情報保護學會論文誌
第18卷 第6(A)號, 2008. 12

인증서가 없는 강한 지정된 검증자 서명기법*

구 영 주†, 천 지 영, 최 규 영, 이 동 훈‡
고려대학교 정보경영 공학전문 대학원

Certificateless Strong Designated Verifier Signature Scheme*

Young Ju Koo†, Ji Young Chen, Kyu Young Choi, Dong Hoon Lee‡
Graduate School of Information Management and Security CIST, Korea University

요 약

일반서명은 누구나 서명의 정당성을 검증 가능하기 때문에 메시지에 대한 인증을 제 3자에게 전가함으로서 프라이버시 침해를 일으키기도 한다. 이에 Jakobsson 등은 지정된 사람만이 서명을 검증할 수 있게 하는 지정된 검증자 서명기법을 처음 소개 하였다. 지정된 검증자 서명기법은 오직 선택된 검증자만이 서명을 검증할 수 있게 하는 기법으로 메시지 인증 성질의 문제점과 프라이버시 침해를 해결한다. 본 논문에서는 '인증서가 없는(Certificateless)' 개념을 기반으로, 서명자의 프라이버시를 강화한 인증서 없는 강한 지정된 검증자 서명기법을 제안하고자 한다. 제안하는 기법은 기존의 인증서 없는 지정된 검증자 서명기법에 비해 다소 효율성이 떨어지나 강한의 성질과 양도성의 문제를 가지지 않는 인증서가 없는 강한 지정된 검증자 서명 기법은 본 논문이 최초이다.

ABSTRACT

In the traditional signature techniques, anyone can verify the signed message. It may cause a problem since a receiver of the signature can transfer the conviction of signature to a third party. In 1996, Jakobsson introduced a designate verifier signature(DVS) which is allowed to verify only specific verifier. DVS is the solution of conflict between authenticity and privacy because it provides message authentication without non-repudiation property. In this paper based on the notion of certificateless, we suggest a certificateless strong designated verifier signature scheme including the notion of strong which provides privacy of the signer. We suggest a scheme which is first trial to propose a certificateless strong designated verifier signature scheme including the notion of strong and non-delegatability, although it is not more efficient than previous one.

Keywords : certificateless, non-delegatability, strong, designated verifier signature

Ⅰ. 서 론

검증자 서명기법은 특정한 검증자만이 그 서명의 정당성을 검증 할 수 있도록 한 기법으로 1996년 Jakobsson 등이 처음 제안하였다.[1] 지정된 검증자 서명기법은 기존의 서명이 가지고 있던 부인방지 성질을 가지지 않고도 메시지 인증을 제공한다. 수신자에게는 송신자가 서

접수일: 2008년 4월 2일; 수정일: 2008년 8월 18일;
채택일: 2008년 10월 9일
* 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(IITA-2008-(C1090-0801-0025))
† 주저자, danmiluv@korea.ac.kr
‡ 교신저자, donghlee@korea.ac.kr

명된 메시지를 생성하였다는 메시지 인증이 가능하면서도 제 3자에게는 자신이 서명을 생성하지 않았다고 부인 할 수 있는 것이다. 예를 들어 지정된 검증자가 서명자로부터 서명된 메시지를 받았다고 할 때, 검증자는 서명자가 보내준 메시지에 대한 서명의 정당성을 확인할 수 있으나 제3자는 서명이 서명자로부터 생성된 것임을 확신할 수 없다. 이는 지정된 검증자가 제3자가 보기에 서명자가 생성한 서명과 구별 불가능 한 트랜스크립트를 만들어낼 능력을 가지고 있기 때문이다.

이처럼 지정된 검증자 서명기법에서는 선택된 검증자만이 서명을 확신할 수 있기 때문에 일반 서명이 가지고 있는 메시지 인증 전가의 문제점을 해결하고, 서명자가 부인 가능하여 서명자에 대한 익명성을 보장해준다. 그러나 지정된 검증자 서명기법은 서명자와 지정된 검증자사이에서의 익명성만을 제공하기 때문에 완벽한 익명성을 제공한다고 하기는 어렵다.

인증서가 없는(CL, Certificateless) 암호기법은 기존의 공개키 시스템에서 가지고 있던 KGC(Key Generation Center)의 키 위탁 문제를 해결하기 위해 2003년 AL-Riyami 등에 의해 제안되었다.[2] ID기반 공개시스템의 경우 KGC가 개인의 개인키를 생성해주어 서명을 생성하는데, 이 KGC가 모든 이의 개인키를 알기 때문에 악의적일 경우 모든 이의 서명을 위조해 낼 수 있다는 키 위탁문제를 가지고 있었다. 인증서가 없는 암호기법은 이러한 KGC가 생성해 주는 개인키 뿐 아니라 서명자도 비밀값을 생성하여 서명을 생성하기 때문에 이러한 키 위탁 문제를 해결한다. 2006년 Huang 등은 '인증서 없는 지정된 검증자 서명기법(CL-DVS, Designated Verifier Signature)'을 최초로 제안하였다.[3] 본 논문에서는 Huang 등의 인증서 없는 지정된 검증자 서명기법에 '강한' 개념을 추가한 인증서 없는 강한 지정된 검증자 서명기법을 제안하고자 한다. 강한 지정된 검증자의 개념은 1996년 Jakobsson등에 의해 정의[1]되었으며, 2003년 Saeednia등이 새로운 강한 지정된 검증자의 개념을 정의하였다[4]. 지정된 검증자 서명기법에서는 제3자가 입장에서 오직 서명자(A)와 검증자(B), 두 측만이 잠재적인 서명자가 될 수 있는데, 만약 A로부터 날아가는 서명을 B가 받기 전에 누군가 가로채 확인하고, B가 서명을 생성하지 않았다는 것을 확인할 수 있다면 A가 실제 서명자라는 걸 알 수 있게 된다. 또한 제3자가 B를 신뢰할 수 있는 경우 제3자는 A가 서명자임을 확신할 것이다. 이와 같이 지정된 검증자 서명은 오직 서명자와 검증자 둘 사이에 대한 익명성만을 제공한다. 그러나 강한 지정된 검증자 서명기법에서는 제 3자가 서명을 보고 어떤 서명자와 검증자 사이에 생성된 서명인지 구별할 수 없다. 따라서 오직 두 사람만이 잠재적인 서명자가 아닌, 모든 사람이 잠재적인 서명자가 될 수 있어 완벽한 익명성을 제공한다고 할 수 있다.[5]

2005년 Lipmaa등[6]은 지정된 검증자 서명에서 고려되어야 할 새로운 안정성 성질인, 비양도성(Non-Delegatability)을 정의하였다. 양도성이란 서명자 또는 검증자가 자신의 비밀키를 노출하지 않고도 제 3자에게 부가적인 정보를 제공으로써 제 3자가 정당한 서명을 생성할 수 있도록 서명권리를 양도하는 것을 말한다. DVS기법이 양도성을 만족하지 않을 경우 심각한 문제를 야기할 수 있다. DVS는 전자투표나 e-옥션,공개입찰 등에 적용될 수 있는데 전자투표에서 DVS가 양도성을 가지는 경우, 서명자는 부가적인 정보만을 제 3자에게 제공으로써 제 3자에게 자신의 투표 권리를 이행하도록 할 수 있다. 이전의 제안된 DVS기법들은 대부분이 양도성을 가지고 있어 최근에는 비양도성을 가지는 DVS 기법들이 제안되고 있다. 기존의 Huang의 CL-DVS기법은 양도성의 문제를 가지고 있으며 이는 결론에서 간략히 보이겠다. 본 논문에서 제안하는 CL-SDVS기법은 양도성을 가지지 않는다.

Ⅱ. 모델 및 보안 요구사항

본 절에서는 CL-SDVS모델과 공격 모델에 대해 정의하고 보안 요구사항에 대하여 서술하겠다. CL-SDVS 기법은 Huang 등의 기법[3]에 "강한" 성질을 추가한 것으로 기존의 CL-DVS기법의 모델과 동일하다.

2.1 CL-SDVS 모델

서명자를 A, 지정된 검증자를 B라고 가정한다. 모델은 8개의 알고리즘으로 구성되어 있으며 각각은 Setup, Partial-Private-Key-Extract, Set-Secret-Value, Set-Public-Key, Set-Private-Key, Sign, Verify, Transcript- Simulation 과 같다. Setup, Partial-Private-Key-Extract 알고리즘은 KGC에 의해 실행되어 부분 개인키를 생성한 후, 사용자는 그 부분 개인키를 받고 자신의 Set- Secret-Value 알고

리즘을 통해 얻은 비밀 값을 이용하여 개인키를 생성한다. 각 알고리즘에 대한 자세한 내용은 다음과 같다.

Setup : 시스템 파라미터 $params$를 받아 시스템 파라미터 $params$와 마스터 키(master key) s를 출력한다.

Partial-Private-Key-Extract : 시스템 파라미터 $params$ $params$, 마스터 키 s와 사용자의 아이디 ID 를 입력받아 부분 개인키 D_{ID}를 출력한다.

Set-Secret-Value : 시스템 파라미터 $params$ 와 사용자의 아이디 ID 를 입력받아 사용자의 비밀 값 x_{ID}를 랜덤하게 출력한다.

Set-Public-Key : 사용자의 비밀값 x_{ID} 를 입력받아 사용자의 공개키 PK_{ID} 를 출력한다.

Set-Private-Key : 사용자의 부분 개인키 D_{ID}, 비밀 값 x_{ID}를 입력받아 사용자의 개인키 S_{ID}를 출력한다.

Sign : 시스템 파라미터 $params$, 메시지 m, 서명자 A의 $(ID_A, S_{ID_A}, x_{ID_A}, D_{ID_A})$, 지정된 검증자 B의 (PK_{ID_B}, ID_B)를 받아 서명σ을 출력한다.

Verify : 메시지m, 서명σ, $params$, 지정된 검증자 B의 $(ID_B, S_{ID_B}, x_{ID_B}, D_{ID_B})$, 서명자 A의 (PK_{ID_A}, ID_A)를 받아 서명의 정당성을 확인하여 정당하면 accept를 출력하고, 그렇지 않으면 reject를 출력한다.

Transcript-Simulation : 서명자 A가 생성한 것과 구별 불가능한 트랜스크립트를 생성하기 위해 지정된 검증자 B에 의해서 실행되는 알고리즘이다.

2.2 보안 요구 사항

CL-SDVS 기법은 지정된 검증자 서명기법의 보안 요구사항을 모두 포함하며 강한 성질을 가지고 있기 때문에서명자의 프라이버시 성질을 추가한 다음과 같은 성질들을 만족해야 한다.

- 정확성(Correctness) : 서명자가 서명 알고리즘을 통하여 정당한 방법으로 서명을 생성하였다면, 그 서명은 반드시 검증 알고리즘을 통과해야 한다.
- 서명의 위조 불가(Unforgeability) : 공격자가 서명자나 지정된 검증자의 개인키에 대한 지식 없이 정당한 서명을 위조하는 것이 계산적으로 불가능 해야 한다. CL-SDVS기법에서는 두 가지 유형의 공격자가 존재하므로 이 두 공격자 모두에 대하여 서명이 위조 불가능 하여야 한다.
- 서명자의 프라이버시(Signer's privacy) : 지정된 검증자와 임의의 메시지에 대한 서명이 존재 할 때, 제 3자는 지정된 검증자 또는 어느 서명자의 비밀키 없이 서명이 어느 서명자와 검증자의 키로 생성된 것 인지 알 수 없어야 한다.
- 서명능력의 비양도성(Non-delegatability) : 서명자 또는 지정된 검증자는 자신의 비밀키를 노출하지 않고도 제 3자에게 서명 권리를 양도하는 것이 불가능해야 한다.

2.3 보안 요구 사항

인증서가 없는 서명 기법에서는 기존의 서명기법과 공격 모델이 다르다. 서명의 위조 불가능을 보이기 위하여 [2][7]에서 정의된 것과 같이 다음과 같은 두 가지 서명 위조 공격자 타입을 고려한다.

유형 I 공격자(Ƒ1) : 이 유형의 위조자 Ƒ1은 master-key에는 접근할 수 없으나, 사용자의 공개키를 위조하는 공격자로 자신이 선택한 값으로 사용자의 공개키를 바꿀 수 있다.

유형 II 공격자(Ƒ2) : 이 유형의 위조자 Ƒ2는 사용자의 공개키는 위조할 수 없지만 master-key를 아는 공격자이다.

유형 I은 시스템 외부에서의 공격을 나타내며, 유형 II는 악의적인 KGC를 가정하였을 때의 공격을 나타낸 것이다. Ƒ1 과 Ƒ2 공격자에 대한 게임은 다음과 같이 이루어진다.

GAME 1. 첫 번째 게임은 인증서가 없는 강한 지정된 검증자 서명 기법에 대한 챌린저 C_I과 공격자 Ƒ1 사이에서 다음과 같이 수행된다.

초기화 : C_I은 $setup$알고리즘을 시행하고 master key와 $params$를 생성한다. $params$는 Ƒ1에게준다. Ƒ1은 master-key는 알지 못한다.

쿼리들 : Ƒ1은 C_I에게 다음과 같은 질문들을 던진다.

- $ExtrPartSK(ID)$: Ƒ1이 사용자 ID에 대한 부분 개인키를 질의하면 C_I은 Partial-Private-Key-Extract 알고리즘을 수행하여 사용자의 부분 개인키 D_{ID}를 생성해 준다.

- $ExtraFullSK(ID)$: $\mathcal{F}1$이 사용자 ID에 대한 개인키를 요청하면 C_I은 Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key 알고리즘을 수행하여 사용자의 개인키 S_{ID}를 생성해 준다.
- $RequestPK(ID)$: $\mathcal{F}1$이 사용자 ID에 대한 공개키를 요청하면 C_I은 Set-Secret-Value, Set-Public-Key 알고리즘을 수행하여 사용자의 공개키 PK_{ID}를 생성해 준다.
- $ReplacePK(ID)$: $\mathcal{F}1$은 PK_{ID}를 자신이 선택한 새로운 공개키 PK'_{ID}로 대치할 수 있다.
- $Sign(m,ID_i,ID_j)$: $\mathcal{F}1$은 임의의 지정된 검증자의 아이디 ID_j에 대해, 메시지 m에 대한 임의의 서명자 ID_i의 서명을 요청한다. C_I은 $Sign$알고리즘을 실행하여 정당한 서명 σ 를 생성 해준다.
- $Verify(\sigma,ID_i,ID_j)$: $\mathcal{F}1$이 서명자 ID_i와 지정된 검증자 ID_j에 대한 서명 σ을 검증 요청하면 C_I은 $Verify$알고리즘을 실행하여 정당하면 accept를, 그렇지 않으면 reject를 출력한다.

출력 : $\mathcal{F}1$이 위조된 서명 (m',σ',ID_A',ID_B')를 내놓는다. 다음의 조건을 만족한다면 $\mathcal{F}1$이 게임에서 이긴다.

1) $ExtrPartSK(ID_A')$, $ExtrPartSK(ID_B')$, $Extra$, $FullSK(ID_A')$, $ExtraFullSK(ID_B')$, $Sign(m',ID_A',ID_B')$, $Sign(m',ID_B',ID_A')$가 한 번도 질의되지 않았을 때
2) $Verify$ 알고리즘이 위조된 서명에 대해 accept를 출력할 때 (이때, ID_A',ID_B'의 공개키는 공격자에 의해 대체 되었을 수 도 있다)

GAME 2. 두 번째 게임은 인증서가 없는 강한 지정된 검증자 서명기법에 대한 챌린저 C_{II} 와 공격자 $\mathcal{F}2$ 사이에 다음과 같이 실행된다.

초기화 : C_{II}는 $setup$알고리즘을 시행하고 master key와 $params$를 생성하여 $params$와 master key를 $\mathcal{F}2$에게 준다.

쿼리들 : $\mathcal{F}2$는 C_{II}에게 다음과 같은 질문들을 던진다.

- $ExtraFullSK(ID)$: $\mathcal{F}2$가 사용자 ID에 대한 개인키를 요청하면 C_{II}는 Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key 알고리즘을 수행하여 사용자의 개인키 S_{ID}를 생성해 준다.
- $RequestPK(ID)$: $\mathcal{F}2$가 사용자 ID에 대한 공개키를 요청하면 C_{II}는 Set-Secret-Value, Set-Public-Key 알고리즘을 수행하여 사용자의 공개키 PK'_{ID}를 생성해 준다.
- $Sign(m,ID_i,ID_j)$: $\mathcal{F}2$는 지정된 검증자의 ID_j에 대해, 메시지 m에 대한 서명자ID_i의 서명을 요청한다. C_{II}는 Sign 알고리즘을 실행하여 정당한 서명 σ을 생성 해준다.
- $Verify(\sigma,ID_i,ID_j)$: $\mathcal{F}2$가 서명자 ID_i와 지정된 검증자 ID_j에 대한 서명 σ을 검증 요청하면 C_{II}는 $Verify$알고리즘을 실행하여 정당하면 accept를, 그렇지 않으면 reject를 출력한다.

출력 : 마침내 $\mathcal{F}2$ 가 위조된 서명(m',σ',ID_A',ID_B')를 내놓는다. 다음의 조건을 만족한다면 $\mathcal{F}2$가 게임에서 이긴다.

1) $ExtrPartSK(ID_A')$, $ExtrPartSK(ID_B')$, $Extra$, $FullSK(ID_A')$, $ExtraFullSK(ID_B')$, $Sign(m',ID_A',ID_B')$, $Sign(m',ID_B',ID_A')$가 한 번도 질의되지 않았을 때
2) $Verify$ 알고리즘이 위조된 서명에 대해 accept를 출력할 때

Bilinear Map과 암호학적 가정들

안전성을 증명하기 위한 몇 가지 정의와 암호학적 가정에 대하여 살펴본다.

G_1은 P를 생성자로 하는 위수가 q인 덧셈 연산군이라 하고, G_2를 위수가 q인 곱셈 연산 군이라 하자. 이 때 G_1, G_2에서 이산 대수 문제(Discrete Logarithm Problem, DLP)는 어렵다고 가정한다. 임의의 $P,Q\in G_1$와 $a,b\in Z_q^*$에 대하여 다음을 만족하는 함수 $e: G_1\times G_1\rightarrow G_2$을 admis- sible bilinear map 이라 한다.

- Bilinear : $e(aP,bQ)=e(P,Q)^{ab}$
- Non-degenerate : $e(P,Q)\neq 1$을 만족하는 $P,Q\in G_1$가 존재한다.
- Computable : $e(P,Q)$을 계산할 수 있는 효율적인 알고리즘이 존재한다.

이 bilinear map은 Weil pairing이나 Tate pairing을

사용하여 실제화 될 수 있다. 자세한 적용 방법은 [8]에 언급되어 있다.

Computational Diffie-Hellman(CDH) Problem : $a,b \in Z_q^*$에 대하여 P, aP, bP 가 주어졌을 때 abP를 계산하는 문제이다. 본 논문에서는 취약하지 않은 확률로 다항시간 안에 CDH문제를 해결하는 알고리즘은 없다고 가정한다.

Ⅲ. 제안하는 CL-SDVS 기법

이 절에서는 CL-SDVS기법을 제안하고 그에 대한 안전성을 분석한다.

3.1 제안하는 CL-SDVS 기법

A를 서명자, B를 지정된 검증자라고 하자. Setup, Partial-Private-Key-Extract 알고리즘은 KGC에 의해 실행된다.

Setup : KGC는 다음과 같이 시스템 파라미터와 master-key를 생성한다.

1) G_1, G_2는 소수 q를 c위수로 갖는 순환군이고 e는 admissible bilinear map일 때, (G_1, G_2, e)를 생성한다.
2) 랜덤한 $s \in Z_q^*$를 고르고 G_1의 생성원 P를 선택하여 $P_{pub} = sP$를 계산한다.
3) 암호학적 일방향 해쉬함수 $H_1 : \{0,1\}^* \rightarrow G_1$, $H_2 : G_1 \rightarrow Z_q^*, H_3 : \{0,1\}^* \times G_1 \times G_1 \rightarrow Z_q^*$를 생성한다.

master-key는 s가 되고, 시스템 파라미터 $params = \{e, G_1, G_2, q, g, P, P_{pub}, H_1, H_2, H_3\}$를 공개한다.

Partial-Private-Key-Extract : $params$, master key와 사용자의 아이디 ID_A를 입력받아 $Q_{ID_A} = H_1(ID_A)$를 계산 후, 부분 개인키 $D_{ID_A} = sQ_{ID_A}$를 출력한다.

Set-Secret-Value : 보안 파라미터 k와 사용자의 아이디 D_{ID_A}를 입력받아 사용자의 비밀 값 x_{ID_A}를 출력한다.

Set-Public-Key : 사용자의 비밀값 x_{ID_A} 를 입력받아 사용자의 공개키 $PK_{ID_A} = x_{ID_A}P$ 를 출력한다.

Set-Private-Key : 사용자의 부분 개인키 D_{ID_A}, 공개키 PK_{ID_A}, 비밀 값 x_{ID_A}를 입력받아 $y_{ID_A} = H_2(PK_{ID_A})$를 계산 후, 사용자의 개인키 $S_{ID_A} = (\frac{1}{x_{ID_A} + y_{ID_A}}) D_{ID_A}$를 출력한다.

Sign : 메시지 m에 대한 서명을 다음과 같이 생성한다.

1) $U_B \in G_1$와 랜덤한 $r \in Z_q^*$을 선택한다.
2) $k = x_{ID_A} PK_{ID_B} = x_{ID_A} x_{ID_B} P$를 계산하고, $h_B = H_3(m, U_B, k)$ 값을 계산한다.
3) $U_A = rQ_{ID_A} - (U_B + h_B Q_{ID_B})$를 계산한다.
4) $\partial_B \in Z_q^*$를 ``선택하여 $V_B = \partial_B P$ 를 계산한다.
5) $h_A = H_3(m, U_A, k)$값을 계산한다.
6) $V_A = (-\frac{1}{x_{ID_A} + y_{ID_A}}) \partial_B (PK_{ID_B} + y_{ID_B} P) + (r + h_A) S_{ID_A}$ 를 계산한다.
7) 메시지 m에 대한 서명은 $\sigma = (U_A, U_B, V_A, V_B)$이다.

Verify : 메시지 m에 대한 서명σ 을 다음과 같이 검증한다.

$$e(V_A, PK_{ID_A} + y_{ID_A} P) e(V_B, PK_{ID_B} + y_{ID_B} P) = e(U_A + U_B + h_A Q_{ID_A} + h_B Q_{ID_B}, P_{pub})$$

이를 계산하여 등호가 성립하면 accept, 그렇지 않으면 reject를 출력한다.

Transcript-Simulation : 지정된 검증자 B는 제3자가 입장에서 서명자 A가 생성한 것과 구별 불가능한 서명을 생성할 수 있어야 한다. 구별 불가능한 트랜스크립트는 다음과 같이 생성한다.

1) $U_A' \in G_1$와 랜덤한 $r' \in Z_q^*$을 선택한다.
2) $k = x_{ID_B} PK_{ID_A} = x_{ID_B} x_{ID_A} P$를 계산하고, $h_A' = H_3(m, U_A', k)$ 값을 계산한다.
3) $U_B' = r' Q_{ID_B} - (U_A' + h_A' Q_{ID_A})$를 계산한다.
4) $\partial_A \in Z_q^*$를 선택하여 $V_A' = \partial_A P$를 계산한다.
5) $h_B' = H_3(m, U_B', k)$ 값을 계산한다.
6) $V_B' = (-\frac{1}{x_{ID_B} + y_{ID_B}}) \partial_A (PK_{ID_A} + y_{ID_A} P) + (r' + h_B') S_{ID_B}$를 계산한다.
7) 메시지 m에 대한 트랜스크립트는 $\sigma = (U_A', U_B', V_A', V_B')$이다.

물론 B의 트랜스크립트 또한 A가 생성한 서명과 같은 방법으로 검증 가능해야 한다.

3.2 제안하는 CL-SDVS 기법

제안하는 CL-SDVS에 대한 안전성은 정확성과 위조불가, 서명자의 프라이버시, 그리고 비양도성으로 보일 수 있다. 서명자가 적절한 방법으로 생성한 서명은 반드시 검증알고리즘을 통과하여야 하는데 이러한 정확성 성질은 다음과 같이 만족한다.

$$
\begin{aligned}
&e(V_A, PK_{ID_A}+y_{ID_A}P)e(V_B, PK_{ID_B}+y_{ID_B}P)\\
&=e(U_A+U_B+h_AQ_{ID_A}+h_BQ_{ID_B}, P_{pub})\\
&\text{좌변})e(V_A, PK_{ID_A}+y_{ID_A}P)e(V_B, PK_{ID_B}+y_{ID_B}P)\\
&=e(-\frac{1}{x_{ID_A}+y_{ID_A}}\partial_B(PK_{ID_B}+y_{ID_B}P)+(r+h_A)S_{ID_A}\\
&,(x_{ID_A}+y_{ID_A})P)\ e(\partial_BP, PK_{ID_B}+y_BP)\\
&=e(-\frac{1}{x_{ID_A}+y_{ID_A}}\partial_B(PK_{ID_B}+y_{ID_B}P)+(r+h_A)\cdot\\
&\frac{1}{x_{ID_A}+y_{ID_A}}D_{ID_A}, (x_{ID_A}+y_{ID_A})P)\ e(\partial_BP,\\
&PK_{ID_B}+y_{ID_B}P)\\
&=e(-\partial_BPK_{ID_A}-\partial_By_{ID_B}P+(r+h_A)D_{ID_A}, P)\ e(P,\\
&\partial_B(PK_{ID_B}+y_{ID_B}P))\\
&=e((r+h_A)sQ_{ID_A}, P)\\
&\text{우변})e(U_A+U_B+h_AQ_A+h_BQ_{B,}P_{pub})\\
&=e(rQ_A-(U_B+h_BQ_B)+U_B+h_AQ_A+h_BQ_{B,}P_{pub})\\
&=e((r+h_A)Q_A, sP)=e((r+h_A)sQ_A, P)
\end{aligned}
$$

인증서 없는 서명 기법에서는 서명 위조성질에 관한 두 가지 공격자 유형에 대하여 고려한다. 이는 이미 2.1에서 정의하였다. 이 장에서는 암호학적으로 어려운 Computational Diffie-Hellman문제에 기반하여 사용자의 공개키를 변경할 수 있는 공격자 유형 I과 KGC의 마스터 키 정보를 알고 있는 공격자 유형 II에 대한 안전성을 보인다. 제안하는 CL-SDVS기법의 안전성은 랜덤 오라클 모델을 가정하여 안전성을 증명한다.

정리 1. *램덤오라클 모델에서 제안하는 CL-SDVS기법은 CDH문제가 어렵다고 가정할 때 공격자 유형 I($\mathcal{F}1$)에 대하여 적합하게 선택된 메시지 공격에 안전하다.*

증명. 사용자의 공개키를 바꿀 수 있는 능력을 가진 공격자 $\mathcal{F}1$이 존재한다고 가정하자. 이점(advantage)을 갖는 공격자 $\mathcal{F}1$ 이 존재 할 때, Computation Diffie-Hellman 문제를 푸는 알고리즘 C_I이 존재함을 보일 것이다. C_I이 CDH문제를 풀 수 있도록 임의의 $a,b\in Z_q^*$와 $P\in G$를 선택하여 CDH문제의 입력값 (P,aP,bP)을 전달한다. C_I은 CDH문제를 풀기 위하여 공격자 $\mathcal{F}1$이 알아채지 못하도록 공격 환경을 시뮬레이션 해주고, 공격자 $\mathcal{F}1$이 주는 정보를 이용한다. C_I은 $P_{pub}=aP$로 설정하여 $\mathcal{F}1$에게 시스템 파라미터를 전달한다. 따라서 여기서 a는 마스터 키로 C_I은 마스터 키의 정보를 모르며 aP로부터 알아낼 수도 없다. 우리는 RequstPK 질의는 H_2질의 전에 이루어진다고 가정하며 일반성을 잃지 않고 ExtrPartSK, ExtrFullSK, Sign, Verify 질의를 요청하기 전에 사전에 H_1과 H_2, H_3 해쉬 질의를 한다고 하자. 또한 게임 1 에서만 존재하는 ReplacePK 질의는 Sign, Verify 질의 전에 행해진다고 가정한다. 질의에 일관성 있게 답하기 위하여 C_I은 $L_{H_1}, L_{H_2}, L_{H_3}, L_K$ 리스트를 유지한다. 이들 리스트는 초기에는 비어있다.(null) C_I은 $\mathcal{F}1$의 오라클 질의에 다음과 같이 시뮬레이션 한다.

- ***H_1*해쉬질의** : $\mathcal{F}1$이 최대 qH개의 H_1질의를 요청한다고 하자. C_I은 $A,B\in[1....q_{H_1}]$를 선택한다. $\mathcal{F}1$이 $ID_i(1\le i\le q_{H_1})$에 대하여 H_1질의를 할 때, 만약 $i=A$라면 $Q_{ID_i}=bP$로 대답하고 그렇지 않으면 임의의 $t\in Z_q^*$를 선택하여 $Q_{ID_i}=t_iP\,(i\neq A)$로 대답한다. 두 경우 모두 $<ID_i, t_i>$를 L_{H_1}리스트에 추가한다.
- ***H_2*해쉬질의** : $\mathcal{F}1$이 PK_{ID_i}에 대한 H_2질의를 요청하면, 임의의 $y_{ID_i}\in Z_q^*$를 선택하여 답하고, $L_{H_2}=<PK_{ID_i}, y_{ID_i}>$리스트에 추가한다.
- ***H_3*해쉬질의** : $\mathcal{F}1$이 (m_i, U_i, k_i)를 주면, $h_i\in Z_q^*$를 임의로 선택하여 답하고, $L_{H_3}=<m_i, U_i, k_i, h_i>$ 리스트에 추가한다.
- **ExtrPartSK질의** : $\mathcal{F}1$이 ID_i에 대해 ExtrPartSK 질의를 요청할 때, 만약 $ID_i\neq A$라면 L_{H_1} 리스트를 찾아 ID_i에 대한 $<ID_i, t_i>$를 찾은 후, $D_{ID_i}=t_iP_{pub}$를 응답한다. 그 외의 경우 C_I는 $Fail$을 출력 후 시뮬레이션을 중단한다.
- **RequestPK질의** : $\mathcal{F}1$이 ID_i에 대한 RequestPK 질의

를 요청할 때, $< ID_i, PK_{ID_i}, x_{ID_i}, c >$가 L_K 리스트에 있다면 리스트에 있는 정보로 대답하고, 리스트에 없다면 임의의 $x_{ID_i} \in Z_q^*$를 선택하여 $PK_{ID_i} = x_{ID_i} P$ 를 응답한다. 그리고 응답한 $< ID_i, PK_{ID_i}, x_{ID_i}, c >$를 L_K리스트에 추가한다. 여기서 c는 모두 1로 설정한다.

- **ExtraFullSK질의** : $\mathcal{F}1$이 ID_i에 대한 Extra FullSK 질의를 요청할 때,

1) $ID_i \neq A$일 경우, C_I은 먼저 L_{H_1}리스트를 찾아 ID_i에 대한 t_i값을 얻고, L_{H_1}리스트에 ID_i정보가 존재하지 않는다면 스스로 H_1질의를 시행하고 리스트에 저장한다. ID_i에 대한 L_K리스트를 찾아 $< ID_i, PK_{ID_i}, x_{ID_i}, c >$정보를 얻은 후, PK_{ID_i}로 L_{H_2}리스트를 검색하여 해당되는 y_{ID_i}를 얻는다. 만약 리스트들에 ID_i에 대한 정보가 없다면 스스로 질의를 하여 생성 후 리스트에 저장한다. 모은 정보들을 이용하여 정당한 S_{ID_i}를 계산하여 대답한다.
2) $ID_i = A$일 경우, $Fail$을 출력하고 시뮬레이션을 중단한다. 우리는 서명자를 A로, 지정된 검증자를 B라고 가정하였기 때문에 가정대로 C_I이 A와 B에 대하여 위조된 서명을 내놓는다고 하면, C_I은 그 이전에 A와 B에 대한 ExtraPartSK 질의와 ExtraFullSK 질의를 하지 않았을 것이다.(게임에서 출력 단계 1번에서 언급함) 따라서 ExtraPartSK 질의와 ExtraFullSK 질의에서의 $Fail$은 의미가 없다.

- **ReplacePK 질의** : $\mathcal{F}1$이 (ID_i, PK'_{ID_i})를 주면, C_I는 일치하는 정보가 있는지 L_K리스트를 찾고, 해당하는 ID_i에 대한 $< ID_i, PK_{ID_i}, x_{ID_i}, c >$가 리스트에 존재한다면 $PK_{ID_i} = PK'_{ID_i}$로 대체하고, $c = 0$으로 설정한다. 만약 리스트에 해당 ID_i에 대하여 일치하는 정보가 없을 경우 $\mathcal{F}1$ 으로부터 받은 ID_i, PK'_{ID_i}에 대하여 $< ID_i, PK'_{ID_i}, x_{ID_i}, c >$를 만들어 리스트에 추가한다.
- **Sign질의** : $\mathcal{F}1$이 (m_i, ID_i, ID_j)에 대한 Sign질의를 할 때,

1) $ID_i \neq A$라면 C_I은 ID_i, ID_j에 대하여 k_{ij}를 계산하고 임의로 $U_j, h_j \in Z_q^*$를 선택하여 $L_{H_3} = < m_i, U_i, k_i, h_i >$ 리스트에 추가한다. 스스로 ID_i에 대한 ExtraFullSK 질의를 하여 (m_i, ID_i, ID_j)에 대한 서명을 생성하여 대답한다.
2) $ID_i = A$일 경우, 이때 A에 대한 Full-Secret Key를 알 수 없으므로 지정된 검증자 ID_j 의 Transcript-Simulation 알고리즘을 이용하여 서명을 생성 후 답한다. 트랜스크립트를 생성하는데 필요한 ID_j의 Full-Secret Key는 스스로 ID_j에 대한 ExtraFullSK 질의를 하여 얻는다. 마찬가지로 서명 시 만들어지는 U_j, h_j는 $L_{H_3} = < m_i, U_i, k_i, h_i >$리스트에 추가한다.

- **Verify질의** : $\mathcal{F}1$이 $(\sigma_i, m_i, ID_i, ID_j)$에 대한 Verify 질의를 하면, 올바른 답을 해주기 위하여 h_i, h_j 가 필요하다.

1) 만약 $\mathcal{F}1$이 ID_i, ID_j에 대하여 H_3와 $RequestPK$ 질의를 한 경우, L_{H_3}리스트에서 $(m, U_i), (m, U_j)$이 같은 $< m, U_i, k, h_i >, < m, U_j, k, h_j >$ 목록들을 찾는다. 찾은 항들은 k값이 모두 다를 수도 있고 같은 것이 있을 수도 있다. 올바르게 시뮬레이션 해주기 위해서는 이들 리스트 중에서 $(m, U_i, k), (m, U_j, k)$에 대한 올바른 해쉬 값 h_i, h_j을 찾아야 하는데 이를 위해 k가 필요하다. 따라서 $< ID, PK_{ID}, x_{ID}, c >$형태의 L_K리스트를 찾고 이로부터 ID_i, ID_j에 대한 k값을 계산한다. 만약 $c = 0$이라면 PK_{ID}가 $\mathcal{F}1$에 의해 대체 된 것이므로 $\mathcal{F}1$에게 PK_{ID}에 대한 비밀 값 x_{ID}를 요청하여 정보를 얻는다. 이렇게 얻은 k값으로 위에서 찾은 L_{H_3} 리스트 항들과 비교하여 같은 k값을 갖는 $< m, U_i, k, h_i >, < m, U_j, k, h_j >$항을 찾고, 이 h_i, h_j정보를 이용해 검증 알고리즘을 통하여 서명을 검증한다. $\mathcal{F}1$이 내놓은 서명이 정당 하다면 accept를, 그렇지 않다면 reject를 출력해준다.
2) $\mathcal{F}1$이 ID_i, ID_j에 대하여 H_3와 $RequestPK$ 질의를 하지 않아 L_{H_3}리스트와 L_K리스트에 저장되어 있지 않은 경우, $\mathcal{F}1$이 내놓은 서명에 관하여 $null$ 을 출력한다.

마침내, $\mathcal{F}1$은 위조된 서명 (σ, m, ID_s, ID_v)을 출력한다. 만약 $ID_s, ID_v \neq ID_A, ID_B$라면 C_I은 $Fail$을 출력하고

시뮬레이션을 중단한다. 그렇지 않다면 C_I은 *forking lemma* [9]에서와 같이 랜덤 테이프를 유지한 채, 다른 랜덤 오라클 H_4를 선택하여 $\mathcal{F}1$에 대한 시뮬레이션을 재 수행한다. 다른 랜덤 오라클 H_4를 사용하였으므로 C_I은 $(ID_s, ID_v, m, h_s, h_v, U_s, U_v, V_s, V_v)$와 $(ID_s, ID_v, m, h_s', h_v', U_s', U_v, V_s', V_v)$를 얻을 수 있다. 만약 시뮬레이션이 성공하고 공격자 $\mathcal{F}1$이 정당한 두 서명값을 내놓는다면 L_{H_2}, L_{H_3}, L_K리스트로부터 $< PK_{ID}, y_{ID} >, < m, U, k, h > < ID, PK_{ID}, x_{ID}, c >$형태를 찾아 다음과 같이 CDH문제를 계산할 수 있다.

$$(x_{ID_s} + y_{ID_s})\frac{V_s - V_s'}{h_s - h_s'} = D_{ID_s} = abP$$

따라서 제안한 CL-SDVS기법에 대해 정당한 서명을 위조해 낼 능력을 가진 공격자 $\mathcal{F}1$이 존재한다면, CDH 문제를 계산할 수 있는 PPT(Probablistic Polynomial Time)공격자가 존재하게 된다.

정리 2. *랜덤오라클 모델에서 제안하는 CL-SDVS기법은 CDH문제가 어렵다고 가정할 때, 공격자 유형 II ($\mathcal{F}2$)에 대하여 적합하게 선택된 메시지 공격에 안전하다.*

증명. 사용자의 마스터키를 아는 공격자 $\mathcal{F}2$가 존재한다고 가정하자. 이점을 갖는 공격자 $\mathcal{F}2$가 존재 할 때, Computational Diffie-Hellman 문제를 푸는 알고리즘 C_{II}가 존재함을 보일 것이다. 정리 1에서와 마찬가지로 C_{II}가 CDH문제를 풀 수 있도록 임의의 $a, b \in Z_q^*$ 와 $P \in G$를 선택하여 CDH문제의 입력값 (P, aP, bP)을 전달한다. C_{II}는 CDH문제를 풀기 위하여 공격자 $\mathcal{F}2$가 알아채지 못하도록 공격 환경을 시뮬레이션 해줌으로써 공격자 $\mathcal{F}2$가 주는 정보를 이용한다. C_{II}는 $\mathcal{F}2$에게 시스템 파라미터를 전달한다. C_{II}는 마스터키 s를 알고 있으므로 이 게임에서는 ExtrPartSK와 ReplacePK 오라클은 필요하지 않다. 우리는 RequstPK 질의는 H_2질의 전에 이루어진다고 가정하며 일반성을 잃지 않고 ExtrFullSK, Sign, Verify 질의를 요청하기 전에 사전에 H_1과 H_2, H_3 해쉬 질의를 한다고 하자. 질의에 일관성 있게 답하기 위하여 C_{II}는 $L_{H_1}, L_{H_2}, L_{H_3}, L_K$ 리스트와 L_S리스트를 유지한다. 이들 리스트는 초기에는 비어있다.(null) C_{II}는 $\mathcal{F}2$의 오라클 질의에 다음과 같이 시뮬레이션 한다.

- ***H_1* 질의** : $\mathcal{F}2$가 최대 qH개의 H_1질의를 요청한다고 하자. C_{II}는 게임 1에서와 마찬가지로 $A, B \in [1....q_{H_1}]$를 선택한다. $\mathcal{F}2$가 $ID_i (1 \le i \le q_{H_1})$에 대하여 H_1질의를 하면, 임의의 $t \in Z_q^*$를 선택하여 $Q_{ID_i} = t_i P (i \ne A)$로 대답한다. 그리고 이를 $< ID_i, t_i >$형태로 L_{H_1}리스트에 추가한다.
- ***H_2, H_3* 질의** : H_2 질의와 H_3질의는 게임 1과 동일하게 시뮬레이션 한다.
- **RequestPK 질의** : $\mathcal{F}2$가 ID_i에 대한 RequestPK 질의를 요청할 때,

1) 만약 $i = A$라면 $PK_{ID_A} = aP$로 $i = B$라면 $PK_{ID_B} = bP$를 준다.
2) 그렇지 않을 경우, $< ID_i, PK_{ID_i}, x_{ID_i} >$가 L_K 리스트에 있다면 리스트에 있는 정보로 대답하고, 리스트에 없다면 임의의 $x_{ID_i} \in Z_q^*$를 선택하여 $PK_{ID_i} = x_{ID_i} P$ 를 응답한다. 그리고 모든 경우를 $< ID_i, PK_{ID_i}, x_{ID_i} >$형태로 L_K리스트에 추가한다.

- **ExtraFullSK 질의** : $\mathcal{F}2$가 ID_i에 대해 질의를 요청하면,

1) 만약 $ID_i \ne A, B$라면, C_{II}은 먼저 L_{H_1}리스트를 찾아 ID_i에 대한 t_i값을 얻고, L_{H_1}리스트에 ID_i 정보가 존재하지 않는다면 스스로 H_1질의를 시행하고 리스트에 저장한다. ID_i에 대한 L_K리스트를 찾아 $< ID_i, PK_{ID_i}, x_{ID_i}, c >$정보를 얻은 후, PK_{ID_i}로 L_{H_2}리스트를 검색하여 해당되는 y_{ID_i}를 얻는다. 만약 리스트들에 ID_i에 대한 정보가 없다면 스스로 질의를 하여 생성 후 리스트에 저장한다. 모은 정보들을 이용하여 정당한 S_{ID_i}를 계산하여 대답한다.
2) 만약 $ID_i = A$ or B라면 $Fail$을 출력하고 시뮬레이션을 중단한다.

- **Sign 질의** : $\mathcal{F}2$가 (m_i, ID_i, ID_j)에 대한 Sign 질의를 할 때,

1) $ID_i \ne A, ID_i \ne B$일 경우, C_{II}는 ID_i, ID_j에 대하여 k_{ij}를 계산하고 임의로 $U_j, h_j \in Z_q^*$를 선택

하여 $L_{H_3}=<m_i,U_i,k_i,h_i>$리스트에 추가한다. 스스로 ID_i에 대한 ExtraFullSK 질의를 하여 (m_i,ID_i,ID_j)에 대한 서명을 생성하여 대답한다.

2) $ID_i=A(\text{or}B)$일 경우, 이때는 ID_j에 따라 두 가지 경우로 나누어 진다. $ID_i=A(B), ID_j\neq B(A)$일 때는, $A(B)$에 대한 Full-Secret Key를 알 수 없으므로 지정된 검증자 ID_j 의 Transcript-Simulation 알고리즘을 이용하여 서명을 생성 후 답한다. 트랜스크립트를 생성하는데 필요한 ID_j의 Full-Secret Key는 스스로 ID_j에 대한 ExtraFullSK 질의를 하여 얻는다. 마찬가지로 서명 시 만들어지는 U_j, h_j는 $L_{H_3}=<m_i,U_i,k_i,h_i>$ 리스트에 추가한다.

3) $ID_i=A(B), ID_j=B(A)$일 경우, A,B의 Full-Secret Key를 모두 알 수 없으므로 임의로 $U_i,U_j,V_i,V_j\in Z_q^*$ 를 선택하여 답하고 $L_S=<U_i,U_j,V_i,V_j>$리스트에 추가한다.

4) CL-SDVS기법으로 생성된 서명 값의 분포는 랜덤 값을 임의로 뽑을 때의 분포와 같기 때문에 공격자는 임의로 선택하여 답해주더라도 구별할 수 없다.

• **Verify 질의** : ℱ2가 (σ_i,m_i,ID_i,ID_j)에 대한 Verify 질의를 하면, 먼저 L_S 리스트를 확인하여 리스트에 저장된 정보와 일치하는 것인지 확인한다.

1) 일치하는 것이 있을 경우, Sign질의에서 답해준 서명이므로 Verify의 답으로 accept를 출력한다.

2) 일치하는 정보가 없을 경우, 게임 1에서와 같이 동일하게 시뮬레이션 한다. 올바른 답을 해주기 위하여 h_i,h_j 가 필요하다.

- 만약 ℱ2가 ID_i,ID_j에 대하여 H_3질의와 RequestPK 질의를 한 경우, L_{H_3}리스트에서 $(m,U_i),(m,U_j)$이 같은 $<m,U_i,k,h_i>$, $<m,U_j,k,h_j>$목록들을 찾는다. 찾은 항들은 k값이 모두 다를 수도 있고 같은 것이 있을 수도 있다. 올바르게 시뮬레이션 해주기 위해서는 이들 리스트 중에서 $(m,U_i,k),(m,U_j,k)$에 대한 올바른 해쉬값 h_i,h_j을 찾아야 하는데 이를 위해 k가 필요하다. 따라서 $<ID,PK_{ID},x_{ID}>$형태의 L_K리스트를 찾고 이로부터 ID_i,ID_j에 대한 k값을 계산한다. 이를 위에서 찾은 L_{H_3} 리스트 항들과 비교하여 k값이 같은 $<m,U_i,k,h_i>,<m,U_j,k,h_j>$ 항을 찾고, 찾은 h_i,h_j정보를 이용해 검증 알고리즘으로 서명을 검증한다. ℱ2가 내놓은 서명이 정당하다면 accept, 그렇지 않다면 reject를 출력한다.

- ℱ2가 ID_i,ID_j에 대하여 H_3질의와 RequestPK 질의를 하지 않은 경우, C_{II}는 ℱ2가 질의를 하지 않았으므로 L_{H_3}리스트와 L_K리스트로부터 필요한 정보를 찾을 수 없다. 그러나 ℱ2가 위조된 서명을 하기 위해서는 반드시 H_3해쉬 값과 PK_{ID}가 필요하므로, 이를 정당하게 만들었다면 반드시 질의를 했을 것이다. 따라서 리스트에 없는 경우는 Verify질의에 대하여 $null$을 출력한다.

마침내, ℱ2는 위조된 서명 (σ,m,ID_s,ID_v)을 출력한다. ℱ2는 서명을 위조하기 위해 위조된 서명에 대해 높은 확률로 해쉬 질의를 했을 것이다. 만약 $ID_s,ID_v\neq ID_A,ID_B$라면 C_{II}는 $Fail$을 출력하고 시뮬레이션을 중단한다. 그렇지 않을 경우, C_{II}는 H_3리스트에서 $<m,U_s,k_s,h_s>$, $<m,U_v,k_v,h_v>$를 찾아 $k_s(k_v)$를 CDH문제의 답으로 출력한다.

$$k_s(k_v)=aPK_{ID_v}(bPK_{ID_s})=abP$$

따라서 제안한 CL-SDVS기법에 대해 정당한 서명을 위조해 낼 능력을 가진 공격자 ℱ2가 존재한다면, CDH 문제를 계산할 수 있는 PPT(Probablistic Polynomial Time)공격자가 존재하게 된다.

정리 3. *우리의 CL-SDVS기법은 강한 지정된 검증자 서명기법이다.*

지정된 검증자 기법에서는 실제 서명자가 생성한 서명과 구별 불가능한 트랜스크립트를 지정된 검증자가 생성할 수 있어야 한다. 지정된 검증자는 트랜스크립트 알고리즘을 통해 서명자가 생성한 서명과 똑같은 방법으로 검증되는 서명을 생성할 수 있다. 따라서 제 3자는 서명에 대하여 지정된 검증자 생성한 것인지 실제 서명자가 생성한 것인지 확신할 수 없으므로 지정된 검증자

기법이라 할 수 있다.

또한 제안하는 CL-SDVS기법은 서명자의 프라이버시를 보호한다. 서명을 검증하기 위해서는 지정된 검증자의 비밀 값과 개인키가 필요하기 때문에 지정된 검증자 이외에는 서명을 검증 할 수 없다. 또한 서명자를 구분하기 위해서는 서명자의 공개키와 같이 서명자를 구분하기 위한 정보를 서명으로부터 유추할 수 있어야 하는데, 서명의 각 항에는 $U_B \in G_1$, $r \in Z_q^*, \partial_B \in Z_q^*$와 같이 랜덤수가 섞여있고 이 값들은 모두 균등하게(uniformly) 선택된다. 따라서 서명의 각 항은 모두 균일하게 랜덤하게 된다. 제 3자는 서명자 또는 검증자의 비밀키 정보 없이는 서명에 대하여 의미있는 정보를 얻을 수 없으며 누가 서명자인지 구별할 수 없다.

정리 4. *우리의 CL-SDVS기법은 서명능력의 양도성을 가지지 않는다.*

정당하게 검증되는 서명(U_A, U_B, V_A, V_B)을 생성하기 위해서는, V_A항에 서명자의 개인키 S_{ID_A}가 필요하다. 따라서 서명자의 개인키를 노출하지 않고 제 3자에게 서명 능력을 양도하는 것은 불가능 하다. 검증자의 경우, 트랜스크립트 생성시 V_B' 항의 계산에 있어 반드시 검증자의 개인키S_{ID_B}가 직접적으로 요구되므로 검증자 또한 개인키의 노출 없이 제 3자에게 구별 불가능한 트랜스크립트의 생성 능력을 양도할 수 없다. 따라서 우리의 CL-SDVS기법에서는 양도성의 문제점을 가지지 않는다.

Ⅳ. 결 론

본 논문에서는 인증서가 없는 강한 지정된 검증자 서명 기법을 최초로 제안하였다. 인증서가 필요 없는 공개키 기반의 전자 서명 기법은 인증서 기반의 인증서 관리 문제나 ID기반의 키 위탁 문제 등을 수반하지 않는 새로운 형태의 전자서명이다. 2006년 Huang 등이 인증서가 없는 지정된 검증자 서명 기법을 제안[3]하였으나 Huang 등의 CL-DVS기법은 같은 메시지에 대하여 매번 동일한 서명이 나오는 특징을 가지고 있어 서명자와 검증자가 동일한 메시지를 반복해서 주고 받을 경우 공격자는 메시지와 서명의 한 쌍만을 저장하고도 어떤 서명자와 검증자 사이의 서명인지 알 수 있으며, 만약 공격자가 서명된 메시지를 지정된 검증자에게 도착하기 전에 얻는다면 그 서명이 지정된 검증자가 생성한 것이 아닌 서명자가 생성했음을 알 수 있다. 따라서 이러한 같은 메시지에 대해 매 번 동일한 서명이 나오는 특징은 문제가 될 수 있다. 또한 Huang 등의 CL-DVS기법은 양도성의 문제점을 가지는데, Huang 등의 CL-DVS 서명은 다음과 같으며

$$\sigma = H(m \parallel e(Q_A, D_B + x_B Y_A))$$

검증자 B에 대한 부분정보 $e(Q_A, D_B + x_B Y_A)$ 값을 제 3자에게 주게 되면 제 3자는 지정된 검증자 B에 대하여 서명자 또는 검증자의 비밀키를 모르고도 임의의 메시지에 대한 정당한 서명을 생성할 수 있다. 제안하는 기법은 이러한 양도성의 문제점을 가지지 않는다.

본 논문에서는 같은 메시지에 대해 동일한 서명이 생성되는 성질을 없애고 강한 성질을 추가하여 서명자의 프라이버시 보호를 강화한 인증서가 없는 지정된 검증자 서명기법을 최초로 제안하였다. 기존의 Huang등의 CL-DVS기법에 비하여 다소 효율적이지 않으나 서명자의 프라이버시를 보호하고 Haung등의 기법이 갖고 있는 서명권리 양도의 문제점을 가지지 않는다.

참고문헌

[1] M.Jakobsson, K.Sako, R.Impagliazzo, Designated verifier proofs and their applications, in: Advances in Cryptology Eurocrypt96, LNCS, vol.1070, Springer Verlag,1996, pp. 143-154.

[2] S.S. Al-Riyami, and K.G. Paterson, Certificateless Public Key Cryptography, Advances in Cryptology ASIACRYPT 2003, LNCS 2894, pp. 452-473, Springer-Verlag, 2003.

[3] X. Huang, W. Susilo, Y. Mu, and F. Zhang, Certificateless Designated Verifier Signature Schemes, The 20th International Conference on Advanced Information Networking and Applications 2006 (AINA2006), 2006.

[4] S. Saeednia, S. Kramer, and O. Markovitch, An Efficient Strong Designated Verifier Signature Scheme, The 6th International Conference on Information Security and Cryptology 2003

(ICISD 2003), LNCS 2971, pp. 40-54, Springer-Verlag, 2003.

[5] W. Susilo, F. Zang, and Y.Mu, Identity-based Strong Designated Verifier Signature Schemes, The 9th Australasian Conference on Information Security and Privacy 2004 (ACISP 2004), LNCS 3108,pp. 313-324, Springer-Verlag, 2004.

[6] H.Lipmaa, G.Wang and F.Bao, Designated verifier signature schemes: attacks, new security notions and a new construction, in: ICAP 2005, LNCS, vol. 3580, Springer-Verlag, 2004, pp. 459-471.

[7] X. Huang, W. Susilo, Y. Mu, and F. Zhang, On the security of certificateless signature schemes from Asiacrypt 2003., CANS 2005, LNCS 3810, pp. 13-25, Springer-Verlag, 2005.

[8] D. Boneh and M. Franklin, “Identity-based encryption from the Weil pairing”, Crypto'01, LNCS 2139, pp. 213-229, 2001.

[9] D. Pointcheval and J. Stern, Security arguments for digital signature and blind signature., J. of Cryptology, Vol. 13, pp. 361-396, 2000.

<著 者 紹 介>

구 영 주 (Young ju Koo) 학생회원
2007년 2월 : 숭실대학교 수학과 학사
2007년 3월~현재 : 고려대학교 정보보호대학원 석사과정
<관심분야> 암호프로토콜, 암호 이론

천 지 영 (Ji Young Chun) 학생회원
1997년 2월 : 이화여자대학교 수학과 학사
2006년 2월 : 단국대학교 수학과 석사
2006년 3월~현재 : 고려대학교 정보보호대학원 박사과정
<관심분야> 암호 이론, PET 기술, 유비쿼터스 보안

최 규 영 (Kyu Young Choi) 학생회원
2002년 2월 : 고려대학교 수학과 학사
2004년 8월 : 고려대학교 정보경영공학전문대학원 석사
2004년 9월~현재 : 고려대학교 정보보호대학원 박사과정
<관심분야> 암호 프로토콜, 암호 이론, 그룹키 연구

이 동 훈 (Dong Hoon Lee) 종신회원
1983년 8월 : 고려대학교 경제학사
1987년 12월 : Oklahoma University 전산학 석사
1992년 5월 : Oklahoma University 전산학 박사
1993년 3월~1997년 2월 : 고려대학교 전산학과 조교수
1997년 3월~2001년 2월 : 고려대학교 전산학과 부교수
2001년 2월~현재 : 고려대학교 정보보호대학원 교수
<관심분야> 암호프로토콜, 암호이론, USN 이론, 키 교환, 익명성 연구, PET 기술