K. Ohta, Universal Electronic Cash. In Advances in Cryptology-Crypto'91, pages 324-337. Springer-Verlag, 1992.
[38] T. Eng and T. Okamoto, Single-Term Divisible Electronic Coins, in Pre-Proceedings of Eurocrypt'94, pages 313-323. Springer-Verlag, 1994.
[39] T. Okamoto, An Efficient Divisible Electronic Cash Scheme, in Advances in Cryptology-Crypto'95, pages 438- 451. Springer-Verlag, 1995.
[40] D. Chaum, A. Fiat, and M. Noar. Untraceable Electronic Cash, In Advances in Cryptology-Crypto'88, pages 319-327. Springer-Verlag,1990.

부 록

A. 사생활보호

현재까지 제안된 많은 종류의 초소액 지불 시스템은 사생활 보호 기능을 갖지 않는다. 그리고 제안한 시스템 역시 동일하다. 이 장에서는 NetCash에서 적용한 방법 [15]을 이용하여 제안 시스템을 사생활 보호 기능을 갖도록 확장하는 방법을 다룬다.

A.1 관련 연구

일반적인 지불 시스템에서 사생활 보호와 관련된 요소는 다음의 두가지로 구분하여 생각할 수 있다.

- 거래에 대한 사용자 정보 노출 방지(익명성): 지불 프로토콜에서 사용된 데이타로부터 상품 구매자를 확인

할 수 없어야 한다.

- 거래 데이타에 대한 연관성 방지(연관성 방지): 거래 데이타들로부터 동일한 사용자가 사용한 것임을 확인할 수 없어야 한다.

일반적으로 익명성 조건을 만족하지 못할 경우, 연관성 방지 기능은 만족할 수 없다. 그러나 익명성 조건을 만족하는 것이 반드시 연관성 방지를 의미하지는 않는다. 이러한 특성은 분할 가능한 전자현금의 경우에서 찾을 수 있다[37, 38, 39].

일반적인 지불 시스템에서 익명성을 얻기 위해 사용하는 방법은 크게 다음의 두가지로 구분할 수 있다.

- 완전한 익명성: D. Chaum이 제안한 익명서명(blind signature)를 사용하는 방법이다[40]. 자세한 언급은 본 논문의 범위를 벗어나므로 생략한다.

- 기관 신뢰에 기반한 익명성: 전자현금의 일종인 NetCash 설계에 사용되었던 방법이다. 구성 방식은 구입한 전자현금을 다른 기관에서 새로운 전자현금으로 교환하는 방법이다. 이 방식을 적용하였을 경우, 사용되는 전자현금과 관련된 모든 기관이 연합하여야만 사용자의 신원이 공개된다. 따라서 관련 기관 중, 적어도 하나의 기관이 정직하게 동작할 경우, 사용자의 신원이 공개되지 않는다. 자세한 내용은 [15]를 참조 바란다.

본 논문에서는 NetCash에 적용된 방식을 이용하여 제안 시스템이 익명성을 갖을 수 있도록 확장한다.

A.2 초소액 지불 시스템의 익명성

초소액 지불 시스템의 익명성은 연동되는 일반적인 지불 시스템의 익명성과 밀접한 관계를 갖는다. 만일 연동되는 시스템이 익명성을 지원할 경우, 일반적으로 초소액 지불 시스템은 동일한 익명성을 갖는다. 그러나 그렇지 않을 경우, 별도의 방법이 요구된다.

제안 시스템의 경우, 거래 데이타에서 사용자의 신원과 직접적인 관련이 있는 값은 존재하지 않는다. 그러나 사용자가 쿠폰묶음 사용 인증서를 구입한 브로커는 쿠폰묶음 인증서를 발급할 당시 규정된 데이타들인 C_U, $PsID_U$, PK_U, Ex_{C_U}, Ex_{DH}, MAX 이외에 사용자의 실제 식별번호를 저장할 수 있다. 이 경우, 브로커는 지불 정산과정에서 C_U를 이용하여 사용자의 실제 신원을 확인할 수 있다. 이 문제를 해결하기 위해 NetCash에서 적용하였던 방법과 같이 다음의 프로토콜을 사용할 수 있다.

step 1. 사용자는 별도의 브로커(*Broker'*)와 암호화 통신 프로토콜을 이용하여 연결한다.

step 2. 사용자는 이중 해쉬 체인을 생성한 후, 사용 가능한 최대 금액을 사용자의 새로운 유사 개인 식별번호($PsID_U'$)과 함께 *Broker'*에게 지불한다.

step 3. *Broker'*는 유사 개인 식별번호가 $PsID_U'$인 새로운 쿠폰 묶음 사용 인증서를 사용자에게 제공한다.

사용자는 최종적으로 쿠폰을 사용하기 이전에 이 과정을 여러 브로커에게 수행할 수 있다. 이 경우, 브로커가 정직할 경우, 브로커는 사용자의 이전 유사개인 식별번호와 자신이 발급한 쿠폰묶음 사용 인증서에 포함된 유사개인 식별번호의 관계를 공개하지 않고, 또한 사용자와의 모든 통신은 암호화 되어 있으므로 사용자는 자신이 접속한 모든 종류의 브로커가 정직하지 않을 경우를 제외하고는 거래로부터 자신의 신원이 공개되는 것을 막을 수 있다.

A.3 토 의

이 장에서는 NetCash에서 사용하였던 방법을 이용하여 제안 시스템을 익명성을 갖도록 확장하는 방법을 다루었다. 그러나 제안한 방식은 연관성 방지 기능은 지원하지 않는다. 앞으로 이 문제를 해결할 수 있는 방법에 대한 연구가 요구된다.

신 준 범

1995년 한국과학기술원 수학과 졸업(학사). 1998년 한국과학기술원 수학과 졸업(석사). 1998년 ~ 현재 한국과학기술원 전자전산학과 전산학전공 박사과정. 관심분야는 암호 프로토콜 및 알고리즘, 전자상거래, 인터넷 보안, 퍼지이론

이 광 형

1978년 서울공대 산업공학 학사. 1980년 한국과학원 산업공학 석사. 1982년 프랑스 INSA 전산학과 석사(DEA). 1985년 프랑스 INSA 전산학과 공학박사. 1988년 1월 프랑스 국가박사(전산학INSA-LYON1대). 1985년 1995년 한국과학기술원 전산학과 조교수 및 부교수. 1995년 ~ 현재 한국과학기술원 전자전산학과 교수. 1985년 프랑스 INSA. 1995년 미국 Stranford Research Institute. 관심분야는 퍼지이론 및 응용, 인공지능, 전문가 시스템 등