표준 모델에서 안전한 Diffie-Hellman 키 교환 프로토콜

(A Diffie-Hellman Key Exchange Protocol in the Standard Model)

정 익 래 [†] 권 정 옥 [††] 이 동 훈 [††] 홍 도 원 [†††]
(Ik Rae Jeong) (Jeong Ok Kwon) (Dong Hoon Lee) (Dowon Hong)

요 약 MQV 프로토콜은 가장 효율적인 Diffie-Hellman 키 교환 프로토콜로 여겨지고 있으며, 미국 NSA를 비롯한 많은 기관들에서 표준으로 채택되었다. Crypto 2005에서 Hugo Krawczyk는 MQV의 약점들을 보였으며, MQV를 변형한 HMQV를 제안했다. HMQV는 MQV와 비슷한 계산량을 요구하는 반면 다양한 안전성을 만족하며, 랜덤 오라클 모델에서 안전성 증명이 가능하다. 이 논문에서 HMQV가 제공하는 다양한 안전성을 만족하면서도 랜덤 오라클을 사용하지 않는 Diffie-Hellman 키 교환 프로토콜을 제안한다. 지금까지는 랜덤 오라클을 사용하지 않으면서 HMQV가 제공하는 다양한 안전성을 보장하는 Diffie-Hellman 키 교환 프로토콜은 존재하지 않았다.

키워드 : 키 교환, Diffie-Hellman, 전방위 안전성, 개인키 사용 위장 공격

Abstract The MQV protocol has been regarded as the most efficient authenticated Diffie-Hellman key exchange protocol, and standardized by many organizations including the US NSA. In Crypto 2005, Hugo Krawczyk showed vulnerabilities of MQV to several attacks and suggested a hashed variant of MQV, called HMQV, which provides the same superb performance of MQV and provable security in the random oracle model. In this paper we suggest an efficient authenticated Diffie-Hellman key exchange protocol providing the same functionalities and security of HMQV without random oracles. So far there are no authenticated Diffie-Hellman protocols which are provably secure without using random oracles and achieve the same level of security goals of HMQV efficiently yet.

Key words : Key exchange, Diffie-Hellman, Strong forward secrecy, Key compromise impersonation

1. 서 론

키 교환 프로토콜은 가장 널리 사용되는 암호학 프로토콜 중에 하나이다. 키 교환 프로토콜은 사용자들간에 공통된 세션키를 만들며, 이런 세션키들은 안전하지 않은 네트워크를 통해서도 안전한 통신을 가능하게 해준다. 그러므로 안전한 키 교환 프로토콜은 좀 더 복잡하고 안전한 상위 레벨의 어플리케이션을 위한 기본적인 툴로 사용될 수 있다. 이 논문에서는 인증된 키 교환 프로토콜에 국한하여 논의한다. 인증된 키 교환을 위한 일반적인 방법은 각 사용자가 인증된 공개키-개인키 쌍을 가지고 키 교환을 하는 방식이다.

기존 연구와의 관련성을 논하기 전에 먼저 키 교환 프로토콜에서의 안전성 개념에 대해서 살펴보자. 엄밀한 정의는 다음절에 나온다. 가장 기본적으로 인증된 키 교환 프로토콜은 세션키에 대한 비밀성을 보장해야 한다. 이런 안전성 개념은 어떤 공격에 대해서 안전한지를 말

· 본 연구는 지식경제부 및 정보통신연구진흥원의 IT신성장동력핵심기술 개발사업의 일환으로 수행하였음. [2005-Y001-04, 차세대 시큐리티 기술 개발]. 이 연구에 참여한 연구자 중 일부는 '2단계 BK21사업'의 지원비를 받았음

† 정 회 원 : 고려대학교 정보경영공학부
irjeong@korea.ac.kr
†† 정 회 원 : 고려대학교 정보경영공학전문대학원
pitapat@korea.ac.kr
donghlee@korea.ac.kr
††† 정 회 원 : 한국전자통신연구원 암호기술연구팀 팀장
dwhong@etri.re.kr

논문접수 : 2007년 7월 3일
심사완료 : 2008년 9월 22일

정보과학회논문지: 정보통신 제35권 제6호(2008.12)

해야 한다. 키 독립성(security against known-key attacks)은 세션키들이 공격자에게는 독립적으로 보여야 한다는 것을 의미하며, Denning-Sacco 공격[1]에 대해 안전해야 한다는 것을 말한다.

약한 전방위 안전성(weak forward secrecy)은 공격자의 *간섭 없이* 만들어진 세션키는 사용자의 개인키를 가진 공격자라고 할지라도 세션키를 알 수 없다는 것을 말한다. 강한 전방위 안전성(strong forward secrecy)은 비록 공격자의 간섭으로 만들어진 세션키라도, 세션키가 만들어진 이후에 사용자의 개인키가 노출된다면, 그 개인키를 아는 공격자라 할지라도 세션키를 알 수 없다는 것을 말한다.

세션 상태 노출에 대한 안전성(security against Session State Reveal)은 공격자가 세션키를 만드는 데 사용되는 난수값을 가지고서도 세션키를 알 수 없어야 한다는 것을 말한다. 세션 상태 노출 공격은 사용자가 중요시하는 롱텀키인 개인키보다는 일회용 비밀값인 난수들이 더욱 쉽게 노출될 수 있다는 관점에서 제안된 개념이다[2,3].

이 밖에 개인키 사용 위장에 대한 안전성(security against Key Compromise Impersonation)과 파트너 혼돈 공격에 대한 안전성(security against Unknown Key Share)이 존재한다. A의 개인키가 노출된다면, 공격자는 A의 개인키를 가지고서 B에게 A인척 위장할 수 있다. 개인키 사용 위장에 대한 안전성이란 A의 개인키가 노출되더라도 A의 개인키을 가진 공격자가 A에게 B인척 위장할 수 없어야 한다는 것을 말한다. 파트너 혼돈 공격에 대한 안전성이란 A와 B가 똑같은 세션키를 계산했으면 A는 현재 B와 키 교환하고 있다고 인식해야 하며, B 또한 A와 키 교환하고 있다고 인식해야 한다는 것을 말한다. III장에서 강한 전방위 안전성이 세션 상태 노출에 대한 안전성 이외의 모든 안전성 개념들을 내포한다는 것을 증명한다.

1.1 기존 연구

[4,5]에서, Menezes 등이 MQV라는 양자간 키 교환 프로토콜을 제안했다. MQV는 현재까지 가장 효율적인 Diffie-Hellman 키 교환 프로토콜로 여겨지고 있으며, 미국 NSA를 비롯한 많은 기관에서 표준화되었다[6-11]. [3]에서 Krawczyk가 MQV의 약점들을 지적했으며, HMQV라는 MQV의 변형을 제안하였으며, HMQV가 약한 전방위 안전성과 세션 상태 노출에 대한 안전성 등을 만족함을 증명하였다. 또한 강한 전방위 안전성을 제공하기 위해서 HMQV-C를 제안하였다[12].

랜덤 오라클 모델은 스탠다드 모델에 비해서 프로토콜 설계나 안전성 증명을 더욱 쉽게 할 수 있다. 하지만 랜덤 오라클 모델에서 안전한 프로토콜은 랜덤 오라클을 실제 존재하는 함수로 대체하였을 경우에 안전하지 않을 수 있다[13-15]. 따라서 스탠다드 모델에서 안전성 증명이 가능한 프로토콜이 중요하며, 이런 스탠다드 모델에서 안전성 증명이 된 여러 프로토콜들이 존재한다[2,16-19]. 하지만 현재까지 나와 있는 스탠다드 모델에서 안전한 프로토콜들 모두는 세션키를 계산할 때 난수들만을 사용하므로, 세션 상태 노출 공격에 안전하지 않다. 이 논문에서 스탠다드 모델에서 안전성 증명이 가능하며, 강한 전방위 안전성과 세션 상태 노출에 대한 안전성 등 HMQV-C가 제공하는 모든 안전성을 제공하는 키 교환 프로토콜을 제시한다.

2. 암호학적 툴

2.1 HDH(Hash Diffie-Hellman) 문제[20,21]

θ를 안전성 파라미터라고 하고 $H: \{0,1\}^* \rightarrow \{0,1\}^\theta$를 해쉬함수라고 하자. GG를 그룹 오더가 q인 그룹 G와 그룹 생성자 g를 생성하는 그룹 생성 알고리즘이라고 하자. 다음의 실험을 생각해 보자.

$\mathrm{Exp}_A^{HDH}(\theta)$
$(G,q,g) \leftarrow GG(1^\theta)$
$(u_1,u_2) \leftarrow [1,q]$
$U_1 = g^{u_1};\ U_2 = g^{u_2};\ d \leftarrow [0,1]$
만약 $d=1$, $W = H(g^{u_1 u_2})$
그렇지 않으면, $W \leftarrow \{0,1\}^\theta$
$d' \leftarrow A(G,q,g,U_1,U_2,W)$

공격자 A의 어드벤티지는 다음과 같이 정의된다:

$$Adv_A^{HDH} = 2\Pr[d=d'] - 1.$$

어드벤티지 함수는 다음과 같이 정의된다:

$$Adv^{HDH}(\theta,t) = \max_A \{Adv_A^{HDH}\}.$$

여기서 A는 시간 복잡도가 t이다. 만약 어드벤티지 함수가 네글리져블(negligible)[1]하면, HDH 문제는 어려운 문제이다. H가 SHA-1[22]와 같은 암호학적 해쉬 함수라면, HDH 문제는 어려운 문제라고 여겨진다.

2.2 ODH(Oracle Diffie-Hellman) 문제[20,21]

θ를 안전성 파라미터라고 하고 $H: \{0,1\}^* \rightarrow \{0,1\}^\theta$를 해쉬함수라고 하자. GG를 그룹 오더가 q인 그룹 G와 그룹 생성자 g를 생성하는 그룹 생성 알고리즘이라고 하자. 다음의 실험을 생각해 보자.

1) 어떤 함수 f가 네글리저블하면, f는 충분히 큰 값들에 대해, 모든 다항함수(polynomial)의 역수보다 더 빨리 감소한다. 즉, 모든 상수 c에 대해서, 다음을 만족하는 자연수 N이 존재한다: N보다 큰 모든 n에 대해서, $f(n) < \frac{1}{n^c}$.

$\mathrm{Exp}_A^{ODH}(\theta)$
$(G,q,g) \leftarrow GG(1^\theta)$
$(u_1,u_2) \leftarrow [1,q]$
$U_1 = g^{u_1}$; $U_2 = g^{u_2}$; $d \leftarrow [0,1]$
만약 $d=1$, $W = H(g^{u_1 u_2})$
그렇지 않으면, $W \leftarrow \{0,1\}^\theta$
$d' \leftarrow A^{H_{u_1}(\cdot)}(G,q,g,U_1,U_2,W)$

위 실험에서 A는 오라클 $H_{u_2}(\cdot)$에게 임의의 X를 쿼리해서 $H_{u_2}(X) = X^{u_2}$을 돌려받는다. 단, A는 U_1을 오라클 $H_{u_2}(\cdot)$에게 쿼리할 수 없다. 공격자 A의 어드벤티지는 다음과 같이 정의된다:

$$Adv_A^{ODH} = 2\Pr[d=d'] - 1.$$

어드벤티지 함수는 다음과 같이 정의된다:

$$Adv^{ODH}(\theta,t) = \max_A \{Adv_A^{ODH}\}.$$

여기서 A는 시간 복잡도가 t이다. 만약 어드벤티지 함수가 네글리져블하면, ODH 문제는 어려운 문제이다. 만약 H가 SHA-1[22]와 같은 암호학적 해쉬 함수라면, ODH 문제는 어려운 문제라고 여겨진다.

2.3 MAC(Message Authentication Code)

MAC 스킴 M은 두가지 알고리즘 (Mac, Vfy)로 구성된다. Mac는 주어진 메시지를 위한 MAC 값을 만드는 알고리즘이며, Vfy는 메시지와 MAC 값이 올바른가를 확인하는 알고리즘이다. θ를 안전성 파라미터라고 할 때, 다음의 실험을 생각해 보자.

$\mathrm{Exp}_{M,A}^{SUF}(\theta)$
$sk \leftarrow \{0,1\}^\theta$
$(M,\tau) \leftarrow A^{Mac_{sk}(\cdot),Vfy_{sk}(\cdot,\cdot)}(1^\theta)$
만약 $Vfy_{sk}(M,\tau)=1$이고 $Mac_{sk}(\cdot)$이 메시지 M에 대한 MAC값으로 τ를 반환한 적이 없다면, 1을 반환한다. 그렇지 않은 경우, 0을 반환한다.

공격자 A의 어드벤티지는 다음과 같이 정의된다:

$$Adv_{M,A}^{SUF} = \Pr[\mathrm{Exp}_A^{SUF}(\theta)=1].$$

어드벤티지 함수는 다음과 같이 정의된다:

$$Adv_M^{SUF}(\theta,t,q_s) = \max_A \{Adv_{M,A}^{SUF}\}.$$

여기서 A는 시간 복잡도가 t이며, 최대 q_s번의 오라클 쿼리를 던진다. 만약 어드벤티지 함수가 네글리져블하면, MAC 스킴 M은 SUF 안전성을 제공한다.

3. 키 교환의 안전성 모델

듀플렉스(duplex) 채널로 통신하는 두 사용자는 동시에 메시지를 주고받을 수 있다. 본 논문에서는 [18]에 나와 있는 듀플렉스 채널에서의 안전성 모델을 확장한다. [18]에서는 듀플렉스 채널 하에서 키 독립성(security against Known-Key)과 약한 전방위 안전성(Weak Forward Secrecy)만을 정의하고 있다. 본 논문에서는 [18]에서 정의된 안전성들 이외에 듀플렉스 채널 하에서 강한 전방위 안전성(Strong Forward Secrecy), 세션 상태 노출에 대한 안전성(security against Session State Reveal), 개인키 사용 위장에 대한 안전성(security against Key Compromise Impersonation)과 파트너 혼돈 공격에 대한 안전성(security against Unknown Key Share)을 제공하는 키 교환 스킴을 제안하는 것이 목적이므로 이러한 안전성들 증명에 필요한 안정성 모델을 부가적으로 정의한다.

안전성 모델에서 P_i는 사용자 ID를 의미하며, 각 P_i는 공개키-개인키 쌍을 가지고 있으며 키 교환 프로토콜에서 인증을 위해서 사용한다. 본 논문에서는 양자간 키 교환 프로토콜을 고려하며, 이를 위한 모델을 설명한다. 오라클 $\prod_i^k$는 P_i의 k번째 인스턴스를 의미한다. 키 교환 프로토콜이 종료하면 $\prod_i^k$는 세션키 sk_i^k를 생성한다. 세션 식별자 sid_i^k는 세션들 중에서 하나의 세션을 의미할 수 있는 스트링이다. sid_i^k는 $\prod_i^k$가 세션에서 보는 메시지들의 결합이라고 정의할 수 있다. 메시지들의 결합은 사용자들의 ID의 사전순서(lexicographic order)에 의해서 결합할 수 있다. (이 논문에서는 통신하는 두 파티가 동시에 메시지를 보낼 수 있으므로, 시간의 흐름상 먼저 나타나는 순서에 따라서 메시지를 결합 순서를 정할 수 없다.) 오라클의 *파트너*는 오라클이 통신하고 있다고 믿는 상대방의 ID를 의미한다. 만약 $sid_i^k = sid_j^{k'}$이며, $\prod_i^k$의 파트너가 P_j이고 $\prod_j^{k'}$의 파트너가 P_j이면, $\prod_i^k$와 $\prod_j^{k'}$은 *매칭*이라고 한다. 모든 안전한 프로토콜은 다음을 만족해야 한다: 만약 두 오라클이 매칭이면, 두 오라클은 같은 세션키를 계산해야 한다. 안전성 개념을 정의하기 위해서 먼저 공격자의 능력을 정의한다. 공격자가 일련의 오라클 쿼리(query)를 통해서 통신상에서 흐르는 모든 메시지의 흐름을 제어한다고 가정한다. 공격자의 성공 확률인 어드벤티지를 정의하기 위해서 실험을 수행하며, 이 실험에서 공격자는 오라클들에게 쿼리를 던지며, 오라클들은 이 쿼리에 응답한다. 오라클 쿼리들은 현실 세계에서 공격자가 행하는 공격을 모델링한다. 이 논문에서는 다음과 같은 쿼리들을 고려한다.

- Initiate(i,j): 이 쿼리는 P_i로 하여금 P_j와 키 교환 프로토콜을 수행하게 한다. P_i는 이 쿼리의 응답으로서 키 교환 프로토콜의 첫 번째 메시지를 공격자에게 돌려준다.
- Send(i,k,M): 이 쿼리는 메시지 M을 오라클 $\prod_i^k$에게 보낸다. $\prod_i^k$는 M을 받고서 프로토콜에 따라서 반응한다. 이 쿼리를 통해서 현실 세계에서의 능동적 공격(active attack)을 모델링할 수 있다.
- Reveal(i,k): 이 쿼리는 기존 세션의 세션키가 알려질 경우를 모델링한다. 즉, Denning-Sacco 공격을 모델링한다. 이 쿼리의 응답으로 세션키 sk_i^k를 돌려준다.
- Corrupt(i): 이 쿼리는 사용자 P_i의 비밀키가 노출되었을 경우를 모델링한다. 하지만 공격자는 P_i의 비밀키는 알 수 있더라도 공격자의 행동까지 컨트롤하지는 못한다. 물론 공격자는 이 비밀키를 가지고서 다른 사용자에게 P_i인척 할 수 있다(impersonation attack).
- State(i,k): 이 쿼리는 오라클 $\prod_i^k$의 세션 상태가 노출되었을 경우를 모델링한다. 이 쿼리의 응답으로 세션키 sk_i^k를 만들 때 사용되는 모든 난수값들을 돌려준다.
- Test(i,k) : 이 쿼리는 공격자의 성공 능력을 정의하기 위해서 사용한다. 공격자가 이 쿼리를 던지면 동전 던지기를 수행한다. 이 결과를 b라고 하자. 만약 b가 1이면 세션키 sk_i^k를 공격자에게 돌려주고, 아니면 길이가 θ-비트인 랜덤값을 뽑아서 돌려준다. 여기서 θ는 안전성 파라미터이다. 공격자는 하나의 Test 쿼리를 프레쉬(fresh) 오라클에게 던질수 있다. 이제 프레쉬 오라클을 정의한다.

정의 1. 실험이 끝날 때까지 다음의 조건들을 만족하는 오라클 $\prod_i^k$는 프레쉬하다.

(a) 공격자는 Reveal(i,k) 쿼리를 하지 않았다.

(b) 만약 $\prod_i^k$와 매칭되는 오라클인 $\prod_j^{k'}$가 존재하면, 공격자는 Reveal(j,k') 쿼리를 하지 않았다.

(c) $\prod_i^k$의 파트너는 *내부 공모자*[2]가 아니다.

다음은 키 교환 프로토콜이 얻을 수 있는 안전성들의 종류다.

(1) 키 독립성(security against Known-Key): 공격자는 Corrupt 쿼리나 State 쿼리는 할 수 없으나, Reveal 쿼리는 할 수 있다.

(2) 약한 전방위 안전성(Weak Forward Secrecy): 공격자는 State 쿼리는 할 수 없으나, 다음의 제약 조건하에서 Reveal 쿼리와 Corrupt 쿼리 둘 다 할 수 있다: P_j가 $\prod_i^k$의 파트너라 할 때, 만약 공격자가 Corrupt(i) 또는 Corrupt(j) 쿼리를 던졌다면, $\prod_i^k$와 매칭되는 오라클 $\prod_j^{k'}$가 존재해야만, 오라클 $\prod_i^k$는 프레쉬하다.

(3) 강한 전방위 안전성(Strong Forward Secrecy): 공격자는 State 쿼리는 할 수 없으나, 다음의 제약 조건하에서 Reveal 쿼리와 Corrupt 쿼리 둘 다 할 수 있다: 만약 공격자가 Corrupt(j)와 Send(i,k,*) 쿼리를 던졌다면, Corrupt(j)쿼리는 모든 Send(i,k,*) 쿼리 이후에 발생해야만, 오라클 $\prod_i^k$는 프레쉬하다.

(4) 세션 상태 노출에 대한 안전성(security against Session State Reveal): 공격자는 Reveal 쿼리와 Corrupt 쿼리는 할 수 없으나, State 쿼리는 할 수 있다.

(5) 개인키 사용 위장에 대한 안전성(security against Key Compromise Impersonation): 공격자는 State 쿼리는 할 수 없으나, 다음의 제약 조건하에서 Reveal 쿼리와 Corrupt 쿼리는 할 수 있다: $\prod_i^k$의 파트너 P_j가 Corrupt되지 않아야만, $\prod_i^k$는 프레쉬하다.

(6) 파트너 혼돈 공격에 대한 안전성(security against Unknown Key Share): 공격자는 Reveal 쿼리, Corrupt 쿼리, State 쿼리를 할 수 없다.

파트너 혼돈 공격에 대한 공격이 끝났을 경우, 공격자는 (i,k)와 (j,k')을 출력하고 멈춘다. 이 때 파트너 혼돈 공격에 대한 공격자의 어드벤티지는 $Adv_A^{UKS}(\theta)=\Pr[sk_i^k=sk_j^{k'} \wedge (P_j$가 $\prod_i^k$의 파트너가 아님 $\vee$ P_i가 $\prod_j^{k'}$의 파트너가 아님)]이 된다. 파트너 혼돈 공격 이외의 공격들에서는, 공격자는 실험이 끝날 때 b'을 출력한다. 공격자 A의 공격 성공 확률은 어드벤티지로 정의되며 다음과 같다: $Adv_A^{XX}(\theta)=2\Pr[b'=b]-1$. 여기서 XX는 KK(키 독립성), w-FS(약한 전방위 안전성), s-FS(강한 전방위 안전성), SSR(세션 상태 노출에 대한 안전성), KCI(개인키 사용 위장에 대한 안전성), UKS(파트너 혼돈 공격에 대한 안전성) 중에 하나이다. 프로토콜 P에 대해 공격자가 할 수 있는 공격 종류에 따라서 어드벤티지를 다음과 같이 표기한다: $Adv_P^{XX}(\theta,t)=\max_A\{Adv_A^{XX}(\theta)\}$. θ는 안전성 파라미터이며, t는 공격자에게 허용되는 시간이다. 만약 모든 다항시간 공격자들에 대해서 $Adv_P^{XX}(\theta,t)$가 네글리져블하다면 P는 XX-안전성을 보장한다고 말한다.

본 논문의 정의에서는 s-FS가 w-FS와 KCI를 내포

2) 내부 공모자들의 개인키와 공개키는 공격자에 의해서 생성되며, 그들의 내부 상태나 행동 방식도 공격자에 의해서 결정된다.

하며, w-FS가 KK를 내포한다. KK가 UKS를 내포한다는 것을 다음의 정리에서 증명한다.

정리 1. KK는 UKS를 내포한다.

정리 1 증명. UKS에 대한 공격자가 A가 (i,k)와 (j,k')을 출력하고 멈추었으며, 이 공격이 성공했다고 가정하자. 그러면 $sk_i^k = sk_j^{k'}$이 되고, $\prod_i^k$와 $\prod_j^{k'}$는 매칭이 아니다. 그러므로 공격자 A를 사용해서 KK를 성공적으로 공격하는 공격자 B를 다음처럼 만들 수 있다: (1) A가 멈춘 후, B는 Reveal(j,k') 쿼리를 던져서 $\prod_j^{k'}$의 세션키 $sk_j^{k'}$를 얻는다. (2) B는 Test(i,k) 쿼리를 던져서 δ를 얻는다. (3) B는 $sk_j^{k'} = \delta$이면 1을 출력하고, 그렇지 않으면 0을 출력한다.

A의 공격이 성공하면, B의 공격도 성공함을 알 수 있다. 따라서 어떤 프로토콜이 UKS를 제공하지 못하면 KK를 제공하지 못한다. □

4. 키교환 프로토콜

본 논문에서 제시하는 프로토콜은 다음을 가정한다. 사용자들은 그들의 ID에 의해서 사전 순서대로 정렬될 수 있으며, 만약 P_i가 P_j에 앞설 경우 $P_i < P_j$로 표기한다. θ는 안전성 파라미터이며, G는 소수 q를 오더(order)로 가지는 그룹이며, g는 그룹의 제너레이터(generator)이다. H는 임의의 스트링을 θ-비트 스트링으로 맵핑하는 해쉬 함수다. 각각의 사용자 P_i는 공개키-개인키 쌍 $(y_i = g^{x_i}, x_i)$를 가진다.

4.1 KAM

셋업 : P_i와 $P_j(\neq P_i)$가 세션키를 만들려 하며, $P_i < P_j$를 가정한다. P_i의 입장에서 프로토콜을 기술하지만, 파트너인 P_j도 유사하게 행동한다.

라운드 1: P_i는 난수 α_i를 Z_q로부터 뽑아서 $Z_i = g^{\alpha_i}$를 계산한 다음, P_j에게 Z_i를 보낸다.

라운드 2: P_i는 Z_j의 그룹 멤버쉽 테스트를 수행한다. 테스트가 성공적이면, P_i는 MAC 키인 $k_{i,j} = H(Z_j^{x_i}) = H(g^{x_i\alpha_j})$를 계산하며, $\tau_i = MAC_{k_{i,j}}(i\|j\|Z_i)$를 계산한 다음, P_j에게 τ_i를 보낸다.

세션키 계산: P_i는 $k_{j,i} = H(y_j^{\alpha_i}) = H(g^{x_j\alpha_i})$를 계산한 다음, $Vfy(\tau_j, j\|i\|Z_j) = 1$인지를 확인한다. 여기서 $k_{i,j} \neq k_{j,i}$이다. 만약 MAC값이 틀리다면, 세션키는 만들어지지 않는다. MAC 값이 올바르면, P_i는 세션키 $sk = H(y_j^{x_i}) \oplus H(Z_j^{\alpha_i}) = H(g^{x_jx_i}) \oplus H(g^{\alpha_i\alpha_j})$를 만든다.

	$P_1(x_1, y_2)$	$P_2(x_2, y_1)$
라운드 1	g^{α_1}	g^{α_2}
라운드 2	τ_1	τ_2

$k_{1,2} = H(g^{\alpha_2x_1}); k_{2,1} = H(g^{\alpha_1x_2})$(여기서 $k_{1,2} \neq k_{2,1}$)

$\tau_1 \leftarrow Mac_{k_{1,2}}(1\|2\|g^{\alpha_1}); \tau_2 \leftarrow Mac_{k_{2,1}}(2\|1\|g^{\alpha_2})$

$sk = H(g^{x_1x_2}) \oplus H(g^{\alpha_1\alpha_2})$

그림 1 KAM의 실행 예

KAM의 실행 예가 그림 1에 나와 있다.

다음의 정리는 KAM의 안전성에 관한 것이다.

정리 2. 만약 MAC 스킴이 위조 불가능성을 만족하고, G에서 ODH 문제와 HDH 문제가 어려운 문제면, KAM은 s-FS와 SSR을 보장한다. KAM이 s-FS를 제공하면, KAM은 w-FS, KCI, KK, UKS를 제공한다 (앞 절 참조).

정리 2 증명. 정리 2의 증명을 다음의 Lemma 1과 Lemma 2에 의해서 증명한다. Lemma 1에서는 KAM이 전방위 안전성을 제공함을 증명하며, Lemma 2에서는 KAM이 세션 상태 노출 공격에 안전함을 증명한다. □

Lemma 1. 만약 MAC 스킴이 위조 불가능성을 만족하고, G에서 ODH 문제와 HDH 문제가 어려운 문제면, KAM은 s-FS를 보장한다. 좀 더 구체적으로,

$$Adv_{KAM}^{s-FS}(\theta,t) \le \frac{2q_s^2}{q} + 2N^2q_s \cdot Adv_{MI}^{SUF}(\theta,t) + 2N^2q_s \cdot Adv^{ODH}(\theta,t) + (Nq_s)^2 \cdot Adv^{HDH}(\theta,t),$$

여기서 θ는 안전성 파라미터이며, t는 공격자의 실행 시간을 포함한 총 실험시간이다. N은 실험에서 *정직한 사용자*[3]의 최대 수이며, q_s는 실험에서 최대 세션의 수이다.

Lemma 1 증명. Lemma 1에서 어떤 공격자가 전방위 안전성을 깨트리면, 그 공격자를 이용해서 MAC 스킴을 깨거나, ODH 문제나 HDH 문제를 풀 수 있음을 보인다.

g^{α_i}가 두 번 반복되는 사건을 col이라고 하자. 그리고 MAC 위조가 발생한 사건을 forge라고 하자. 그러면

$Pr_A[b' = b] \le Pr_A[col] + Pr_A[forge]$

$+ Pr_A[b' = b \wedge \overline{col} \wedge \overline{forge}]$ 임을 알 수 있다.

다음과 같은 세 개의 Claim들에 의해서 위의 확률들의 최대값을 제한할 수 있다.

3) 내부 공모자가 아닌 사용자를 *정직한 사용자*라 한다.

Claim 1. $\Pr_A[col] \le \frac{q_s^2}{q}$.

Claim 2. $Pr_A[forge] \le N^2 q_s \cdot (Adv^{ODH} + Adv_M^{SUF})$.

Claim 3.

$$Pr_A[b' = b \wedge \overline{col} \wedge \overline{forge}] \le \frac{(Nq_s)^2 \cdot Adv^{HDH} + 1}{2}.$$

위의 클레임들로부터 다음을 알 수 있다.

$$Pr_A[b' = b] \le \frac{q_s^2}{q} + N^2 q_s \cdot (Adv^{ODH} + Adv_M^{SUF}) + \frac{(Nq_s)^2 \cdot Adv^{HDH} + 1}{2}.$$

따라서 식 $Adv^{s-FS}_{KAM,A} = 2\Pr_A[b' = b] - 1$으로부터 Lemma 1이 증명된다. □

이제 다음 세 개의 Claim들을 차례대로 증명한다.

Claim 1 증명. 구간 $[1,q]$ 사이에서 똑같은 난수를 뽑을 확률은 잘 알려진 "생일 패러독스(birthday paradox)"에 의해서 $\frac{q_s^2}{q}$ 로 제한된다. □

Claim 2 증명. 먼저 일련의 연속된 게임들인 ($Game_0$,

$Game_{1,1,2}, Game_{1,1,3}, \ldots, Game_{1,1,N}$,

$Game_{1,2,2}, Game_{1,2,3}, \ldots, Game_{1,2,N}, \ldots$,

$Game_{1,q_s,2}, Game_{1,q_s,3}, \ldots, Game_{1,q_s,N}$,

$Game_{2,1,1}, Game_{2,1,3}, \ldots, Game_{2,1,N}, \ldots$,

$Game_{2,q_s,1}, Game_{2,q_s,3}, \ldots, Game_{2,q_s,N}, \ldots$,

$Game_{N,q_s,1}, Game_{N,q_s,2}, \ldots, Game_{N,q_s,N-1}$)[4]를 다음과 같이 정의한다.

- $Game_0$에서는 제안한 프로토콜인 KAM을 그대로 따라서 공격자의 쿼리들에 응답한다.
- $Game_{1,1,2}$에서는 만약 $\prod_1^1$의 파트너가 P_2이면, $\prod_1^1$에서 사용되는 MAC 키인 $k_{2,1}$을 θ-비트 스트링 공간에서 랜덤하게 뽑은 $k'_{2,1}$로 대체한다.
- $Game_{i,\ell,j}$는 이 게임의 바로 이전 게임과 다음을 제외하면 똑같다: 만약 $\prod_i^\ell$의 파트너가 P_j이면, $\prod_i^\ell$에서 사용되는 MAC 키인 $k_{j,i}$을 θ-비트 스트링 공간에서 랜덤하게 뽑은 $k'_{j,i}$로 대체한다.

Claim 2는 다음의 Claim 2.1과 Claim 2.2로부터 증명된다.

Claim 2.1 $\Pr[\text{forge in } Game_0] - \Pr[\text{forge in } Game_{N,q_s,N-1}] \le N^2 q_s Adv^{ODH}$.

Claim 2.2 $\Pr[\text{forge in } Game_{N,q_s,N-1}] \le N^2 q_s Adv_M^{SUF}$. □

이제 Claim 2.1과 Claim 2.2를 증명한다.

Claim 2.1 증명. $Game_{i',\ell',j'}$과 $Game_{i^*,\ell^*,j^*}$을 두 개의 연속된 게임이라고 하자. 만약 이 두개의 연속된 게임에서 사건 forge가 일어날 확률의 차이가 네글리져블하지 않으면, ODH 문제를 푸는 알고리즘 D를 만들 수 있음을 보인다. 먼저 $\Pr[\text{forge}] = \Pr[\text{forge} | \prod_{i^*}^{\ell^*}\text{의 파트너} = P_{j^*}] + \Pr[\text{forge} | \prod_{i^*}^{\ell^*}\text{의 파트너} \neq P_{j^*}]$가 된다. 만약 $\prod_{i^*}^{\ell^*}$의 파트너$\neq P_{j^*}$이면, $Game_{i',\ell',j'}$과 $Game_{i^*,\ell^*,j^*}$은 똑같은 게임이 되므로 forge가 일어날 확률도 같게 된다. 따라서 만약 두 게임에서 forge의 발생 확률 차이가 네글리져블하지 않다면, 두 게임에서 $\Pr[\text{forge} | \prod_{i^*}^{\ell^*}\text{의 파트너} = P_{j^*}]$의 차이가 네글리져블하지 않음을 알 수 있다. 이 확률 차이를 사용해서 ODH 문제를 푸는 알고리즘 D를 다음과 같이 구성할 수 있다.

1. D는 (G, q, g, U_1, U_2, W)가 주어지며, $H_{u_2}(\cdot)$ 오라클이 주어진다. D는 U_2를 P_{j^*}의 공개키로 사용한다. P_{j^*}를 제외한 다른 파티를 위해서는 정상적으로 개인키-공개키 쌍을 만든다.
2. 공격자 A의 쿼리들에 대해서, D는 다음을 제외하고는 $Game_{i',\ell',j'}$에서와 같이 A의 쿼리들에 응답한다:
   - Initiate 쿼리와 Send 쿼리:
     - D는 U_1을 $\prod_{i^*}^{\ell^*}$의 일회용 Diffie-Hellman 메시지로 사용한다.
     - 만약 P_{j^*}가 $\prod_{i^*}^{\ell^*}$의 파트너이면, D는 W를 MAC 키인 k_{j^*,i^*}로 사용하고, $H(y_{i^*}^{\alpha_{i^*}'})$를 MAC 키인 k_{i^*,j^*}로 사용한다.
     - 만약 P_{j^*}가 $\prod_{i^*}^{\ell^*}$ 이외의 인스턴스의 파트너이면, D는 $H_{u_2}(Z_i)$를 MAC 키인 $k_{j^*,i}$로 사용하고, $H(y_i^{\alpha_i'})$를 MAC 키인 k_{i,j^*}로 사용한다.
   - Reveal(i,k) 쿼리와 Test(i,k) 쿼리:
     - 만약 P_{j^*}가 $\prod_{i^*}^{\ell^*}$의 파트너이면, D는 세션키 $sk_{i^*,j^*} = H(U_2^{\alpha_{i^*}'}) \oplus H(U_1^{\alpha_{i^*}'})$를 만든다.
     - 만약 P_{j^*}가 $\prod_{i^*}^{\ell^*}$ 이외의 인스턴스의 파트너이면, D는 세션키 $sk_{i,j^*} = H_{u_2}(y_i) \oplus H(Z_i^{\alpha_i'})$를 만든다.
   - Corrupt(i) 쿼리: 만약 $i = j^*$이면, D는 알고리즘을 멈춘다.
3. 만약 사건 forge가 발생하면, D는 1을 출력하고 멈춘다. 만약 공격자 A의 실행이 끝나도 forge가 일어나지 않으면, D는 0을 출력하고 멈춘다.

4) $Game_{i,*,i}$는 정의되지 않는다.

D는 입력값인 $(U_1=g^{u_1}, U_2=g^{u_2}, W)$에서 W가 $H(g^{u_1u_2})$인지 아닌지에 따라서, $Game_{i',\ell',j'}$ 또는 $Game_{i^*,\ell^*,j^*}$을 시뮬레이션하게 된다. 따라서 다음과 같은 등식이 성립한다.

$$\begin{aligned}Adv_D^{HDH} &= \Pr[D^{H_k}(U_1,U_2,W)=1 \mid \\ &\qquad U_1=g^{u_1}, U_2=g^{u_2}, W=H(g^{u_1u_2})] \\ &\quad - \Pr[D^{H_k}(U_1,U_2,W)=1 \mid \\ &\qquad U_1=g^{u_1}, U_2=g^{u_2}, W=\{0,1\}^\theta]\end{aligned}$$

$= \Pr[\text{forge in } Game_{i',\ell',j'} \wedge \prod_{i^*}^{\ell^*} \text{의 파트너}=P_{j^*}]$

$\quad - \Pr[\text{forge in } Game_{i^*,\ell^*,j^*} \wedge \prod_{i^*}^{\ell^*} \text{의 파트너}=P_{j^*}]$

$= \Pr[\text{forge in } Game_{i',\ell',j'}]$

$\quad - \Pr[\text{forge in } Game_{i^*,\ell^*,j^*}]$.

이로부터 표준적인 하이브리드 논법(hybrid argument)에 의해서 Claim 2.1이 증명된다. □

Claim 2.2 증명. 만약 $Game_{N,q_s,N-1}$에서 Pr[forge]가 네글리져블하지 않으면, MAC 스킴을 깨는 알고리즘 F를 만들 수 있음을 보인다. 알고리즘 F는 다음과 같다.

1. F는 $Mac_{sk}(\cdot)$ 오라클과 $Vfy_{sk}(\cdot)$오라클이 주어진다. F는 정직한 사용자를 위해서 정상적으로 개인키-공개키 쌍을 만든다. F는 i^*,j^*를 구간 $[1,N]$에서 랜덤하게 뽑고, ℓ^*를 구간 $[1,q_s]$에서 랜덤하게 뽑는다.
2. 공격자 A의 쿼리들에 대해서, F는 다음을 제외하고는 $Game_{N,q_s,N-1}$에서와 같이 A의 쿼리들에 응답한다:
   - Send 쿼리: F는 만약 P_{j^*}가 $\prod_{i^*}^{\ell^*}$의 파트너이고 $\prod_{i^*}^{\ell^*}$가 k'_{j^*,i^*}를 가지고 MAC을 생성하거나 확인해야 할 경우, F는 $Mac_{sk}(\cdot)$ 오라클과 $Vfy_{sk}(\cdot)$ 오라클을 사용한다.
3. 만약 P_{j^*}가 $\prod_{i^*}^{\ell^*}$의 파트너이고 $\prod_{i^*}^{\ell^*}$의 k'_{j^*,i^*}에 대해서 사건 forge가 발생하면, F는 위조된 MAC을 출력하고 멈춘다. 그렇지 않으면 F는 그냥 멈춘다.

만약 F가 i^*,j^*,ℓ^*를 올바르게 추측하면, F는 실패하지 않으므로, F의 성공 확률은 다음과 같다:

$$Adv_{M,F}^{SUF} \geq \frac{1}{N^2 q_s}\Pr[\text{forge in } Game_{N,q_s,N-1}].$$

따라서 Claim 2.2가 증명된다. □

Claim 3 증명. Claim 3를 증명하기 위해서, 만약 사건 col이나 forge없이 공격자 A가 KAM의 전방위 안전성을 깬다면, 이를 이용해서 HDH 문제를 푸는 알고리즘 D를 만들 수 있음을 보인다.

1. D는 (G,q,g,U_1,U_2,W)가 주어진다. D는 정직한 사용자를 위해서 정상적으로 개인키-공개키 쌍을 만든다. D는 i^*,j^*를 구간 $[1,N]$에서 랜덤하게 뽑고, t_1,t_2를 구간 $[1,q_s]$에서 랜덤하게 뽑는다. $i^*<j^*$라고 가정한다.
2. 공격자 A의 쿼리들에 대해서, D는 다음을 제외하고는 KAM에 의해서 A의 쿼리들에 응답한다:
   - Initiate 쿼리: D는 U_1을 $\prod_{i^*}^{t_1}$의 일회용 Diffie-Hellman 메시지로 사용하며, U_2을 $\prod_{j^*}^{t_2}$의 일회용 Diffie-Hellman 메시지로 사용한다. U_1을 위한 MAC 값을 τ_1이라 하고, U_2를 위한 MAC 값을 τ_2라 하자.
   - Reveal(i,k) 쿼리: 만약 $\prod_i^k$가 $\prod_{j^*}^{t_1}$이거나 $\prod_{j^*}^{t_2}$이면, D는 그냥 멈춘다.
   - Test(i,k) 쿼리: 만약 $\prod_i^k$가 $\prod_{j^*}^{t_1}$이거나 $\prod_{j^*}^{t_2}$이고, $sid_i^k = U_1\|U_2\|\tau_1\|\tau_2$이면 D는 $sk=H(g^{x_{i'}x_{j'}})\oplus W$를 A에게 돌려준다.
3. A가 b'을 출력하고 멈추면, D도 b'을 출력하고 멈춘다.

D는 입력값인 $(U_1=g^{u_1}, U_2=g^{u_2}, W)$에서 W가 $H(g^{u_1u_2})$인지 아닌지에 따라서, A에게 "Test쿼리에서 sk가 KAM을 따라서 만들어지는 게임"을 시뮬레이션하거나 또는 "Test쿼리에서 sk가 랜덤하게 뽑히는 게임"을 시뮬레이션하게 된다. 따라서 D가 i^*,j^*,t_1,t_2를 올바르게 추측했는지의 여부에 따라 다음과 같은 등식이 성립한다.

$$\begin{aligned}&Adv_D^{HDH} = \\ &\Pr[D(U_1,U_2,W)=1 \mid U_1=g^{u_1}, U_2=g^{u_2}, W=H(g^{u_1u_2})] - \\ &\Pr[D(U_1,U_2,W)=1 \mid U_1=g^{u_1}, U_2=g^{u_2}, W=\{0,1\}^\theta] \\ &\geq \frac{1}{(Nq_s)^2}(\Pr[A()=1 \mid \text{Test쿼리에서 } sk\text{가 KAM을 따라서 만들어짐}] - \\ &\Pr[A()=1 \mid \text{Test쿼리에서 } sk\text{가 랜덤하게 뽑힘}]) = \frac{1}{(Nq_s)^2}(2\Pr_A[b'=b \wedge \overline{col} \wedge \overline{forge}]-1).\end{aligned}$$

이로부터 Claim 3이 증명된다. □

Lemma 2. 만약 G에서 HDH 문제가 어려운 문제면, KAM은 SSR을 보장한다. 좀 더 구체적으로,

$$Adv_{KAM}^{SSR}(\theta,t) \leq N^2 \cdot Adv^{HDH}(\theta,t) \text{ 이다.}$$

여기서 θ는 안전성 파라미터이며, t는 공격자의 실행 시간을 포함한 총 실험시간이다. N은 실험에서 정직한 사용자의 최대 수이다.

Lemma 2 증명. Lemma 2를 증명하기 위해서, 공격자 A가 KAM의 SSR 안전성을 깬다면, 이를 이용해서 HDH 문제를 푸는 알고리즘 D를 만들 수 있음을 보인다.

1. D는 (G,q,g,U_1,U_2,W)가 주어진다. D는 i^*,j^*를 구간 $[1,N]$에서 랜덤하게 뽑고, U_1을 P_{i^*}의 공개키인 y_{i^*}로 사용하고, U_2를 P_{j^*}의 공개키인 y_{j^*}로 사용한다. D는 P_{i^*}와 P_{j^*}를 제외한 다른 모든 정직한 사용자들을 위해

서 정상적으로 개인키-공개키 쌍을 만든다.

2. 공격자 A의 쿼리들에 대해서, D는 다음을 제외하고는 KAM에 의해서 A의 쿼리들에 응답한다:
   - Test(i,k) 쿼리: $\prod_i^k$의 파트너를 P_j라고 하자. 만약 $\{P_i, P_j\} \neq \{P_{i^*}, P_{j^*}\}$이면 D는 그냥 멈춘다. $\{P_i, P_j\} = \{P_{i^*}, P_{j^*}\}$이면 D는 $sk = W \oplus H(g^{\alpha_{i^*}\alpha_{j^*}})$를 A에게 돌려준다.
3. A가 b'을 출력하고 멈추면, D도 b'을 출력하고 멈춘다. D는 입력값인 $(U_1 = g^{u_1}, U_2 = g^{u_2}, W)$에서 W가 $H(g^{u_1 u_2})$인지 아닌지에 따라서, A에게 "Test쿼리에서 sk가 KAM을 따라서 만들어지는 게임"을 시뮬레이션하거나 또는 "Test쿼리에서 sk가 랜덤하게 뽑히는 게임"을 시뮬레이션하게 된다. 따라서 D가 i^*, j^*를 올바르게 추측했는지의 여부에 따라 다음과 같은 등식이 성립한다.

$Adv_D^{HDH} =$

$\Pr[D(U_1, U_2, W) = 1 | U_1 = g^{u_1}, U_2 = g^{u_2}, W = H(g^{u_1 u_2})] -$

$\Pr[D(U_1, U_2, W) = 1 | U_1 = g^{u_1}, U_2 = g^{u_2}, W = \{0,1\}^\theta]$

$\geq \frac{1}{N^2}$(Pr[A()=1 | Test쿼리에서 sk가 KAM을 따라서 만들어짐] - Pr[A()=1 | Test쿼리에서 sk가 랜덤하게 뽑힘])

$= \frac{1}{N^2} Adv_A^{SSR}$.

이로부터 Lemma 2가 증명된다.

5. 결 론

본 논문에서는 키 교환 프로토콜인 KAM을 제안하며 안전성을 증명한다. KAM은 스탠다드 모델에서 안전성 증명이 가능하며, 강한 전방위 안전성과 세션 상태 노출에 대한 안전성 등 HMQV-C가 제공하는 모든 안전성을 제공한다. KAM과 세션 상태 노출에 안전한 프로토콜들 간의 비교가 표 1에 나와 있다.

표 1 세션 상태 노출에 안전한 프로토콜들 간의 비교

스킴	안전성	계산량	모델
2-message HMQV [3]	- 약한 전방위 안전성 - 개인키 사용 위장에 대한 안전성 - 세션 상태 노출에 대한 안전성	3.5 지수승	랜덤 오라클
3-message HMQV-C [3]	- 강한 전방위 안전성 - 세션 상태 노출에 대한 안전성	3.5 지수승	랜덤 오라클
KAM (제안 프로토콜)	- 강한 전방위 안전성 - 세션 상태 노출에 대한 안전성	6 지수승	스탠다드

* 강한 전방위 안전성은 약한 전방위 안전성, 개인키 사용 위장에 대한 안전성, 키 독립성, 파트너 혼돈 공격에 대한 안전성을 내포한다.

참 고 문 헌

[1] Denning, D. and Sacco, G. M., "Timestamps in Key Distribution Protocols," Comm. ACM, Vol.24, No.8, pp. 533-536, 1981.

[2] Canetti, R. and Krawczyk, H., "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels," EUROCRYPT 2001, LNCS 2045, pp. 453-474, 2001.

[3] Krawczyk, H., "HMQV: A High-Performance Secure Diffie-Hellman Protocol," CRYPTO '05, LNCS 3621, pp. 546-566, 2005.

[4] Law, L., Menezes, A., Qu, M., Solinas, J., Vanstone, S., "An Efficient Protocol for Authenticated Key Agreement," Designs Codes and Cryptography, Vol.28, pp. 119-134, 2003.

[5] Menezes, A., Qu, M., Vanstone, S., "Some new key agreement protocols providing mutual implicit authentication," SAC '95, pp. 22-32, 1995.

[6] American National Standard (ANSI) X9.42-2001. Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography".

[7] American National Standard (ANSI) X9.63. Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography.

[8] IEEE 1363-2000: Standard Specifications for Public Key Cryptography.

[9] ISO/IEC IS 15946-3 Information technology-Security techniques: Cryptographic techniques based on elliptic curves-Part 3: Key establishment, 2002.

[10] NIST Special Publication 800-56 (DRAFT): Recommendation on Key Establishment Schemes. Draft 2, Jan. 2003.

[11] NSAs Elliptic Curve Licensing Agreement, presentation by Mr. John Stasak (Cryptography Office, National Security Agency) to the IETF's Security Area Advisory Group, Nov 2004. http://www.machshav.com/~smb/saag-11-2004/NSA-EC-License.pdf

[12] Krawczyk, H., "HMQV: A High-Performance Secure Diffie-Hellman Protocol," Full version of [15], in: eprint.iacr.org/2005/176, 2005.

[13] Bellare, M., Boldyreva, A., Palacio, A., "An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem," EUROCRYPT 2004, LNCS 3027, pp. 171-188. 2004.

[14] Canetti, R., Goldreich, O., Halevi, S., "The random oracle methodology, revisited," STOC '98, ACM, pp. 209-218, 1998.

[15] Goldwasser, S. and Tauman, Y., "On the (In)security of the Fiat-Shamir Paradigm," FOCS '03, pp.102, 2003.

[16] Canetti, R. and Krawczyk, H., "Security Analysis of IKE's Signature-Based Key-Exchange Protocol,"

CRYPTO '02, LNCS 2442, pp. 143-161, 2002.
[17] Diffie, W., Oorschot, P. C. van, Wiener, M. J., "Authentication and Authenticated Key Exchanges," Designs, Codes and Cryptography, Vol.2, pp. 107-125, 1992.
[18] Jeong, I. R., Katz, J., Lee. D. H., "One-Round Protocols for Two-Party Authenticated Key Exchange," ACNS '04, LNCS 3089, pp.220-232, 2004.
[19] Shoup. V., "On Formal Models for Secure Key Exchange," Available at http://eprint.iacr.org.
[20] Abdalla, M., Bellare, M., Rogaway, P., "DHAES: an encryption scheme based on the Diffie-Hellman problem," Submission to IEEE P1363, 1998.
[21] Abdalla, M., Bellare, M., Rogaway, P., "The oracle Diffie-Hellman assumption and an analysis of DHIES," CT-RSA '01, LNCS 2020, pp.143-158, 2001.
[22] Secure hash standard. National Institute of Standards and Technology, NIST FIPS PUB 180-1, U.S. Department of Commerce, Apr. 1995.

정 익 래

1998년 2월 고려대학교 전산학과(학사) 2000년 2월 고려대학교 전산학과(석사) 2004년 8월 고려대학교 정보보호대학원(박사). 2006년 6월~2008년 2월 한국전자통신연구원 암호기술연구팀 선임연구원. 2008년 3월~현재 고려대학교 정보경영공학부 조교수. 관심분야는 암호프로토콜, 암호이론, 계산이론

권 정 옥

2000년 8월 동덕여자대학교 전자계산학과(학사). 2003년 2월 고려대학교 정보보호기술협동과정(석사). 2007년 2월 고려대학교 정보경영공학전문대학원(박사). 2007년 3월~2007년 8월 고려대학교 정보보호기술연구센터 박사후연구원. 2007년 9월~현재 고려대학교 BK21 유비쿼터스 정보보호 사업단 연구교수. 관심분야는 암호프로토콜, 암호이론

이 동 훈

1983년 8월 고려대학교 경제학과(학사) 1987년 12월 Oklahoma University 전산학과(석사). 1992년 5월 Oklahoma University 전산학과(박사). 1993년 3월~1997년 2월 고려대학교 전산학과 조교수. 1997년 3월~2001년 2월 고려대학교 전산학과 부교수. 2001년 2월~현재 고려대학교 정보경영공학전문대학원 교수. 관심분야는 암호프로토콜, RFID/USN 보안, 프라이버시 보호기술

홍 도 원

1994년 2월 고려대학교 수학과(학사). 1996년 2월 고려대학교 수학과(석사). 2000년 2월 고려대학교 수학과(박사). 2000년 4월~현재 한국전자통신연구원 암호기술연구팀 팀장. 관심분야는 암호프로토콜, 암호이론, 프라이버시 보호기술