韓國 컴퓨터情報學會 論文誌
第8卷 第3號, 2003. 9.

2003-8-3-3-1

# 효율적 사용자 인증을 위한 SRP 기반의 독립적 인증 프로토콜 설계

정 경 숙*, 정 태 충**

# Design of SRP based Independent authentication protocol for efficient user authentication

Kyoungsook Jung*, TaeChoong Chung**

## 요 약

본 논문은 클라이언트-서버 환경이 발달되어 있는 현재의 시스템들에서 사용자 인증을 효율적으로 할 수 있는 프로토콜 설계를 제안한다. 기존의 패스워드 기반 프로토콜들은 클라이언트와 서버 사이에 인증기관(CA)을 통하여 사용자를 인증하는 데에 반해, 본 논문에서는 사용자와 서버가 독립적으로 키 교환 및 인증을 하는 패스워드 기반 프로토콜을 제안함으로써 사용자 인증을 효율적으로 할 수 있도록 하였다. 패스워드는 충분한 랜덤성을 가지지 못할 뿐만 아니라 패스워드의 길이가 짧기 때문에 오로지 패스워드 만을 이용해 인증 및 키 교환을 하는 것은 많은 주의를 요한다. 그러므로 Diffie-Hellman 키교환 방식에 기반한 SRP 프로토콜과 ECDSA의 서명 기법을 적용하여 안전성이 높은 프로토콜을 제안한다. 또한 기존의 다른 프로토콜과의 라운드 횟수 및 해쉬 함수의 연산과 지수 연산의 횟수를 비교 분석함으로써 제안하는 프로토콜의 효율성을 설명하였다.

## Abstract

This paper proposes protocol design that can do user authentication efficiently in current systems that client-server environment is developed. And proposes a password-based authentication protocol suitable to certification through trustless network or key exchange. While the existing password-base protocols certify users through certification authority (CA) between client and server, the proposed protocol in this paper, users and server exchange keys and perform authentication without help of CA. To ameliorate the drawback of password-based protocols causing by the short length and randomness of password, the proposed protocol uses the signature techniques of ECDSA and the SRP protocol based on Diffie-Hellman key exchange method. Also, by with compare to round number and Hash function number and exponential operation of existing protocols, we explained efficiency of proposed protocol.

▸ Keywords : user authentication, key exchange, ECDSA

*, ** 경희대학교 컴퓨터공학과

# Ⅰ. 서 론

인터넷이 발달함에 따라, 지식 정보화 사회에서는 온라인상에 노출되는 정보들의 불법적인 위조와 변조, 신분위장 등 각종 위협들이 예상되어 지고 있다. 따라서 인터넷상에서 사용자와 정보 제공자 간의 정보 보호를 위해서는 상호간의 인증이 중요한 문제로 대두되었다. PKI에서는 사용자의 신상 정보와 공개키를 확인할 수 있게 하기 위해서 제 3자인 인증기관으로부터 인증서를 발급 받는다. 그러나 빈번한 인증서의 발급으로 인한 통화량의 증가, 비용과 시간의 소모, 키 관리 등 다양한 문제가 발생하고 있다. 따라서 사용자간의 통신과 전자상거래 시 제3의 신뢰기관과의 접촉 없이 독립적으로 사용자 인증 및 키 분배를 안전할 수 있는 시스템에 대한 연구가 필요하게 되었다. 기존에 연구된 프로토콜은 이러한 목적으로 다양하게 설계되었다. 그러나 패스워드를 이용하는 인증 프로토콜은 사용자가 자신이 설정한 패스워드를 이용해 인증할 수 있기 때문에 가장 효율적인 방법으로 연구되고 있다[1].

그러나 패스워드는 충분한 랜덤성을 가지지 못할 뿐만 아니라 패스워드의 길이가 짧기 때문에 오로지 패스워드만을 이용해 인증이나 키 교환을 하는 것은 많은 주의를 요한다.[2,3] 그러므로 패스워드 기반 프로토콜들은 다양한 공격들로부터 대응할 수 있도록 구성되어야 한다.

본 논문에서 제안하는 프로토콜은 기존의 제 3의 신뢰기관에 접근하지 않고, 독립적으로 클라이언트와 서버간의 사용자 인증과 키 교환이 가능한 패스워드 기반 프로토콜이다. 이렇게 설계함으로써 서버의 사용자 인증을 효율적으로 할 수 있는 프로토콜을 설계하고자 하였다. 또한 ECDSA 서명기법을 이용하여 신뢰할 수 없는 네트워크를 통해서도 사용자를 인증하거나 키를 교환할 때 안전성을 향상시키는 것을 목적으로 하고 있다.

본 논문은 2장에서 패스워드 기반 인증 프로토콜과 ECDSA의 절차를 간단히 설명하고, 3장에서는 ECDSA기법을 이용한 패스워드 기반 인증 및 키 교환 프로토콜을 제안한다. 4장에서는 제안 프로토콜의 특징, 안전성 및 효율성에 대하여 설명하고, 5장에서는 결론 및 향후 연구 방안에 대하여 논한다.

# Ⅱ. 관련 연구

## 1. 패스워드 기반 프로토콜

패스워드 기반의 프로토콜은 이산 대수 문제의 어려움을 기반으로 세션키를 생성하고, 상호 인증을 한다. 패스워드는 이들 프로토콜에서 상호 인증하는 수단으로 사용되며 공통의 키를 생성하는데 사용된다. 프로토콜 내에서 패스워드의 용도를 살펴보면 주고받는 메시지의 암호화키로 사용되어진다. 또는 프로토콜 실행에서 수학적 계산에서 그룹의 생성자로 사용하거나 지수연산에서 지수로 사용된다. 이들의 대표적인 프로토콜로는 EKE, A-EKE, PAK, B-SPEKE, SRP 등이 있다[4, 5, 6].

### 1.1 A-EKE

EKE(*Encrypted Key Exchange*)는 패스워드를 암호화키로 이용하여 사용자의 임의의 생성 값을 서버에게 전달하고, 패스워드를 공유한 서버는 이를 이용하여 세션키를 만들고, 만들어진 세션키를 이용하여 인증하는 방법이다.

A-EKE는 이 방법과 거의 유사하지만 패스워드를 해쉬하여 암호화키로 사용하는 것과 1회의 통신 회수를 추가하여 총 5회로 상호 인증과 세션키를 교환하는 방법이다.

### 1.2 SNAPI-X

SNAPI-X는 공개키 암호 알고리즘인 RSA를 사용하였다. 전체적인 프로토콜의 흐름은 OKE와 매우 유사하다. SNAPI-X의 다른 점은 서버가 생성한 키 쌍을 이용한 것으로 사용자는 서버의 공개키를 이용하여 암호화하고 서버는 비밀키를 이용하여 복호화 한다. 서버는 패스워드를 저장하는 것이 아니라 검증자를 저장하게 되어 서버의 파일이 노출되어도 사용자로 위장하여 공격하는 것을 방지할 수 있다. 총 소요되는 통신회수는 5회로써 비교적 많으며, RSA 암호화, 복호화시 실행되는 지수연산을 제외하더라도 총 8회의 지수연산을 실행함으로써 프로토콜의 실행시간이 많이 걸린다는 단점이 있다.

### 1.3 OKE

OKE는 SNAPI-X와 같이 RSA 공개키 암호 알고리즘을 이용하였다. 사용자는 사용자와 서버 사이의 임의의 키 쌍을 생성하고, 서버에게 자신의 공개키를 전송한다. 서버는 사용자의 공개키를 이용하여 서버가 생성한 임의의 세션키를 암호화하여 전송한다. 사용자는 이 키를 복호화한 다음, 임의의 키 값을 해쉬하여 다시 서버에게 전송하고, 이 결과를 비교한 서버는 사용자를 인증하게 된다. 서버는 다시 해쉬를 하여 사용자에게 전송하고, 사용자는 결과를 비교하여 서버를 인증하여 상호인증을 하게 된다.

이와 같이 OKE 알고리즘은 공개키 알고리즘의 암호화, 복호화를 이용한 세션키 생성과 상호 인증 프로토콜로써 총 네 번의 통신을 수행한다. 하지만, 세션키를 공유하는 방법은 사용자와 서버 간에 키 교환을 통한 세션키 생성이 아닌 서버에 의한 일방적인 임의의 값 선택에 의한 것이다. 그리고 사용자에 의해 선택되어지는 키 쌍은 임의로 생성하게 된다. 또한 서버의 패스워드 파일에는 검증자가 저장되는 것이 아니라 패스워드가 직접 저장되기 때문에 파일이 노출되었을 때 프로토콜은 공격이 가능하게 된다.

### 1.4 B-SPEKE

B-SPEKE는 기본적으로 SPEKE 형식을 따르고 있다. SPEKE는 패스워드의 정보를 그룹의 생성자로 사용해 세션키를 생성하며, 이를 이용하여 상호 인증을 하는 방식이다. 만약, 상호 인증을 하는 검증자가 노출되었을 때 위장 공격이 가능하여 상호 인증과 세션키가 정상적으로 실행되는 문제를 갖고 있다. 이런 SPEKE의 문제점을 보완하기 위해서 B-SPEKE가 제안되었다. B-SPEKE는 SPEKE에 별도의 인증 값을 추가하여 문제점을 보완하였으나, 사용자와 서버 간의 통신회수는 4회로 유지하였으며, 지수연산은 사용자와 서버에서 각각 3회, 4회를 실행하고 있다.

### 1.5 SRP

SRP(Secure Remote Password) 프로토콜은 Diffie-Hellman 키교환 방식에 기반한 프로토콜로 두 참여자의 키 교환 설정단계에서 이산대수 문제를 이용하여 구성하고, 두 참여자간의 상호인증은 해쉬 함수를 이용하여 구성된다[6]. SRP 프로토콜에서는 개인키와 공개키 대신 패스워드(Password)와 검증자(Verifier)라는 용어를 쓰고 있다. 패스워드는 개인키와 달리 사용자가 암기 가능한 크기여야하며, 검증자는 공개키와 달리 공개되지 않으며 서버의 데이터베이스에 안전하게 저장된 후 사용자를 인증하기 위한 기반 정보로 쓰이게 된다. 엄밀하게는 패스워드가 개인키 역할을 하는 것은 아니며 패스워드를 이용하여 사용자의 개인키가 유도되어진다. 본 논문에서는 일방향 인증을 수행하기 위해 상호 연동적으로 실행되는 SRP 프로토콜을 이용하며, 다음과 같이 프로토콜을 실행한다. 〈그림 1〉은 SRP 프로토콜의 실행 과정을 간략하게 나타내고 있다.

▶ 시스템 설정

n이 큰 소수(large prime number)이라고 할 때, 소수가 되는 $q = 2*n + 1$ , $p = k*q + 1$ 을 선택한다.(k는 짝수) $g \in NZp^*$는 order q의 원소라고 하면, $g^q \equiv 1 \bmod p$ 와 같은 형태를 갖게 된다. 여기에서 밑수(base) g의 이산대수를 계산하는 것은 불가능하다고 가정한다.

▶ 실행 단계

① 우선 클라이언트가 pwd를 설정한다.

② salt값 s를 선택하고 $x = Hash(s,pwd)$, $v = g^x \bmod p$를 계산한다.

③ 클라이언트는 사전지식으로 (pwd, q, p, g)를 갖고, 서버에게 (s, v)를 전달한다.

④ 클라이언트는 난수 $a \in NZq$를 생성하고, 자신의 임시 공개키 $A = g^a$를 계산한 다음 서버에게 전송한다.

⑤ 서버에서는 난수 $b, u \in NZq$를 생성하여 자신의 임시 공개키 $B = v + g^b$를 계산해 (B, u)를 클라이언트에 전송한다.

⑥ 클라이언트와 서버는 각각 지니고 있는 값을 이용해 공통 지수값 S를 계산하고, $SK = Hash(S)$를 계산해 상호 동의된 세션키를 설정한다.

⑦ 클라이언트는 서버에게 정확한 S를 지니고 있음을 증명하는 M을 구성해 전달하면 서버가 검증을 통해 확인한다.

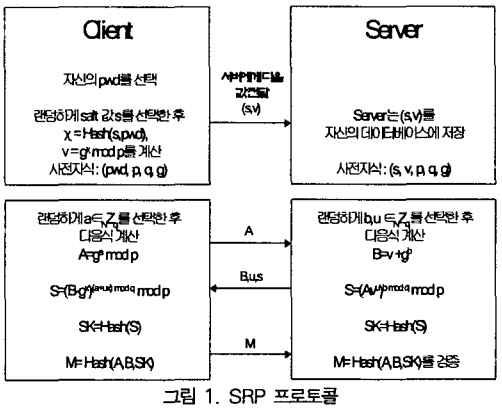


그림 1. SRP 프로토콜
Fig. 1 SRP Protocol

## 2. 디지털 서명

암호의 사용이 확산되고, 상거래나 개인의 목적으로 사용된다면, 그때 전자적인 메시지와 문서는 종이 문서에 사용된 서명과 동등한 무언가가 필요하게 될 것이다. 그것은 디지털 메시지가 특정한 사람에 의해 보내졌다는 것을 모든 사람이 만족하도록 약정할 수 있는 방법이다. 이 방법이 디지털 서명이다.

디지털 서명의 개념은 자료에 부착되거나 논리적으로 결합된 전자적 형태의 자료로서 서명자의 신원을 확인하고 자료의 내용에 대한 그 사람의 승인을 나타낼 목적으로 사용된 것을 의미한다. 디지털 서명은 그 디지털 서명에 작성자로 기재된 사람이 그 전자 문서를 작성하였다는 사실과 작성 내용이 송·수신 과정에서 위·변조되지 않았다는 사실을 증명하고 또한 작성자가 그 전자문서 작성 사실을 나중에 부인할 수 없게 하는 역할을 한다.

### 2.1 DSA(Digital Signature Algorithm)

DSA는 이산 대수 문제에 기반을 두고 있는 공개키 알고리즘이다. 본 논문에서 DSA를 언급하는 이유는 ECC가 기반하고 있는 수학적 문제인 타원 곡선 이산 대수 문제가 이산 대수 문제에 근원을 두고 있을 뿐 아니라 이산 대수 문제를 이용해 개발된 대부분의 프로토콜이 타원 곡선 이산 대수 문제에 적용이 가능하기 때문이다[7].

여기서 살펴보는 DSA는 ECDSA(Elliptic Curve Digital Signature Algorithm)으로 확장되어 진다. 전자서명 알고리즘은 키 생성과정, 서명 생성과정, 서명 확인 과정의 3부분으로 나누어지며 DSA의 각 부분은 다음과 같다.

▶ 키 생성

① 소수 q를 선택한다.
② q|p-1을 만족하는 소수 p를 선택한다.
③ $h \in Zp^*$를 선택하고 $g = h(p-1)/q \bmod p$을 계산한다. $g \neq 1$ 일때까지 반복한다(g는 군 $Zp^*$에 대해 위수(order) q를 갖는 생성자이다).
④ [1, q-1]내에 존재하는 난수 x를 선택한다.
⑤ $y=gx \bmod p$를 계산한다.
⑥ 생성된 공개키와 비밀키는 다음과 같다.
클라이언트의 공개키 : (p,q,g,y)
클라이언트의 개인키 : x

▶ 서명 생성

① [1, q-1]내에 존재하는 난수 k를 선택한다.
② $r = (gk \bmod p) \bmod q$를 계산한다.
③ k-1 mod q를 계산한다.
④ $s=k-1\{h(m)+xr\} \bmod q$를 계산한다. 여기서 h는 해쉬 함수이다.
⑤ 만일 s=0이면 ①부터 다시 반복한다.
⑥ m에 대한 서명쌍 : (r, s)

▶ 서명확인

① 클라이언트의 공개키 (p, q, g, y)를 획득한다.
② $w = s - 1 \bmod q$와 h(m)을 계산한다.
③ $u1 = h(m)w \bmod q$와 $u2 = rw \bmod q$를 계산한다.
④ $v = (gu1yu2 \bmod p) \bmod q$를 계산한다.
⑤ v = r인 경우에만 정당한 서명문으로 인정한다.

키 생성과정에서 DSA는 연산과정은 군 $Zp^*$에서 이루어지기 때문에 군의 정의에 따라 x값에 대한 y는 항상 존재하게 된다.

또한 공개키인 (p, q, g, y)가 알려진다 하더라도 개인키 x를 알 수 없다. 왜냐하면 앞서 말했듯이 x를 구하는 문제는 수학적인 어려움으로 이를 구하기 위한 효율적인 알고리즘이 존재하지 않기 때문이다.

서명문 생성과정에서 s를 구하기 위해서는 반드시 개인키인 x를 알아야 하며 x대신 임의의 값을 사용할 경우 서명확인과정에서 ⑤의 등식이 성립하지 않게 된다. 서명문 확인과정에서는 개인키가 필요하지 않으며 공개키만으로 서명의 진위를 확인 할 수 있다.

### 2.2 ECDSA

ECDSA는 DSA를 타원곡선 알고리즘으로 옮긴 것으로 ANSI X9.62로 표준화되었다[8].

ECDSA와 DSA의 중요 차이점은 $r$의 생성에 있다. DSA는 $r$을 임의의 $g^k \bmod p$를 선택하고 계산한 후, $\bmod q$를 계산하여 얻는다. 그러나 ECDSA에서 $r$은 임의의 점 $kP$의 $x$좌표를 하여 얻는다. ECDSA가 160비트 $q$와 1024비트 $p$를 가진 DSA와 비슷한 안전도를 갖기 위해서는 매개변수 $n$이 약 160비트이면 된다. 이 경우 DSA와 ECDSA는 같은 서명길이(320비트)를 갖지만, 기존의 DSA알고리즘보다는 ECDSA를 사용하는 것이 키 bit당 암호화 정도가 강하다고 할 수 있다[9]. 다음 <표 1>은 DSA와 ECDSA의 기호를 비교한 것이다.

표 1. DSA와 ECDSA 기호 비교
Table 1. Compare DSA and ECDSA notation

| DSA | ECDSA |
|---|---|
| q | n |
| g | P |
| x | d |
| y | Q |

▶ 실행 단계

(1) ECDSA 키 생성
① Zp에서 정의된 타원곡선 E를 선택한다.
② 위수가 n인 점 P∈E(Zp)를 선택한다.
③ 구간 [2,n-2]에서 난수 d를 선택한다.
④ Q = dP를 계산한다.
⑤ 사용자의 공개키는 (E, P, n, Q)이며 사용자의 비밀키는 d이다.

(2) ECDSA 서명 생성
① 구간 [1, n-1]에서 난수 a를 선택한다.
② aP=(x1, y1)과 r = x1 mod n을 계산한다.(x1은 정수)
③ r=0이면, ①단계로 되돌아간다.
④ a-1 mod n을 계산한다.
⑤ s=a-1{h(m)+dr} mod n을 계산한다.
(h : 메세지 m의 SHA-1 해쉬 알고리즘)
⑥ s=0이면, ①단계로 되돌아간다.
⑦ 메세지 m에 대한 서명은 (r, s)이다.

(3) ECDSA 서명 검증
① 사용자의 서명 (r, s)를 검증하기 위해서 서버는 사용자의 인증된 공개키 (E, P, n, Q)를 얻는다.
② r과 s가 [1,n-1]에 있는 지 확인한다.
③ w = s - 1 mod n과 h(m)을 계산한다.
④ u1 = h(m)w mod n와 u2 = rw mod n을 계산한다.
⑤ u1P + u2Q = (x0,y0)를 계산 v = x0 mod n
⑥ 만약 v=r이면 올바른 서명이다.

## III. 제안 프로토콜

기존의 SRP는 키 교환 설정 단계가 이산대수문제에 근거하므로 타원곡선을 적용할 수 있다. 제안한 프로토콜은 그림 2.와 같이 안전성 향상 및 프로토콜의 간략화를 위해 일방향 최적화된 SRP에 ECDSA를 적용하였다.

본 논문에서 사용된 용어의 표기법은 〈표 2〉와 같다.

표 2. 용어 설명
Table 2. Notation

| 파라미터 | 설 명 |
|---|---|
| s | 클라이언트의 salt 값 |
| pwd | 클라이언트의 패스워드 |
| x | 개인키 |
| u | 랜덤 은닉 파라메터, 공개됨 |
| a,b | 랜덤 생성된 개인키, 공개되지 않음 |
| A,B | 동의된 공개키 |
| h() | 일방향 해쉬함수 |
| g | 곱셈군 Zp*의 생성자 |
| p,q | 임의의 소수, p=2*q+1 |
| P,Q | 타원곡선 상의 임의의 점 |

▶ 설 정

사전 공격을 보다 어렵게 하기 위하여 본 논문에서는 salt 값을 추가해 pwd와 연접시켜 일방향 함수에 입력한다 [10].

$$x = \mathrm{Hash}(salt, pwd)$$

타원곡선 E를 단순화하기 위해 K = Fp = Zp(p : 소수, p개의 원소를 갖는 유한체)로, 타원곡선은 $y2 = x3 + ax + b(a,b \in Zp)(4a3+27b2 \neq 0(\mathrm{mod}\ p))$와 무한원점(0)을 말하기로 한다. n은 160비트 이상의 크기를 갖는 소수, 타원곡선 E(Zp)는 $y2 = x3 + ax + b(a,b \in Zp)$의 방정식을 만족하는 Zp상의 점들과 무한점으로 이루어진 집합을 의미한다.

① Zp위에 정의된 타원곡선 E(Zp)를 선택한다(E(Zp)는 큰 소수 n에 의해 나누어져야 한다).

② 위수가 n인 p∈E(Zp)를 선택한다.
③ 안전한 일방향 해쉬함수 h를 선택한다.
④ 클라이언트가 pwd를 설정한다.
⑤ salt값 s를 선택하고 x = h(s,pwd)를 계산하고 점 Q = xP를 계산한다.
⑥ 클라이언트는 사전지식으로 (E, P, Q, pwd, q, p) 를 갖고, 서버에게 (E, P, n, Q, s)를 전달한다.

▶ 실행 단계

① 클라이언트는 구간 [1, n-1]에서 임의의 난수 a를 선택하고, A = aP = (x1, y1)를 서버로 전송한다.
② 서버는 난수 b, u ∈ [1, n-1]를 선택해, B = bP = (x2, y2)를 계산하고, (B, u)를 클라이언트로 전송한다.
③ 클라이언트와 서버는 각각 공통 값 S 계산한다.
④ i = a - 1{h(S) + xr}를 계산한다. 그리하여 서명 쌍은 (A, i)가 되고, 서버에 i값을 전달한다.
⑤ 서버에서 w = i - 1 mod n을 계산한 후, u1 = h(S)w mod n, u2 = Aw mod n를 계산한다.
⑥ u1P + u2Q = (x0, y0), v = x0 mod n을 계산한 후 v = A이면 올바른 서명이다.
⑦ 클라이언트와 서버는 세션키 SK = h(S)를 생성한다.

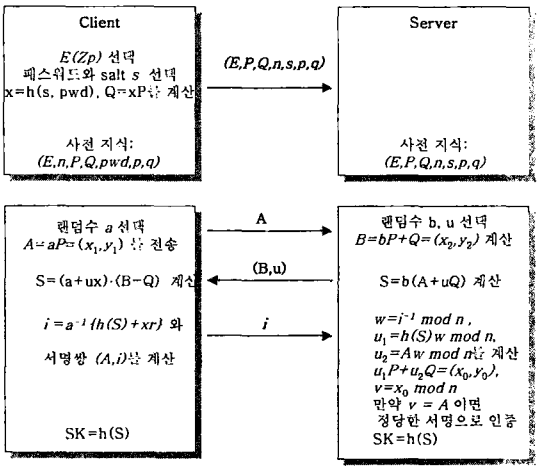


그림 2. 제안 프로토콜
Fig 2. Proposed Protocol

실행 단계에서는 Diffie-Hellman 키 교환 방법을 제안 프로토콜에 맞게 변형하여 클라이언트와 서버가 안전하게 키를 교환하도록 하였다.

클라이언트와 서버가 키 교환을 하게 되면, 클라이언트는 난수 a를 선택하고, 공개키 A = aP = (x1, y1)를 계산한다. 서버도 독립적으로 난수 b와 u를 선택하고, 공개키 B = bP + Q = (x2, y2)를 계산한다. 클라이언트와 서버는 a, b 값은 각자 보관하고, A와 B값을 상대편에게 전송한다. 이제 클라이언트는 키를 S = (a + ux) · (B - Q)로 계산하고, 서버는 키를 S=b(A+uQ)로 계산한다.

이 두 가지 계산법은 동일한 결과를 가지므로 서버와 클라이언트는 동일한 키를 공유할 수 있게 된다.

$$
\begin{aligned}
S &= (a + ux) \cdot (B - Q) \\
&= aB + uxB - aQ - uxQ \\
&= a \cdot (bP + Q) + ux \cdot (bP + Q) - axP - ux2P \\
&= abP + axP + uxbP + ux2P - axP - ux2P \\
&= abP + uxbP \\
&= b \cdot (aP + uxP) \\
&= b \cdot (A + uQ)
\end{aligned}
$$

## Ⅳ. 제안 프로토콜 분석

### 1. 제안 프로토콜의 특징

네트워크 상에서의 통신 회수는 네트워크 자원의 효율성과 네트워크상의 지연 등을 고려할 때 적을수록 장점을 갖는다. 그러나 통신 횟수가 줄어도 보안의 안전도는 변함이 없어야 한다.

제안 프로토콜은 사용자의 서명을 이용함으로써 사용자의 부인 방지를 제공하고, 인증에 사용되는 키 쌍 (A, i)는 두 통신자 사이의 키 교환에 의해 생성된다. 그림 2.를 이용해 설명하면, A와 i값은 사용자에 의해 생성되는 서명 값으로 사용자가 자신의 비밀키 x = h(s, pwd)를 이용해 서명 값을 생성하고, 서버는 사용자의 공개키(E, P, Q, n, s)를 이용해 서명을 검증하여 인증하게 됨으로 생성한 세션키 S에 대한 사용자의 부인을 방지할 수 있다. 그리고, 사용자의 인증에 사용되는 키 쌍은 매 세션마다 새롭게 생성되는 값이며, 세션키도 매 세션 마다 생성된다.

뿐만 아니라, 표 4.에서와 같이 pass의 횟수를 비교해 보면 SRP나 A-EKE 등의 라운드 횟수보다 더 적은 3 라운드에 의해 안전성을 제공하고 있음을 알 수 있다.

## 2. 제안 프로토콜의 안전성

### ▶ 사전 공격

사전공격(dictionary attack)은 공격자가 사용자의 패스워드를 추측하여 실제 메시지에서 드러나는 값에 대입하여 결과를 비교해 실제 패스워드를 찾는 방법이다. 제안 프로토콜에서는 사용자가 자신의 패스워드와 서버로부터 전송받은 salt값에 기반하여 개인키를 유도해낸다. 여기서 사용된 salt값은 패스워드의 랜덤성을 보장하기 위해서 사용되었다. 공격자는 사용자의 개인키가 공격자가 원하는 값을 갖도록 salt값을 조작할 수 없다. 그 이유는 공격자는 사용자의 패스워드를 알지 못하며 개인키를 유도하는 패스워드와 salt값을 일방향 해쉬 함수에 적용했기 때문이다. 공격자가 salt값을 조작할 경우라도, 사용자의 개인키가 잘못된 값으로 계산되게 함으로써 인증이 성립되지 않기 때문에 사전 공격을 차단할 수 있다.

### ▶ 재실행 공격

공격자가 사용자의 메시지를 재전송하여 이미 정상적인 사용자에 의해 생성된 이전키(old session key)를 다시 생성하기 위함이다.

이 공격 방법은 사용자와 서버 간에 항상 임의의 값을 사용하기 때문에 불가능 하다. 제안 프로토콜에서는 클라이언트에서 생성되는 임의의 값 a와 서버에서 생성되는 임의의 값 b가 매 세션마다 새롭게 생성됨으로 반복에 의한 이전 키의 생성이 불가능하다.

### ▶ Perfect Forward Secrecy

Perfect Forward Secrecy는 공격자가 비밀키를 알아낸다고 하더라도, 현재의 세션키를 유추할 수 없어야 함을 나타낸다. 이것을 제공하기 위해서는 클라이언트와 서버 간에 주고받는 메시지의 내용이 이전 키 생성을 위한 정보와 관련이 없어야 한다. 이를 위해 생성되는 세션키 값이 항상 임의의 값으로 생성되어야 한다. 제안 프로토콜에서는 세션키 값이 랜덤하게 생성되는 a와 b를 사용하므로 Perfect Forward Secrecy를 제공한다.

### ▶ Denning-Sacco attack

Denning-Sacco attack는 이전키를 알아내 패스워드를 알아내는 방법이다. 이 공격은 세션키가 임의의 값으로 매 세션마다 생성되고, 만약 S값이 알려지더라도 S나 SK로부터 새로운 정보를 유도하지 못한다. 또한 클라이언트의 패스워드에 대한 어떠한 정보를 갖고 있지 않기 때문에 Denning-Sacco attack으로부터 안전하다.

### ▶ 위장 공격

서버의 패스워드 파일이 노출되었을 때 공격자가 프로토콜을 공격하는 것이 위장공격(Impersonation attack)이다. 제안 프로토콜에서는 패스워드를 직접 저장하지 않고 salt값을 추가해 검증자를 생성하여 검증자가 저장된다. 검증자 자체가 사전 공격으로 공격이 가능하지만 앞에서와 같이 차단이 가능하므로 위장 공격은 고려할 의미가 없다.

## 3. 성능분석

프로토콜 수행과정에서 수행속도와 밀접한 관련이 있는 연산은 해쉬 함수 연산과 지수연산의 회수이다. 제안 프로토콜에서는 표 3.에서와 같이 해쉬 함수 연산은 클라이언트에서 2번, 서버에서 1번 수행되며, 지수연산은 클라이언트 2번, 서버에서 2번 이루어진다. 연산이 이루어지는 부분은 다음과 같다. 표 4.는 각 프로토콜의 pass와 계산량을 비교한 것이다.

표 3. 해쉬 함수와 지수 연산 회수
Table 3. Hash Function and Exp. Operation

| | 해쉬함수연산 | 지수연산 |
|---|---|---|
| Client | x=h(s, pwd) | A=aP=(x1,y1) |
| | i=a-1(h(S)+xr) | S=(a+ux)·(B-Q) |
| Server | u1=h(S)·w | B=bP+Q=(x2,y2) |
| | | S=b(A+uQ) |

표 4 . pass와 계산량 비교
Table 4. Compare pass and calculating

| 구분 | 프로 토콜 | Pass | 암호화 | | Hash 함수 | | 지수 연산 | | 랜덤 생성 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Client | Server | Client | Server | Client | Server | Client | Server |
| 공개키 | A-EKE | 5 | RSA서명 | | x | x | 2 | 2 | 1 | 1 |
| | SNAPI-K | 5 | RSA | | 4 | 3 | 3 | 4 | 2 | 2 |
| | AMP | 2 | DSS | | 3 | 2 | 2 | 2 | 1 | 1 |
| | AKEECC | 4 | ECDSA | | 3 | 3 | 2 | 2 | 1 | 1 |
| | A-EKE | 5 | 3 | 3 | 1 | 1 | 2 | 2 | 1 | 1 |
| DH | B-SPEKE | 4 | x | x | 1 | 1 | 3 | 4 | 1 | 2 |
| | AMP | 4 | x | x | 4 | 4 | 2 | 2 | 1 | 1 |
| | SRP | 4 | x | x | 3 | 2 | 3 | 3 | 1 | 1 |
| | 제안 프로토콜 | 3 | ECDSA | | 2 | 1 | 2 | 2 | 1 | 1 |

## V. 결론 및 향후 연구 방향

본 논문에서는 클라이언트-서버 환경에서 서버가 사용자 인증을 효율적으로 할 수 있는 프로토콜 설계를 제안하였다. 또한 패스워드가 랜덤하게 생성됨으로 인해 발생하는 문제점과 패스워드 길이가 짧기 때문에 발생할 수 있는 문제점을 보완하여 패스워드 기반 인증 및 키 교환을 보완하고 안전성을 높이는 프로토콜을 제안하였다. 이 제안 프로토콜은 기존의 패스워드 프로토콜이 클라이언트와 서버 사이에 인증기관을 통하여 인증하던 방식과 달리 사용자와 서버가 독립적으로 키 교환 및 인증을 하는 패스워드 기반 프로토콜을 하도록 Diffie-Hellman 키교환 방식에 기반한 SRP 프로토콜을 사용하고, 안전성과 효율을 높이기 위해 ECDSA의 서명 기법을 적용하여 표4에서 보는 바와 같이 SRP와 ECDSA를 이용함으로써 계산량과 통신량의 효율성을 높였다. 제안된 프로토콜의 안전성은 타원곡선 이산대수 문제와 Diffie-Hellman 문제의 어려움에 기반을 두고 있다. 또한 제안된 프로토콜은 사전 공격, 공모 공격, 위장 공격, 알려지지 않은 키 공유 공격에 강하며, 또한 perfect forward secrecy를 보장한다.

## 참고문헌


[1] Katz, R Q:trovsky and M Yang, "Efficient Password Authenticated Key Exchange Using Human-Memorable Passwords", Eurocrypt'01, LNCS, VoL 2045, pp. 475-494, Springer, Verlag, 2001.

[2] R. Anderson and T. Lomas, "Fortifying Key negotiation schemes with poorly chosen passwords", Electronics Letters, 1994, Vol. 30, No. 13,

[3] V. Boyko, P. MacKenzie, and S. Patel, "Provably Secure Password, Authenticated Key Exchange Using Diffie-Hellman", Eurocrypt'00, LNCS VoL 1807, pp. 156-171, Springer, Verlag, 2000.

[4] S. Bellovin and M. Merritt, "Encrypted Key exchange: password based protocols secure against dictionary attacks", IEEE Comp. Society Symp. on Research in Security and Privacy, 1992. page.72-81.

[5] D. Jablon, "Strong Password-only Authenticated Key Exchange", Computer Comm Review, ACM SIGCOMM, vol.26, no. 5, page. 5-26, 1996.

[6] Thomas Wu, "The Secure Remote Password Protocol", Internet Society Symp. Network and Distributed Systems Security Symposium, 1998, page. 97-111.

[7] I.Biehl, B.Meyer, V.Muller, Differential Fault attacks on elliptic curve cryptosystems, Proceedings of Crypto 2000, LNCS 1880, Springer-Verlag, 2000. 131-146

[8] ANSI X9.62, The elliptic curve digital signature algorithm(ECDSA), draft standard, 1997.

[9] Miller, V., Uses of elliptic curves in cryptography, Advances in Cryptology, CRYPTO 85-Lecture Notes in Computer Science, Volume 218, Springer-Verlag, pages 417-426, 1986.

[10] S. Even, O. Goldreich, A. Lempel "A Randomized Protocol for Signing Contracts", Communications of the ACM, 28, 1985, page. 637-647.


### 저 자 소 개


정 경 숙

1995. 2 경희대학교 수학과 졸업
1997. 8 경희대학교 컴퓨터공학과 석사
1999. 3 ~ 현재 경희대학교 컴퓨터공학과 박사수료
2000. 3 ~ 현재 용인송담대학 컴퓨터소프트웨어학과 겸임 교수
〈관심분야〉 정보보호, 인공지능, 전자상거래, 기계학습

정 태 충

1980. 2 서울대학교 전자공학과 졸업
1982. 2 한국과학기술원 전자계산공학과 석사
1987. 2 한국과학기술원 전자계산공학과 박사
1987. 9 ~ 1988. 3 KIST 시스템 공학센터 선임 연구원
1988. 3 ~ 현재 경희대학교 컴퓨터공학과 정교수
〈관심분야〉 인공지능, 자연어처리, 로봇 에이전트, 정보보호