情報保護學會論文誌
第 21 卷 第 5 號, 2011. 10

고정된 검사자를 고려한 메시지 동일성 검사 공개키 암호시스템*

이 영 민,[1†] 구 우 권,[1] 이 현 숙,[2] 이 동 훈[1‡]
[1]고려대학교 정보보호대학원, [2]삼성전자

Public Key Encryption with Equality Test with Designated Tester*

Young Min Lee,[1†] Woo Kwon Koo,[1] Hyun Sook Rhee,[2] Dong Hoon Lee[1‡]
[1]Graduate School of Information Security, Korea University, [2]Samsung Electronics

요 약

2004년 Boneh et. al. 은 송신자가 전송한 검색어에 대한 암호문과 수신자가 제공한 쿼리 생성에 사용된 검색어의 동일성을 서버가 검사할 수 있는 PEKS 스킴을 제안하였다. 이후 Yang et. al.은 서로 다른 공개키로 암호화된 암호문의 비교를 통해 암호화된 메시지의 동일성을 검사하는 기법인 Probabilistic Public Key Encryption with Equality Test(PEET)을 제안하였다. PEET 기술은 메시지 부분을 검색어로 대체해서 암호화할 경우 Keyword guessing attacks에 대한 안전성을 보장하지 못하고 검색가능 암호화 기법들에서 일반적으로 고려되고 있는 안전성의 기준인 IND-CPA 안전성을 제공하지 못한다. 본 논문에서는 Keyword guessing attacks에 안전하며 IND-CPA를 만족하는 고정된 검사자를 고려한 메시지 동일성 검사 공개키 암호시스템(public key encryption with equality test with designated tester, dPEET)을 제안한다.

ABSTRACT

In 2004, Boneh et.al. proposed a public key encryption with keyword search (PEKS) scheme which enables a server to test whether a keyword used in generating a ciphertext by a sender is identical to a keyword used in generating a query by a receiver or not. Yang et. al. proposed a probabilistic public key encryption with equality test (PEET) scheme which enables to test whether one message of ciphertext generated by one public key is identical to the other message generated by the other public key or not. If the message is replaced to a keyword, PEET is not secure against keyword guessing attacks and does not satisfy IND-CPA security which is generally considered in searchable encryption schemes. In this paper, we propose a public key encryption with equality test with designated tester (dPEET) which is secure against keyword guessing attacks and achieves IND-CPA security.

Keywords: Keyword search on encrypted data, Database security and privacy, Keyword guessing attack, Searchable encryption

접수일(2010년 10월 14일), 게재확정일(2011년 2월 14일)
* 이 논문은 2010년 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (한국연구재단-2010-0003388)
* 이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2011-0029831).
† 주저자, oueoue@nate.com
‡ 교신저자, donghlee@korea.ac.kr

I. 서 론

.컴퓨터와 인터넷, 스마트 폰 기술의 발전은 개인이 소유하고 있는 정보의 저장장소를 개인의 컴퓨터에서 누구나 접근이 용이한 외부저장 공간으로 확장시키고 있다. 외부 저장 공간에 저장된 개인정보들은 사용자의 요구에 따라 다양한 기기로 전송되어질 수 있으며, 전송된 정보들은 가공되어 다시 저장 되거나 다른 기기로 전송된다. 이렇듯 정보통신기기의 발전으로 개인이 접하는 정보의 양은 급격히 증가하고 있지만 정작 사용자의 대다수는 자신의 민감한 정보가 외부 저장 공간에서 어떻게 관리되고 있는지 알 수 없다. 외부 저장 공간에 보관중인 민감한 정보가 시스템 관리자에게 노출이 되는지 또는 저장 공간 외부로 정보가 누출되었는지 확인 할 수 없다. 최근 유명 백화점 등 25개 사이트의 개인정보 누출사건은 이러한 문제점을 보여주는 예라고 할 수 있다[14]. 수년전 정보 누출사건이 있었지만 사용자들은 그 내용을 알지 못해 명의도용 등 2차 3차 피해에 노출 되어 있었고 특히 개인정보가 암호화되지 않은 상태에서 유출돼 추가피해의 가능성은 더욱 높아졌다. 이러한 정보 누출 방지에 안전하도록 데이터를 저장하기 위해서는 개인정보를 사용자만이 아는 개인키로 암호화하여 저장하는 것을 생각할 수 있다. 하지만 데이터를 암호화하여 저장한다면 사용자가 원하는 데이터를 얻기 위해 외부 저장 공간은 암호문을 복호화 하여 검색하여야 한다. 이는 개인만이 알고 있는 개인키를 외부 저장 공간과 공유해야하는 문제점을 발생시킨다.

이러한 문제점을 해결하기 위해 많은 연구가 진행되었고 2000년 Song et. al.[11]은 대칭키 방식에 기반하여 암호화되어 저장된 데이터들로부터 평문의 정보를 유출하지 않으며 검색어를 이용해 원하는 데이터만 찾을 수 있는 프로토콜을 제안하였다. 그 후 Boneh et. al.[2] 은 처음으로 공개키 기반 환경에서 키워드를 이용한 검색기술인 PEKS(public key encryption with keyword search)를 제안하였다. PEKS는 송신자의 공개키 정보를 이용하여 누구나 암호문을 생성할 수 있기 때문에 송신자가 수신자에게 메시지를 전송하는 이메일 등과 같은 환경에서 사용 할 수가 있다. 이메일 시스템에서 PEKS는 다음과 같이 이루어진다. 송신자가 수신자의 공개키로 메시지를 암호화하고 PEKS 암호문(암호화된 검색어)을 첨부한 형태로 수신자에게 메일을 발송한다. 그 후 수신자는 수신된 메일 중 자신이 원하는 검색어를 포함한 메일을 찾고 싶을 때 검색어와 자신의 개인키를 결합해 검색용 트랩도어(trapdoor)를 만들어 이메일 서버에 전송한다. 이메일 서버는 트랩도어와 PEKS 암호문에 대한 Test 함수의 결과를 얻어서 그 결과가 '1'이면 트랩도어 생성에 사용된 검색어와 PEKS 암호문 생성에 사용된 검색어가 일치하는 것으로 PEKS 암호문에 대응하는 암호화된 메시지를 수신자에게 제공한다. 만일 결과가 '0'이면 트랩도어 생성에 사용된 검색어와 PEKS 암호문 생성에 사용된 검색어가 불일치하는 것으로 Test 함수를 종료한다.

PEKS와 같이 송신자와 수신자간의 사전 동의 없이 검색어를 이용한 검색이 가능하기 위해서는 추측 가능한 검색어가 이용되어야 한다. 공개키 기반 환경에서 키워드를 이용한 검색기술인 PEKS는 수신자의 공개키를 이용해 누구나 PEKS 암호문을 만들어 볼 수 있기 때문에 만일 공격자에게 정당한 트랩도어가 주어진다면 트랩도어에 사용된 검색어를 추측해 PEKS 암호문을 만들고 공개된 Test 함수를 이용한다면 트랩도어 생성에 사용된 검색어를 추측 볼 수 있다.

Byun et. al.[5] 은 이러한 문제점을 지적하며 PEKS 스킴이 Keyword guessing attack에 안전하지 않음을 보이며 Keyword guessing attack에 안전한 PEKS를 만드는 것을 Open problem으로 남겨 두었다. 이후 Jeong et. al.[9] 은 Keyword guessing attack에 안전한 PEKS를 만드는 것은 불가능하다는 것을 보였다. 2009년 Rhee et. al.[10] 은 수신자와 서버의 공개키로 키워드를 암호화 하여 서버만이 암호화된 키워드와 트랩도어를 비교할 수 있는 Keyword guessing attack에 안전한 dPEKS(searchable public key encryption for designated tester)기법을 제안하였다. dPEKS란 기존의 PEKS와는 달리 고정된 검사자(designated tester)만이 Test를 할 수 있는 기법으로 수신자의 공개키와 서버의 공개키를 이용해 PEKS 암호문을 만들어 수신자로부터 트랩도어를 받은 서버만이 서버의 개인키를 이용해 Test를 해 볼 수 있는 기법이다.

2010년 Yang et. al.[13] 은 수신자의 비밀값으로 만든 트랩도어의 비교가 아닌 암호문의 비교를 통해 암호문에 사용된 메시지의 동일성을 검사하는 기법인 PEET(probabilistic public key encryption with equality test)을 제안하였다. 이 기법은 PEKS와 달리 서로 다른 공개키로 암호화된 암호문

에서도 두 개의 암호문을 이용해 메시지의 동일성을 검사해 볼 수 있다. 또한 키워드가 아닌 메시지에 대한 동일성을 검사하기 때문에 Keyword guessing attack에 대한 안전성을 고려하지 않아도 된다. 하지만 PEET 기법은 공개키 방식에 기반 하여 구현되었기 때문에 누구나 암호문을 만들어 볼 수 있고 공개되어있는 Test 함수를 사용한다면 두 개의 암호문에 대한 메시지의 동일성을 검사해 볼 수 있어 암호문의 구별불가능성(ciphertext indistinguishability)을 보장하지 못한다는 단점이 있다. Yang et. al. 은 One-Way CCA(one way adaptive chosen ciphertext attack)를 통해 증명하였지만 이것은 공격자의 입장에서 구별불가능성 보다 더욱 강력한 제약이다. 구별불가능성이란 암호문을 통해 평문의 어떤 부분정보도 얻기 힘든 경우로 공격자 A에 의해 생성된 길이가 같은 메시지 m_0, m_1 중 하나의 메시지를 선택하여 암호문$C_i(i\in\{0,1\})$을 만든 다음 공격자 A에게 C_i에 사용된 메시지가 m_0, m_1 중 어떠한 것인지 구별하라는 문제이다. 이론적으로 안전한 기법이 되기 위해서는 공격자에게 유리한 환경을 조성한 상태에도 공격이 성공하지 못하는 기법이어야 한다. 하지만 One-Way CCA는 평문의 부분정보가 아닌 평문 전체의 정보를 맞추라는 개념으로 본 논문에서는 Yang et. al. 이 제안한 PEET기법에 고정된 검사자를 고려한 PEET기법을 제안하고 이를 dPEET 기법이라 부를 것이다. dPEET기법은 PEET기법과는 다르게 검색가능한 암호화 (searchable encryption) 기법들이 주로 고려하고 있는 암호문의 기밀성 (IND-CPA security) 을 제공하며 제안된 기법이 IND-CPA를 만족한다는 것을 제안기법-1과 제안기법-2를 통해 보인다. 제안기법-1은 계산적 Diffie-Hellman(CDH) 문제의 가정에 기반을 두고 랜덤 오라클(random oracle) 모델[3,4]에 안전함을 증명을 하였으며 제안기법-2는 결정적 겹선형 Diffie-Hellman(DBDH) 문제의 가정에 기반을 두고 랜덤오라클을 사용하지 않는 스탠다드 모델(standard model)[1,8,12]에 안전함을 증명하였다.

본 논문의 구성은 다음과 같다. 2장에서는 제안 기법을 위한 배경지식을 설명한다. 3장에서 제안기법인 dPEET기법의 형식적 정의와 안전성 모델을 정의한다. 4장에서는 프로토콜을 제안하고 안전성을 증명한다. 마지막으로 5장에서는 결론을 맺는다.

II. 배경지식

본 장에서는 제안 기법의 구성 및 안전성 증명에 필요한 겹선형 함수(bilinear map), CDH 가정과 DBDH 가정을 살펴보고, Yang et. al.이 제안한 메시지 동일성 검사 공개키 암호시스템을 설명한다.

2.1 겹선형 함수 및 CDH, DBDH 가정

겹선형 함수(Bilinear Maps). G_1과 G_2가 위수를 소수 q로 갖는 순환 군(group)이라고 하자. 군 G_1과 G_2에서 모두 이산대수문제(Discrete Logarithm Problem)가 어렵다고 가정하자. 겹선형 함수(bilinear map)은 다음과 같은 성질을 갖는 $G_1\times G_1$에서 군 G_2 위로 맵핑되는 함수 $e: G_1\times G_1\rightarrow G_2$이다.

(1) **겹선형성 (Bilinearity)**: 임의의 군 원소 $g\in G_1$와 $a,b\in Z_q^*$에 대하여 $e(g^a,g^b)=e(g,g)^{ab}$을 만족한다.

(2) **비소실성 (Non-degeneracy)**: $e(g,g)\neq 1$을 만족시키는 $g\in G_1$가 존재한다.

(3) **계산 가능성 (Computability)**: 임의의 $g_1, g_2\in G_1$에 대해서 $e(g_1,g_2)$를 계산하는 효율적인 알고리즘이 존재한다.

본 논문에서 사용되는 겹선형 함수는 위와 같은 성질을 가지는 $e: G_1\times G_1\rightarrow G_2$로 정의한다.

계산적 Diffie-Hellman 가정 (Computational Diffie-Hellman Assumption. CDH). 소수 위수 p인 (순환) 군 G_1와 G_1의 생성원 g이 주어졌다고 가정하자. 임의의 난수 $a,b\in Z_q^*$에 대해 g^a, g^b가 주어졌을 때 CDH 문제는 g^{ab} 값을 계산하는 문제이다. CDH 문제를 해결하는 알고리즘 A의 이점(advantage)은 다음과 같은 확률 값으로 정의된다.

$$\Pr\left[g^{ab}\leftarrow A(G_1, g, g^a, g^b)\right]\geq\epsilon$$

g^{ab}을 의미 있는 확률로 계산할 수 있는 다항식 시간 (polynomial time) 알고리즘 A가 존재하지 않는다면 군 G_1은 CDH 가정을 만족한다고 한다. 일반적으로, 겹선형 함수(bilinear map)가 정의되는 군은 CDH 가정을 만족한다는 것이 알려져 있다.

결정적 겹선형 Diffie-Hellman 문제 및 가정 (Decision Bilinear Diffie-Hellman Assumption. DBDH). 주어진 G_1의 생성원 g에 대해 g^a, g^b, g^c, T가 주어졌을 때 DBDH 가정이란 $T=e(g,g)^{abc}$인지 T가 임의의 난수인지를 의미 있는 확률로 판단할 수 있는 알고리즘 A이 존재하지 않음을 말한다. 알고리즘 A의 이점(advantage)은 다음과 같은 확률 값으로 정의된다.

$$\Pr[A(g, g^a, g^b, g^c) = e(g, g^{abc})] \geq \epsilon$$

2.2 메시지 동일성 검사 암호 시스템(PEET)

Yang et. al. 이 제안한 메시지 동일성 검사 암호 시스템은 공개키로 암호화된 두 개의 암호문을 이용하여 암호문의 복호화 과정 없이 암호문에 사용된 메시지의 동일성을 검사하는 시스템이다. 이 시스템은 송신자와 수신자, 그리고 서버로 구성되며, 송신자는 수신자의 공개키로 메시지를 암호화한 후 서버에 전송하면, 서버는 저장된 암호문들과 수신자로부터 받은 검색용 암호문을 이용해 메시지의 동일성을 검사한다. PEET 기법은 다음과 같은 4개의 다항식 시간(polynomial-time) 알고리즘들로 구성된다.

- $Setup(1^k)$: 셋업 알고리즘은 보안상수 k를 입력으로 받아 공개상수 pp와 수신자의 공개키, 개인키 쌍 (pk_r, sk_r)을 생성한다.
- $Encryption(pp, pk_r, m)$: 암호화 알고리즘은 공개상수 pp와 수신자의 공개키 pk_r , 메시지 m을 입력받은 후 암호문 C를 출력한다.
- $Decryption(pp, sk_r, CT)$: 복호화 알고리즘은 공개상수 pp와 수신자의 개인키 sk_r , 암호문 CT을 입력으로 받은 후 암호문에 대한 메시지 m을 출력한다.
- $Test(CT, CT')$: 테스트 알고리즘은 두 개의 암호문 CT과 CT'을 입력으로 받아 $m=m'$이라면 1을, 아니면 0을 반환한다.

III. 안전성 모델 및 정의

Yang et. al. 이 제안한 PEET기법은 공개된 Test 함수로 인해 CPA 공격자에 대한 구별불가능성을 보이지 못한다. 공개키 기반 환경에서 설계된 PEET기법은 누구나 수신자의 공개키를 이용해 암호문을 만들어볼 수 있으며 공개된 Test 함수를 이용한다면 암호문에 사용된 메시지를 구별해 낼 수 있다. 이것은 다음과 같은 단계를 통해 보일 수 있다. 먼저 공격자 A는 메시지 m_0, m_1을 생성하여 *Challenger*에게 준다. 두 개의 메시지를 받은 *Challenger*는 그 중 하나를 선택하여 공개키 pk를 이용하여 암호문 $C_i (i \in \{0,1\})$을 만들어 A에게 준다. 암호문 C_i를 받은 A는 공개키 $pk' (pk \neq pk')$와 m_0, m_1중 하나인 m_0 선택하여 암호문 C'를 만든다. 이후 A는 공개되어있는 Test 함수를 이용하여 $Test(C_i, C')$를 해본다. 만일 반환값이 1이라면 C_i에 사용된 메시지는 m_0이고 그렇지 않다면 C_i에 사용된 메시지는 m_1이다. Yang et. al.은 이러한 이유로 One-Way CCA를 통해 증명하였지만 이것은 구별불가능성보다 강력한 제약이기 때문에 이론적으로 약한 안전성을 갖는다. 본 장에서는 서버와 수신자를 제외한 외부공격자에 대해 CPA 안전성을 제공하는 dPEET 기법의 형식적 정의와 암호문의 기밀성을 제공하는 안전성 모델에 대해 설명한다.

3.1 dPEET 기법의 형식적 정의

dPEET 기법은 송신자, 수신자, 그리고 서버로 구성되며 수신자에게 전송하기 위한 암호문은 수신자와 서버의 공개키를 이용해 생성된다. 메시지의 동일성 검사는 서버만이 할 수 있으며 이를 위해 서버는 수신자로부터 검색용 암호문을 받아 테스트를 한다. dPEET 기법은 다음과 같은 다항식 시간 알고리즘들로 구성된다.

- $Setup(1^k)$: 셋업 알고리즘은 보안상수 k를 입력으로 받아 공개상수 pp를 생성한다.
- $KeyGen_s(pp)$: 서버 키 생성 알고리즘은 공개상수 pp를 입력으로 받아 테스트 서버의 공개키와 개인키 쌍 (pk_s, sk_s)을 생성한다.
- $KeyGen_r(pp)$: 수신자 키 생성 알고리즘은 공개상수 pp를 입력으로 받아 수신자의 공개키와 개인키 쌍 (pk_r, sk_r)을 생성한다.
- $Encryption(pp, pk_s, pk_r, m)$: 암호화 알고리즘은 공개상수 pp와 서버의 공개키 pk_s, 수신자의 공개키 pk_r, 메시지 m을 입력받은 후 암호문 CT를 출력한다.
- $Decryption(pp, sk_r, CT)$: 복호화 알고리즘은 공개

상수 pp와 수신자의 개인키 sk_r, 암호문 CT을 입력으로 받은 후 암호문에 대한 메시지 m을 출력한다.

- $Test(CT, CT')$: 테스트 알고리즘은 두 개의 암호문 CT과 CT'을 입력으로 받아 $m=m'$이라면 1을, 아니면 0을 반환한다.

3.2 dPEET 기법의 안전성 모델

제안기법의 안전성을 정의하기 위해 챌린저(challenger) C와 외부 공격자(adversary) A 사이의 상호적으로 수행되는 게임(Game)을 고려한다. 외부 공격자 A는 서버만이 수행할 수 있는 $Test$를 시도하며 암호문에 대한 메시지 m을 얻기 위해 공격한다. 이때 서버와 수신자는 공격자에서 제외된다.

$$\mathbf{Game}(\mathrm{Exp}_{A}^{Game-cpa}(k))$$

- $Setup$: 챌린저(challenger) C는 보안상수 k을 선택하고 공개상수 pp, 서버와 수신자의 공개키/개인키 쌍 (pk_s, sk_s), (pk_r, sk_r)을 얻기 위해 $Setup$ 알고리즘을 실행한다. 챌린저 C는 공개상수 pp와 서버의 공개키 pk_s, 수신자의 공개키 pk_r을 공격자 A에게 준다.
- $Challenge$: 공격자 A는 평문 m_0, m_1을 생성하여 챌린저 C에게 준다. 챌린저 C는 $i \in \{0,1\}$을 선택하고 CT_i^*를 생성하여 공격자 A에게 준다.
- $Guess$: 공격자 A는 $i' \in \{0,1\}$을 출력하고 $i'=i$인 경우 공격자 A가 게임에서 이긴다.

이러한 $Game-cpa$의 이점(advantage)을 다음과 같이 정의 한다.

$$Adv_{Game-cpa,A} = \left| \Pr[i=i'] - \frac{1}{2} \right|$$

정의 1. 어떠한 다항 함수 시간에 동작하는 공격자 A에 대하여 공격자 A가 위에서 정의된 게임에서 이길 확률이 무시할(negligible)만 하다면 제안된 기법은 IND-CPA 관점에서 안전하다고 정의한다.

IV. 제안하는 기법

제안기법-1은 CDH 문제의 가정에 기반을 두고 랜덤 오라클 모델에 안전함을 증명을 하고 제안기법-2는 DBDH 문제의 가정에 기반을 두고 랜덤 오라클을 사용하지 않는 스탠다드 모델에 안전함을 증명한다. 랜덤 오라클 모델이란 충돌쌍을 생성하지 않는 이상적인 랜덤 함수의 존재를 가정해야 증명이 가능 하지만 스탠다드 모델보다 기법의 연산 효율성이 뛰어나다. 스탠다드 모델은 이상적인 랜덤 함수의 존재를 가정할 필요 없이 안전성 증명이 가능하다. 이는 스탠다드 모델이 랜덤 오라클 모델 보다 더 적은 가정을 사용함을 의미한다. 따라서 스탠다드 모델에서 안전성 증명이 가능하다면 안전성이 더 뛰어나다고 할 수 있다. 본 논문에서는 랜덤 오라클 모델에서 안전성 증명이 가능한 기법을 4.1절에서 제안하고, 스탠다드 모델에서 안전성 증명이 가능한 기법을 4.3절에서 제안한다.

4.1 제안기법-1

- $Setup(1^k)$: 셋업 알고리즘은 보안상수 $k \in Z^+$를 입력으로 받고 다음과 같이 작동한다. 적절한 크기의 소수 q를 생성하고 (G_1, G_2, e)을 생성한다. 이때 G_1과 G_2는 소수 q을 위수로 갖는 순환군들이다. 다음으로 G_1에서 임의의 생성원 g와 암호학적 해쉬함수 $H_1 : \{0,1\}^* \rightarrow \{0,1\}^n$, $H_2 : G_1 \rightarrow G_1$을 선택한다. 그 후 공개 파라미터 pp를 $pp = (q, G_1, G_2, e, g, H_1, H_2)$과 같이 설정한다.
- $KeyGen_s(pp)$: 서버 키 생성 알고리즘은 공개상수 pp을 입력으로 받아 임의의 난수 $\alpha \leftarrow Z_q^*$을 선택하여 g^α을 계산한다. 테스트 서버의 공개키와 개인키$(pk_s = g^\alpha, sk_s = \alpha)$를 출력한다.
- $KeyGen_r(pp)$: 수신자 키 생성 알고리즘은 공개상수 pp을 입력으로 받아 임의의 난수 $\beta \leftarrow Z_q^*$을 선택하여 g^β을 계산한다. 수신자의 공개키와 개인키$(pk_r = g^\beta, sk_r = \beta)$를 출력한다.
- $Encryption(pp, pk_s, pk_r, m)$: 암호화 알고리즘은 공개 파라미터 pp와 테스트 서버의 공개키 pk_s, 수신자의 공개키 pk_r, 메시지 $m \in G_1^* (\stackrel{\text{def}}{=} G_1 \setminus \{1\})$을 입력으로 받는다. 메시지 m에 대한 암호문은 다음과 같이 계산한다. 임의의 $r \leftarrow Z_q^*$, $\delta \leftarrow Z_q^*$을 선택한 후 다음과 같이 암호문 $CT = (C_1, C_2, C_3, C_4)$를 계산한다.

$$C_1 = g^\delta,$$
$$C_2 = H_1(m)^r,$$
$$C_3 = g^r \oplus H_2(g^{\alpha\delta}),$$
$$C_4 = m \oplus H_2(g^{\beta\delta})$$

• $Decryption(pp, sk_r, CT)$: 복호화 알고리즘은 공개 파라미터 pp, 수신자의 개인키 $sk_r = \beta$, 암호문 $CT = (C_1, C_2, C_3, C_4)$을 입력으로 받는다. 다음을 계산하여 메시지 m을 얻는다.

$$m = H_2(C_1^{\beta}) \oplus C_4$$

• $Test(sk_s, CT, CT')$: 테스트 알고리즘은 테스트 서버의 개인키 $sk_s = \alpha$와 두 개의 암호문 $CT = (C_1, C_2, C_3, C_4)$와 $CT' = (C_1', C_2', C_3', C_4')$을 입력으로 받는다. 각각의 암호문에서 $g^r = H_2(C_1^{\alpha}) \oplus C_3, g^{r'} = H_2(C_1'^{\alpha}) \oplus C_3'$을 계산한 후 $e(g^r, C_2') = e(g^{r'}, C_2)$을 통해 두 암호문에 대한 메시지의 동일성을 확인한다. 만일 위의 등식이 성립하면 1을 반환하고 그렇지 않다면 0을 반환한다.

정확성(Correctness). 위의 제안 기법에 대한 정확성은 다음과 같이 보일 수 있다.

- 복호화과정.

주어진 암호문

$CT = (C_1, C_2, C_3, C_4)$

$= (g^{\delta}, H_1(m)^r, g^r \oplus H_2(g^{\alpha\delta}), m \oplus H_2(g^{\beta\delta}))$와 수신자의 개인키$sk_r = \beta$을 이용하여 메시지 m을 얻을 수 있다.

$$\begin{aligned} m &= H_2(C_1^{\beta}) \oplus C_4 \\ &= H_2(g^{\delta\beta}) \oplus H_2(g^{\beta\delta}) \oplus m \end{aligned}$$

- $Test$과정.

주어진 암호문

$CT = (C_1, C_2, C_3, C_4)$

$= (g^{\delta}, H_1(m)^r, g^r \oplus H_2(g^{\alpha\delta}), m \oplus H_2(g^{\beta\delta}))$,

$CT' = (C_1', C_2', C_3', C_4')$

$= (g^{\delta'}, H_1(m)^{r'}, g^{r'} \oplus H_2(g^{\alpha\delta'}), m \oplus H_2(g^{\beta'\delta'}))$와 테스트 서버의 개인키$sk_s = \alpha$을 이용하여 $Test$ 알고리즘의 정확성을 보일 수 있다. 먼저 아래와 같은 계산을 통해 $g^r, g^{r'}$를 구한다.

$$\begin{aligned} g^r &= C_3 \oplus H_2(C_1^{\alpha}) \\ &= g^r \oplus H_2(g^{\alpha\delta}) \oplus H_2(g^{\delta\alpha}) \end{aligned}$$

$$\begin{aligned} g^{r'} &= C_3' \oplus H_2(C_1'^{\alpha}) \\ &= g^{r'} \oplus H_2(g^{\alpha\delta'}) \oplus H_2(g^{\delta'\alpha}) \end{aligned}$$

다음으로 $g^r, g^{r'}, C_3, C_3'$을 두 암호문에 대한 메시지 동일성 $Test$과정을 수행한다.

$$\begin{aligned} & e(g^r, C_2') \stackrel{?}{=} e(g^{r'}, C_2) \\ \Leftrightarrow\ & e(g^r, H_1(m)^{r'}) = e(g^{r'}, H_1(m)^r) \\ \Leftrightarrow\ & e(g, H_1(m))^{rr'} = e(g, H_1(m))^{rr'} \end{aligned}$$

4.2 안전성 증명

제안된 기법의 증명은 하이브리드 게임(hybrid game)[6]을 통해 증명된다. 실제 공격 환경에서 공격자에게 주어지는 챌린지 암호문을 (C_1, C_2, C_3, C_4)라 하고 G_1에서의 임의의 값 R', R''을 선택한다. 우리는 다음과 같은 세 가지 하이브리드 게임 Γ_0, Γ_1, Γ_2을 정의한다.

Γ_0: 챌린지 암호문 CT_0: (C_1, C_2, C_3, C_4)

Γ_1: 챌린지 암호문 CT_1: (C_1, C_2, R', C_4)

Γ_2: 챌린지 암호문 CT_2: (C_1, C_2, R', R'')

위의 게임 Γ_2에서의 챌린지 암호문는 결국 4개의 랜덤한 그룹 원소로 구성되어 있으므로 메시지에 대한 어떠한 정보를 노출하지 않음을 알 수 있다. 즉, 게임 Γ_0에서의 챌린지 암호문은 실제 공격에서 공격자에게 주어지는 올바른 형태(well-formed)이고 게임 Γ_2에서는 랜덤한 챌린지 암호문이 공격자에게 주어진다. 우리는 다음 정리 1과 정리 2 을 통하여 공격자가 게임 Γ_0와 게임 Γ_1, Γ_1와 Γ_2를 의미있는 확률(non-negligible advantage)로 구별할 수 없음을 보인다.

정리 1. CDH 가정과 랜덤오라클 모델 하에서 게임 Γ_0와 게임 Γ_1을 의미있는 확률로 구별할 수 있는 공격자는 존재하지 않는다.

게임 Γ_0와 게임 Γ_1을 의미있는 확률로 구별할 수 있는 공격자 A가 존재한다고 가정하자. 그러면 A를 이용하여 CDH 문제를 효율적으로 해결할 수 있는 알고리즘 B가 존재함을 보일 것이다. B는 (g, g^a, g^b)을 입력 받는다. B의 목적은 g^{ab}을 계산 하는 것이다. A의 질의에 대하여 충돌과 모순이 없는 응답을 하기 위하여 B는 초기에 비어있는 리스트 $L_{H_2} = \{\langle c_j, h_j \rangle\}$을 관리한다. B는 A을 하위 루틴으로 실행하고 A의 공격 환경을 다음과 같이 시뮬레이션 한다.

- *Setup*(k) : B는 공개파라미터를 생성하기 위해 $pk_s = g^a$, 임의의 $\beta \leftarrow Z_q^*$를 선택하여 $pk_r = g^\beta$ 로 설정하고 공개상수 $pp = (q, G_1, G_2, e, g, H_1, H_2)$와 pk_s, pk_r 을 A에게 준다.
- $H_2 - query$: A가 $c_k[c_j | j = 1, ..., q_h]$에 대한 H_2 질의를 요청할 때, 만약 리스트 L_{H_2}에 $\langle c_k, h_k \rangle$가 포함되어 있다면, B는 h_k를 반환한다. 그렇지 않으면 B는 $e(g^a, g^b) = ^? e(c_k, g)$를 확인한다. 등식이 성립한다면 B는 c_k값을 CDH 결과 값으로 출력한다. 그렇지 않다면 임의의 $h_k \in Z_q^*$을 선택하여 h_k를 반환한 후 L_{H_2}에 $\langle c_k, h_k \rangle$를 추가한다.
- *Challenge* : A는 m_0, m_1을 생성하여 B에게 준다. B는 $i \in \{0,1\}$을 선택하고 $C_1 = g^b$, $C_2 = H_1(m_i)^r$, $C_4 = m_i \oplus H_2(g^{b\beta})$ 을 계산하고 C_3을 임의의 값 R'로 선택하여 $CT_i^* = (C_1, C_2, C_3, C_4)$를 A에게 준다.
- *Guess* : A는 자신의 추측 비트값 $i' \in \{0,1\}$을 출력한다.

A는 *Challenge* 단계 이후에도 챌린지 암호문에 대한 비트값 $i' \in \{0,1\}$을 추측하기 위해 $H_2 - query$ 질의를 반복적으로 한다. 이 때 B는 A가 질의한 c_k이 CDH 값 g^{ab}가 맞는지 $e(g^a, g^b) = ^? e(c_k, g)$를 계산하여 확인한다. 만약 모든 $H_2 - query$ 질의에 대해 등식이 성립하지 않는다면 B는 A을 통해 어떠한 이점도 얻을 수 없다. 그러므로 A가 ϵ의 확률로 공격에 성공한다면 B도 ϵ의 확률로 CDH 문제를 풀게 된다.

정리 2. CDH 가정 하에서 게임 Γ_2와 게임 Γ_3을 의미있는 확률로 구별할 수 있는 공격자는 존재하지 않는다.

정리 2의 증명은 **정리 1**의 증명과 매우 동일하므로 자세한 증명은 생략한다. 정리 1과의 차이점은 정리 1에서는 *Setup* 단계에서 서버의 공개키를 $pk_s = g^a$로 설정하고 임의의 $\beta \leftarrow Z_q^*$을 선택하여 수신자의 공개키를 $pk_r = g^\beta$로 설정하여 A에게 공개파라미터를 생성해 주고 *Challenge* 단계에서 $C_1 = g^b$, $C_2 = H_1(m_i)^r$, $C_3 = R'$, $C_4 = m_i \oplus H_2(g^{b\beta})$을 계산하여 A에게 주었다. 그러나 정리 2에서는 수신자의 공개키를 $pk_r = g^a$로 설정하고 임의의 $\alpha \leftarrow Z_q^*$을 선택하여 서버의 공개키를 $pk_s = g^\alpha$로 설정하여 공개파라미터를 생성한다. *Challenge* 단계에서는 $C_1 = g^b$, $C_2 = H_1(m_i)^r$, $C_3 = R'$ $C_4 = R'$을 계산하여 A에게 준다.

4.3 제안기법-2

- *Setup*(1^k) : 셋업 알고리즘은 보안상수 $k \in Z^+$를 입력으로 받고 다음과 같이 작동한다. 적절한 크기의 소수 q을 생성하고 (G_1, G_2, e)을 생성한다. 이때 G_1과 G_2는 소수 q을 위수로 갖는 순환군들이다. 다음으로 G_1에서 임의의 생성원 g와 암호학적 해쉬함수 $H_1 : \{0,1\}^* \rightarrow \{0,1\}^n$, $H_2 : G_1 \rightarrow G_1$을 선택한다. 그 후 공개 파라미터 pp을 $pp = (q, G_1, G_2, e, g, H_1, H_2)$과 같이 설정한다.
- *KeyGen*$_s(pp)$: 서버 키 생성 알고리즘은 공개상수 pp을 입력으로 받아 임의의 난수 $\alpha \leftarrow Z_q^*$을 선택하여 g^α을 계산한다. 테스트 서버의 공개키와 개인키$(pk_s = g^\alpha, sk_s = \alpha)$를 출력한다.
- *KeyGen*$_r(pp)$: 수신자 키 생성 알고리즘은 공개상수 pp을 입력으로 받아 임의의 난수 $\beta \leftarrow Z_q^*$을 선택하여 g^β을 계산한다. 수신자의 공개키와 개인키$(pk_r = g^\beta, sk_r = \beta)$를 출력한다.
- *Encryption*(pp, pk_s, pk_r, m) : 암호화 알고리즘은 공개 파라미터 pp와 테스트 서버의 공개키 pk_s, 수신자의 공개키 pk_r, 메시지 $m \in G_1^* (\stackrel{\text{def}}{=} G_1 \setminus \{1\})$을 입력으로 받는다. 메시지 m에 대한 암호문은 다음과 같이 계산한다. 임의의 $r, \delta_1, \delta_2 \leftarrow Z_q^*$을 선택한 후 다음과 같이 암호문 $CT = (C_1, C_2, C_3, C_4, C_5)$를 계산한다.

$$C_1 = g^{\delta_1},$$
$$C_2 = g^{\delta_2},$$
$$C_3 = H_1(m)^r,$$
$$C_4 = g^r \cdot e(g^\alpha, g^{\delta_1})^{\delta_2},$$
$$C_5 = m \cdot e(g^\beta, g^{\delta_1})^{\delta_2}$$

- *Decryption*(pp, sk_r, CT) : 복호화 알고리즘은 공개파라미터 pp, 수신자의 개인키 $sk_r = \beta$, 암호문 $CT = (C_1, C_2, C_3, C_4, C_5)$을 입력으로 받는다. 다음을 계산하여 메시지 m을 얻는다.

$$m = C_5 / e(C_1, C_2)^\beta$$

- $Test(sk_s, CT, CT')$: 테스트 알고리즘은 테스트 서버의 개인키 $sk_s = \alpha$와 두 개의 암호문 $CT = (C_1, C_2, C_3, C_4, C_5)$와 $CT' = (C_1', C_2', C_3', C_4', C_5')$을 입력으로 받는다. 각각의 암호문에서 $g^r = C_4/e(C_1, C_2)^\alpha$와 $g^{r'} = C_4'/e(C_1', C_2')^\alpha$을 계산한 후 $e(g^r, C_3') = e(g^{r'}, C_3)$을 통해 두 암호문에 대한 메시지의 동일성을 확인한다. 만일 위의 등식이 성립하면 1을 반환하고 그렇지 않다면 0을 반환한다.

정확성(Correctness). 위의 제안 기법에 대한 정확성은 다음과 같이 보일 수 있다.

- 복호화과정.

주어진 암호문

$CT = (C_1, C_2, C_3, C_4, C_5)$

$= \left(g^{\delta_1}, g^{\delta_2}, H_1(m)^r, g^r \cdot e(g^\alpha, g^{\delta_1})^{\delta_2}, m \cdot e(g^\beta, g^{\delta_1})^{\delta_2}\right)$와 수신자의 개인키$sk_r = \beta$을 이용하여 메시지 m을 얻을 수 있다.

$$m = C_5/e(C_1, C_2)^\beta$$
$$= m \cdot e(g^\beta, g^{\delta_1})^{\delta_2}/e(g^{\delta_1}, g^{\delta_2})^\beta$$

- $Test$과정.

주어진 암호문

$CT = (C_1, C_2, C_3, C_4, C_5)$

$= \begin{pmatrix} g^{\delta_1}, g^{\delta_2}, H_1(m)^r, g^r \cdot e(g^\alpha, g^{\delta_1})^{\delta_2}, \\ m \cdot e(g^\beta, g^{\delta_1})^{\delta_2} \end{pmatrix}$,

$CT = (C_1', C_2', C_3', C_4', C_5')$

$= \begin{pmatrix} g^{\delta_1'}, g^{\delta_2'}, H_1(m)^{r'}, g^{r'} \cdot e(g^\alpha, g^{\delta_1'})^{\delta_2'}, \\ m \cdot e(g^\beta, g^{\delta_1'})^{\delta_2'} \end{pmatrix}$ 와 테스트 서버의 개인키$sk_s = \alpha$을 이용하여 $Test$ 알고리즘의 정확성을 보일 수 있다. 먼저 아래와 같은 계산을 통해 $g^r, g^{r'}$를 구한다.

$$g^r = C_4/e(C_1, C_2)^\alpha$$
$$= g^r \cdot e(g^\alpha, g^{\delta_1})^{\delta_2}/e(g^{\delta_1}, g^{\delta_2})^\alpha$$

$$g^{r'} = C_4'/e(C_1', C_2')^\alpha$$
$$= g^{r'} \cdot e(g^\alpha, g^{\delta_1'})^{\delta_2'}/e(g^{\delta_1'}, g^{\delta_2'})^\alpha$$

다음으로 $g^r, g^{r'}, C_3, C_3'$을 두 암호문에 대한 메시지 동일성 $Test$과정을 수행한다.

$$e(g^r, C_3') \stackrel{?}{=} e(g^{r'}, C_3)$$
$$\Leftrightarrow e(g^r, H_1(m)^{r'}) = e(g^{r'}, H_1(m)^r)$$
$$\Leftrightarrow e(g, H_1(m))^{rr'} = e(g, H_1(m))^{rr'}$$

4.4 안전성 증명

제안된 기법의 증명은 하이브리드 게임(hybrid game)을 통해 증명된다. 실제 공격 환경에서 공격자에게 주어지는 챌린지 암호문을 $(C_1, C_2, C_3, C_4, C_5)$라 하고 G_1에서의 임의의 값 R', R''을 선택한다. 우리는 다음과 같은 세 가지 하이브리드 게임 Γ_0, Γ_1, Γ_2을 정의한다.

Γ_0: 챌린지 암호문 CT_0: $(C_1, C_2, C_3, C_4, C_5)$
Γ_1: 챌린지 암호문 CT_1: (C_1, C_2, C_3, R', C_5)
Γ_2: 챌린지 암호문 CT_2: (C_1, C_2, C_3, R', R'')

위의 게임 Γ_2에서의 챌린지 암호문는 결국 5개의 랜덤한 그룹 원소로 구성되어 있으므로 메시지에 대한 어떠한 정보를 노출하지 않음을 알 수 있다. 즉, 게임 Γ_0에서의 챌린지 암호문은 실제 공격에서 공격자에게 주어지는 올바른 형태(well-formed)이고 게임 Γ_2에서는 랜덤한 챌린지 암호문이 공격자에게 주어진다. 우리는 다음 정리 3과 정리 4 을 통하여 공격자가 게임 Γ_0와 게임 Γ_1, Γ_1와 Γ_2를 의미있는 확률로 구별할 수 없음을 보인다.

정리 3. DBDH 가정 하에서 게임 Γ_0와 게임 Γ_1을 의미있는 확률로 구별할 수 있는 공격자는 존재하지 않는다.

게임 Γ_0와 게임 Γ_1을 의미있는 확률로 구별할 수 있는 공격자 A가 존재한다고 가정하자. 그러면 A를 이용하여 DBDH 문제를 효율적으로 해결할 수 있는 알고리즘 B가 존재함을 보일 것이다. B는 (g, g^a, g^b, g^c, T)을 입력 받는다. B의 목적은 $T \stackrel{?}{=} e(g,g)^{abc}$인지 구별하는 것이다. B는 A을 하위 루틴으로 실행하고 A의 공격 환경을 다음과 같이 시뮬레이션 한다.

- $Setup(k)$: B는 공개파라미터를 생성하기 위해 $pk_s = g^a$, 임의의 $\beta \leftarrow Z_q^*$를 선택하여 $pk_r = g^\beta$ 로 설정하고 공개상수 $pp = (q, G_1, G_2, e, g, H_1, H_2)$와 pk_s, pk_r 을 A에게 준다.
- $Challenge$: A는 m_0, m_1을 생성하여 B에게 준다. B는 $i \in \{0,1\}$와 임의의 $r \leftarrow Z_q^*$을 선택하고 $C_1 = g^b$, $C_2 = g^c$, $C_3 = H_1(m_i)^r$, $C_4 = g^r \cdot T$ $C_5 = m_i \cdot e(g^b, g^c)^\beta$을 계산하여 $CT_i^* = (C_1, C_2, C_3, C_4, C_5)$를 A

에게 준다.

- *Guess* : A는 자신의 추측 비트값 $i' \in \{0,1\}$을 출력한다.

B는 출력된 비트값 i'을 i와 비교한다. 만일 $i'=i$인 경우 B는 자신의 추측 비트값으로 i'를 출력한다. 만일 $i' \neq i$인 경우는 B는 임의의 비트 값을 출력한다.

다음을 사항을 관찰해 보자. 만일, T가 임의의 난수값이면 공격자에게 챌린지 암호문은 난수값으로 보이므로 공격자의 이점을 이용할 수 없다. 만일 $T=e(g,g)^{abc}$이면 위의 시뮬레이션은 완전한 공격 시뮬레이션을 제공한다. 즉, 제안 기법-2의 실행환경과 동일하다. 따라서, 의미있는 확률로 게임 Γ_0와 게임 Γ_1을 의미있는 확률로 구별할 수 있는 공격자 A가 존재하면, *DBDH* 문제를 해결할 수 있는 효율적인 알고리즘이 존재한다.

정리 4. DBDH 가정 하에서 게임 Γ_2와 게임 Γ_3을 의미있는 확률로 구별할 수 있는 공격자는 존재하지 않는다.

정리 2의 증명은 정리 1의 증명과 매우 동일하므로 자세한 증명은 생략한다. 정리 1과의 차이점은 정리 1에서는 *Setup* 과정에서 서버의 공개키를 $pk_s=g^a$로 설정하고 임의의 $\beta \leftarrow Z_q^*$를 선택하여 수신자의 공개키를 $pk_r=g^\beta$로 설정하여 A에게 공개파라미터를 생성해 주고 *Challenge* 단계에서 $C_1=g^b$, $C_2=g^c$, $C_3=H_1(m_i)^r$, $C_4=g^r \cdot T$, $C_5=m_i \cdot e(g^b,g^c)^\beta$을 계산하여 A에게 주었다. 그러나 정리 2에서는 수신자의 공개키를 $pk_r=g^a$로 설정하고 임의의 $\alpha \leftarrow Z_q^*$를 선택하여 서버의 공개키를 $pk_s=g^\alpha$로 설정하여 공개파라미터를 생성한다. *Challenge* 단계에서는 $C_1=g^b$, $C_2=g^c$, $C_3=H_1(m_i)^r$, $C_4=R$, $C_5=m_i \cdot T$을 계산하여 A에게 준다.

V. 결 론

본 논문에서는 고정된 검사자를 고려한 메시지 동일성 검사 공개키 암호시스템(dPEET)을 제안하였으며 제안기법-1은 CDH 문제의 가정에 기반을 두고 랜덤 오라클 모델에 안전함을 증명하였고 제안기법-2는 DBDH 문제의 가정에 기반을 두고 스탠다드 모델에 안전함을 증명하였다. 본 논문에서 제안한 기법은 메시지에 대한 동일성 검사를 수행하기 때문에 Keyword guessing attack에 안전하며 기존의 PEET 기법이 보이지 못한 IND-CPA를 만족한다. 또한 본 논문에서 제안한 기법에 Fujisaki-Okamoto 변환 기법[7]을 적용하면 CCA 안전성을 갖는 dPEET 기법을 쉽게 설계할 수 있다.

참고문헌

[1] D. Boneh and X. Boyen, "Secure Identity-based encryption without random oracles," Advances in Cryptology - CRYPTO 2004, LNCS 3152, pp.443-459, 2004.

[2] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," Proc. EUROCRYPT 2004, LNCS 3027, pp.506-522, 2004.

[3] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," Pro. CRYPTO 2001, LNCS 2139, pp.213-229, 2001.

[4] M. Bellare and P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols," In ACM CCS, pp.62-73, November. 1993.

[5] J.W. Byun, H.S. Rhee, H. Park, and D.H. Lee, "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," SDM 2006, LNCS 4165, pp. 75 - 83, 2006.

[6] X. Boyen and D. Boneh, "Anonymous Hierarchical Identity-based encryption (without random oracles)," Advances in Cryptology - CRYPTO 2006, LNCS 4117, pp.290-307, 2006.

[7] E. Fujisaki and T. Okamoto, "Secure integration of asymmetric and symmetric encryption schemes," Proc. Crypto 99, LNCS 1666, pp. 537-554, 1999.

[8] C. Gentry, "Practical Identity-Based Encryption Without Random Oracles," Advances in Cryptology - EUROCRYPTO 2006, LNCS 4004, pp.445-464, 2006.

[9] I. R. Jeong, J. O. Kwon, and D. H. Lee, "Constructing PEKS Schemes Secure against Keyword Guessing Attacks Is Possible?," Computer Communications, Elsevier, Vol. 32, pp.394-396, Feb. 2009.

[10] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Improved searchable Public key encryption with designated tester," Proc. ASIACCS'09(to appear), March. 2009.

[11] D. Song, D. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," Proceedings of IEEE Symposium on Security and Privacy, pp. 44-55, May. 2000.

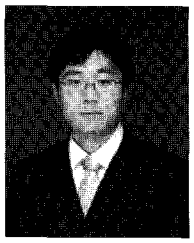
[12] B. Warters. "Efficient Identity-Based Encryption Without Random Oracles," Advances in Cryptology - EUROCRYPTO 2005, LNCS 4005, pp.114-127, 2005.

[13] G. Yang, C. H. Tan, Q. Huang, and D. S. Wong, "Probabilistic Public Key Encryption With Equality Test," Proc. CT-RSA2010, LNCS 5985, pp.119-131, 2010.

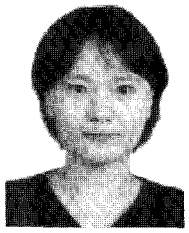
[14] 매일경제신문, "2천만 정보 유출". (http://mbn.mk.co.kr/pages/news/newsView.php?news_seq_no=492145&category=mbn00000)

〈著者紹介〉

이 영 민 (Young min Lee) 학생회원
2009년 8월: 단국대학교 컴퓨터과학과 학사 졸업
2009년 9월~현재: 고려대학교 정보보호학과 석사과정
〈관심분야〉 정보보호 이론, 암호 프로토콜, 프라이버시향상기술(PET)



구 우 권 (Woo Kwon Koo) 학생회원
2006년 2월: 고려대학교 수학과 학사 졸업
2008년 2월: 고려대학교 정보경영공학과 공학 석사 졸업
2008년 3월~현재: 고려대학교 정보보호학과 박사과정
〈관심분야〉 정보보호이론, 암호 프로토콜, 프라이버시향상기술(PET)



이 현 숙 (hyun Sook Rhee) 정회원
1998년 2월: 단국대학교 수학과 학사 졸업
2000년 2월: 단국대학교 응용수학과 이학 석사 졸업
2008년 2월: 고려대학교 정보경영공학과 공학 박사 졸업
2008년 3월: 고려대학교 정보경영공학전문대학원, 박사후 연구원
2011년 1월: 삼성전자 책임, 인증 및 보안솔루션 개발
〈관심분야〉 정보보호 이론, IPTV 와 Smart Card 관련 보안기술, 프라이버시향상기술(PET)

이 동 훈 (Dong Hoon Lee) 정회원
1983년: 고려대학교 경제학과 학사 졸업
1987년: Oklahoma University 전산학 석사 졸업
1992년: Oklahoma University 전산학 박사 졸업
1993년~1997년: 고려대학교 전산학과 조교수
1997년~2001년: 고려대학교 전산학과 부교수
2001년~현재: 고려대학교 정보보호대학원 교수
〈관심분야〉 정보보호이론, 암호 프로토콜, USN, 키 교환, 프라이버시향상기술(PET), 익명성 연구