인증 및 키 합의를 위한 무인증서 서명 기술*

김준식*, 엄지은*, 이동훈*
*고려대학교 정보보호대학원
e-mail: ha29re@korea.ac.kr

Certificateless Signature for Authentication and Key Agreement

Joon Sik Kim*, Jieun Eom*, Dong Hoon Lee*
*Graduate School of Information Security, Korea University

요 약

사용자 인증 및 키 합의 프로토콜은 두 사용자의 안전한 통신에 필수적인 세션키를 생성하는 프로토콜이다. 전자서명과 디피-헬만(Diffie-Hellman) 키 합의 프로토콜을 이용하여 인증 및 키 합의를 수행할 수 있으나, 각각의 파라미터 정보를 모두 공유해야한다는 단점이 있다. 이에 ID 기반 서명을 이용하여 인증과 키 합의를 동시에 수행할 수 있는 프로토콜이 제안되었는데, 기본적으로 ID 기반 서명은 키 위탁(key escrow) 문제가 있다. 본 논문에서는 이러한 문제를 해결하기 위해 상호 인증과 키 합의를 동시에 수행할 수 있는 무인증서 (certificateless, CL) 서명 기법을 설계하고, 이를 이용한 인증 및 키 합의 프로토콜을 제안한다.

1. 서론

개체 인증 및 키 합의 프로토콜(authentication and key agreement protocol)은 통신하고자 하는 상대의 신원을 파악하여 안전한 통신을 가능하게 하는 프로토콜로써 서로의 공개키를 이용해서 상대방을 인증하고, 인증이 완료되면 암호화 통신을 위한 비밀키를 생성한다. 일반적인 서명을 이용하여 인증을 수행하는 경우, 키 합의를 위한 추가적인 정보 공유가 필요하기 때문에 인증 및 키 합의가 각각 독립적으로 수행된다. 한 예로, 이산대수(discrete logarithm, DL) 구조를 가지는 서명의 경우에는 디피-헬만(Diffie-Hellman) 키 합의를 위한 추가적인 정보 공유가 필요하지 않지만, RSA 구조를 가지는 서명의 경우에는 사용하는 군(group)의 형태가 다르기 때문에 두 종류의 파라미터 정보가 필요하게 된다. 따라서, 인증을 수행하는 단계에서 생성되는 정보를 이용하여 키 합의를 한 번에 수행할 수 있다면, 추가적인 정보 공유 없이 보다 효율적인 인증 및 키 합의가 가능하다.

RSA 구조의 ID 기반 서명을 이용한 인증 및 키 합의 프로토콜은 서명 기법의 파라미터 내에서 키 합의를 위한 군 정보를 추가적으로 생성함으로써 가능해졌다 [6.3.13]. 그러나 기본적으로 ID 기반 암호 시스템은 키 생성기관(Key Generation Center, KGC)이 모든 키를 관리하는 키 위탁(key escrow) 문제가 존재한다. ID 기반 암호의 키 위탁 문제에 대한 하나의 해결책인 Certificateless(CL) 암호 시스템을 이용한 서명을 이용한다면, 더 높은 안전성을 가지는 인증 및 키 합의 프로토콜을 설계할 수 있다. 본 논문에서는 RSA 구조의 CL 서명을 기반으로 하는 사용자 인증 및 키 합의 프로토콜을 제안한다.

* 이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2015-0-00320, 계층적 식별자를 가진 인터넷 개체의 공개키 인증 구조 연구)

2. 관련 연구

1989년 Okamoto와 Tanaka는 RSA 구조의 ID 기반 인증 및 키 합의 프로토콜을 제시하였다[6]. 이후 2010년 Gennaro 등은 Okamoto-Tanaka 프로토콜의 취약점을 보완하는 기법을 제안하고 안전성을 표준 모델에서 분석하였다[3]. 두 기법은 서명의 형태를 그대로 가져가지 않고, 전송된 값에 대한 검증 단계를 거치지 않기 때문에 중간에 공격자가 전송되는 값을 변조하게 되면 통신하는 두 사용자는 서로 다른 세션키를 생성하게 된다[5]. 따라서 키 합의을 위한 정보를 교환할 때 반드시 서명의 구조를 따라야 한다. 이에 Eom 등은 Okamoto-Tanaka 프로토콜과 유사한 방식으로 인증 및 키 합의를 수행할 수 있는 RSA 구조의 ID 서명을 제안하였다[13]. 그러나 기본적으로 ID 기반 암호 시스템에서는 키 생성기관이 모든 사용

자의 비밀키(서명키, 복호화키 등)를 알고 있다는 한계가 있기 때문에, 사용자가 자신만의 비밀 정보를 생성하여 서명 생성 또는 복호화에 이용하는 Certificateless(CL) 암호 또는 서명을 기반으로 인증 및 키 합의를 수행하는 것이 보다 높은 안전성을 제공할 수 있다.

CL 서명은 2003년에 Al-Riyami 와 Paterson이 처음 제시한[1] 이후로 CL 서명에 대한 여러 연구가 진행되어 왔다. 2004년 Yum과 Lee는 CL 서명을 일반적으로 설계하는 방법(generic construction)을 제시했고[10], 2006년 Zhang 등은 페어링(pairing)을 이용해 CL 서명을 설계하는 방법을 발표했다[12]. 2012년 Zhang과 Mao는 페어링을 사용하지 않고, RSA 기반의 효율적인 CL 서명을 제시하였다[11]. 이후, He 등은 Zhang과 Mao의 기법이 공개키 교체(Replace public key) 공격에 취약함을 지적하고[4], 이를 바탕으로 Sharma 등은 향상된 기법을 제안하였다[8]. 그러나 해당 공격은 공개키 e가 작아서 지수승 연산에 사용되는 값들을 나눌 수 있을 때 가능한 공격이며, 이는 e의 크기를 충분히 크도록 하면 바로 해결된다. 특히, CL 서명에서 기반하고 있는 ID 기반 서명의 안전성은 이미 해당 지수값이 e보다 작다는 가정 하에 암호학적으로 증명이 되어 있기 때문에, 실제로 지적한 문제점은 발생하지 않는다.

3. 배경지식

3.1 암호학적 가정

RSA 문제와 이산 대수(Discrete Logarithm, DL) 문제에 따른 암호학적 가정은 다음과 같이 정의된다.

- **RSA 문제** : 소수 p,q에 대해 $n=pq$와 위수를 n으로 가지는 군 G의 임의의 원소 z가 있을 때, $z=u^e$인 u를 찾는 문제이다.
- **RSA 가정** : 다항식 시간 안에 RSA 문제를 푸는 알고리즘 A가 존재할 때, 문제는 푸는 A의 이점(advantage)이 충분히 작다면(negligible) RSA 문제는 풀기 어렵다고 정의한다. 이때 A의 이점을 다음과 같이 정의한다.

$$Adv_A^{RSA}=|\Pr[A(n,e,z)=u]=1|\le\epsilon$$

- **DL 문제** : 두 소수 $p=2p'+1, q=2q'+1$과 $n=pq$, 그리고 위수가 $p'q'$인 Z_n^*의 생성원 g에 대하여 (g,y,n)이 주어졌을 때 $y=g^x \bmod n$인 x를 찾는 문제이다.
- **DL 가정** : 다항식 시간 안에 DL 문제를 풀어내는 알고리즘 A가 존재할 때, 문제를 풀어내는 A의 이점이 충분히 작다면(negligible) DL 문제는 풀기 어렵다고 정의한다. 이때 A의 이점을 다음과 같이 정의한다.

$$Adv_A^{DL}=|\Pr[A(g,y,n)=x]=1|\le\epsilon$$

3.2 CL 서명 시스템 모델

CL 서명 시스템에는 키 생성기관(Key Generation Center, KGC)과 사용자가 참여하고, 동작하는 과정은 다음과 같다. KGC는 시스템 셋업(Setup) 단계에서 공개파라미터(public parameters, pp)와 마스터키(master key, msk)를 생성하고, 키 발급(KeyGen) 단계에서 사용자의 *ID*를 이용해 개인키 sk_{ID}를 생성하고 발급한다. 사용자는 자신이 선택한 비밀값 s_{ID}(SetSec)를 이용하여 공개키 pk_{ID}를 등록(SetPub)하고, 발급받은 개인키와 자신의 비밀 정보를 이용해 서명키 sk_{ID}를 생성한다. 서명키를 가진 사용자는 메시지 m에 대한 서명을 생성(Sign)할 수 있다. 서명 검증(Verify)을 위해서는 메시지 m과 사용자의 *ID*, 그리고 공개키를 이용하여 검증식 만족 여부를 확인한다.

3.3 CL 서명 안전성 모델

일반적으로 서명에 대한 안전성은 서명키를 모르는 공격자가 임의의 메시지에 대한 정당한/유효한 서명을 생성(위조)하는 것이 어렵다는 것(existential unforgeability)을 보임으로써 증명된다. CL 서명에서는 KGC와 사용자 모두 서명키를 생성하는 과정에 관여하기 때문에 일반적인 공격자를 의미하는 Type 1 (T1) 공격자와 악의적인 KGC를 의미하는 Type 2 (T2) 공격자의 두 가지 공격자 유형이 존재한다. 안전성 모델은 공격자와 공격 환경을 제공하는 챌린저(challenger) 간의 게임(game)으로 구성된다.

Game 1: T1 공격자는 KGC의 msk는 알 수 없지만 임의의 사용자의 공개키를 교체할 수 있고, 챌린저로부터 원하는 메시지의 서명과 *ID*에 대응하는 개인키를 자유롭게 질의하여 얻을 수 있다. 공격자는 개인키 질의와 공개키를 교체한적 없는 *ID*와 서명 질의를 하지 않은 메시지 m에 대한 위조를 하면 게임에서 이긴다.

Game 2: T2 공격자는 공개키를 교체할 수는 없지만 msk를 알고 있으며, 챌린저에게 원하는 메시지에 대한 서명과 *ID*에 대응하는 서명키를 자유롭게 질의할 수 있다. 공격자는 서명키 질의를 한적 없는 *ID*와 서명 질의를 한적 없는 메시지 m에 대한 위조를 하면 게임에서 이긴다.

4. CL 서명 기법

CL 서명 기법은 (Setup, KeyGen, SetSec, SetPub, Sign, Verify)의 6개의 알고리즘으로 구성되며, RSA 구조의 CL 서명 기법은 다음과 같다.

- **Setup**$(1^\lambda)\rightarrow pp,msk$: 보안상수 1^λ를 입력으로 하여

소수 p'과 q'에 대하여 $p=2p'+1$과 $q=2q'+1$을 만족하는 RSA 파라미터 (n,p,q,e,d)를 생성한다. 이때 $n=pq$이고, 오일러 함수 $\phi(n)$을 이용해 $e<\phi(n)$와 $ed\equiv 1 \bmod \phi(n)$을 만족하도록 e와 d를 생성한다. 위수가 $p'q'$인 군 G의 생성원 g를 선택하여 공개파라미터 $pp=(n,e,g,H,h)$와 마스터키 $msk=(n,p,q,d)$를 출력한다. 여기서 $H:\{0,1\}^* \rightarrow Z_n^*$와 $h: Z_n^4 \times \{0,1\}^* \rightarrow \{0,1\}^\lambda$는 암호학적 해쉬 함수이다.

- **KeyGen**$(ID,msk,pp)\rightarrow x_{ID}$: 사용자의 ID와 마스터키 msk, 그리고 공개파라미터 pp를 입력으로 하여, 사용자 개인키 $x_{ID}=H(ID)^{-d} \bmod n$를 출력한다.
- **SetSec**$(pp)\rightarrow s_{ID}$: 공개파라미터 pp를 입력으로 하여 임의의 $s_{ID} \in Z_{2^{|n|/2-1}}$를 비밀값으로 출력한다. 여기서, $|n|$은 n의 비트 길이를 의미하고, 사용자의 서명키는 $sk_{ID}=(x_{ID}, s_{ID})$로 설정한다.
- **SetPub**$(s_{ID},pp)\rightarrow pk_{ID}$: 사용자의 비밀값 s_{ID}와 공개파라미터 pp를 입력으로 하여, 사용자 검증키 $vk_{ID}=g^{s_{ID}} \bmod n$를 출력한다.
- **Sign**$(m,sk_{ID},pp)\rightarrow \sigma_{ID}$: 메시지 m과 서명키 sk_{ID}, 그리고 공개파라미터 pp를 입력으로 하여, 임의의 $r_1, r_2 \in Z_{2^{|n|/2-1}}$를 선택하고, $c=h(g^{er_1}, g^{r_2}, vk_{ID}, ID, m)$를 계산한다. 그리고 다음과 같이 서명 σ_{ID}을 출력한다.

$$\sigma_{ID}=(z,s,c)=(sk_{ID}^c \cdot g^{r_1}, r_2 - s_{ID}c, c)$$

- **Verify**$(\sigma_{ID},m,ID,vk_{ID},pp)\rightarrow 1/0$: 서명 $\sigma_{ID}=(\sigma_1,\sigma_2,\sigma_3)$, 메시지 m, 아이디 ID, 검증키 vk_{ID}, 그리고 공개파라미터 pp를 입력으로 하여 다음의 과정을 수행한다.

1. $R_1=\sigma_1^e \cdot H(ID)^{\sigma_3}$와 $R_2=g^{\sigma_2} \cdot vk_{ID}^{\sigma_3}$을 계산한다.
2. $R_3=h(R_1,R_2,vk_{ID},ID,m)$을 계산하여, $R_3=\sigma_3$이면 1을 출력하고, 그렇지 않으면 0을 출력한다.

정확성. 사용자의 서명키 sk_{ID}와 서명 $\sigma_{ID}=(\sigma_1,\sigma_2,\sigma_3)$, 그리고 검증키 vk_{ID}에 대한 검증은 아래와 같은 식을 통해 기법의 정확성을 확인할 수 있다.

$$\begin{aligned} R_1 &= \sigma_1^e \cdot H(ID)^c = (x_{ID}^c \cdot g^{r_1})^e \cdot H(ID)^c \\ &= (H(ID)^{-cd} \cdot g^{r_1})^e \cdot H(ID)^c \\ &= (H(ID)^{-c} \cdot g^{er_1}) \cdot H(ID)^c = g^{er_1} \\ R_2 &= g^{\sigma_2} \cdot vk_{ID}^c = g^{r_2 - s_{ID}c} \cdot g^{s_{ID}c} = g^{r_2} \end{aligned}$$

4.1 CL 서명의 안전성 증명

정리 1: RSA 문제와 이산대수 문제가 어렵다고 가정할 때, 제안하는 RSA 구조의 CL 서명은 랜덤 오라클(random oracle) 모델에서 존재적 위조 공격에 안전하다(existentially unforgeable).

증명 : 안전성 증명은 챌린저 C1이 T1 공격자를 이용하여 DL 문제를 푸는 Game1과 챌린저 C2가 T2 공격자를 이용하여 RSA 문제를 푸는 Game2로 이루어진다.

Game1에서 C1은 자신이 받은 문제 $(n,g,y\equiv g^x \bmod n)$을 이용하여 T1 공격자에게 서명 및 서명키 정보를 생성해주며, 최종적으로 각 공격자는 위조된 서명 튜플 $(ID^*, pk^*, m^*, \sigma_1^*)$을 출력한다. 이때, 챌린저는 Forking Lemma[7]에 따라 공격자로부터 동일한 난수이지만 다른 랜덤 오라클을 사용하여 생성된 서명 $(ID^*, pk^*, m^*, \sigma_2^*)$을 얻는다. 그리고 σ_1^*과 σ_2^*로부터 C1은 x를 계산한다.

Game2에서도 마찬가지로 C2는 자신이 받은 문제 $(n,e,z\equiv u^e \bmod n))$를 이용하여 T2 공격자에게 서명 및 서명키 정보를 생성해주며, Forking Lemma에 따라 위조된 서명 튜플 $(ID^*, pk^*, m^*, \sigma_1^*)$와 $(ID^*, pk^*, m^*, \sigma_2^*)$를 T2 공격자로부터 얻는다. 그리고 C2는 σ_1^*과 σ_2^*로부터 u를 계산한다. 자세한 증명 과정은 생략한다. □

5. CL 서명을 이용한 인증 및 키 합의 프로토콜

CL 서명을 이용한 인증 및 키 합의 프로토콜은 다음과 같이 수행된다.

- **시스템 셋업 단계 :** KGC는 CL 서명 기법의 Setup 알고리즘을 통해 공개파라미터 $pp=(n,e,g,H,h,f)$와 마스터 키 $msk=(n,p,q,d)$를 생성한다. $f:\{0,1\}^* \rightarrow \{0,1\}^k$는 키 합의를 위한 추가적인 해시 함수 이다.
- **키 발급 단계 :** 사용자는 자신의 ID에 대한 개인키를 KGC에 요청하고, KGC는 CL 서명 기법의 KeyGen 알고리즘을 이용하여 각 사용자의 x_{ID}를 생성한다. 두 사용자 A와 B는 KGC로부터 각각 개인키 $x_A=H(ID_A)^d \bmod n$과 $x_B=H(ID_B)^d \bmod n$을 안전한 채널을 통해 발급받는다.
- **키 생성 단계 :** 사용자는 SetSec과 SetPub 알고리즘을 이용하여 자신의 서명키 sk_{ID}와 검증키 vk_{ID}를 생성한다. 사용자 A와 B의 서명키와 검증키는 각각 (sk_A, vk_A)와 (sk_B, vk_B)이다.
- **인증 및 키 합의 단계 :** 두 사용자 A와 B의 인증 및 키 합의 프로토콜은 다음과 같이 수행된다.

1. A는 먼저 챌린지에 해당하는 난수 $N_A \in \{0,1\}^*$을 선택하여 (ID_A, ID_B)와 함께 B에게 전달한다.
2. B는 전송받은 (ID_A, ID_B)를 확인하고 챌린지에 해당하는 난수 $N_B \in \{0,1\}^*$를 선택한다. 그리고 아래와 같이 CL 서명 기법의 Sign 알고리즘을 이용하여 메시지 $m_B=ID_B \| ID_A \| N_B \| N_A$에 대한 서명 σ_B를 생성하고, m_B와 함께 A에게 전송한다.

① $r_{B,1}, r_{B,2} \leftarrow Z_{2^{|n|/2-1}}, R_{B,1} \leftarrow g^{er_{B,1}} \bmod n,\ R_{B,2} \leftarrow g^{r_{B,1}} \bmod n$
② $c_B = h(R_{B,1}, R_{B,2}, vk_B, B, m_B)$
③ $\sigma_B = (\sigma_1, \sigma_2, \sigma_3) = (sk_B^{c_B} \cdot g^{r_{B,1}}, r_{B,2} - s_B c_B, c_B)$

3. A는 CL 서명 기법의 Verify 알고리즘을 이용하여 전송받은 서명 σ_B를 아래와 같이 검증한다.

① $R_{B,1} = \sigma_1^e \cdot H(B)^{\sigma_3},\ R_{B,2} = g^{\sigma_2} \cdot vk_B^{\sigma_3}$
② $R_3 = h(R_{B,1}, R_{B,2}, vk_B, B, m_B)$에 대해 등식이 성립하면 1을 출력하고, 그렇지 않으면 0을 출력한다.
출력값이 0이면 A는 프로토콜을 중단하고, 1이면 Sign 알고리즘을 이용하여 $m_A = ID_A \| ID_B \| N_A \| N_B$에 대한 서명 σ_A를 생성하여 m_A와 함께 B에게 전송한다.

① $r_{A,1}, r_{A,2} \leftarrow Z_{2^{|n|/2-1}}, R_{A,1} \leftarrow g^{er_{A,1}} \bmod n,\ R_{A,2} \leftarrow g^{r_{A,1}} \bmod n$
② $c_A = h(R_{A,1}, R_{A,2}, vk_A, B, m_A)$
③ $\sigma_A = (\sigma_1, \sigma_2, \sigma_3) = (sk_A^{c_A} \cdot g^{r_{A,1}}, r_{A,2} - s_A c_A, c_A)$
그리고 검증 과정에서 계산한 $R_{B,1}, R_{B,2}$와 서명 과정에서 선택한 난수 $r_{A,1}, r_{A,2}$를 이용하여 세션키(session key)를 다음과 같이 계산한다.

① $K_A = R_{B,1}^{r_{A,1}} \| R_{B,2}^{r_{A,2}}$
② $ssk = (K_A \| ID_A \| ID_B)$

4. B는 CL 서명 기법의 Verify 알고리즘을 이용하여 전송받은 서명 σ_A를 아래와 같이 검증한다.

① $R_{A,1} = \sigma_1^e \cdot H(A)^{\sigma_3},\ R_{A,2} = g^{\sigma_2} \cdot vk_A^{\sigma_3}$
② $R_3 = h(R_{A,1}, R_{A,2}, vk_A, A, m_A)$에 대해 등식이 성립하면 1을 출력하고, 그렇지 않으면 0을 출력한다.
출력값이 0이면 B는 프로토콜을 중단하고, 1이면 검증 과정에서 계산한 $R_{A,1}, R_{A,2}$와 서명 과정에서 선택한 난수 $r_{B,1}, r_{B,2}$를 이용하여 세션키(session key)를 다음과 같이 계산한다.

① $K_B = R_{A,1}^{r_{B,1}} \| R_{A,2}^{r_{B,2}}$
② $ssk = (K_B \| ID_A \| ID_B)$

5.1 CL 인증 및 키 합의 프로토콜의 안전성 분석

CL 인증 및 키 합의 프로토콜의 안전성은 CL 타입의 Canetti-Krawczyk(CK) 모델에서 증명할 수 있다. CL 서명의 안전성 모델과 같이 두 가지 유형의 공격자가 존재하며, ID에 대응하는 개인키를 얻거나 공개키를 자유롭게 교체할 수 있다. 공격자의 최종 목표는 자신이 선택한 사용자 간의 세션키 정보를 얻는 것이다.

정리 2: Computational Diffie-Hellman(CDH) 문제[9]가 어렵다고 가정할 때, 제안하는 인증 및 키 합의 프로토콜은 CL-CK 모델에서 안전하다.

6. 결론

본 논문에서는 기존 RSA 구조의 CL 서명 연구의 잘못된 분석을 지적하고, 인증과 키 합의를 동시에 수행할 수 있는 RSA 구조의 CL 서명을 제안하였다. 제안한 서명 기법은 약한 가정 하에서 안전성이 증명되며, 인증 및 키 합의 프로토콜로 쉽게 확장 가능하다. 인증을 수행하는 과정에서 계산되는 정보를 이용하여 키 합의를 바로 수행할 수 있기 때문에 서명을 이용한 일반적인 인증 및 키 합의 프로토콜보다 효율적이다.

참고문헌

[1] Sattam S. Al-Riyami and Kenneth G. Paterson. "Certificateless public key cryptography." Asiacrypt 2003
[2] Dario Fiore and Rosario Gennaro, "Identity-based key exchange protocols without pairings", Transactions on computational science X. 42-77, 2010
[3] Rosario Gennaro, Hugo Krawczyk, and Tal Rabin, "Okamoto-Tanaka Revisited: Fully Authenticated Diffie-Hellman with Minimal Overhead." ACNS 2010
[4] Debiao He, Muhammad K. Khan, and Shuhua Wu. "On the Security of a RSA-based Certificateless Signature Scheme." IJ Network Security 16:78-80, 2014
[5] Dheerendra Mishra and Sourav Mukhopadhyay, "Cryptanalysis of Pairing-Free Identity-Based Authenticated Key Agreement Protocols", ICISS 2013
[6] Eiji Okamoto and Kazue Tanaka, "Key Distribution System Based on Identification Information", IEEE Journal on Selected Areas in Communications, 7(4):481-485, 1989.
[7] David Pointcheval and Jacques Stern, "Security proofs for signature schemes", Eurocrypt 1996
[8] Gaurav Sharma, Suman Bala, and Anil K. Verma, "An Improved RSA-based Certificateless Signature Scheme for Wireless Sensor Networks", IJ Network Security 18(1):82-89, 2016
[9] Zahava Shmuely, "Composite Diffie-Hellman Public-Key Generating Systems are Hard to Break", Technical Report, Department of Computer Science, 356, 1985
[10] Dae Hyun Yum and Pil Joong Lee, "Generic construction of certificateless signature", ACISP 2004
[11] Jianhong Zhang and Mao Jane, "An efficient RSA-based certificateless signature scheme", Journal of Systems and Software, 85(3): 638-642., 2012
[12] Zhenfeng Zhang, Duncan S. Wong, Jing Xu, and Dengguo Feng, "Certificateless public-key signature: security model and efficient construction", ACNS, 2006
[13] 엄지은, 서민혜, 박종환, 이동훈. "효율적인 ID 기반 인증 및 키 교환 프로토콜." 정보보호학회논문지 26(6): 1387-1399, 2016