한국통신학회논문지 '13-03 Vol.38B No.03
http://dx.doi.org/10.7840/kics.2013.38B.3.190

다중서버 인증을 위한 스마트카드 기반 중재 인증 기법 연구

김 명 선◆

A Brokered Authentication Scheme Based on Smart-Card for Multi-Server Authentication

Myungsun Kim◆

요 약

사용자가 원하는 서비스가 여러 개의 서버에 분산되어 있을 수 있기 때문에 다수의 서버가 존재하는 다중서버 환경을 위한 인증기법은 웹서비스를 이용할 때 반드시 필요하다. 일반적으로 Password를 사용하는 방법이 적용되나 안전성 측면에서 취약하고 서버마다 다른 Identity(ID)와 Password를 사용하는 것은 불편하다. 그래서 사용자가 사용하는 여러 서버에 접속할 때 항상 동일한 ID를 사용하는 것이 허용되나, 다양한 공격에 노출될 수 있다. 본 논문에서는 서버가 여러 개 존재하는 환경에서 원격지에 있는 사용자는 하나의 스마트카드만 사용하여 다양한 서비스를 편리하고 안전하게 받을 수 있는 인증기법을 제안한다. 추가로 제안한 기법의 안전성을 공격 유형별로 나누어 분석하고, 기존 방법과 성능비교를 통하여 제안하는 기법이 효율적임을 보인다.

Key Words : Brokered Authentication, Multi-server Authentication, Security Token, Smart-card

ABSTRACT

Since the facilities for the remote users tend to be deployed in distributed manner, authentication schemes for multi-server communication settings, which provide various web services, are required for real-world applications. A typical way to authenticate a remote user relies on password authentication mostly. However, this method is vulnerable to attacks and inconvenient as the system requires users to maintain different identities and corresponding passwords. On the other hand, the user can make use of a single password for all servers, but she may be exposed to variants of malicious attacks. In this paper, we propose an efficient and secure authentication scheme based on a brokered authentication along with smart-cards in multi-server environment. Further we show that our scheme is secure against possible attacks and analyze its performance with respect to communication and computational cost.

Ⅰ. 서 론

프로세스의 성능 향상과 비용감소에 의해 스마트카드가 폭넓게 사용되는 것이 가능해지고 있다. 예를 들면 신분증과 결합되거나 다기능 교통카드가 대표적이다. 동시에 IT 기술의 발전으로 편리한 서비스나 자원이 여러 개의 서버에 분산되어 존재하는 추세이다. 사용자는 좀 더 효율적으로 분산된 자원에 접근할 수 있게 되었으나, 여러 개의 서버에 접근해야 할 필요성이 있다. 이러한 상황에서 사용자에게 편리성을 제공하기 위해 저렴한 스마트카드를 사용하여 인증하는 방법이 도입되었다[23,4,11]. 그

◆ 주저자 : 수원대학교 IT대학 정보보호학과, msunkim@suwon.ac.kr, 정회원
논문번호 : 2013-02-103, 접수일자 : 2013년 2월 26일, 최종논문접수일자 : 2013년 3월 12일

래서 사용자는 권한이 있는 서버가 발행한 한 장의 스마트카드를 사용해서 여러 개의 서버에 접속하여 다양한 서비스를 편리하게 받을 수 있을 것이다. 사용자 입장에서는 서버마다 별도의 스마트카드를 사용하는 대신 한 장의 스마트카드를 사용함으로써 얻는 경제적인 이점도 있다.

이러한 이유로 사용자는 여러 서버에 한 장의 스마트카드를 사용할 때 각기 다른 서비스에 대하여 한 개의 패스워드만 사용하면 되므로 사용자는 매우 편리하다. 그러나 원격지 사용자가, 여러 서버가 제공하는 상이한 네트워크 서비스를 한 장의 스마트카드를 사용하여 이용하려는 경우에 해당 서비스에 자신의 아이디와 패스워드를 먼저 등록해야 한다. 즉 사용자가 여러 서비스를 사용하려면 매번 자신의 아이디와 패스워드를 해당 서버에 제시해야한다. 이러한 상황에서 개인의 비밀정보는 공격자의 추적에 노출되고 시스템의 개인정보가 해킹될 수 있다.

다양한 연구들이 이러한 보안문제를 다루기 위해 제안되었다[16,6,15,25]. 그 후 Chang과 Lee[4], Juang[11]는 기존 기법의 효율성 개선에 기여하였다. 이러한 기존 연구들의 공통 특징은 인증을 위하여 검증테이블(Verification Table)을 서버가 저장해야 한다. 그러나 이들은 Stolen Verifier Attack이나 Insider Attack에 취약하다. 이 문제를 해결하기 위해 검증테이블 대신 해시함수(Hash Function)을 사용하는 기법이 Tsai[22], Yoon과 Yoo [27]에 의해 제안되었다. 그러나 이들의 기법은 단순한 Replay Attack에 취약함이 밝혀졌다[5]. 그래서 Timestamp를 이용하여[26,24] Replay Attack을 방지하는 방법들이 제안되었다[20,5].

본 논문은 이러한 기존의 보안문제를 해결하기 위한 새로운 다중서버 인증기법을 제안한다. 좀 더 구체적으로 이야기하면 본 연구에서는 인증서버(Authentication Server, AS)에 인증을 중재하는 기능을 도입하여 사용자와 서버간의 직접 연결을 제거한다. 인증이 중재되는 경우 이것을 중재인증(Brokered Authentication, BA)이라 한다[17]. 구체적인 기술로는 보안 토큰 서비스(Security Token Service, STS)가 쓰인다. (STS가 편한 독자는 BA 대신 사용해도 무방하다.) 중재인증을 사용하여 얻을 수 있는 또 다른 장점은 이기종 네트워크간의 사용자 인증이 용이하게 구현될 수 있다. 이 시스템에서 AS는 사용자 인증에 사용될 보안 토큰을 발행하고 접근제어 기능을 제공한다. 특히 본 연구에서는 연산능력이 제한된 휴대용기기에서 사용하는 것이 가능하도록 연산량을 개선하였다. 또한 저장량을 줄이면서 보안성을 유지하기 위해 타원곡선기반의 기법을 사용한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 연구의 이해를 돕기 위한 배경지식을 기술하고 3장에서 본 논문에서 제안하는 기법을 자세히 다룬다. 끝으로 4장에서 안전성 및 성능의 분석을 제시하고 5장의 맺음말로 정리한다.

Ⅱ. 배경지식 및 정의

이번 장에서는 본 연구의 주요 도구인 타원곡선 기반의 암호 시스템, 제안한 인증 기법이 기반하는 암호학적 가정과 중재 인증에 대해서 좀 더 자세히 기술한다.

2.1. 타원곡선 암호

공개키 암호 알고리즘은 인수분해[21] 또는 이산대수[9] 문제의 어려움에 기반을 두고 있다. (다음 장에서 자세히 다룬다.) 이때 문제를 풀기 어려운 것이지 불가능한 것이 아니므로, 주어진 수는 인수분해나 또는 주어진 군(Group)에서 이산대수 문제를 풀기 어려울 정도의 크기이어야 한다. 예를 들면 RSA 암호 알고리즘은 현재 2048 비트 크기의 정수를 키(Private Key)로 사용한다[21]. 더구나 컴퓨터의 연산능력이 점점 향상되기 때문에 점점 더 큰 정수나 위수(Order)가 더 큰 군을 사용하게 된다. 그러나 경우에 따라서는 이렇게 큰 정수를 저장하거나 다루는데 문제가 되는 기기가 존재한다. 그래서 동일한 안전성을 제공하면서 짧은 키를 사용하는 타원곡선 암호 기법이 제안되었다[18,13].

타원곡선 암호 기법은 Weierstrass 방정식을 사용하며 암호 알고리즘이 동작할 수 있는 유한순환군(Finite Cyclic Group)을 만든다[2]. 이렇게 만들어진 유한순환군이 큰 소수에 기반을 두어 만들어지는 군을 대체하는 것이다. 차이점은 타원곡선을 사용하여 만들어진 유한순환군은 덧셈군(Additive Group)이며 기존 유한순환군은 곱셈군이다. 타원곡선 암호의 연산규칙, 안전성은 각각 [3,2]과 [7] 및 해당 논문의 참고문헌을 참고하고 효율적 연산은 [12,8] 등을 참고한다.

타원곡선 기반의 암호화(Encryption) 알고리즘은 표준 X9.63[1]에 구체적으로 제정되어 있으며 서명과 키교환(Key Exchange) 및 키합의(Key

Agreement) 기법은 표준 P1363[10]에 정의되어 있다. 본 논문에서는 암호화 알고리즘으로 ECIES(Elliptic Curve Integrated Encryption Scheme), 키합의 기법으로 ECDH(Elliptic Curve Diffie-Hellman)을 사용한다[1].

2.2. 암호학적 가정

본 논문에서 제안하는 암호 알고리즘은 타원곡선 기반 Diffie-Hellman(DH) 문제의 어려움에 기반을 둔다. 이산대수 문제와도 밀첩하게 관련된 DH 문제는 Diffie와 Hellman에[9] 의해 설계된 공개키 암호에 처음 적용된 것으로 다음과 같이 정의된다.

정의 1. (**Diffie-Hellman 문제**) G를 유한순환군이라 하고 g를 유한순환군 G의 모든 원소를 생성하는 생성자라 하고 $G=\langle g\rangle$라 표기하자. 이때 G의 임의의 두 원소 g^a와 g^b가 주어졌을 때 g^{ab}를 계산하는 문제를 Diffie-Hellman 문제라 한다. 여기서 a,b는 정수이다.

정의에서와 같이 군 G의 원소 g,g^a,g^b가 주어졌을 때 이것들로부터 g^{ab}를 계산하기 어렵다는 것이 **Diffie-Hellman 가정**이다. 현재 곱셈군 G를 사용하여 표기한 것을 덧셈군으로 표현하면, 지수 형식은 상수 곱셈(Scalar Product)으로 표현하고 곱셈 형식은 덧셈으로 표현한다.

2.3. 중재 인증

이미 잘 알려진 바와 같이 보안 토큰 서비스(Security Token Service, STS)는 분산 네트워크에서 보안토큰을 발행하는 웹서비스(Web Service)이다. 이러한 네트워크에 있는 사용자들 간의 신뢰할 수 있는 통신을 제공하기 위해 신뢰할 수 있는 개체가 이기종 네트워크 내에 존재하는 사용자들에게 보안토큰을 발행한다. 이러한 기술을 일반적으로 중재된 인증 또는 중재인증(Brokered Authentication, BA)라 부른다[17]. BA는 인증중재자(Authentication Broker)에게 사용자의 신원을 증명하는 방법을 명시하고 있으며, 실제로는 인증중재자가 보안토큰을 발행한다.

BA의 이점은 웹서비스에서 요구되는 인증, 권한관리(Authorization)와 세션키(Session Key) 교환을 통합적으로 지원하는 시스템 구현이 가능하고 이기종 플랫폼간의 상호호환성을 제공한다.

BA에서 인증 시스템은 사용자, 서버, 인증중재자 및 신원저장소(Identity Store)로 구성된다. 그래서 사용자의 계정이나 ID를, 인가되지 않은 내부 관리자가 수정하는 보안문제가 발생할 수 있다. 이런 보안문제를 해결하기 위해 신원저장소 또는 검증테이블(Verification Table)을 별도로 저장하지 않고 인증 서버(Authentication Server)가 발행한 보안토큰을 사용하는 방법을 이용한다.

본 논문에서는 이러한 중재인증의 개념을 다중서버 인증에 도입하려는 것이며 이를 위해 다음과 같은 개체를 재정의한다.

- 사용자(User). 사용자는 웹서비스에 접근하여 웹서비스에서 요구하는 인증 정보를 제공한다.
- 인증서버(Authentication Server, AS). AS는 사용자가 입력한 로그인(Login) 정보를 검증하여 인증한다. 또한 인증에 성공한 사용자와 그가 접근하려는 서버를 위해 보안토큰을 발행한다.
- 대상서버(Target Server, TS). TS는 사용자가 접근하려는 특정 웹서비스를 제공하는 웹서버(Web Server)이다. 해당 웹서비스를 이용하기 위해 TS에 접근하려면 인증이 필요하다.

2.4. 다중서버 인증 기법

본 장에서는 다중서버 인증 기법을 좀 더 형식적으로 정의하고자 한다. 먼저 다중서버 인증에 포함된 TS의 수를 m이라 하고, 웹서비스를 이용하려는 사용자의 수를 n이라 하자.

다중서버인증 기법은 다음과 같이 5단계로 구성된다.

(i) 설정(Setup) 단계. 사용자, AS와 TS 간의 인증을 수행하기 위해 필요한 시스템 매개변수와 해시함수를 결정하는 단계이다.

(ii) 등록(Registration) 단계. 모든 n명의 사용자와 m개의 TS는 AS에 정당한 사용자 및 서버로 등록을 한다.

(iii) 로그인(Login) 단계. 임의의 사용자 u_i가 TS가 제공하는 웹서비스를 이용하기 위해서 로그인 단계를 수행한다.

(iv) 서버인증(TS-Authentication) 단계. 이 단계에서 AS는 사용자가 서비스를 요청하는 TS를 먼저 인증하고 보안토큰을 생성한다.

(v) 상호인증(Mutual Authentication) 단계. AS로부터 토큰을 받은 사용자 u_i는 임의의 TS S_j와 상호인증을 수행한다.

Ⅲ. 제안하는 방법

3장에서는 본 논문에서 제안하는 기법을 자세히 설명한다. 이를 위해 먼저 표기법을 간략히 정리하고 제안하는 기법을 구체적으로 설명한다.

3.1. 표기법

본 논문에서는 인증에 참여하는 개체 간에 인증된 비밀키(Secret Key)인 세션키를 공유하는 것을 최종 목적으로 한다. 이를 위하여 타원곡선 기반 DH-키합의(Key Agreement) 기법을 이용한다. 먼저, 사용할 타원곡선과 관련된 표기법을 기술한다. 유한체(Finite Field) $\mathbb{F}_q$위에서 정의된 타원곡선을 $E(\mathbb{F}_q)$로 표기한다. 여기서 어떤 수 k, 소수 p에 대하여 $q=p^k$이다. 본 논문에서는 구현의 편의를 위해 $p=2$를 사용하므로, $\mathbb{F}_{2^k}$ 위에서 정의되는 다음과 같은 타원곡선 E를 사용한다:

$$E(\mathrm{F}_{2^k}) : y^2+xy=x^3+ax^2+b,$$

여기서 $a,b\in\mathbb{F}_{2^k}$ 이며, $b\neq 0$이다. 특히 안전성을 만족하는 타원곡선을 사용해야 하며[14] 본 논문에서는 $k=163$인 Koblitz 곡선을 사용한다. 추가로 기본점(Base Point)을 P로 표기하자.

각 사용자는 u_i, 각 TS는 S_j로 표기한다. 또한 해시함수는 H로 표기하고 T_u, T_s를 각각 사용자와 TS의 Timestamp라 하자. 사용자 u_i의 ID는 UID_i, TS S_j의 ID는 SID_j, AS의 ID는 SID_{AS}로 표기한다. 2.4장에서 언급한 바와 같이, 사용자의 수는 n, TS의 수는 m이므로 본 논문 전체에서 사용하는 첨자는 $1\le i\le n, 1\le j\le m$이다.

3.2. 제안하는 인증 기법

본 장에서는 전술한 표기법을 사용하여 본 논문에서 제안하는 인증기법을 자세히 설명한다.

3.2.1. 설정 단계

이 단계에서는 DH-키합의를 위해 필요한 타원곡선의 매개변수 및 난수의 크기를 결정한다. 추가로 사용할 해시함수를 결정하는데 암호학적 해시함수를 선택하여 사용한다 (예, SHA1[19]).

타원곡선 암호의 매개변수가 결정되면, 각 사용자와 TS는 ECIES[1]에서 정의된 암호화 알고리즘을 위한 자신의 공개키와 개인키 쌍을 계산한다. (R_{u_i}, d_i)는 사용자 u_i의 공개키/개인키 쌍이고, (R_{S_j}, e_j)는 TS의 공개키/개인키 쌍이다. AS도 공개키와 개인키 쌍 (pk, sk)을 생성하여 공개한다.

3.2.2. 등록 단계

이 단계에서는 AS에 접근하여 다중서버 네트워크에 사용자와 TS가 유효한(Valid) 개체로 각각을 등록한다. 이를 위해서 사용자 u_i는 자신의 UID_i와 암호 w_i를 안전한 채널을 통해서 AS에게 전송한다. 사용자 u_i가 AS의 공개키 pk를 사용하여 암호화한 후 전송하는 것으로 안전한 채널을 만들 수 있다. DH 가정에 의해 이러한 암호화 알고리즘은 안전하다.

AS는 sk를 사용하여 (UID_i, w_i)를 복구한 후

$$\alpha = H(\mathrm{UID}_i \| \theta_u)$$

와

$$\beta = \alpha \oplus H(w_i)$$

를 계산한다, 여기서 θ_u는 사용자용 난수이며 $\oplus$는 비트 XOR 연산을 의미한다. 이렇게 계산한 β 값만 스마트카드에 저장한 후, 이것을 사용자 u_i에게 발행한다. 단 AS는 w_i를 저장하지 않는다.

TS S_j가 등록하는 과정도 사용자가 등록하는 과정과 유사하다. 그러나 TS는 유효개체로 등록만 하면 되므로 Password를 사용할 필요는 없다. 그래서 S_j가 공개된 채널로 SID_j를 전송하면 AS는 TS용 난수 θ_s를 선택한 후,

$$\gamma = H(\mathrm{SID}_j \| \theta_s)$$

을 계산한다. AS는 γ를 안전한 채널로 TS에게 전송한다. 이때 TS S_j의 공개키 R_{S_j}를 이용한다. TS는 자신의 개인키 e_j를 사용해 복호화한다.

3.2.3. 로그인 단계

이 단계에서는 사용자가 임의의 TS가 제공하는 웹서비스를 사용하려 하는 것으로 시작한다. 이를 위하여 사용자는 자신의 Password w_i를 스마트카드에 제시하여 다음을 얻는다:

$$\alpha = \beta \oplus H(w_i).$$

위에서 계산한 α와 새로운 난수 r'_i를 이용해

$$x = H(\alpha \| r'_i) = H(H(\mathrm{UID}_i \| \theta_u) \| r'_i)$$

를 계산한다. TS S_j의 서비스를 사용하려면 AS에

게 메시지 $(x, \mathrm{SID}_j, \mathrm{UID}_i, r'_i)$를 전송한다.

AS는 u_i가 등록된 사용자인지 확인하기 위해 자신이 저장하고 있는 u_i의 $(\mathrm{UID}_i, \theta_u)$와 수신한 난수 r'_i를 사용하여

$$x' = H(H(\mathrm{UID}_i \| \theta_u) \| r'_i)$$

을 계산하고 $x = x'$을 만족하는지 검사한다. 만족한다면 등록된 사용자로 판단하고 그렇지 않으면 실패메시지를 전송하고 종료한다.

등록된 사용자일 경우 AS는 $(\mathrm{UID}_i, \mathrm{SID}_{AS})$를 TS에게 전송하여 사용자의 접속요청을 알린다.

TS는 AS로부터 요청을 받으면 난수 r'_j를 생성한 후

$$y = H(\gamma \| r'_j) = H(H(\mathrm{SID}_j \| \theta_s) \| r'_j)$$

을 계산하여 (y, r'_j)를 로그인 승인 메시지로 사용하도록 AS에게 전송한다.

3.2.4. 서버인증 단계

이 단계에서 AS는 TS를 인증한 후 성공하면 보안토큰을 생성한다.

우선 AS는 다음을 계산하여 TS가 등록된 서버인지 검증한다. 이때 AS가 저장한 $(\mathrm{SID}_j, \theta_s)$를 사용한다.

$$y' = H(H(\mathrm{SID}_j \| \theta_s) \| r'_j)$$

만약 $y = y'$을 만족하면 다음 식을 사용하여 임시키(Ephemeral Key)[1] K_1, K_2를 생성한다:

$$\begin{aligned} K_1 &= H(\gamma \| r'_i \| r'_j) \\ &= H(H(\mathrm{SID}_j \| \theta_s) \| r'_i \| r'_j) \end{aligned}$$

와

$$\begin{aligned} K_2 &= H(\alpha \| r'_i \| r'_j) \\ &= H(H(\mathrm{UID}_i \| \theta_u) \| r'_i \| r'_j). \end{aligned}$$

임시키 생성을 완료하면 보안토큰을 생성하기 위해 $z_1 = K_1 \oplus K_2$와 $z_2 = H(K_1 \| K_2)$를 계산한다. AS는 사용자 u_i의 보안토큰 $\delta_{u,s} = z_1 \| z_2$, TS S_j의 보안토큰 $\delta_{u,s} = z_1 \| z_2$, $\sigma = r'_i \| r'_j$을 계산하여 u_i에게 $(\delta_{u,s}, \sigma)$를 전송하고 S_j에게 $(\delta_{u,s}, \sigma)$를 전송한다.

3.2.5. 상호인증 단계

마지막으로 사용자 u_i와 TS S_j간의 상호인증을 수행하는 단계를 설명한다.

우선 등록된 사용자 u_i는 AS에게 받은 난수값 σ와 자신의 Password를 사용하여, 이번에 수신한 보안토큰이 유효한지 검사한다. 사용자 u_i는 앞에서 계산한 α를 이용하여 차례로

$$\begin{aligned} U_1 &= H(\alpha \| r'_i \| r'_j) \\ &= H(H(\mathrm{UID}_i \| \theta_u) \| r'_i \| r'_j) \end{aligned}$$

와

$$U_2 = z_1 \oplus U_1$$

을 계산한다. 여기서 z_1은 $\delta_{u,s}$에서 얻고 r'_j는 σ에서 얻는다. 그리고 $z'_2 = H(U_1 \| U_2)$를 계산하여 $\delta_{u,s}$에서 얻은 z_2와 비교한다. 만약 같으면 다음을 단계를 진행하고 $z'_2 \neq z_2$이면 AS에게 통보하고 종료한다.

TS S_j도 비슷한 과정을 수행하여 자신이 수신한 보안토큰이 유효한지 검증한다. 이를 위해 AS가 전송한 σ로부터 r'_i를 추출하여

$$\begin{aligned} V_1 &= H(\gamma \| r'_i \| r'_j) \\ &= H(H(\mathrm{SID}_j \| \theta_s) \| r'_i \| r'_j) \end{aligned}$$

와

$$V_2 = z_1 \oplus V_1$$

을 차례로 계산한다. 다음에 $z'_2 = H(V_1 \| V_2)$를 계산하고, u_i처럼 $\delta_{u,s}$로부터 추출한 z_2와 비교한다. 만약 $z'_2 \neq z_2$이면 AS에게 이 사실을 통보하고 종료한다.

이제 상호인증을 위해 먼저 u_i가 DH-키합의를 위한 준비를 한다. 이를 위해 먼저 DH-키합의를 위한 공개값 $pk_u = r_u \cdot P$를 계산한다, 여기서 r_u는 사용자의 비밀값이며, $\cdot$은 기본점 P에 대한 상수배(Scalar Product) 연산을 의미한다. 추가로

$$W_u = H(pk_u \| \mathrm{UID_i} \| \mathrm{SID_j}) \oplus \mathrm{r'_i}$$

를 계산한다. 사용자 u_i는 (pk_u, W_u, T_1)을 S_j에게 전송한다, 여기서 T_1은 u_i의 Timestamp이다.

TS S_j가 이것을 수신하면 서버인증 단계에서 얻은 $(pk_u, \mathrm{UID_i}, \mathrm{r'_i})$를 이용하여

$$\widetilde{W_u} = H(pk_u \| \mathrm{UID_i} \| \mathrm{SID_j}) \oplus \mathrm{r'_i}$$

을 계산하여, $\widetilde{W}_u = W_u$을 만족하는지 검증한다. 만족하지 않으면 u_i에게 통보하고 종료한다. 만족하면 S_j는 DH 키합의를 위해 난수 r_s를 선택한 후

1) 임시키는 키합의 과정에서 임시로 만들어지는 비밀값을 의미한다.

자신의 공개값 $pk_s = r_s \cdot P$를 계산한다. 계산을 완료한 후, 자신의 Timestamp T_2를 생성하여 사용자 u_i에게 (pk_s, W_s, T_2)를 전송한다, 여기서

$$W_s = H(pk_s \| \mathrm{UID_i} \| \mathrm{SID_j}) \oplus \mathrm{r'_j}.$$

이것을 수신한 후, 사용자 u_i는 먼저 T_2를 추출하여 $\Delta_t = T_2 - T_1$를 계산한다. 만약 Δ_t가 사전에 정한 임계값(Threshold)보다 크면 S_j에게 통보하고 종료한다. 임계값보다 작으면 $(pk_s, \mathrm{SID_j}, \mathrm{r'_j})$를 사용하여

$$\widetilde{W_s} = H(pk_s \| \mathrm{UID_i} \| \mathrm{SID_j}) \oplus \mathrm{r'_j}$$

를 계산한다. 만약 $\widetilde{W_s} = W_s$인지 확인한 후 이것을 만족하지 않으면 S_j에게 통보하고 종료한다. 만족한다면 사용자 u_i는 TS S_j 간의 세션키 K를 다음과 같이 계산한다:

$$\begin{aligned} K &= r_u \cdot pk_s \\ &= r_u \cdot r_s \cdot P. \end{aligned}$$

실제로 S_j도 동일한 세션키 K를 갖는다. 왜냐하면

$$K = r_s \cdot pk_u = r_s \cdot r_u \cdot P$$

이므로 동일한 값임을 알 수 있다.

이제 사용자 u_i와 TS S_j는 정해진 Session 동안 세션키 K를 사용하여 안전한 채널을 만들어 사용할 수 있다.

Ⅳ. 제안하는 방법 분석

본 장에서는 제안한 다중서버 인증기법의 안전성과 성능을 분석한다. 안전성 분석을 위해 가능한 공격들을 나열하고 제안한 기법이 공격을 어떻게 방어하는지 기술한다. 성능 분석을 위해 각 단계별로 필요한 모든 연산의 횟수를 계산한다.

4.1. 안전성 분석

현재 다중서버 인증에 대한 안전성 증명은 엄밀한 수학적 증명보다는 가능한 공격에 대한 안전성을 보임으로써 이루어진다. 현재 다른 연구 결과의 접근법처럼, 다중서버 인증기법에 가능한 공격을 나열하고 제안한 기법의 안전성을 비교하여 기술한다.

Stolen Verifier Attack. 공격자는 TS에 저장된 Password의 해시값을 얻어내는 것이 가능하다. 그러나 본 논문에서 제안한 기법은 AS나 TS에 어떠한 Password도 저장하지 않는다.

Insider Attack. 사용자 u_i가 AS에게 자신의 UID_i와 Password w_i를 안전한 채널로 전송하면, AS는 $\beta = \alpha \oplus H(w_i)$를 계산한 후, β만 u_i에게 발행할 스마트카드에 저장한다. 그 후 사용자는 AS나 TS가 접근할 수 없는 환경에서 Password를 스마트카드에 제시한다. 그러므로 서버들은 u_i의 Password를 얻어낼 수 없다.

Replay Attack. 공격자는 전송 채널을 도청하여 송수신되는 메시지를 저장한 후 나중에 동일한 메시지를 반복적으로 사용하여 사용자나 TS의 ID에 대응하는 Password를 알아낼 수 있다. 본 논문에서는 이러한 Replay Attack을 방지하기 위해 Timestamp를 사용한다. 사용자와 TS간에 상호인증을 수행할 때 Timestamp를 전송하도록 하고 수신한 Timestamp 값 사이의 차이를 계산하여 임계값의 범위를 초과하지 않는 경우에만 성공하도록 인증기법을 설계하였다.

Man-in-the-middle Attack. 공격자가 중간에서 송신자의 메시지를 가로챈 후 자신의 메시지로 대체하여 지정된 수신자에게 재전송할 수 있다. 그러나 본 논문에서 제안한 기법은 Diffie-Hellman 문제가 어렵다는 가정 하에서 공격자가 세션키 K를 알아내는 것은 불가능하다. 그래서 공격자는 K에 대한 정보가 없기 때문에 사용자 u_i의 메시지를 볼 수는 있으나 TS S_j가 정상적으로 복호화할 수 있는 메시지를 만들 수 없다. 또한 모든 난수는 암호학적 해시함수의 입력값으로 이용되므로 해시함수의 안전성에 의해 공격자는 난수값을 알 수 없다.

Impersonation Attack. 공격자는 정상적인 사용자로 가장하기 위해 로그인 단계에서 사용자가 전송하는 정보를 수집할 수 있다. 그러나 제안한 기법은 사용자가 로그인할 때 난수 θ_u를 사용하여 α를 계산하였고 x을 계산할 때 또 다른 난수 r'_i를 사용한다. 공격자가 α나 x을 계산하려면 (θ_u, r'_i)를 모두 알아내야 한다. 그러나 암호학적 해시 함수의 안전성에 의해 이것은 불가능하다.

Password-guessing Attack. 공격자는 공개된 사용자의 ID UID_i를 사용하여, 자신이 선택한 임

의의 Password를 입력하여 로그인을 지속적으로 시도할 수 있다. 그러나 본 논문에서 제안한 기법은 사용자가 AS에 로그인할 때 Password를 입력하지 않는다. 대신 등록 단계에서 사용한 난수 θ_u와 로그인 단계에서 새롭게 생성한 난수 r'_i만 사용한다. 그러므로 Password를 추측으로 알아내는 공격을 시도할 수 없다.

Server-spoofing Attack. 공격자는 사용자가 등록이나 로그인하려는 단계에서 AS를 가장하는 공격을 할 수 있다. AS를 위장하기 위해서는 로그인 단계에서 각 사용자에게 전송할 γ를 계산하는데 필요한 유효한 θ_s를 알아야 한다.

TS에 대해서도 공격자는 AS를 가장하는 것이 불가능하다. 왜냐하면 TS도 γ를 이용하여 AS를 인증하므로 θ_s를 모르면 정당한 γ를 계산할 수 없다.

Forward Secrecy. 공격자가 과거의 세션키를 알아내는 것이 가능해도 사용자 u_i와 TS S_j는 매번 DH 키합의 기법을 사용하여 새로운 Session마다 새로운 세션키를 생성한다. 그래서 본 논문에서 제안한 기법은 Forward Secrecy를 제공한다.

4.2. 성능 분석

우선 각 단계별로 통신량을 분석한다. 먼저 사용자와 TS가 사용하는 ID의 비트 길이는 128 비트이다. 암호학적 해시 함수의 출력의 비트 길이는 160 비트이다. 모든 난수의 길이도 160 비트로 가정한다. 설정 단계에서는 AS가 매개변수 및 공개키를 게시판에 공지하는 것이므로 계산에서 제외한다. ECIES가 사용하는 대칭키 암호기법은 128 비트, 해시 함수는 160 비트를 각각 사용한다. 그러므로 나머지 단계의 통신량을 계산하면 표 1과 같으며 전체 통신량은 4920 비트이다.

이번에는 제안한 기법에 의해서 요구되는 연산량을 계산한다. 표기의 편의를 위해 타원곡선 암호화 및 복호화 연산을 각각 E와 D로 나타내고 암호학적 해시 함수 연산을 H로 나타내자. 단계별 연산량을 표 2로 나타내었다. 덧셈군에서 상수배 연산은 M으로 표기하고, 타원곡선이 사용하는 유한체에서의 곱셈은 m으로 나타낸다.

표 1. 제안한 기법의 통신량

Steps	Users	Comm. Complexity
Registration	$u_i \leftrightarrow$ AS	774 bits
	AS $\leftrightarrow S_j$	742 bits
Log-in	$u_i \leftrightarrow$ AS	576 bits
	AS $\leftrightarrow S_j$	576 bits
TS Auth.	$u_i \leftrightarrow$ AS	640 bits
	AS $\leftrightarrow S_j$	640 bits
Mutual Auth.	$u_i \leftrightarrow S_j$	972 bits

표 2. 제안한 기법의 연산량

Steps	Total Computation Cost
Setup	1H
Registration	3H+2E+2D
Log-in	4H
TS Authentication	4H
Mutual Authentication	6H+2M

끝으로 본 논문에서 제안한 인증기법과 기존 연구 결과의 연산량을 기준으로 성능을 분석하면 다음 표 3과 같다. 본 논문에서 제안하는 기법과 같이 기존 다중서버 인증 기법은 안전한 채널 형성을 위해 공개키 연산을 사용자와 TS가 수행해야 하고, 마지막에 DH 키합의를 수행해야 한다. 그래서 공평한 비교를 위해 공개키 연산과 상수배 연산을 제외하고 비교한다.

표 3. 연산량 비교

	Our Scheme	[22]	[27]
Computational Complexity	18H	27H	26H

Wang과 Ma가 제안한 기법은[23] 공개키 연산을 수행하지 않기 때문에 위의 비교와 다르다. 그러나 본 논문에서 제안한 기법과 동일하게 타원곡선 연산을 사용하기 때문에 암호화 및 복호화 회수를 계산하는 대신 상수배 연산의 개수를 직접 세어 연산량을 비교한다.

조금 더 구체적으로 이야기하면 ECIES는 암호화하는데 2번의 상수배 곱셈, 복호화하는데 1번의 상수배 연산을 요구한다. 제안한 기법은 2번의 암호화와 복호화 연산과 2번의 상수배 곱셈을 요구하므로 총 8번의 상수배 연산을 요구한다. 그러나 나머지

모든 연산은 해시함수와 XOR 연산만으로 구성된다. 더구나 본 논문에서 제안한 기법은 등록단계에서 공개키 연산을 1회 수행하는데 반해, [23]에서 제안한 기법은 인증단계에서도 요구하는 문제가 있다. 특히 [23]에서 제안한 기법은 유한체에서의 곱셈 연산을 6번 요구하는데 비해, 본 논문에서 제안한 기법은 유한체 곱셈 연산을 요구하지 않는다. 정리하면 다음 표 4와 같다.

표 4. 연산량 비교

	Our Scheme	[23]
Hash Function	18H	30H
Scalar Product over $E(\mathrm{F}_{2^k})$	8M	5M
Finite Field Multiplication	0	6m

위 결과로부터 본 논문에서 제안하는 기법은 1회 인증을 수행하는데 필요한 전체 통신량은 4830 비트이다. 특히 기존 기법에 비하여 수행해야 하는 암호학적 해시 함수의 연산량을 31% 개선한 것을 알 수 있다.

Ⅴ. 결론 및 향후 연구내용

본 논문에서는 중재인증 기법을 사용하여 다중서버를 인증하는 기법을 제안하였다. 특히 중재인증 기법의 하나인 보안토큰을 사용하여 다중서버 인증 기법을 구성하는 방법을 구체적으로 제시하였다. 본 논문에서 제안한 기법이 기존 보안 문제를 어떻게 해결하는지 분석하였고, 기존 기법과의 성능비교를 위해 필요한 통신량과 연산량을 제시하였다.

그러나 본 논문에서 제시한 인증기법이 안전하다는 것을 수학적으로 엄밀하게 증명할 필요가 있고, 구체적인 구현을 통하여 실험적으로 성능을 분석할 필요가 있다.

References

[1] ANSI, *Public-key Cryptography for the Financial Services Industry: Elliptic Curve Key Agreement and Key Transport Schemes*, ANSI X.963, 1998.

[2] R. Avanzi, C. Doche, T. Lange, K. Nguyen and F. Vercauteren, *Handbook of Elliptic and Hiperelliptic Curve Cryptography*, Chapman & Hall/CRC Press, 2006.

[3] I. Blake, G. Seroussi and N. Smart, *Elliptic Curve in Cryptography*, Cambridge Press, 1999.

[4] C. Chang and J. Lee, “An efficient and secure multi-server password authentication scheme using smart cards,”in *Proc. 2004 Int. Conf. Cyberworlds* (*CW’04*), pp. 417-422, Tokyo, Japan, Nov. 2004.

[5] K. Chatterjee, A. De, and D. Gupta, “Timestamp based authentication protocol for smart card using ECC,” in *Proc. Web Inform. Syst. Mining (WISM)*, pp. 368-375, Taiyuan, China, Sep. 2011.

[6] Y. Chen, C. Huang and J. Chou. (2009, Apr 21). *A novel multi-server authentication protocol* [Online], retrieved (2013, January 11), available: http:// eprint.iacr.org/2009/176.

[7] J. H. Cheon, H. Kim, S. G. Hahn, and C. Park, “On the discrete logarithm of an elliptic curve,” *Korean Inst. Inform. Security (KIISC)*, vol. 8, no. 3, pp. 95-104, Sep. 1998.

[8] S. M. Cho, S. C. Seo, T. H. Kim, Y. H. Park, and S. Hong, “New efficient scalar multiplication algorithms based on Montgomery ladder method for elliptic curve cryptosystems,” *Korean Inst. Inform. Security (KIISC)*, vol. 19, no. 4, pp. 3-19, Aug. 2009.

[9] W. Diffie and M. Hellman, “New directions in cryptography,” *IEEE Tran. Inform. Theory*, vol. 22, no. 6, pp. 644-654, Nov. 1976.

[10] *IEEE, Standard specifications for public-key cryptography*, IEEE P1363, 1999.

[11] W. Juang, “Efficient multi-server password authenticated key agreement using smart cards,” *IEEE Trans. Comsum. Electron.*, vol. 50, no. 1, pp. 252-255, Feb. 2004.

[12] Y. H. Kim, Y. H. Park, S. Lee, J. Y. Hwang, C. H. Kim, and J. Lim, “An

improved method of scalar multiplication on elliptic curve cryptosystems over small fields of odd characteristic," *Korean Inst. Inform. Security (KIISC)*, vol. 12, no. 6, pp. 105-113, Dec. 2002.

[13] N. Koblitz, "Elliptic curve cryptosystems," *Math. Comp.*, vol. 48, no. 177, pp. 203-209, Jan. 1987.

[14] KISA (2009, Apr 21), *Development of improved Korean digital signature algorithm and standard* [Online], retrieved (2012, November 21), available: http://www.kisa.or.kr/.

[15] Y. Lao and S. Wang, "A secure dynamic ID based remote user authentication scheme for multi-server environment," *Computer Standards and Interfaces*, vol. 13, no. 1, pp. 24-29, Jan. 2009.

[16] I. Lin, M. Hwang and L. Li, "A new remote user authentication scheme for multi-server architecture," *Future Generation Computer Systems*, vol. 19, no. 1, pp. 13-22, Jan. 2003.

[17] Microsoft Corporation (2005), *Web service security: scenarios, patterns and implementation guidance for web service enhancement (WSE)* [Online], retrieved (2012, Oct. 12), available: http://msdn.microsoft.com/en-us.

[18] V. Miller, "Use of elliptic curves in cryptography," in *Proc. Advances Cryptology (CRYPTO 2005)*, pp. 417-426, L.A., U.S.A., Aug. 1985.

[19] *NIST*, *Secure hash standard*, NIST FIPS 180-4, 2012.

[20] A. Pathan and C. Hong, "An improved timestamp-based password authentication scheme," *The 9th Int. Conf. Advanced Commun. Technol. (ICACT 2007),* pp. 804-809, Gangwon, Korea, Feb. 2007.

[21] R. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Comm. ACM*, vol. 21, no. 2, pp. 120-126, Feb. 1978.

[22] J. Tsai, "Efficient multi-server authentication scheme based on one-way hash function without verification table," *Comput. Security*, vol. 27, no 3-4. pp. 115-121, June 2008.

[23] B. Wang and M. Ma, "A smart card based efficient and secured multi-server authentication scheme," *Wireless Pes. Commun.*, vol. 63, no. 3, pp. 361-378, Jan. 2013.

[24] X. Wang, J. Zhang, W. Zhang, and M. Khan, "Cryptanalysis and improvement on two efficient remote user authentication schemes using smart cards," *Computer Standards and Interfaces*, vol. 29, no. 52, pp. 507-512, July 2007.

김 명 선 (Myungsun Kim)

1994년 2월 서강대학교 전자계산학과 졸업

2002년 8월 한국정보통신대학교 정보통신공학과 석사

2012년 8월 서울대학교 수학과 박사

2012년 9월~현재 수원대학교 정보보호학과 조교수

<관심분야> 암호학, 다자간 연산