Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.25, NO.3, Jun. 2015 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2015.25.3.531

전방향 안전성을 만족하며 인증서 기반이 아닌 비대화형 키 교환 프로토콜*

이 영 경,† 엄 지 은, 서 승 현, 이 동 훈‡
고려대학교

Certificateless Non-Interactive Key Exchange Protocol with Forward Secrecy*

Young Kyung Lee,† Ji Eun Eom, Seung-Hyun Seo, Dong Hoon Lee‡
Korea University

요 약

비대화형 키 교환 프로토콜은 키 교환을 위한 별도의 통신을 없애 시스템 내 효율성을 높일 수 있는 키 교환 프로토콜이다. 그러나 개인키 업데이트를 하지 않는 일반적인 비대화형 키 교환 프로토콜은 임시키를 사용하지 않기 때문에 전방향 안전성(forward secrecy)을 만족하지 못한다. 2012년 Sang 등은 인증서를 사용하지 않는 공개키 기반의 비대화형 키 교환(CL-NIKE : Certificateless Non-Interactive Key Exchange) 프로토콜을 제안하였지만, 이에 대한 안전성 증명이 없고 전방향 안전성을 제공하지 않는다. 본 논문에서는 새로운 CL-NIKE와 그에 대한 안전성 모델을 제안하고, 제안한 프로토콜이 해당 모델에서 안전함을 DBDH 가정(Decision Bilinear Diffie-Hellman Assumption)을 기반으로 증명한다. 추가적으로 다선형 함수(Multilinear map)를 사용한 개인키 업데이트를 통해 전방향 안전성을 만족하는 인증서 기반이 아닌 비대화형 키 교환 프로토콜을 제안하고 증명한다.

ABSTRACT

A non-interactive key exchange protocol provides an efficiency of overall system by eliminating additional communication. However, traditional non-interactive key exchange protocols without updating a private key fail to provide forward secrecy, since there is no usage of ephemeral key for randomness of session key. In 2012, Sang et al. proposed a certificateless non-interactive key exchange(CL-NIKE) protocol, but they do not prove the security of the protocol and it does not provide forward secrecy. In this paper, we propose a new CL-NIKE protocol and it's security model. Then we prove the proposed protocol is secure under the security model based on DBDH(Decision Bilinear Diffie-Hellman) assumption. Moreover, we propose a CL-NIKE protocol with forward secrecy which updates user's private key by using multilinear map and prove it's security.

Keywords: Certificateless, Non-Interactive Key Exchange, Security Model, Forward Secrecy

I. 서 론

1.1 개요

비대화형 키 교환(NIKE : Non-Interactive Key Exchange) 프로토콜은 통신하고자 하는 상대방의 공개키를 이용하여, 별도의 통신 없이 기밀 통신을 위한 공통키를 생성할 수 있는 키 교환 프로토콜이다.

접수일(2015년 1월 13일), 수정일(2015년 4월 16일), 게재확정일(2015년 5월 22일)
* 이 논문은 2015년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.R0126-15-1090, 계층적 식별자를 가진 인터넷 개체의 공개키 인증 구조 연구)
† 주저자, dudrudve@korea.ac.kr
‡ 교신저자, donghlee@korea.ac.kr(Corresponding author)

즉, 비대화형 키 교환 프로토콜에서는 기밀 통신용 공통키를 계산하기 위해 상대방의 공개키 정보와 본인의 개인키 정보만 사용함으로써, 기존의 키 교환 프로토콜들에서 수반되었던 통신 부담을 효과적으로 제거하였다. 따라서 시스템 내의 전체통신량이 부담되는 환경이나 전송횟수에 민감한 환경에서 비대화형 키 교환 프로토콜이 효율적으로 사용될 수 있다.

대표적인 비대화형 키 교환 프로토콜은 Diffie-Hellman 키 교환 프로토콜[9]로, 상대방의 공개키에 자신의 개인키를 지수승 연산하여 공통된 키를 계산한다. 2006년, Bernstein[6]에 의해서, PKI (Public Key Infrastructure) 기반 비대화형 키 교환 프로토콜의 안전성 모델이 제안된 이후로 비대화형 키 교환 프로토콜에 관한 연구가 활발하게 진행되었다[5,7,8,10]. 그러나 일반적인 비대화형 키 교환 프로토콜은 키 교환 시 매번 동일한 개인키를 사용하기 때문에 사용자의 개인키가 노출되었을 때 이전에 생성한 세션키(session key)에 대한 정보를 알 수 없어야 하는 전방향 안전성(forward secrecy)을 만족하지 못한다. 2014년, Pointcheval과 Sanders의 연구[8]에서 다선형 함수(multilinear map)를 이용하여 전방향 안전성을 만족하는 비대화형 키 교환 프로토콜이 제안되었다. 그러나 PKI 기반의 비대화형 키 교환 프로토콜은 인증서 관리(인증서 저장 및 분배, 폐기, 검증 계산 등)에 대한 부담을 가지고 있다.

1984년 Shamir는 신원정보(ID : identity)를 기반으로 한 공개키 암호시스템[3]을 제안하여, PKI 기반 공개키 암호시스템의 인증서 관리 부담 문제를 해결하였다. ID 기반 공개키 암호시스템 (Identity-based Public Key Cryptography)은 사용자의 e-mail 주소나 IP 주소 등 사용자의 신원을 나타낼 수 정보들을 공개키로 사용함으로써, 기존 PKI 기반 공개키 암호시스템이 인증서를 이용하여 사용자와 공개키의 소유관계를 증명하는 관리적 부담을 제거하였다. 그러나 ID 기반 공개키 암호시스템에서는 키 생성 기관(KGC : Key Generation Center)이 마스터키(master key)와 등록된 사용자의 ID 정보를 이용하여, 모든 사용자들의 개인키를 생성하기 때문에 키 위탁(key escrow) 문제가 있다. 즉, KGC가 마스터키를 이용하여 사용자들의 암호문을 복호화하거나 사용자대신 서명을 생성하는 등의 문제가 발생할 수 있다.

인증서 관리의 부담과 키 위탁 문제를 해결하고자, 2003년에 Al-Riyami와 Paterson은 기존의 PKI 기반 공개키 암호시스템과 ID 기반 공개키 암호시스템의 장점을 결합하여, 인증서를 사용하지 않는 공개키 암호시스템 (CL-PKC : Certificateless Public Key Cryptography)[1]을 제안하였다. 인증서를 사용하지 않는 공개키 암호시스템에서는 사용자의 개인키가 KGC에 의해 생성된 부분 개인키(partial private key)와 사용자가 직접 생성한 비밀값(secret value)의 특별한 조합으로 구성됨으로써, 공개키 인증서의 검증 없이 사용자의 공개키가 해당 프로토콜을 수행함으로써 검증된다. 또한 KGC는 사용자가 생성한 비밀값을 알 수 없고, 오직 정당한 사용자만이 부분 개인키와 비밀값을 가지고 있기 때문에 키 위탁 문제가 해결된다.

본 논문에서는 인증서를 사용하지 않는 공개키 암호시스템을 기반으로 하여, 공개키 인증서 관리 부담과 키 위탁 문제를 해결하는 비대화형 키 교환 프로토콜(이하 CL-NIKE)을 제안한다. 제안하는 CL-NIKE의 안전성을 보이기 위해, CL-NIKE의 안전성 모델을 제시하고, DBDH 문제의 어려움에 기반하여 프로토콜의 안전성을 수학적으로 증명한다. 또한, 다선형 함수를 이용하여 전방향 안전성을 만족하는 CL-NIKE 프로토콜(이하 CL-NIKE-fs)을 제안하고, n-MDDH 문제에 기반하여 안전성을 증명한다.

1.2 기여도

CL-NIKE는 2012년, Sang 등의 연구[4]에서 처음 제안되었다. 그러나 [4]에서는 안전성 증명을 위한 모델이 정의되지 않았고, 단순한 안전성 분석만 제시하였다. 안전성 모델에 기반한 안전성 증명이 없다면 프로토콜 자체의 보안 취약점이 존재할 수 있기 때문에 단순한 안전성 분석만으로는 프로토콜의 안전성을 보장할 수 없다.

본 논문에서는 CL-NIKE의 안전성 모델을 제안하였다. 그리고 설계한 CL-NIKE 프로토콜의 안전성을 제안한 안전성 모델에서 DBDH(Decisional Bilinear Diffie-Hellman) 문제의 어려움에 기반하여 증명하였다. 추가로 다선형 함수를 사용하여 전방향 안전성을 만족하는 CL-NIKE(CL-NIKE-fs) 프로토콜을 설계하였고, CL-NIKE-fs의 안전성 모델과 제안한 프로토콜의 안전성을 n-MDDH(n-Multilinear Decisional Diffie-Hellman) 문제의 어려움에 기반하여 증명하였다.

제안된 프로토콜들은 인증서를 사용하지 않는 공개키 암호시스템에서 부가적인 통신이 필요 없는 키 교환 프로토콜로써 시스템 전체의 통신부담을 개선시킬 수 있는 키 교환 프로토콜이다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 관련 연구를 살펴보고, Ⅲ장에서는 배경지식을 다룬다. Ⅳ장에서는 CL-NIKE 프로토콜과 안전성 모델을 제시하고, 안전성을 증명한다. Ⅴ장에서는 CL-NIKE-fs 프로토콜을 제안하고 안전성을 증명한다. 마지막으로 Ⅵ장에서 결론을 맺는다.

II. 관련 연구

2.1 비대화형 키 교환 프로토콜

비대화형 키 교환 프로토콜에서는 상대방의 공개키를 알고 있다는 가정 하에 추가적인 정보의 전송 없이 키 교환이 가능하다. 가장 기본적이고 널리 알려진 것은 Diffie-Hellman 키 교환 프로토콜이다[9]. 상대방의 공개키에 자신의 개인키를 지수연산 하는 형태로 단순하면서 효율적인 키 교환 프로토콜이다.

PKI 기반의 비대화형 키 교환 프로토콜의 안전성 모델은 2006년에 Bernstein[6]에 의해 처음 제안되었다. 2008년, Cash 등[7]은 생성된 세션키와 난수의 구분불가능성(indistinguishability)을 이용한 안전성 모델을 제안하였다. 생성된 세션키와 난수의 구분불가능성은 세션키의 어떠한 정보도 노출되지 않음을 의미한다. 이후 2013년, Freire 등[10]은 공격자가 선택한 공개키로 시스템에 등록할 수 있고, 등록된 공개키가 사용된 다른 사용자와의 세션키를 얻어볼 수 있는 능력을 추가한 안전성 모델을 제안하였다.

위 관련 연구들을 참고하여 본 논문에서는 세션키와 난수의 구분불가능성으로 안전성을 보이고, [10]에서 추가된 공격자의 능력을 고려하여 안전성 모델을 제안하였다.

2.2 비대화형 키 교환에서의 전방향 안전성

키 교환 프로토콜에서 전방향 안전성이란 사용자의 개인키가 노출되어도 이전에 맺었던 세션키의 안전성이 보장된다는 개념으로, 키 교환 프로토콜에서 전방향 안전성은 보안사고 피해를 감소시키는 중요한 보안 요구사항이다. 기존의 대화형 키 교환 (interactive key exchange) 프로토콜에서는 사용자들이 키 교환을 시행하기 전에 각각 임시(ephemeral) 공개키/개인키 쌍을 만들고, 생성된 임시 공개키를 교환하고 임시키를 세션키 생성 과정에 관여시킨다. 그러므로 사용자의 개인키가 노출되어도 공격자가 이전에 맺은 세션에서 사용된 임시 개인키를 모르면 세션키의 정보를 알아낼 수 없다.

임시키를 사용하지 않는 비대화형 키 교환 프로토콜에서는 전방향 안전성을 제공하기 위해 개인키를 주기적으로 업데이트하는 방법이 있다. 업데이트 이후에 개인키가 노출되어도 이전에 맺은 세션키 생성에 사용된 개인키를 알지 못하면 세션키의 정보를 알아낼 수 없다.

2014년, Pointcheval 과 Sanders의 연구[8]에서는 PKI 기반의 비대화형 키 교환 프로토콜에 전방향 안전성을 제공하기 위해 다선형 함수를 사용하여 개인키 업데이트를 수행한다. 다선형 함수는 일방향 함수로써 업데이트된 개인키로부터 업데이트 이전의 개인키를 계산할 수 없는 기능을 제공한다. 또한 개인키 업데이트 이후에도 사용자 공개키에서 지수에 해당하는 비밀값을 지속적으로 개인키의 지수로 유지할 수 있는 특성을 갖고 있다. 이러한 특성 덕분에 개인키 업데이트 이후에도 공개키의 변형 없이 두 사용자 간의 동일한 세션키 생성이 가능하다.

III. 배경지식

3.1 겹선형 함수(bilinear map)

위수가 큰 소수 q로 같은 두 곱셈 연산군 G_1과 G_2가 있을 때 군 G_1, G_2에서 모두 이산대수문제(discrete logarithm problem)가 어렵다고 할 수 있다. 이 때, 다음과 같은 조건을 만족하는 함수 $e : G_1 \times G_1 \rightarrow G_2$를 겹선형 함수라 한다.

- 겹선형성(bilinearity) : 임의의 군 원소 $g_1, g_2 \in G_1$와 $a, b \in Z_q^*$에 대해 $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$를 만족한다.
- 비소실성(non-degeneracy) : $e(g,g) \neq 1$을 만족하는 $g \in G_1$이 존재 한다.
- 계산 가능성(computability) : 임의의 $g_1, g_2 \in G_1$에 대해서 $e(g_1, g_2)$을 연산하는 효율적인 알고리즘이 존재한다.

3.2 DBDH(Decisional Bilinear Diffie-Hellman) 문제 및 가정

문제 : 위수가 큰 소수 q로 같은 두 곱셈 연산군 G_1와 G_2가 있고 겹선형 함수 $e: G_1 \times G_1 \rightarrow G_2$가 있을 때 $g \in G_1$을 G_1의 생성자라 하자. 이 때, DBDH 문제는 주어진 $< g, g^a, g^b, g^c, G >$ $(a,b,c \in Z_q^*)$에서 $G \stackrel{?}{=} e(g,g)^{abc}$를 결정하는 것이다.

가정 : 알고리즘 A는 $G = e(g,g)^{abc}$인 경우 1을, $G \neq e(g,g)^{abc}$인 경우 0을 출력하는 알고리즘이다. 이 때, 다항식 시간 안에 의미 있는(non-negligible) 확률로 계산할 수 있는 알고리즘 A가 존재하지 않는다면 DBDH 문제는 풀기 어렵다고 정의하며, 알고리즘 A의 이점(advantage) adv_A^{DBDH}는 다음과 같이 정의된다.

$$adv_A^{DBDH} := \left| \begin{array}{r} \Pr[A(g, g^a, g^b, g^c, e(g,g)^{abc}) = 0] \\ -\Pr[A(g, g^a, g^b, g^c, T) = 0] \end{array} \right|$$

$$where\ T \in G_2 / \{e(g,g)^{abc}\}$$

3.3 다선형 함수(leveled multilinear map)

위수가 소수 p로 동일한 n개의 순환군 $G_1, ..., G_n$을 가정하자. 이 때, 겹선형 함수 $e_{i,j}: G_i \times G_j \rightarrow G_{i+j}$ $(i,j \geq 1,\ i+j \leq n)$가 임의의 $g_i \in G_i, g_j \in G_j$와 $a,b \in Z_p$에 대해서 $e_{i,j}(g_i^a, g_j^b) = e_{i,j}(g_i, g_j)^{a \cdot b}$를 만족할 때 다선형 함수라 한다. 간략한 표기를 위해 $e_{i,j}$일 때 i,j가 명확하다면 e로 표기하고, $e(g_1, e(g_2, ..., g_n))$ 대신 $e(g_1, g_2, ..., g_n)$로 표기한다.

3.4 n-MDDH(n-Multilinear Decisional Diffie-Hellman) 문제 및 가정

문제 : $< g, g^{x_1}, ..., g^{x_{n+1}}, G >$ (g는 G_1의 생성자, $G \in G_n$)가 주어졌을 때 n-MDDH 문제는 $G \stackrel{?}{=} e(g^{x_1}, ..., g^{x_n})^{x_{n+1}}$를 결정하는 것이다.

가정 : 알고리즘 A는 $G = e(g^{x_1}, ..., g^{x_n})^{x_{n+1}}$인 경우 1을 $G \neq e(g^{x_1}, ..., g^{x_n})^{x_{n+1}}$인 경우 0을 출력하는 알고리즘이다. 이 때, 다항식 시간 안에 의미 있는 확률로 계산할 수 있는 알고리즘 A가 존재하지 않는다면 n-MDDH 문제는 풀기 어렵다고 정의하며, 알고리즘 A의 이점 adv_A^{n-MDDH}는 다음과 같이 정의된다.

$$adv_A^{n-MDDH} := \left| \begin{array}{r} \Pr[A(g, g^{x_1}, ..., g^{x_{n+1}}, e(g^{x_1}, ..., g^{x_n})^{x_{n+1}}) = 0] \\ -\Pr[A(g, g^{x_1}, ..., g^{x_{n+1}}, T)] \end{array} \right|$$

$$where\ T \in G_n / \{e(g^{x_1}, ..., g^{x_n})^{x_{n+1}})\}$$

IV. CL-NIKE 프로토콜

본 장에서는 CL-NIKE 프로토콜에서 사용되는 알고리즘을 정의하고, 안전성 모델을 제안한다. 다음으로 CL-NIKE 프로토콜을 설계하고 제안한 모델에서의 안전성 증명을 제시한다.

4.1 알고리즘

CL-NIKE 프로토콜은 다음과 같은 6개의 알고리즘으로 구성된다.

- 설정(*Setup*) : KGC에서 보안 상수(security parameter)를 입력으로 받아 시스템 내에 사용되는 파라미터(parameter)를 생성하는 알고리즘이다.
- 부분키 추출(*PartialPrivateKeyExtract*) : KGC에서 사용자의 *ID*를 입력으로 하여 해당 *ID*의 부분 개인키(*PartialPrivateKey*)를 생성하는 알고리즘이다.
- 비밀값 설정(*SetSecretValue*) : 각 사용자의 고유한 비밀값(*SecretValue*)을 생성하는 알고리즘이다.
- 공개키 설정(*SetPublicKey*) : 각 사용자가 자신이 생성한 비밀값과 공개 파라미터를 이용하여 공개키를 생성하는 알고리즘이다.
- 개인키 설정(*SetPrivateKey*) : 사용자가 KGC로부터 받은 부분 개인키와 자신이 생성한 비밀값을 입력으로 하여 사용자의 개인키를 생성하는 알고리즘이다.
- 세션키 생성(*SessionKeyGen*) : 두 사용자가 공유하는 세션키를 생성하는 알고리즘으로 한 사용자의 개인키와 상대방의 공개키를 입력으로 한다. 상대방 역시 자신의 개인키와 상대방의 공개키를 입력으로 생성하는데 이 때, 두 사용자가 생성한

세션키는 동일해야 한다.

4.2 안전성 모델

인증서를 사용하지 않는 키 교환 프로토콜의 안전성 모델에서 공격자는 크게 두 가지 유형으로 나눠진다[2]. KGC의 마스터키를 알고 있는 공격자는 명백히 위장공격(impersonation attack), 중간자공격(man-in-the-middle attack)이 가능하기 때문에 공격자 유형 1,2로 나누어 안전성 모델을 설계한다. 공격자 유형 1은 KGC의 마스터키에 대한 정보를 알 수 없고 사용자들의 공개키를 교체할 수 있는 능력이 있다. 공격자 유형 2는 사용자들의 공개키를 교체할 수 있는 능력이 없지만 KGC의 마스터키를 얻어낼 수 있다. 공격자의 목표는 특정 세션에서 자신이 선택한 두 ID 간에 생성되는 세션키가 정당한 세션키인지 아닌지를 구분하는 것이다.

프로토콜의 안전성을 증명하기 위해 가상의 게임(simulation)을 수행한다. 챌린저 S는 공격자를 이용하여 자신에게 주어진 NP(Nondeterministic-polynomial) 문제를 풀어야 하며, 프로토콜의 공격자가 실제 공격 환경과 가상의 게임을 구분하지 못하도록 해야 한다. 게임의 방식은 다음과 같다. 먼저 챌린저 S가 시스템을 설정(Setup)하여 공격자에게 공개 파라미터를 제공한다. 이후에 공격자는 챌린저 S에게 임의의 세션에 대한 세션키 또는 ID에 대응하는 개인키 등과 같이 공격에 필요한 정보를 질의하여 얻어낸다. 마지막으로 Test 단계에서 챌린저로부터 주어진 값이 공격 목표의 세션키인지 랜덤한 값인지를 판단한다.

4.2.1 공격자 유형 1

공격자 유형 1의 경우 공격자는 KGC의 마스터키에 대한 정보를 알 수 없지만 사용자들의 공개키를 교체할 수 있는 능력이 주어진다. ICN^1 (IND-CL-NIKE) 게임은 다음과 같이 정의된다.

- $Setup(k)$: S는 보안 상수k를 입력으로 파라미터를 생성하여 공개 파라미터는 공격자에게 전송하고 마스터키는 보관한다.
- $SetTarget(ID_A, ID_B)$: 공격자는 공격목표(ID_A, ID_B)를 선택하여 S에게 전송한다.
- $ReplacePublicKey(ID_i, PK_i)$: ID_i의 공개키를 PK_i로 교체하는 질의로 S는 사용자 집합에서 ID_i에 해당하는 공개키를 PK_i로 교체한다.
- $RevealSecretValue(ID_i)$: ID_i에 해당하는 비밀값을 얻어볼 수 있는 질의로 S는 ID_i에 해당하는 비밀값을 공격자에게 전송한다.
- $RevealPartialPrivareKey(ID_i)$: ID_i에 해당하는 부분 개인키를 얻어볼 수 있는 질의로 S는 해당 ID_i의 부분 개인키를 공격자에게 전송한다. 공격자 유형 1의 경우 공격목표로 선택한 ID_A, ID_B에 대해선 본 질의를 제한한다.
- $RevealSessionKey(ID_i, ID_j)$: 사용자 ID_i, ID_j 사이의 세션키를 얻어볼 수 있는 질의로 S는 (ID_i, ID_j)사이에서 생성되는 세션키를 공격자에게 전송한다. 목표(ID_A, ID_B)을 제외한 모든 경우의 질의에 대한 답을 전송해야 한다.
- $Test(ID_A, ID_B)$: 공격자는 본 질의를 통해 챌린지(challenge) 단계로 진입한다. S는 본 질의를 받으면 $b \in \{0,1\}$를 선택하는데 $b=0$인 경우 공격목표인 A와 B의 세션키를 전송하고, $b=1$인 경우에는 세션키 공간에서 랜덤값을 선택하여 전송한다. 공격자는 S로부터 받은 값이 공격목표의 세션키라고 판단하면 $guess=0$을 답하고, 랜덤값이라 판단하면 $guess=1$을 답한다. 이 때, 공격자 유형 1(E_1)의 이점(advantage)을 다음과 같이 정의한다.

$$Adv_{E_1}^{ICN_1}(k) = \left| \Pr[guess = b] - \frac{1}{2} \right|$$

4.2.2 공격자 유형 2

공격자 유형 2의 경우 공격자가 KGC의 마스터키를 아는 대신 사용자들의 공개키를 교체할 수 있는 능력을 제한한다. ICN^2 게임은 다음과 같이 정의된다.

- $Setup(k)$: 공격자 유형 1에서와 동일하다.
- $SetTarget(ID_A, ID_B)$: 공격자 유형 1에서와 동일하다.
- $RevealMasterKey()$: KGC의 마스터키를 얻을 수 있는 질의로, S는 본 질의를 받으면 KGC의 마스터키를 공격자에게 전송한다.

- $RevealSecretValue(ID_i)$: ID_i에 해당하는 비밀값을 얻어볼 수 있는 질의로 S는 ID_i에 해당하는 비밀값을 공격자에게 전송한다. 공격자 유형 2의 경우 공격목표로 선택한 ID_A, ID_B에 대해선 본 질의를 제한한다.
- $RevealSessionKey(ID_i, ID_j)$: 공격자 유형 1에서와 동일하다.
- $Test(ID_A, ID_B)$: 공격자 유형 1에서와 동일하다. 공격자 유형 2(E_2)의 이점을 다음과 같이 정의한다.

$$Adv_{E_2}^{ICN_2}(k) = \left| \Pr[guess = b] - \frac{1}{2} \right|$$

CL-NIKE 안전성 모델에서 공격자 유형 1,2에 대해, 다항식 시간 안에 상당한 이점을 가지고 게임을 이기는 알고리즘이 존재하지 않는다면, 즉, $Adv_{E_1}^{ICN_1}(k)$와 $Adv_{E_2}^{ICN_2}(k)$이 모두 무시할 정도로 작다면(negligible) CL-NIKE 프로토콜은 정의된 안전성 모델에서 안전하다.

4.3 제안하는 프로토콜

- 설정($Setup$) : KGC에서 보안 상수 k를 입력으로 하여 다음과 같은 파라미터를 생성한다.
  (1) 위수가 소수 q인 군 G_1, G_2와 생성자 $g \in G_1$, 겹선형 함수 $e : G_1 \times G_1 \rightarrow G_2$를 선택한다.
  (2) 해시함수 $H_1 : \{0,1\}^* \rightarrow G_1, H_2 : \{0,1\}^* \rightarrow G_1$, $H_3 : \{0,1\}^* \rightarrow \{0,1\}^k$를 생성한다.
  (3) 임의의 마스터키 $MK(= s \in Z_q^*)$를 선택한다. 마스터키를 제외한 모든 값과 함수들은 공개 파라미터로 사용자들에게 공개된다.
- 부분키 추출($PartialPrivateKeyExtract$) : KGC는 사용자(i)의 고유한 ID_i와 마스터키 s로 사용자의 부분 개인키 $D_i = Q_1^s (Q_i = H_1(ID_i))$를 생성한다. 생성한 D_i를 안전한 경로를 통해 사용자에게 전송한다.
- 비밀값 설정($SetSecretValue$) : 사용자는 임의의 비밀값 $x_i \in Z_q^*$를 선택한다.
- 공개키 설정($SetPublicKey$) : 사용자는 비밀값과 공개 파라미터를 이용하여 공개키 $PK_i = g^{x_i}$를 생성한다.

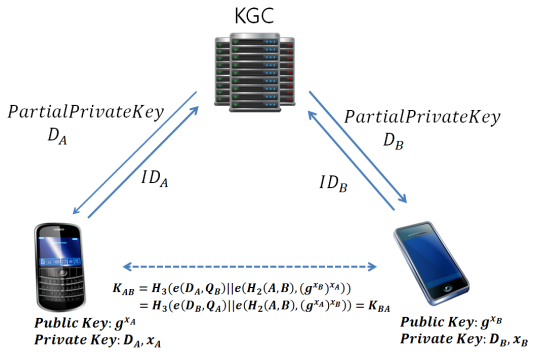

Fig. 1. CL-NIKE

- 개인키 설정($SetPrivateKey$) : 사용자는 부분 비밀키와 선택한 비밀값으로 비밀키 $< D_i, x_i >$를 설정한다.
- 세션키 생성($SessionKeyGen$) : 임의의 사용자 i와 사용자 j의 세션키는 다음과 같은 형태로 생성된다.
  (1) i는 자신의 D_i와 j의 Q_j값을 통해 $e(D_i, Q_j)$를 계산한다.
  (2) 자신의 ID_i, x_i값과 상대방의 ID_j, PK_j를 통해 $e(H_2(ID_i, ID_j), PK_j^{x_i})$값을 구한다.
  (3) 위에서 계산한 두 값을 해시함수 H_3로 계산한 $H_3(e(D_i, Q_j) \| e(H_2(ID_i, ID_j), PK_j^{x_i}))$값이 i와 j의 세션키가 된다.

이 때, i와 j가 생성한 세션키가 동일하다는 것을 다음수식을 통해 확인할 수 있다.

$$\begin{aligned} K_{i,j} &= H_3(e(D_i, Q_j) \| e(H_2(ID_i, ID_j), PK_j^{x_i})) \\ &= H_3(e(Q_i, Q_j)^s \| e(H_2(ID_i, ID_j), g^{x_j x_i})) \\ &= H_3(e(Q_i, Q_j^s) \| e(H_2(ID_i, ID_j), (g^{x_i})^{x_j})) \\ &= H_3(e(D_j, Q_i) \| e(H_2(ID_i, ID_j), PK_i^{x_j})) = K_{j,i} \end{aligned}$$

4.4 안전성 증명

정리 1. 제안된 CL-NIKE 프로토콜은 공격자 유형 1로부터 DBDH 문제가 어렵다는 가정 하에 안전하다. E_1을 제안된 CL-NIKE 프로토콜을 공격하는 공격자 유형 1이라 하면 E_1의 이점은 다음과 같이 수렴한다.

$$Adv_S^{DBDH}(k) \geq Adv_{E_1}^{ICN_1}(k)$$

증명. E_1을 이용해 DBDH 문제를 효율적으로 풀 수 있는 챌린저 S를 설계 할 것이다. 챌린저 S는 DBDH 문제 $< g, g^a, g^b, g^c, G >$를 받고, E_1과 ICN[1] 게임을 수행한다.

◇ *Setup* : 챌린저 S는 겹선형 함수 e, 군 G_1, G_2, 그리고 생성자 $g \in G_1$를 생성하고 랜덤 오라클 $H_1 : \{0,1\}^* \rightarrow G_1$, $H_2 : \{0,1\}^* \rightarrow G_1$, $H_3 : \{0,1\}^* \rightarrow \{0,1\}^k$를 생성한다. 위에서 생성한 공개 파라미터들을 공격자에게 전송한다.

- H_1 : E_1은 S에게 언제든지 H_1-질의를 할 수 있다. S는 질의에 ID_i가 입력으로 들어오면 H_1^{list}에서 ID_i에 대한 리스트 항목이 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 임의의 $l_i \in Z_q^*$를 선택한 뒤 $H_1(ID_i) = Q_i = g^{l_i}$를 답한다. 이후 응답을 위해 H_1^{list}에 $< ID_i, Q_i, l_i >$를 기록한다. 공격목표로 지목한 ID_A, ID_B에 대해서는 각각 DBDH 문제로 받은 g^a, g^b값을 답해주고 $< ID_A, g^a, null >, < ID_B, g^b, null >$를 H_1^{list}에 기록한다.
- H_2 : E_1은 S에게 언제든지 H_2-질의를 할 수 있다. S는 질의에 (ID_i, ID_j)가 입력으로 들어오면 H_2^{list}에서 (ID_i, ID_j)에 대한 리스트 항목이 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 임의의 $r_i \in Z_q^*$를 선택한 뒤 $h_{i,j} = g^{r_i}$를 답한다. 이후 응답을 위해 $< (ID_i, ID_j), h_{i,j}, r_i >$를 H_2^{list}에 기록한다.
- H_3 : E_1은 S에게 언제든지 H_3-질의를 할 수 있다. S는 질의에 $e(D_i, Q_j) \| e(H_2(ID_i, ID_j), PK_j^{x_i})$가 입력으로 들어오면 H_3^{list}에 리스트 항목이 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 임의의 값 $H_I \in \{0,1\}^n$를 답한다. 이후 응답을 위해 $< e(D_i, Q_j) \| e(H_2(ID_i, ID_j), PK_j^{x_i}), H_I >$를 H_3^{list}에 기록한다.(I는 *Index*)
- *UserSet* : S는 *ReplacePublicKey*(ID_i, PK_i), *RevealPartialPrivareKey*(ID_i), *RevealSessionKey*(ID_i, ID_j) 질의를 통해 들어오는 ID들을 사용자 집합 리스트(U^{list})를 통해 관리한다. 만약 위의 질의들에서 나타난 ID가 U^{list}에 없다면 S는 *SetSecretValue*, *SetPublicKey*, *Partial−PrivateKeyExtract*, *SetPrivateKey* 알고리즘을 사용하여 ID에 대한 PK, D, x를 계산한 뒤 U^{list}에 등록한다. $< ID_i, PK_i, D_i, x_i > \in U^{list}$

공개 파라미터를 받은 공격자는
SetTarget(ID_A, ID_B)
질의를 통해 공격목표를 선택한다. 이 때, S는 A, B의 개인키, 공개키를 생성 하여 사용자 집합에 등록한다. $< ID_A, g^{x_a}, D_A, x_a > \in U^{list}$,
$< ID_B, g^{x_b}, D_B, x_b > \in U^{list}$

◇ *phase* 1 : E_1은 위 공격모델에서 정의한 질의들을 사용할 수 있다.

- *ReplacePublicKey*(ID_i, PK_i) : S는 본 질의를 받으면 다음과 같이 수행한다.
  $ID_i \in UserSet$인 경우 기존의 U^{list}에 있는 ID_i에 대응되는 PK 및 SK 대신 질의로 들어온 PK_i로 *UserSet*에 등록한다. $< ID_i, PK_i, D_i, null > \in U^{list}$
  $ID_i \notin UserSet$인 경우 S는 *PartialPrivateKey−Extract*알고리즘을 통해 D_i를 계산한 뒤 ID_i, PK_i, D_i를 등록한다. $< ID_i, PK_i, D_i, null > \in U^{list}$
- *RevealPartialPrivareKey*(ID_i) : S는 본 질의를 받으면 다음과 같이 수행한다.
  $ID_i \notin \{ID_A, ID_B\}$인 경우 DBDH 문제로 받은 g^c에 H_1^{list}로부터 ID_i에 해당하는 l_i값을 찾아 지수연산한 $(g^c)^{l_i} (= g^{l_i c} = Q_i^c = D_i)$ 값을 답한다.
  $ID_i \in \{ID_A, ID_B\}$인 경우 제한된 질의로 S는 게임을 종료한다.
- *RevealSessionKey*(ID_i, ID_j) : S는 본 질의를 받으면 다음과 같이 수행한다.
  $(ID_i, ID_j) \neq (ID_A, ID_B)$인 경우 $ID_i \notin \{ID_A, ID_B\}$의 *PartialPrivateKey*를 구한 다음 $e(D_i, Q_i)$를 계산한다. 다음으로 H_2^{list}에 기록된 (ID_i, ID_j)에 해당하는 $r_{i,j}$값을 사용해 $e(PK_i, PK_j)^{r_{i,j}}$

$(=e(g^{r_{i,j}},PK_j)^{x_i}=e(H_3(ID_i,ID_j),PK_j^{x_i}))$값을 계산한다. 계산된 값을 H_3오라클의 입력으로 넣어 $K_{i,j}=H_3(e(D_i,Q_j)\|e(H_2(ID_i,ID_j),PK_j^{x_i}))$를 계산하여 답해준다.$(ID_i,ID_j)=(ID_A,ID_B)$인 경우는 제한된 질의로 S는 게임을 종료한다.

◇ *challenge* : E_1은 $Test(ID_A,ID_B)$질의를 통해 챌린지(challenge) 단계로 진입한다. $Test(ID_A,ID_B)$ 질의를 받은 S는 $H_3(G\|e(H_2(ID_A,ID_B),PK_B^{x_A}))$를 계산하여 공격자에게 답해준다. E_1이 $guess=0$라 답했을 때, 즉 $H_3(G\|e(H_2(ID_A,ID_B),PK_B^{x_A}))$를 $K_{A,B}$로 판단한 경우에 S는 DBDH 문제의 답으로 $G=e(g,g)^{abc}$라 답한다. E_1이 $guess=1$라 답했을 때, 즉 $H_3(G\|e(H_2(ID_A,ID_B),PK_B^{x_A}))$를 $K_{A,B}$가 아닌 랜덤값으로 판단한 경우에 S는 DBDH 문제의 답으로 $G\neq e(g,g)^{abc}$라 답한다.

공격목표인 A,B의 Q_A,Q_B에 각각 DBDH 문제로 들어온 g^a,g^b값을 넣었고, g^c의 지수인 c값이 KGC의 마스터키로 사용되었다. 그러므로 E_1이 $e(D_A,Q_B)$를 계산했다면 $e(g^{ac},g^b)=e(g,g)^{abc}$를 계산한 것이다. 결과적으로 E_1이 본 게임에서 이긴다면 S는 E_1를 이용해 DBDH 문제를 풀 수 있는 것이다. 다시 말해 DBDH 문제를 푸는 S의 이점이 본 게임에서 E_1의 이점보다 크거나 같다.

$$Adv_S^{DBDH}(k) \geq Adv_{E_1}^{ICN_1}(k) = \left|\Pr[guess=b]-\frac{1}{2}\right|$$

E_1이 본 모델에서 상당한 이점으로 게임을 이기는 알고리즘이라면 "상당한 이점으로 DBDH 문제를 풀 수 있는 알고리즘은 존재하지 않는다." 라는 DBDH 가정에 모순이 되므로 "상당한 이점을 갖는 E_1은 존재하지 않는다." 라 할 수 있다. □

정리 2. 제안된 CL-NIKE 프로토콜은 공격자 유형 2로부터 DBDH 문제가 어렵다는 가정 하에 안전하다. E_2를 제안된 CL-NIKE 프로토콜을 공격하는 공격자 유형 2라 하면 E_2의 이점은 다음과 같이 수렴한다.

$$Adv_S^{DBDH}(k) \geq Adv_{E_2}^{ICN_2}(k)$$

증명. E_2를 이용해 DBDH 문제를 효율적으로 풀 수 있는 챌린저 S를 설계 할 것이다. 챌린저 S는 DBDH 문제 $<g,g^a,g^b,g^c,G>$를 받고 E_2와 ICN^2 게임을 수행한다.

◇ *Setup* : 챌린저 S는 겹선형 함수 e와 군G_1,G_2, 그리고 생성자 $g\in G_1$를 생성하고 랜덤 오라클 $H_1:\{0,1\}^*\rightarrow G_1$, $H_2:\{0,1\}^*\rightarrow G_1$, $H_3:\{0,1\}^*\rightarrow\{0,1\}^k$를 생성한다. 마지막으로 임의의 마스터키 $s\in Z_q^*$를 생성한다. 위에서 생성한 공개 파라미터들을 E_2에게 전송한다.

- H_1 : E_2는 S에게 언제든지 H_1-질의를 할 수 있다. S는 질의에 ID_i가 입력으로 들어오면 H_1^{list}에서 ID_i에 대한 리스트 항목이 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 임의의 $l_i\in Z_q^*$를 선택한 뒤 $H_1(ID_i)=Q_i=g^{l_i}$를 답한다. 이후 응답을 위해 $<ID_i,Q_i,l_i>$를 H_1^{list}에 기록한다.
- H_2 : E_2는 S에게 언제든지 H_2-질의를 할 수 있다. S는 질의에 (ID_i,ID_j)가 입력으로 들어오면 H_2^{list}에서 (ID_i,ID_j)에 대한 리스트 항목이 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 임의의 $r_i\in Z_q^*$를 선택한 뒤 $h_{i,j}=g^{r_i}$를 답한다. 이후 응답을 위해 H_2^{list}에$<(ID_i,ID_j),h_{i,j},r_i>$를 기록한다. 공격목표로 지목받은 (ID_A,ID_B)에 대해서는 DBDH 문제로 받은 g^c를 답해준 뒤 $<(ID_A,ID_B),g^c,null>$를 H_2^{list}에 기록한다.
- H_3 : 정리 1.에서의 H_3와 동일하다.
- *UserSet* : S는 $RevealSecretValue(ID_i)$, $RevealSessionKey(ID_i,ID_j)$질의에 들어오는 ID들을 *UserSet* 리스트를 통해 관리한다. $<ID_i,PK_i,D_i,x_i>\in U^{list}$

  만약 위의 질의들에서 나타난 ID가 U^{list}에 없다면 S는 $SetSecretValue$, $SetPublicKey$,

$Partial-Private\,KeyExtract$, $SetPrivateKey$ 알고리즘을 사용하여 ID에 대한 PK,D,x를 계산한 뒤 U^{list}에 등록한다.

공개 파라미터를 받은 E_2는 $SetTarget(ID_A,ID_B)$를 통해 공격목표를 선택한다. 이 때, S는 A,B의 부분 개인키를 각각 계산하고, DBDH 문제로 받은 g^a,g^b값을 각각 A, B의 공개키로 다음과 같이 $UserSet$에 등록한다. $< ID_A,g^a,D_A,null >, < ID_B,g^b,D_B,null > \in U^{list}$

◇ $phase1$: E_2는 위 공격모델에서 정의한 질의들을 사용할 수 있다.

- $RevealMasterKey()$: 본 질의를 받으면 S는 $Setup$단계에서 생성한 마스터키 s를 답해준다.
- $RevealSecretValue(ID_i)$: S는 본 질의를 받으면 다음과 같이 수행한다.
  $ID_i \notin \{ID_A,ID_B\}$이고 $ID_i \in UserSet$인 경우 기존의 U^{list}에 있는 ID_i에 대응되는 x_i를 답해준다.
  $ID_i \notin \{ID_A,ID_B\}$이고 $ID_i \notin UserSet$인 경우 S는 $SetSecretValue$, $SetPublicKey$, $Partial-PrivateKeyExtract$, $SetPrivateKey$ 알고리즘을 사용하여 ID_i에 대한 PK_i,D_i,x_i를 계산하여 x_i값을 답해준 뒤 U^{list}에 등록한다.
  $< ID_i,PK_i,D_i,x_i > \in U^{list}$
  $ID_i \in \{ID_A,ID_B\}$인 경우 제한된 질의로 S는 게임을 종료한다.
- $RevealSessionKey(ID_i,ID_j)$: 정리 1에서의 $RevealSessionKey(ID_i,ID_j)$와 동일하다.

◇ $challenge$: E_2는 $Test(ID_A,ID_B)$질의를 통해 챌린지(challenge) 단계로 진입한다. $Test(ID_A,ID_B)$질의를 받은 S는 $H_3(e(D_A,Q_B)\|G))$를 계산하여 E_2에게 답해준다. E_2가 $guess=0$이라 답했을 때, 즉 $H_3(e(D_A,Q_B)\|G))$를 $K_{A,B}$로 판단한 경우에 S는 DBDH 문제의 답으로 $G=e(g,g)^{abc}$라 답한다. E_2가 $guess=1$이라 답했을 때, 즉 $H_3(e(D_A,Q_B)\|G))$를 $K_{A,B}$가 아닌 랜던값으로 판단한 경우에 S는 DBDH 문제의 답으로 $G \neq e(g,g)^{abc}$라 답한다.

공격목표인 A,B의 공개키로 각각 DBDH 문제로 들어온 g^a,g^b값을 넣었고, $H_2(ID_A,ID_B)$에는 g^c 값을 넣었다. 그러므로 E_2가 $e(H_2(ID_A,ID_B),PK_B^{r_A})$를 계산했다면 $e(g^c,g^{br_A})=e(g,g)^{abc}$를 계산한 것이다. 결과적으로 E_2가 본 게임에서 이긴다면 S는 E_2를 이용해 DBDH 문제를 풀 수 있는 것이다. 정리 1.에서처럼 E_2의 이점은 다음과 같다.

$$Adv_S^{DBDH}(k) \geq Adv_{E_2}^{ICN_2}(k) = \left|\Pr[guess=b]-\frac{1}{2}\right|$$

□

정리 1, 정리 2에 의해 제안된 CL-NIKE 프로토콜은 제안된 CL-NIKE안전성 모델에서 안전하다.

V. CL-NIKE-fs 프로토콜

본 장에서는 CL-NIKE-fs에서 사용하는 알고리즘을 간략히 설명하고 안전성 모델을 제안한다. 다음으로 프로토콜을 소개하고 제안한 안전성 모델에서의 안전성 증명을 보인다. CL-NIKE에 전방향 안전성을 제공하기 위해 개인키를 시간주기에 맞춰 업데이트하는 방식을 사용한다.

5.1 알고리즘

CL-NIKE-fs 프로토콜은 다음과 같은 7개의 알고리즘으로 이루어진다.

- 설정($Setup$) : KGC에서 보안 상수와 총 시간주기 값을 입력으로 받아 시스템 내에 사용되는 파라미터를 생성하는 알고리즘이다.
- 부분 개인키 추출($PartialPrivateKeyExtract$) : KGC에서 사용자의 ID를 입력으로 하여 ID에 해당하는 $PartialPrivateKey$를 생성하는 알고리즘이다.
- 초기 비밀값 설정($SetInitialSecretValue$) : 각 사용자의 고유한 초기 비밀값을 생성하는 알고리즘이다.
- 공개키 설정($SetPublicKey$) : 각 사용자가 자신이 생성한 초기 비밀값과 공개 파라미터를 이용하여 공개키를 생성하는 알고리즘이다.

- 비밀값 업데이트($SecretValueUpdate$) : 사용자의 현재 주기의 비밀값을 다음 주기로 업데이트하는 알고리즘이다.
- 개인키 설정($SetPrivateKey$) : t번째 주기에 해당하는 비밀값과 사용자가 KGC로부터 받은 $PartialPrivateKey$와 함께 사용자의 개인키를 생성하는 알고리즘이다.
- 세션키 생성($SessionKeyGen$) : 두 사용자가 현재시간 주기에 해당하는 세션키를 생성하는 알고리즘으로 한 사용자의 개인키와 상대방의 공개키를 입력으로 생성한다. 상대방 역시 자신의 개인키와 상대방의 공개키를 입력으로 생성하는데 이 때, 두 사용자가 생성한 세션키는 동일해야 한다.

5.2 안전성 모델

본 절에서는 CL-NIKE-fs 프로토콜의 안전성 모델을 제시한다. 전방향 안전성을 추가한 안전성 모델은 4.2에서 제안한 모델의 큰 틀을 따른다. IND-CL-NIKE-fs의 게임은 ICN-fs로 표기한다.

5.1.1 공격자 유형 1

공격자 유형 1의 경우 KGC의 마스터키에 대한 정보를 알 수 없지만 사용자들의 공개키를 교체할 수 있는 능력이 주어진다. ICN^1-fs 게임은 다음과 같이 정의된다.

- $Setup(k)$: 챌린저 S는 보안 상수 k를 입력으로 공개파라미터를 생성하여 공개파라미터는 공격자에게 전송하고 마스터키는 보관한다.
- $SetTarget(ID_A,ID_B,t^*)$: 공격자는 공격할 목표와 주기(t^*)를 선택하여 S에게 전송한다.
- $ReplacePublicKey(ID_i,PK_i)$: 4.2절 공격자 유형 1에서 $ReplacePublicKey(ID_i,PK_i)$와 동일하다.
- $RevealSecretValue(ID_i,t)$: ID_i에 해당하는 t번째 주기의 비밀값을 얻어볼 수 있는 질의로 S는 ID_i에 해당하는 t번째 주기의 비밀값을 공격자에게 전송한다.
- $RevealPartialPrivareKey(ID_i)$: 4.2절 공격자 유형 1에서 $RevealPartialPrivareKey(ID_i)$와 동일하다.
- $RevealSessionKey(ID_i,ID_j,t)$: t번째 주기에 해당하는 사용자 ID_i,ID_j사이의 세션키를 얻어볼 수 있는 질의로 S는 ID_i,ID_j사이에서 생성되는 t번째 주기의 세션키를 공격자에게 전송한다. 공격목표(ID_A,ID_B,t^*)를 제외한 모든 경우에 질의에 대한 답을 전송해야 한다.
- $Test(ID_A,ID_B,t^*)$: 공격자는 본 질의를 통해 챌린지(challenge) 단계로 진입한다. S는 본 질의를 받으면 $b\in\{0,1\}$를 선택하는데 $b=0$인 경우 공격목표인 A와 B의 t^*번째 주기의 세션키를 전송하고, $b=1$인 경우에는 세션키 공간에서 랜덤값을 선택하여 전송한다. 공격자는 S로부터 받은 값을 세션키라고 판단하면 $guess=0$을 답하고, 랜덤값이라 판단하면 $guess=1$을 답한다. 이 때, 공격자 유형 1(E_1)의 이점을 다음과 같이 정의한다.

$$Adv_{E_1}^{ICN^1-fs}(k)=\left|\Pr[guess=b]-\frac{1}{2}\right|$$

5.1.2 공격자 유형 2

공격자 유형 2의 경우 KGC의 마스터키를 아는 대신 사용자들의 공개키를 교체할 수 있는 능력을 제한한다. ICN^2-fs 게임은 다음과 같이 정의된다.

- $Setup(k)$: 공격자 유형 1에서와 동일하다.
- $SetTarget(ID_A,ID_B,t^*)$: 공격자 유형 1에서와 동일하다.
- $RevealMasterKey()$: 4.2절의 공격자 유형 2에서 $RevealMasterKey()$와 동일하다.
- $RevealSecretValue(ID_i,t)$: ID_i에 해당하는 t번째 주기의 비밀값을 얻어볼 수 있는 질의로 S는 ID_i에 해당하는 t번째 주기의 비밀값을 공격자에게 전송한다. 공격자 유형 2의 경우 공격목표로 선택한 ID_A,ID_B에 대해선 t^*를 포함한 이전의 주기에 대한 질의를 제한한다.
- $RevealSessionKey(ID_i,ID_j)$: 공격자 유형 1에서와 동일하다.
- $Test(ID_A,ID_B,t^*)$: 공격자 유형 1에서와 동일하다. 공격자 유형 2(E_2)의 이점을 다음과 같이 정의한다.

$$Adv_{E_2}^{ICN^2-fs}(k)=\left|\Pr[guess=b]-\frac{1}{2}\right|$$

5.3 제안하는 프로토콜

- 설정(*Setup*) : KGC에서 보안 상수 k를 입력으로 하여 다음과 같은 파라미터를 생성한다.
  (1) 위수가 소수 q인 군 $G_1,...,G_{n+3}$와 생성자 $g\in G_1$를 생성 하고 다선형 함수 e, 임의의 공개 파라미터 $(U_0,...,U_n)$, $(V_1,...,V_n)$를 선택한다.
  (2) 해시함수 $H_1:\{0,1\}^*\rightarrow G_1, H_2:\{0,1\}^*\rightarrow G_1$, $H_3:\{0,1\}^*\rightarrow G_1$, $H_4:\{0,1\}^*\rightarrow\{0,1\}^k$들을 선택한다.
  (3) 임의의 마스터키 MK $(=s\in Z_q^*)$를 선택한다. 마스터키를 제외한 모든 값과 함수들은 공개 파라미터로 사용자들에게 공개된다.
- 부분키 추출(*PartialPrivateKeyExtract*) : KGC는 사용자(i)의 고유한 ID_i와 마스터키s로 사용자의 부분 개인키 $D_i=Q_1^s(Q_i=H_1(ID_i))$를 생성한다. 생성한 D_i를 안전한 경로를 통해 사용자에게 전송한다.
- 초기 비밀값 설정(*SetInitalSecretValue*) : 사용자는 임의의 초기 비밀값(*InitalSecret-Value*) $s_i\in Z_q^*$를 생성한다.
- 공개키 설정(*SetPublicKey*) : 사용자는 초기 비밀값(s_i)과 공개 파라미터 g로 공개키 $PK_i=g^{s_i}$를 생성한다.

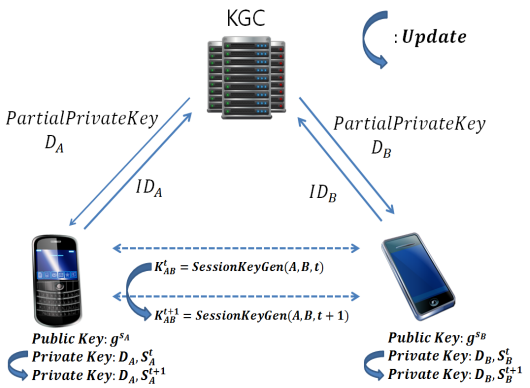

Fig. 2. CL-NIKE-fs

- 개인키 설정(*SetPrivateKey*) :
  (1) 사용자는 초기 비밀값(s_i)과 공개 파라미터 (U_0)로 $S_i^0=u_0^{s_i}$ $(u_0=H_3(U_0)\in G_1)$를 생성한다.
  (2) PK_i, S_i^0 계산을 완료한 뒤에는 초기 비밀값 (s_i)을 안전하게 삭제한다.
  (3) S_i^0를 비밀값 업데이트 단계를 통해 현재주기인 t번째 주기까지 업데이트하여 S_i^t를 얻는다.
  (4) 사용자는 부분 비밀키 D_i와 t번째 주기까지 업데이트된 S_i^t로 사용자 개인키 $<D_i,S_i^t>$를 설정한다.
- 비밀값 업데이트(*SecretValueUpdate*) : $l-1$번째 주기의 개인키 S_i^{l-1}를 l번째 주기로 업데이트 하는 과정은 다음과 같다.
  (1) 사용자는 공개 파라미터 U_l을 해시함수 연산하여 $u_l(=H_3(U_l)\in G_1)$를 계산한다.
  (2) $l-1$번째 주기의 비밀값(S_i^{l-1})와 u_l를 다선형 함수 연산하여 l번째 주기의 비밀값(S_i^l)을 계산한다. 즉, $S_i^l=e(S_i^{l-1},u_l)$가 된다.
  (3) S_i^l 업데이트완료 후 이전 비밀값(S_i^l)은 안전하게 삭제한다.
- 세션키 생성(*SessionKeyGen*) : 사용자 i는 j와의 t번째 주기의 세션키를 생성하기 위해 다음과 같은 연산을 수행한다.
  (1) 자신의 부분 개인키 D_i와 상대방의 Q_j값을 통해 $e(D_i,Q_j)$값을 계산한다.
  (2) 자신의 ID_i, S_i^t, 상대방의 ID_j, PK_j 그리고 $v_{t+1},...,v_n(v_k=H_3(V_k))$를 입력으로 다선형 함수 $e(H_2(ID_i,ID_j),S_i^t,v_{t+1},...,v_n,PK_j)$값을 계산한다.
  (3) 계산한 두 값을 해시함수 H_4로 계산한 값이 i,j의 t번째 주기의 세션키가 된다. 즉, $K_{i,j}=H_4(e(D_i,Q_j)\|e(H_2(ID_i,ID_j),S_i^t,v_{t+1},...,v_n,PK_j))$

이 때, i와 j가 t번째 주기에 생성한 세션키의 동일성을 확인하기 위해 다선형 함수로 계산된 부분이 동일하다는 것을 다음 수식을 통해 확인할 수 있다.

$$e(H_2(ID_i,ID_j),S_i^t,v_{t+1},...,v_n,PK_j))$$
$$= e(H_2(ID_i,ID_j),u_0,...u_t,v_{t+1},...,v_n,PK_j)^{s_i})$$
$$= e(H_2(ID_i,ID_j),u_0,...u_t,v_{t+1},...,v_n,PK_i)^{s_j})$$
$$= e(H_2(ID_i,ID_j),S_j^t,v_{t+1},...,v_n,PK_i))$$

5.4 안전성 증명

정리 3. 제안된 CL-NIKE-fs 프로토콜은 공격자 유형 1로부터 DBDH 문제가 어렵다는 가정 하에 안전하다. E_1을 제안된 CL-NIKE-fs 프로토콜을 공격하는 공격자 유형 1이라 하면 E_1의 이점은 다음과 같이 수렴한다.

$$Adv_S^{DBDH}(k) \geq Adv_{E_1}^{ICN-fs_1}(k)$$

증명. CL-NIKE-fs에서도 공격자 유형 1에 대해서는 정리 1.과 동일하게 공격목표인 A, B의 부분 개인키와 KGC의 마스터키 값에 DBDH 문제로 들어온 값을 넣고 관여시킨다. 증명 방식이 정리 1.과 동일하므로 생략한다.

정리 4. 제안된 CL-NIKE-fs 프로토콜은 공격자 유형 2로부터 n-MDDH 문제가 어렵다는 가정 하에 안전하다. E_2를 제안된 CL-NIKE-fs 프로토콜을 공격하는 공격자 유형 2라 하면 E_2의 이점은 다음과 같이 수렴한다.

$$Adv_S^{n-MDDH}(k) \geq Adv_{E_2}^{ICN-fs_2}(k)$$

증명. E_2를 이용해 n-MDDH 문제를 효율적으로 풀 수 있는 챌린저 S를 설계할 것이다. 챌린저 S는 n-MDDH 문제$< g, g^{x_1},...,g^{x_{n+4}}, G >$를 받고 E_2와 $ICN-fs_2$ 게임을 수행한다.

◇ *Setup* : 챌린저 S는 다선형 함수 e, 공개변수 $(U_0,...,U_n),(V_1,...,V_n)$, 군 $G_1,...,G_{n+3}$, 생성자 g, 랜덤 오라클 $H_1:\{0,1\}^* \rightarrow G_1$, $H_2:\{0,1\}^* \rightarrow G_1$, $H_3:\{0,1\}^* \rightarrow G_1$, $H_4:\{0,1\}^* \rightarrow \{0,1\}^k$를 생성한다. 마지막으로 임의의 마스터키 $s \in Z_q^*$를 생성한다. 위에서 생성한 공개 파라미터들을 E_2에게 전송한다.

- H_1 : 4.4절의 공격자 유형 2에서 H_1과 동일하다.
- H_2 : E_2는 S에게 언제든지 H_2-질의를 할 수 있다. S는 질의에 (ID_i,ID_j)가 입력으로 들어오면 H_2^{list}에서 (ID_i,ID_j)에 대한 리스트 항목이 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 임의의 $r_i \in Z_q^*$를 선택한 뒤 $h_{i,j}=g^{r_i}$를 답한다. 이후 응답을 위해 $<(ID_i,ID_j),h_{i,j},r_i>$를 H_2^{list}에 기록한다. 공격목표로 지목받은 (ID_A,ID_B)에 대해서는 n-MDDH 문제로 받은 $g^{x_{n+4}}$를 답해준 뒤 $<(ID_A,ID_B),g^{x_{n+4}},null>$를 H_2^{list}에 기록한다.
- H_3 : E_1은 S에게 언제든지 H_3-질의를 할 수 있다. S는 들어온 입력이 H_3^{list}에 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 들어오는 입력에 따라 다음과 같은 일을 수행한다.
  임의의 $t^*(1 \leqq t^* \leqq n)$를 선택한 다음 입력으로 $U_0,U_1,...U_{t^*}$값이 들어오면 각각 n-MDDH 문제로 받은 $g^{x_{n+3}},g^{x_1},...,g^{x_{t^*}}$를 답하고 H_3^{list}에 기록한다. (예 $< U_0,g^{x_{n+3}},null > \in H_3^{list}$)
  입력으로 $U_{t^*+1},...,U_n$값이 들어오면 각각 임의의 $e_{t^*+1},...,e_n$를 선택하여 $g^{e_{t^*+1}},...,g^{e_n}$를 답하고 H_3^{list}에 기록한다.(예 $< U_{t^*+1},g^{e_{t^*+1}},e_{t^*+1} > \in H_3^{list}$)
  입력으로 $V_1,...,V_{t^*}$값이 들어오면 각각 임의의 $e_1,...,e_{t^*}$를 선택하여 $g^{e_1},...,g^{e_{t^*}}$를 답하고 H_3^{list}에 기록한다. (예 $< V_1,g^{e_1},e_1 > \in H_3^{list}$)
  입력으로 $V_{t^*+1},...,V_n$값이 들어오면 각각 n-MDDH 문제로 받은 $g^{x_{t^*+1}},...,g^{x_n}$를 답하고 H_3^{list}에 기록한다.(예 $< V_{t^*+1},g^{x_{t^*+1}},null > \in H_3^{list}$)
  공개 파라미터로 지정되지 않은 다른 값들이 입력으로 들어왔을 경우 임의의 값을 답하고 이후 응답을 위해 H_3^{list}에 기록해놓는다.
- H_4 : E_1은 S에게 언제든지 H_4-질의를 할 수 있다. S는 질의에 $e(D_i,Q_j) \| e(H_2(ID_i,ID_j),S_i^{x_i},v_{t+1},...,v_n,PK_j)$가 입력으로 들어오면 H_4^{list}에 리스트 항목이 있는지 찾아본다. 있다면 기록된 값을 답하고, 없다면 임의의 값 $H_I \in \{0,1\}^n (I$는 $Index)$를 답한다. 이후 응답을 위해 H_4^{list}에 기록한다.
- *UserSet* : S는 $RevealSecretValue(ID_i,t)$,

$RevealSessionKey(ID_i,ID_j,t)$ 질의가 들어오는 ID들을 사용자 집합 리스트(U^{list})를 통해 관리한다.
$< ID_i,PK_i,D_i,s_i > \in U^{list}$ (s_i는 초기 비밀값)
만약 위의 질의들에서 나타난 ID가 U^{list}에 없다면 S는 $SetInitialSecretValue$, $SetPublicKey$, $PartialPrivateKeyExtract$, $SetPrivateKey$ 알고리즘을 사용하여 ID에 대한 PK_i,D_i,s_i를 계산한 뒤 U^{list}에 등록한다.

공개 파라미터를 받은 E_2는 $SetTarget(ID_A,ID_B,t)$를 통해 공격목표를 선택한다. 이 때, $t \neq t^*$인 경우 S는 게임을 종료한다. $t=t^*$인 경우 S는 A,B의 부분 개인키를 각각 계산하고 공개키로는 n-MDDH 문제로 받은 $g^{x_{n+1}},g^{x_{n+2}}$값을 넣어 다음과 같이 사용자 집합에 등록한다.

$< ID_A,g^{x_{n+1}},D_A,null > \in U^{list}$,
$< ID_B,g^{x_{n+2}},D_B,null > \in U^{list}$

◇ $phase1$: E_2는 위 공격모델에서 정의한 질의들을 사용할 수 있다.

- $RevealMasterKey()$: 4.4절의 공격자 유형 2에서 $RevealMasterKey()$와 동일하다.
- $RevealSecretValue(ID_i,t)$: S는 본 질의를 받으면 다음과 같이 수행한다.
  $ID_i \in \{ID_A,ID_B\}$이고 $t \leq t^*$인 경우 제한된 질의로 S는 게임을 종료한다.
  $ID_i \in \{ID_A,ID_B\}$이고 $t > t^*$인 경우
  $e(u_o,...,u_{t-1},g^{x_{n+1}} \text{or } g^{x_{n+2}})^{e_t}$
  $e(u_o,...,u_{t-1},g^{x_{n+1}} \text{or } g^{x_{n+2}})^{e_t}$
  $(=e(u_o,...,u_{t-1},g^{e_t})^{x_{n+1} \text{or } x_{n+2}}$
  $=e(u_o,...,u_{t-1},u_t)^{x_{n+1} \text{or } x_{n+2}} = S_A^t \text{ or } S_B^t)$를 답한다.
  $ID_i \notin \{ID_A,ID_B\}$인 경우 U^{list}에서 ID_i에 해당하는 s_i값을 찾아 $e(u_0,...,u_n)^{s_i}$을 계산하여 답한다.
- $RevealSessionKey(ID_i,ID_j,t)$: S는 KGC의 마스터키를 알고 있으므로 모든 입력에 대해 $e(D_i,Q_j)$를 구할 수 있다.
  $(ID_i,ID_j)=(ID_A,ID_B)$이고 $t=t^*$인 경우 제한된 질의로 S는 게임을 종료한다.
  $(ID_i,ID_j)=(ID_A,ID_B)$이고 $t<t^*$인 경우
  $e(H_2(A,B),u_o,...,u_t,g^{x_{n+1}},v_{t+2},...,v_ng^{x_{n+2}})^{e_{t+1}}$
  $(=e(H_2(A,B),u_o,...,u_t,v_{t+1},v_{t+2},...,v_ng^{x_{n+2}})^{x_{n+1}}$
  $=e(H_2(A,B),S_A^t,v_{t+1},v_{t+2},...,v_n,PK_B))$값을 계산 한다.
  $(ID_i,ID_j)=(ID_A,ID_B)$이고 $t>t^*$인 경우
  $RevealSecretValue(ID_i,t)$를 통해 ID_A의 S_A^t값을 계산한 뒤 $e(H_2(A,B),S_A^t,v_{t+1},v_{t+2},...,v_n,PK_B)$값을 계산한다.
  $(ID_i,ID_j) \neq (ID_A,ID_B)$인 경우
  $e(PK_i,u_o,...,u_t,v_{t+1},...,v_nPK_j)^{r_{i,j}}$
  $(=e(g^{s_i},u_o,...,u_t,v_{t+1},...,v_nPK_j)^{r_{i,j}}$
  $=e(g^{r_{i,j}},u_o,...,u_t,v_{t+1},...,v_nPK_j)^{s_i}$
  $=e(H_2(ID_i,ID_j),S_i^t,v_{t+1},...,v_nPK_j))$값을 계산한다.
  (ID_A,ID_B,t^*)가 입력으로 들어오는 경우를 제외한 모든 경우에 $e(D_i,Q_j)$값과 $e(H_2(ID_i,ID_j),S_i^t,v_{t+1},...,v_nPK_j))$값 계산이 가능하다. 계산된 두 값을 입력으로 하여 H_4연산을 통해 얻은 $K_{i,j}^t$를 답한다.

◇ $challenge$: E_2는 $Test(ID_A,ID_B,t^*)$질의를 통해 챌린지(challenge) 단계로 진입한다. $Test(ID_A,ID_B,t^*)$질의를 받은 S는 $H_4(e(D_A,Q_B)\|G))$를 계산하여 E_2에게 답해준다. E_2가 $guess=0$이라 답했을 때, 즉, $H_4(e(D_A,Q_B)\|G))$를 $K_{A,B}^{t^*}$로 판단한 경우에 S는 n-MDDH 문제의 답으로 $G=e(g,...,g)^{x_1...x_{n+4}}$라 답한다. E_2가 $guess=1$이라 답했을 때, 즉, $H_4(e(D_A,Q_B)\|G))$를 $K_{A,B}^{t^*}$가 아닌 랜덤값으로 판단한 경우에 S는 n-MDDH 문제의 답으로 $G \neq e(g,...,g)^{x_1...x_{n+4}}$라 답한다.

공격목표인 A,B의 공개키로 각각 n-MDDH 문제로 들어온 $g^{x_{n+1}},g^{x_{n+2}}$값을 넣었고, $H_2(ID_A,ID_B)$에는 $g^{x_{n+4}}$ 값을 넣었다. 그리고 공개변수의 해시값 $u_0,u_1,...u_{t^*},v_{t^*+1},...,v_n$에는 각각$g^{x_{n+3}},g^{x_1},...,g^{x_n}$을 넣

Table 1. Analysis of efficiency and security m: multilinear map, p: pairing, e: exponent, E: ECC multiplication, h: hash function

		Sang et al[4]	(1)	(2)
efficiency	PartialPrivateKeyExtract(KGC)	1E+1h	1e+1h	1e+1h
	SetPrivate/PublicKey(user)	2E	1e	1e
	SecretValueUpdate(user)	-	-	1p+1h
	PublicKeyVerfying(user)	4p+1h	-	-
	SessionKeyGen(user)	1p+2E+1h	2p+1e+2h	1m+1p+2h
security proof		X	O	O
forward secrecy		X	X	O

었다. 그러므로 E_1이 $e(H_2(ID_A,ID_B),S_A^{t^*},v_{t^*+1},...,v_nPK_B)$를 계산했다면 $e(g^{x_{n+4}},g^{x_{n+3}},g^{x_1},...,g^{x_n},g^{x_{n+2}})^{x_{n+1}}(=e(g,...,g)^{x_1\cdots x_{n+4}})$를 계산한 것이다. t의 범위는 n이므로 E_2가 $t=t^*$로 선택할 확률은 $1/n$이다. 그러므로 본 게임에서 E_2의 이점을 $Adv_{E_2}^{ICN-fs_2}(k)$라 하면 n-MDDH 문제를 푸는 S의 이점은 다음과 같다.

$$Adv_S^{DBDH}(k) \geq \frac{1}{n} \cdot Adv_{E_2}^{ICN-fs_2}(k)$$

□

정리 3, 정리 4에 의해 제안된 CL-NIKE-fs 프로토콜은 제안된 CL-NIKE-fs 안전성 모델에서 안전하다.

5.5 프로토콜 비교

본 절에서는 Sang 등의 연구에서 제안한 프로토콜 [4]와 논문에서 제안한 CL-NIKE 프로토콜(1) 및 CL-NIKE-fs 프로토콜(2)를 효율성 및 안전성 측면에서 비교분석한다. Table 1. 참조

[4]에서는 상대방의 공개키가 KGC로부터 정당하게 생성되었음을 검증하는 공개키 검증단계가 있다. [4]에서 키 교환 시 1번의 다선형 함수연산으로 (1)보다 효율적이지만, 검증 단계 시 4번의 다선형 함수 연산이 필요하기 때문에 전체적으로 필요한 연산량이 (1)에서 더 적다. 또한 (1)은 정의된 안전성 모델에서 프로토콜 자체의 취약점이 존재하지 않다는 것을 증명했기 때문에 [4]보다 더 안전한 프로토콜이다.

(2)는 키 교환 시 다소 무거운 다선형 함수를 사용하는데, 이는 (1)에 전방향 안전성을 제공하기 위함이다. 키 교환을 위한 별도의 통신 없이 전방향 안전성을 갖는 프로토콜을 설계하기 위해선 개인키를 일방향 함수를 통해 업데이트해야 한다. 이 때 공개키와 개인키의 연관성이 사라지지 않도록 다선형 함수를 사용하여 업데이트 한다. 다선형 함수를 사용하지 않고 전방향 안전성을 제공하는 NIKE 프로토콜을 설계하는 것은 향후 도전과제이다.

VI. 결 론

본 논문에서는 인증서를 사용하지 않는 비대화형 키 교환 프로토콜의 안전성 모델을 제안하고 프로토콜을 소개한 뒤 수학적 난제에 기반을 둔 증명을 제시하였다. 본 프로토콜은 인증서를 사용하지 않는 공개키 암호시스템에서 키 교환을 위한 전송량을 최소화 하는 키 교환 프로토콜이다. 2012년, Sang 등의 연구[4]에서 처음으로 인증서를 사용하지 않는 비대화형 키 교환 프로토콜이 제안 되었지만 증명가능한 안전성을 보이기 위해 필요한 안전성 모델을 제시하지 않았다. 본 논문은 인증서를 사용하지 않는 비대화형 키 교환 프로토콜의 증명 가능한 안전성을 제공하기 위한 안전성 모델을 제시하였다. 또한 다선형 함수를 사용하여 전방향 안전성을 만족하는 인증서 기반이 아닌 비대화형 키 교환 프로토콜을 최초로 제안하고 안전성 모델 및 증명을 제시하였다.

References

[1] SS Al-Riyami and KG Paterson, "Certificateless public key cryptography," ASIACRYPT 2003. LNCS, Volume 2894, pp 452-473, 2003.

〔2〕 Colleen Swanson and David Jao, "A Study of Two-Party Certificateless Authenticated Key-Agreement Protocols," INDOCRYPT 2009. LNCS, Volume 5922, pp 57-71, 2009.

〔3〕 A. Shamir. "Identity-based cryptosystems and signature schemes," In Proc. Crypto '84, LNCS 196, pages 47-53, 1984.

〔4〕 Yongxuan Sang, Lili Zhang, Lin You and Zhongwen Li, "Two Non-interactive Key Agreement Protocols under Certificateless Scenarios," International Journal of Advancements in Computing Technology(IJACT), Volume4 Number6,April 2012.

〔5〕 R Steinwandt, AS Corona, "Identity-based non-interactive key distribution with forward security," Designs, Codes and Cryptography, Volume 64, Issue 1-2, pp 195-208, July 2012.

〔6〕 Daniel J. Bernstein, "Curve25519: New Diffie-Hellman Speed Records," Public Key Cryptography-PKC 2006 LNCS Volume 3958, pp 207-228, 2006.

〔7〕 David Cash, Eike Kiltz, and Victor Shoup, "The Twin Diffie-Hellman Problem and Applications" EUROCRYPT 2008. LNCS, vol. 4965, pp. 127 - 145, 2008.

〔8〕 David Pointcheval, Olivier Sanders, "Forward Secure Non-Interactive Key Exchange," Security and Cryptography for Networks, LNCS, Volume 8642, pp 21-39, 2014

〔9〕 W Diffie, ME Hellman, "New directions in cryptography," Information Theory, IEEE Transactions on, Volume22 Issue6, 644 - 654, 1976.

〔10〕 Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, Kenneth G. Paterson. "Non-Interactive Key Exchange," Public-Key Cryptography - PKC 2013 LNCS Volume7778, pp 254-271, 2013.

〈저 자 소 개〉

이 영 경 (Young Kyung Lee) 학생회원
2014년 2월: 고려대학교 수학과 학사 졸업
2014년 3월~현재: 고려대학교 정보보호대학원 석사과정
〈관심분야〉 암호프로토콜, 암호이론, 인증 및 키 교환

엄 지 은 (Ji Eun Eom) 학생회원
2010년 2월: 고려대학교 수학과 학사 졸업
2012년 2월: 고려대학교 정보보호대학원 석사 졸업
2013년 3월~현재: 고려대학교 정보보호대학원 박사과정
〈관심분야〉 암호프로토콜, 암호이론, 클라우드 컴퓨팅, 인증 및 키 교환

서 승 현 (Seung-Hyun Seo) 종신회원
2000년 2월: 이화여대 수학과 학사 졸업
2002년 2월: 이화여대 대학원 컴퓨터학과 석사 졸업
2006년 2월: 이화여대 대학원 컴퓨터학과 박사 졸업
2006년 12월~2010년 2월: 금융보안연구원 주임연구원
2010년 2월~2012년 2월: 한국인터넷진흥원 선임연구원
2012년 2월~2014년 5월: 미국 퍼듀대학교 사이버센터 박사후연구원
2014년 6월~2015년 2월: 고려대학교 정보보호대학원 BK21+ 사업단 연구교수
2015년 3월~ 현재: 고려대학교 세종캠퍼스 수학과 조교수
〈관심분야〉 암호프로토콜, 인증 및 키 교환 기술, 모바일 보안, IoT 보안

이 동 훈 (Dong Hoon Lee) 종신회원
1983년 8월: 고려대학교 경제학과 학사 졸업
1987년 12월: Oklahoma University 전산학과 석사 졸업
1992년 5월: Oklahoma University 전산학과 박사 졸업
1993년 3월~1997년 2월: 고려대학교 전산학과 조교수
1997년 3월~2001년 2월: 고려대학교 전산학과 부교수
2001년 3월~현재: 고려대학교 정보보호대학원 교수
2015년 3월~현재: 고려대학교 정보보호대학원 원장
〈관심분야〉 암호프로토콜, 암호이론, USN이론, 키 교환, 익명성 연구, PET 기술