通信情報保護學會論文誌
第 9 卷 第 1 號, 1999. 3

타원곡선을 이용한 안전한 패스워드 프로토콜

이 용 기*, 이 정 규*

EC-SRP Protocol ; Elliptic Curve Secure Remote Password Protocol

Yong-Ki Lee*, Jong-Kyu Lee*

요 약

본 논문에서는 ECDLP(Elliptic Curve Discrete Logarithm Problem)를 이용한 EC-SRP (Elliptic Curve - Secure Remote Password) 프로토콜을 제안한다. 타원곡선 이산대수 문제를 SRP(Secure Remote Password) 프로토콜에 적용시킴으로써 타원 곡선이 갖는 높은 효율성과 보안성을 갖도록 하였으며, 이와 동시에 타원곡선의 스칼라 곱셈(scalar multiplication)의 회수를 최대한 줄임으로써 최적의 효율성을 갖도록 설계하였다. 또한 랜덤 오라클(random oracle) 모델에서 EC-SRP 프로토콜이 안전한 AKC(Athenticated Key Agreement with Key Confirmation)프로토콜임을 증명하였다.

Abstract

In this paper, we propose an EC-SRP(Elliptic Curve - Secure Remote Password) protocol that uses ECDLP(Elliptic Curve Discrete Logarithm Problem) instead SRP protocols's DLP. Since EC-SRP uses ECDLP, it inherits the high performance and security those are the properties of elliptic curve. And we reduced the number of elliptic curve scalar multiplication to improve EC-SRP protocol's performance. Also we have proved EC-SRP protocol is a secure AKC(Authenticated Key Agreement with Key Confirmation) protocol in a random oracle model.

Ⅰ. 서 론

1990년대에 들어오면서 정보 보안 (Information Security)이라는 말은 도처에서 들을 수 있는 말이 되었으며 정부, 산업계, 개인 등 어느 곳에서나 정보를 전자적 형태로 보존하고

* 한양대학교 전자계산학과

있는 실정이다. 이처럼 정보를 전자적인 형태로 보존하는 전자혁명은 이 전의 방식에 비해 보관, 전송, 검색 등에 있어서 많은 이점을 가져왔지만, 이로 인하여 정보는 더 많은 위협 앞에 놓이게 되었다. 뿐만 아니라 통신의 발달로 인하여 종이에 기반한 문서로 처리되던 많은 업무가 통신을 통해 전자적으로 처리되면서, 안전한 통신을 위한 암호 프로토콜의 개발이 필수적인 요소로 자리잡게 되었다.

암호 프로토콜에서 가장 기본적으로 해결되어야 할 문제가 인증(authentication) 과정으로 현재 통신하고 있는 상대가 실제 의도한 상대인지 확인하는 과정을 의미한다. 따라서 이러한 문제를 해결하기 위한 보다 효율적이고 안전한 프로토콜의 개발이 절실하다. 1997년 Thomas Wu가 제안한 SRP프로토콜 또한 이를 해결하기 위한 프로토콜로서 기존의 패스워드 프로토콜의 장점과 확인자 기반 프로토콜의 장점을 취합한 형태로 설계된 효율적이고 안전한 프로토콜이다[16].

본 논문에서 제안하는 EC-SRP 프로토콜은 SRP 프로토콜이 이산대수(discrete logarithm) 문제를 이용하는 것과는 달리 타원곡선 이산대수(elliptic curve discrete logarithm) 문제를 적용시킴으로써 타원곡선이 갖는 높은 보안성과 효율성의 장점을 계승하도록 하였다. 또한 효율성의 극대화를 위해서 타원곡선 스칼라 곱셈(scalar multiplication)의 회수를 최소화 시켰으며, Bellare-Rogaway 랜덤 오라클 모델을 이용하여 EC-SRP 프로토콜의 안전성을 증명하였다.

Ⅱ 장에서는 SRP 프로토콜을 소개와 함께 EC-SRP 프로토콜에서 쓰이게 될 타원곡선 매개변수에 대해서 기술하고 AKE(Asymmetric Key Exchange) 프로토콜과 EC-AKE(Elliptic Curve - Asymmetric Key Exchange) 프로토콜을 정의한다. 제 Ⅲ 장에서는 본 논문에서 제안하는 EC-SRP 프로토콜의 설계과정을 기술하고 제 Ⅳ 장에서는 EC-SRP 프로토콜의 보안과 성능에 대한 분석한 다음 마지막으로 Ⅴ 장에서 결론을 내린다.

Ⅱ. EC-AKE 프로토콜

1. SRP 프로토콜

사용자 인증 프로토콜은 무엇을 이용하여 인증하느냐에 따라 다음과 같이 세 가지로 분류된다.

- 사용자 자체로서 갖는 특징을 통한 인증 (목소리 식별, 망막 검사 등)
- 사용자가 소유한 물건을 통한 인증 (ID 카드, 스마트 카드 등)
- 사용자가 알고있는 지식을 통한 인증 (패스워드, PIN-Personal Identification Number 등)

이들 모두 개별적으로 인증 프로토콜을 구현하는 데에 쓰일 수 있으나 보다 강력한 인증을 위해서는 두 가지 이상이 혼합되어 사용되기도 한다. 예를 들어 은행 직불카드를 이용하여 ATM(Automated Teller Machine)에서 돈을 인출할 경우, ATM은 사용자의 직불카드뿐 아니라 카드에 해당하는 PIN에 대해서도 함께 인증한다.

첫 번째 방식과 두 번째 방식은 강력한 보안의 효과를 위해 사용되기도 하지만, 그에 따르는 부가적인 하드웨어의 비용 또한 크다. 반면 세 번째 방식은 별도로 필요로 하는 장비가 없기 때문에 큰비용을 들이지 않고 쉽게 사용될 수 있는 방식으로 패스워드 프로토콜이 이에 해당한다.

기존의 널리 쓰이는 패스워드 프로토콜중 challenge-response 방식과 EKE(Encrypted Key Exchange)방식이 있다[2]. 그러나 이러한 방식

은 사용자의 패스워드를 서버에 저장해야 하는 plaintext-equivalence 문제가 있어서, 공격자(adversary)의 공격에 의해 서버에 저장된 사용자 패스워드 파일이 노출될 경우 모든 사용자에 대한 보안이 깨지게 된다. 반면 또 다른 패스워드 프로토콜 방식인 확인자 기반(verifier-based) 프로토콜은 영지식(zero-knowledge)을 이용하기 때문에 사용자의 패스워드가 서버에 존재하지 않을 뿐만 아니라 인증 과정에서 사용자는 패스워드를 서버에 제공하지 않는다. 따라서 서버가 공격자에 의해 공격되더라도 사용자의 확인자만이 공개될 뿐이며, 확인자만으로 서버에게 인증받을 수 없다.

이러한 확인자 기반방식은 공개키(public key) 알고리즘을 사용하며 이에 따라 사용자는 자신의 개인 키(private key)를 소지하고있어야 한다. 향후 20 년간의 안전한 통신을 위해서는 RSA(Rivest Shamir Adleman)와 DSA(Digital Signature Algorithm)의 경우 1024 bit, ECC(Elliptic Curve Cryptosystem)의 경우 160 bit의 랜덤 binary키가 필요하며, 이들 키의 크기는 사용자가 암기하기에는 상당히 크다. 이에 대한 가장 이상적인 대처방안으로 제안되는 것이 스마트 카드(smart card)를 이용하는 것이다. 그러나 스마트 카드와 같이 추가 장비가 필요하다는 것은 시스템 구현상에 비용이 크게 증가하게 된다.

1997년 Thomas Wu가 제안한 SRP(Secure Remote Password) 프로토콜은 이러한 단점을 보완하고 기존의 인증 프로토콜의 장점을 취합한 형태의 패스워드 프로토콜이다. SRP 프로토콜은 이산대수 문제를 이용한 확인자 기반 방식을 쓰고 있으며, 사용자가 개인 키를 암기하거나 이를 저장하기 위한 별도의 장비나 제 3의 신뢰기관(trusted third party)을 필요로 하지 않으면서도 효율적이고 안전한 패스워드 프로토콜을 구사하고 있다.

2. EC-SRP 프로토콜의 타원곡선 매개변수

본 논문에서 제안되는 EC-SRP 프로토콜은 기본적으로 타원곡선 이산대수 문제(ECDLP; Elliptic Curve Discrete Logarithm Problem)에 기반하여 설계된다. 본 절에서는 타원곡선이 암호학적으로 안전하기 위해서 만족해야할 특징들과 EC-SRP 프로토콜에 쓰일 매개변수에 대해 기술하도록 한다.

Characteristic p의 유한체 (finite field) F_q상의 타원곡선 E와 base point $P \in E(F_q)$가 적당하게 선택되어야 한다. ECDLP에 대한 Pollard-rho와 Pohlig-Hellman 알고리즘을 피하기 위해서는 E상의 F_q-rational point의 개수($\#E(F_q)$로 표현됨)가 충분히 큰 소수 n으로 나누어져야 한다. 일반적으로 $n > 2^{160}$이 추천된다[4,5]. 체 F_q가 주어졌을 때, 가능한 한 n이 큰 값을 갖도록 선택되어야 한다. 즉, $n \approx q$이어야 하며 따라서 $E(F_q)$는 almost prime이 된다. 본 논문에서는 $n > 2^{160}$이고 $n > 4\sqrt{q}$ 임을 가정한다. Hasses의 정리에 따르면,

$$(\sqrt{q}-1)^2 \leq \#E(F_q) \leq (\sqrt{q}+1)^2 \qquad (1)$$

이다. 따라서 $n > 4\sqrt{q}$는 n^2이 $\#E(F_q)$을 나누지 못함을 의미하며, $E(F_q)$는 위수(order) n의 유일한 부분군(subgroup)을 갖게 된다. 또한

$$(\sqrt{q}+1)^2 - (\sqrt{q}-1)^2 = 4\sqrt{q} \qquad (2)$$

이므로 $\#E(F_q) = nh$를 만족하는 유일한 정수 h가 존재하며 $h = \lfloor (\sqrt{q}+1)^2 / n \rfloor$이다. 작은 부분군 공격법 (small subgroup attack)으로부터 보호하기 위해서는 점 P가 소수의 위수인 n을 갖도록 해야 한다. reduction 알고리즘을 피하기 위해서는 곡선(curve)이 비초특이(non-

supersingular, 즉 p가 $(q+1-\#E(F_q))$를 나누지 말아야 한다)이어야 한다. 좀 더 일반적으로, 모든 $1 \le k \le C$에 대해 n이 q^k-1를 나누지 말아야 한다. 여기서 C는 F_{q^C}상에서 이산대수(discrete logarithm)가 실행 불가능하도록 충분히 큰 소수이어야 한다($C=20$이면 충분하다[1]). F_q-anomalous 에 대한 Semaev의 공격, Smart의 공격, Satoh and Araki의 공격으로부터 방어하기 위해서는 곡선이 F_q-anomalous이지 말아야 한다(즉, $\#E(F_q) \neq q$).

이러한 공격 뿐 아니라 미래에 발견될 공격을 막는 방법은 $\#E(F_q)$이 큰 소수로 나누어지는 조건하에서 타원곡선 E를 랜덤하게 선택하는 것이다. 랜덤 곡선(random curve)이 이러한 특정 곡선에 대한 공격법 (special-purpose attack)에 깨질 확률은 무시할 수 있을 정도(negligible)로 작다. SHA-1과 같은 일방향 함수(one-way function)를 이용하여 곡선(curve)의 계수를 결정함으로써 랜덤 곡선을 만드는 것도 하나의 방법이 될 수 있다.

타원곡선이 갖는 시스템 매개변수와 이에 대한 확인 과정은 다음과 같다.

시스템 매개변수 (System parameters).

1. 체의 크기 (field size) q; 여기서 q는 소수의 승수이다. 실제로는 $q=p$이거나 $q=2^m$인 경우에 대해 구현된다.
2. 체(field) F_q의 두 원소 a와 b; a와 b는 F_q상의 타원곡선 E의 방정식을 결정하게 된다. ($p>3$인 경우: $y^2=x^3+ax+b$, $p=2$인 경우: $y^2+xy = x^3+ax^2+b$)
3. 체 F_q의 두 원소 x_P와 y_P; 점 $P=(x_P, y_P)$는 $E(F_q)$에서 소수 위수 (prime order)를 갖는 한 점이다.
   ($P \neq O$ 이며, 여기서 O는 무한원점 - point at infinity이다)
4. 점 P의 위수(*order*) n.

시스템 매개변수 확인과정 (System parameter validation).

: 시스템 매개변수 $(q,a,b,P=(x_P,y_P),n)$는 위의 조건을 만족하도록 다음과 같이 검증될 수 있다.

1. q가 소수의 승수 (prime power)임을 검증한다.
2. a, b, x_P, y_P가 F_q의 원소임을 확인한다.
3. a, b에 대한 타원곡선이 비특이(non-singular)임을 확인한다.
   ($p>3$인 경우: $4a^3+27b^2 \neq 0$, $p=2$인 경우: $b \neq 0$)
4. 점 P가 타원곡선 E의 한 점인 것과 $P \neq O$임을 확인한다.
5. $n>4\sqrt{q}$이고, n이 소수이고, n이 충분히 큰 소수($e.g., n>2^{160}$)임을 확인한다.
6. $nP=O$임을 확인한다.
7. 타원곡선의 특정 부류 (special class)에 대해 알려진 공격들을 피하기 위해, 모든 $1 \le k \le 20$에 대해 n이 q^k-1을 나누지 않음과 $n \neq q$임을 확인한다.

시스템 매개변수 (q,a,b,P,n)가 주어졌을 때, A의 개인키 d는 $[1,n-1]$중에서 랜덤하게 선택되며, 공개키는 타원곡선 상의 한 점 $Q=dP$가 된다. 프로토콜의 진행 중 각 개체에 대해 고정(static, long-term) 공개키와 임시(ephemeral, short-term) 공개키가 있다. 생성된 공개키는 다음의 확인과정을 거쳐야 한다.

공개키 확인과정 (Public-key validation).

1. $Q \neq O$임을 확인한다.
2. x_Q와 y_Q가 F_q의 원소임을 확인한다.
3. Q가 방정식 E를 만족하는 점임을 확인한다.
4. $nQ=O$임을 확인한다.

공개키 확인과정에서 많은 계산 량을 요구하는 부분은 과정 4.의 스칼라 곱셈(scalar multiplication)이다. 고정 공개키에 대한 확인(validation)은 키가 만들어질 때 단 한번만 이루어지면 되지만, 프로토콜 진행 중 새롭게 생성되는 임시 공개키는 각 프로토콜마다 확인되어야 한다. 이러한 임시 공개키의 확인과정은 상당히 치명적일 수 있기 때문에 생략되기도 한다. 그 대신 공유된 비밀정보 K에 대한 확인과정을 거친다.

3. Asymmetric key exchange (AKE)

본 절에서는 공개키 알고리즘을 이용하여 양자간의 공유키를 설정하는 프로토콜의 일반적인 형태인 AKE(Asymmetric Key Exchange) 프로토콜을 공식화하고, 제 4절에서 AKE 프로토콜에 타원 곡선을 적용한 EC-AKE(Elliptic Curve - Asymmetric Key Exchange)의 형태를 정의하도록 한다.

먼저 AKE 프로토콜에 사용되는 매개변수 함수의 정의는 〈표 1〉과 같다.

〈표 1〉 AKE의 매개변수

a, b	랜덤 수 : 임시 개인키로 사용
x, y	랜덤 수 : 고정 개인키로 사용
$G(\cdot)$	공개키 생성 함수
$Q_1(\cdot,\cdot)$,$Q_2(\cdot,\cdot)$	개인키 조합 함수
$R_1(\cdot,\cdot)$,$R_2(\cdot,\cdot)$	공개키 조합 함수
$S(\cdot,\cdot)$	비밀정보 생성 함수
K	공유된 비밀정보

AKE 프로토콜이 성립되려면 다음의 등식이 만족해야한다. 그러나 이 방정식 자체로서는 보안성의 의미가 전혀 없다. 보안성은 이 방정식에서 쓰이는 함수 $G()$,$Q_1()$,$Q_2()$,$R_1()$,$R_2()$,$S()$의 선택에 달려 있다.

$$(\forall a,x,b,y)\ S(Q_1(a,x),R_2\ (G(b),G(y))) = S(Q_2(b,y),R_1\ (G(a),G(x))) \quad (3)$$

Carol과 Steve 양자간의 AKE 프로토콜이 이루어진다고 하자. 이때, x,$G(x)$를 Carol의 개인키와 공개키라 하고 y,$G(y)$를 Steve의 개인키와 공개키라고 하면 일반적인 형태의 AKE는 〈그림 1〉과 같다.

Carol		Steve
(랜덤 수 a를 생성)	$G(a)$ $\Rightarrow$	
	$G(b)$ $\Leftarrow$	(랜덤 수 b를 생성)
$K=S(Q_1(a,x),R_2$ $(G(b),G(y)))$		$K=S(Q_2(b,y),$ $R_1(G(a),G(x)))$

〈그림 1〉 AKE 프로토콜

여기에서 쓰인 네 개의 매개변수 중 x와 y는 고정 개인키 (static private key)인 반면, a와 b는 키 설정 세션마다 달라지는 임시 개인키 (ephemeral private key)이다. 프로토콜을 시작하는 개체(initiator, *i.e.* Carol)는 개인키 조합함수와 공개키 조합함수로 $Q_1()$과 $R_2()$를 사용하며, 응답하는 개체(responder, *i.e.* Steve)는 $Q_2()$과 $R_1()$을 사용한다. 양 쪽 개체가 유도해 낸 K가 동일한 값을 갖기 위해서는 $Q_1()$과 $R_1()$ 그리고 $Q_2()$와 $R_2()$가 각각 서로에 대해 대칭적인 구조를 가져야 한다. 다음 절에서 타원곡선 군(group)을 적용하였을 경우 실제로 함수가 어떠한 형태를 갖는지 살펴보도록 하겠다.

4. EC-AKE

a,x,b,y는 프로토콜 진행과정 중 개인키로 쓰이는 매개변수로서 AKE에서와 마찬가지로 a,b는 임시(ephemeral)개인키로 x,y는 고정

(static)개인키로 사용되며 모두 타원곡선 군(group)의 위수(order) n에 대해 $[1,n-1]$의 값을 가져야 한다.

$G()$는 개인키에 대한 공개키를 생성하는 함수로서 타원곡선 군의 생성자(generator)P에 대해 다음과 같이 정의된다.

$$G(d)=dP \tag{4}$$

여기서 d는 $[1,n-1]$중에서 랜덤하게 선택되며, 생성된 공개키 $G(d)$값은 타원곡선 상의 한 점이 된다. 개인키 조합 함수는 다음과 같이 정의된다.

$$Q_1(a,x)=a+u_1x,\ Q_2(b,y)=b+u_2y \tag{5}$$

공개키 조합 함수는 다음과 같이 정의된다.

$$R_1(aP,xP)=aP+u_1\cdot xP,$$
$$R_2(bP,yP)=bP+u_2\cdot yP \tag{6}$$

여기서 u_1과 u_2는 스크램블 인자(scrambling factor)로 Steve와 Carol의 임시 공개키 들의 조합으로 정의할 수 있다. 즉, 다음과 같이 나타낼 수 있다.

$$u_1=g(G(a),G(b))=g(aP,bP),$$
$$u_2=h(G(a),G(b))=h(aP,bP) \tag{7}$$

g와 h에 대한 실제적인 함수정의는 EC-SRP 프로토콜을 설계하는 과정에서 정의하도록 한다. 비밀정보 생성 함수 S는 다음과 같이 정의된다.

* Carol의 경우:

$$K=S(Q_1(a,x),R_2(G(b),G(y)))$$
$$=S(a+u_1x,\ bP+u_2yP)$$
$$=(a+u_1x)(bP+u_2yP)$$
$$=(a+u_1x)(b+u_2y)P \tag{8-1}$$

* Steve의 경우:

$$K=S(Q_2(b,y),R_1(G(a),G(x)))$$
$$=S(b+u_2y,aP+u_1xP)$$
$$=(b+u_2y)(aP+u_1xP)$$
$$=(b+u_2y)(a+u_1x)P \tag{8-2}$$

위의 계산 과정에 따라 Steve와 Carol이 공유하게 되는 비밀정보 K는 동일한 값이 된다.

다음 장에서는 EC-AKE 프로토콜을 토대로 하여 EC-SRP 프로토콜을 설계하도록 한다.

Ⅲ. EC-SRP 프로토콜

SRP 프로토콜은 클라이언트와 서버간의 인증이 이루어지는 비대칭형(asymmetric) 프로토콜로서 프로토콜을 시작하는 개체는 항상 클라이언트 쪽이다. 그리고 서버가 클라이언트(사용자)를 인증하기 위해 사용하는 정보는 클라이언트의 확인자(verifier)이다. 확인자는 일반적인 공개키 알고리즘에서의 공개키에 해당하지만, 공개키와는 달리 공개되지 않으며 서버에 안전하게 저장된다.

앞에서 말했듯이 SRP 프로토콜은 패스워드 기반 프로토콜로서 사용자는 자신의 개인키를 암기하거나 개인키를 저장하기 위한 별도의 장비(*e.g.* 스마트 카드)를 필요로 하지 않는다. 그럼에도 불구하고 SRP 프로토콜이 공개키 알고리즘을 사용한다는 것은 사용자의 개인키가 프로토콜 진행과정 중에 생성됨을 의미한다. 사용자의 개인키는 안전하게 생성되어야 하며 인증 프로토콜이 실행될 때마다 특정 사용자에 대한 개인키는 항상 동일한 값으로 유도되어져야 한다.

SRP 프로토콜이 첫 번째로 수행하는 과정이 바로 사용자의 개인키를 유도해내는 과정으로 수행과정은 〈그림 2〉와 같다.

	Client (Carol)		Server (Steve)
1		*Carol* ⟹	(lookup *s*, *V*)
		s ⟸	
2	$x = H_1(s, Pwd)$		

〈그림 2〉 클라이언트의 개인키 유도과정

1. 클라이언트는 먼저 사용자 이름(username, *e.g. Carol*)을 서버에 전송한다. 서버는 미리 데이터베이스에 저장되어 있던 랜덤 솔트 (random salt) *s*와 Carol의 확인자 *V*를 찾는다. *V*는 비밀정보를 유도하는 과정에서 쓰인다.
2. 서버는 *s*를 클라이언트에 전송하게 되며, 클라이언트는 *s*와 *Pwd*를 기반으로 하여 자신의 개인키 *x*를 유도해 낸다. *s*는 각 클라이언트마다 고유하게 설정된 값으로서 클라이언트의 개인키의 랜덤성을 보장하기 위해 쓰인다. $H_1()$은 해쉬 함수이다.

여기서 서버에 저장되어 있던 *V*는 *x*에 대한 확인자로서, 타원곡선 스칼라 곱셈인 $V = xP$의 식으로부터 유도되어진 값이다. 각각의 사용자에 해당하는 확인자 *V*와 랜덤 솔트 *s*는 사용자 등록과정에서 이미 생성되고 서버에 저장되어진 것으로 가정한다. 서버는 클라이언트의 확인자를 가지고 있을 뿐, 클라이언트의 패스워드나 개인키를 보유하지는 않는다.

1. EC-SRP1 프로토콜

클라이언트의 고정 공개키 쌍(개인키와 공개키)은 이와 같이 온라인(on-line)상에서 유도되지만, 서버의 고정 공개키 쌍은 존재하지 않는다. 이로 인해, 서버는 클라이언트가 클라이언트 자신의 개인키를 가지고 있음을 확인함으로써 클라이언트를 인증하는 반면, 클라이언트는 서버가 클라이언트의 확인자를 가지고 있음을 확인함으로써 서버를 인증하게 된다.

서버의 고정 공개키 쌍이 존재하지 않는다는 것을 EC-AKE에 적용하면, $y = 0$, $G(y) = O$가 된다. 여기서 *O*는 타원곡선 군의 항등원인 무한 원점 (point at infinity)이다. 따라서 비밀정보 생성에 관한 수식들은 다음과 같이 간략화 된다.

$$Q_1(a,x) = a + u_1 x,\ Q_2(b,0) = b + u_2 \cdot 0 = b$$
$$R_1(aP, xP) = aP + u_1 \cdot xP,$$
$$R_2(bP, O) = bP + u_2 \cdot O = bP$$

이때, $Q_2()$의 두 번째 인자가 0으로 고정되고 $R_2()$의 두 번째 인자가 *O*로 고정되므로 서버 측의 스크램블인자인 u_2가 $Q_2(), R_2()$의 결과값에 영향을 주지 않는다. 따라서 $Q_1()$, $Q_2()$,와 $R_2(), R_1()$ 그리고 u_1, u_2를 각기 $Q(), R(), u$로 통합하여 나타낼 수 있으며 함수는 다음과 같이 정의된다.

$$Q(\alpha, \beta) = \alpha + u\beta \tag{9}$$

$$R(\alpha P, \beta P) = \alpha P + u \cdot \beta P \tag{10}$$

$$u = f(G(\alpha), G(\beta)) = f(\alpha P, \beta P) \tag{11}$$

정의된 함수$G(), R()$와 u를 사용하면, EC-SRP에서 공유되는 비밀정보의 생성은 다음과 같다.

* Carol의 경우:

$$K = S(Q(a,x), R(G(b), O))$$
$$= S(a + ux, bP + u \cdot O) = (a + ux)bP \tag{12-1}$$

* Steve의 경우:

$$K = S(Q(b,0), R(G(a), G(x)))$$
$$= S(b + u \cdot 0,\ aP + u \cdot xP)$$
$$= b(aP + uxP) = b(a + ux)P \tag{12-2}$$

EC-SRP 프로토콜의 정의에 필요한 매개변 수와 각각의 역할은 다음과 같다.

〈표 2〉 EC-SRP 매개변수

매개변수	역 할
P	타원곡선 군의 생성자
s	사용자의 salt로 사용되는 random string
Pwd	사용자의 Password
x	사용자의 Pwd와 salt s로부터 계산되어지는 사용자의 개인키
V	서버에 저장되어 있는 해당 사용자에 대한 Verifier, $V = xP$
u	스크램블 인자, $(A+B)$의 x축 좌표값
$\bar{u}$	식 19, 20에 따라 계산되는 스크램블인자 u길이의 1/2을 가지는 값
a,b	임시 개인키로서, random하게 생성되며 공개되지 않는 정보
A,B	a,b에 해당하는 임시 공개키
$H()$	일방향 해쉬 함수
m,n	m 과n string(or quantaty)을 concatenate 시킴
z	K의 x축 좌표값
b',K',Pwd',V',M_1'	공격자의 관점에서 dictionary attack을 위해 추정하는 값
$\hat{a}$	사용자의 관점에서 계산량을 줄이기 위해서 $a+x$로 간주하는 값

이에 기반하여 설계한 EC-SRP1 프로토콜은 〈그림 3〉과 같다.

	Client (Carol)		Server (Steve)
1		$Carol \Rightarrow$	(lookup s, V)
2	$x = H_1(s,Pwd)$	$\Leftarrow s$	
3	$A = aP$	$A \Rightarrow$	
4		$\Leftarrow B$	$B = bP$
5	$K = (a+ux) \cdot B$ $= (a+ux)bP$		$K = b(A+u \cdot V)$ $= b(a+ux)P$
6	$\kappa = H_2(z)$ $\kappa' = H_3(z)$		$\kappa = H_2(z)$ $\kappa' = H_3(z)$
7	$M_1 = MAC_{\kappa'}(A,B)$	$M_1 \Rightarrow$	(verify M_1)
8	(verify M_2)	$\Leftarrow M_2$	$M_2 = MAC_{\kappa'}(A,B,M_1)$

〈그림 3〉 EC-SRP1 프로토콜

EC-SRP1

1. Carol은 Steve에게 사용자이름(e.g. *Carol*)을 보낸다.
2. Steve는 미리 저장된 Carol의 확인자 V와 랜덤 솔트 s를 찾는다. s를 Carol에게 보낸다. Carol은 s와 패스워드 Pwd를 이용하여 자신의 개인키 x를 계산한다.
3. Carol은 $a(1 \le a \le n-1)$을 생성한 후 $A=aP$를 계산하여 Steve에게 보낸다.
4. Steve는 $b(1 \le b \le n-1)$을 생성한 후 $B=bP$를 계산하여 Carol에게 보낸다.
5. Carol과 Steve는 $K=b(a+ux)P$를 계산한다. 여기서 스크램블 인자u는 $(A+B)$의 x축 좌표 값으로 정의한다. A와 B는 모두 공개되는 값이므로 u또한 공개되는 값이다.
6. Carol과 Steve는 $\kappa=H_2(z)$, $\kappa'=H_3(z)$를 각각 계산한다. 여기서 z는 점 K의 x축 좌표 값이다.
7. Carol은 $M_1=MAC_{\kappa'}(A,B)$을 Steve에게 보내고 Steve는 이를 확인한다.
8. Steve는 $M_2=MAC_{\kappa'}(A,B,M_1)$를 계산하여 Carol에게 보내고 Carol은 이를 확인한다.

과정 6.에서 생성된 κ'은 키 확인 (key confirmation)을 하는 과정(7.,8.)에서 MAC(Message Authentication Code) 생성에 쓰이며, κ를 양자간에 공유되는 세션키(session key)로 쓰이게 된다.

과정7.,8.에서 MAC은 $MAC_\alpha(\beta)=(\beta,f_\alpha(\beta))$로 정의되는 함수로서 메시지 β를 키 α로 인증하기 위해 사용되는 함수이다. MAC 함수가 안전하기 위해서는 f가 키 α에 의존적인 일방향 함수 (one-way function)이어야 한다. 본 논문에서 사용되는 β에 해당하는 인자(*i.e.* A와 B 및 M_1)는 이전과정에서 이미 공개되었기 때문에 MAC 전송부분에서 $f_\alpha(\beta)$만 전송하면 된다. 본 논문에서는 해쉬함수를 이용하여 $f_\alpha()$를 설계하도록 하며 그 정의는 다음과 같다.

$$f_\alpha(\beta) = H(\alpha, \beta) \qquad (13)$$

SRP 프로토콜은 일반적인 AKE 프로토콜과 달리 클라이언트의 개인키와 공개키가 암호학적 비도가 비교적 낮은 패스워드에 기반하고 있기 때문에 사전식 공격법 (dictionary attack)에 대한 고려가 있어야 한다. EC-SRP1의 경우, Pwd와 V를 모르는 제 3자(Sue)가 다음의 과정을 거쳐 사전식 공격을 행할 수 있다.

① Carol은 Sue에게 자신의 사용자이름을 보낸다.
② Sue는 Carol에게 미리 도청한 랜덤 솔트 s를 보낸다.
③ Carol은 Sue에게 A를 보낸다.
④ Sue는 자신의 랜덤 수 b를 생성하여 $B=bP$를 계산하고, B를 Carol에게 보낸다.
⑤ Carol은 $K=(a+ux)B$, $\kappa=H_2(z)$, $\kappa'=H_3(z)$를 구하고 K에 대한 인증을 받기 위해 Sue에게 $M_1=MAC_{\kappa'}(A,B)$을 보낸다.
⑥ Sue는 네트워크 장애 (network failure)를 유발시키거나, Carol에게 패스워드가 틀렸다고 알림으로써 프로토콜을 중단한 후 다음을 반복 수행함으로써 Carol의 Pwd를 구한다.
㉠ 패스워드 Pwd를 추측한 값 Pwd'을 생성하여, V'(V를 추측한 값)을 구하고 $K'=b(A+u \cdot V')$을 계산한다.
㉡ K의 값을 증명하는 메시지 M_1과 K'로부터 생성된 M_1'가 동일한지를 검사한다.

2. EC-SRP2 프로토콜

EC-SRP1 프로토콜에서의 사전식 공격법을 막기 위해 B의 값을 다음과 같이 수정하여 보자.

$$B = V + bA \tag{14}$$

그러면, *Carol*은 K를 다음과 같이 계산하게 된다.

$$K = (a+ux) \cdot \frac{(B-xP)}{a} = (a+ux) \cdot bP \tag{15}$$

여기서 a^{-1}를 계산해 내기 위해서는 타원곡선 군의 위수(order)가 반드시 소수이어야 하며, 시스템 매개변수도 이와 같이 설정되어 있다. B의 값을 이와 같이 설정하였을 때, 사전식 공격방법을 시도할 경우 Sue는 다음을 계산해야 한다.

$$K = \frac{b' - x}{a}(A+uV)$$

그러나 Sue는 a를 알 수 없으므로 사전식 공격법을 방지할 수 있다. 이에 대한 프로토콜의 수정된 부분은 〈그림 4〉와 같다.

	Client (Carol)		Server (Steve)
3	$A = aP$	$A \Rightarrow$	
4		$\Leftarrow B$	$B = V + bA$
5	$K = (a+ux) \cdot a^{-1}(B-xP)$ $= (a+ux)bP$		$K = b(A+u \cdot V)$ $= b(a+ux)P$

〈그림 4〉 EC-SRP2 프로토콜

3. EC-SRP3 프로토콜

EC-SRP2에서 Carol의 타원곡선 스칼라 곱셈(elliptic curve scalar multiplication)은 3번 (aP, xP, $(a+ux) \cdot a^{-1}(B-xP)$) 이루어지며, Steve의 경우 또한 3번 (bA, uV, $b(A+uV)$) 이루어진다. 본 절에서는 스칼라 곱셈의 회수를 최소화 하도록 한다.

먼저 Carol의 xP의 계산을 피함으로써 스칼라 곱셈의 회수를 줄일 수 있다.

Carol의 A를 다음과 같이 놓는다.

$$A = \hat{a}P, \quad \hat{a} = a+x \tag{16}$$

여기서 $\hat{a}$는 랜덤하게 생성된 수이며, 암시적으로 랜덤수 a와 고정 개인키 x의 합으로 간주한다(실제로 $a+x$를 계산하는 대신 랜덤수 $\hat{a}$을 랜덤수로 사용하는 이유는, 나중에 프로토콜의 패스 수를 최적화 하였을 경우 A를 생성하는 시점에서 x를 알수 없기 때문이다).

그리고 Steve의 B는 다음과 같이 놓는다.

$$B = b(A-V) = b(\hat{a}-x)P = baP \tag{17}$$

그러면, Carol의 K 계산은 다음과 같다.

$$\begin{aligned} K &= (\hat{a}+(u-1)x)(\hat{a}-x)^{-1}B \\ &= (\hat{a}-x+ux)(\hat{a}-x)^{-1}abP = (a+ux)a^{-1}abP \\ &= (a+ux)bP \end{aligned} \tag{18-1}$$

Steve의 K 계산은 다음과 같다.

$$\begin{aligned} K &= b(A+(u-1)V) \\ &= b(\hat{a}P - xP + uxP) = b(\hat{a}-x+ux)P \\ &= b(a+ux)P \end{aligned} \tag{18-2}$$

이 경우, $B = b'P$를 전송한 후(b'은 랜덤수), K을 계산하여 사전식 공격법을 시도하려면 다음의 K를 구해야하며, 이때 $\hat{a}$의 값을 알아야 하므로 사전식 공격법을 방지할 수 있다.

$$K = \frac{b'}{\hat{a}-x}(A+uV)$$

그리고 스크램블 인자로 사용한 u는 EC-SRP1에서 $(A+B)$의 x축 좌표 값으로 정의하였다. 여기서 u를 $\bar{u}$로 놓고 다음과 같이 정하도록 한다.

$$\bar{u} = \overline{A+B} \tag{19}$$

임의의 타원곡선상의 한 점 R(O가 아닌 한 점)에 대해 $\bar{R}$는 다음과 같이 정의된다.

$$\bar{R} = (\bar{x} \bmod 2^{\lceil l/2 \rceil}) + 2^{\lceil l/2 \rceil} \tag{20}$$

l은 생성자(generator) P의 위수(order)인 n의 비트 길이(즉, $l = \lfloor \log_2 n \rfloor + 1$)이며 $\bar{x}$는 R의 x축 좌표 값을 이진수(binary)로 표현한 값이다. $2^{\lceil l/2 \rceil}$항을 더한 것은 $(\bar{R} \bmod n) \neq 0$ 임을 보장하기 위함이다.

스크램블 인자 $\bar{u}$의 비트 길이는 랜덤수 비트 길이의 반에 해당하므로, 스칼라 곱셈의 회수에 대해 0.5회의 이득이 있으며 암호학적 비도에는 영향이 없다[9]. 따라서, 프로토콜의 수정된 부분은 〈그림 5〉과 같다.

	Client (Carol)		Server (Steve)
3	$A = \hat{a}P$	$A \Longrightarrow$	
4		$B \Longleftarrow$	$B = b(A-V)$
5	$K = (\hat{a}+(\bar{u}-1)x)(\hat{a}-x)^{-1}B$ $= (a+\bar{u}x)bP$		$K = b(A+(\bar{u}-1)V)$ $= b(a+\bar{u}x)P$

〈그림 5〉 EC-SRP3 프로토콜

EC-SRP3에서 Carol의 타원곡선 스칼라 곱셈이 수행되는 곳은 $A = \hat{a}P$와 $K = (\hat{a}+(\bar{u}-1)x)(\hat{a}-x)^{-1}B$으로 총 2회이고, Steve의 스칼라 곱셈이 수행되는 곳은 bA, $\bar{u}V$, $b(A+\bar{u}V)$으로 총 2.5회이다.

4. EC-SRP4 프로토콜

EC-SRP3에서 사용된 B는 $b(A-V) = abP$가 되기 때문에, 공유되는 비밀정보 K의 값을 다음과 같이 나타낼 수 있다.

$$K = (a+ux)bP = abP+ubxP = B+ubxP \tag{21}$$

즉 abP항이 B로서 공개되는 것이다. 이에 따라 a가 갖는 임시 개인키로서의 역할을 상실하는 역효과가 발생한다. 이를 보완하기 위해서 B를 다음과 같이 수정하도록 한다.

$$B = b(A-V+P) = abP+bP \tag{22}$$

여기서 추가된 bP는 노출되는 값이 아니므로 abP의 값 또한 숨길 수 있다. $(a+x)P$와 $(a+1)bP$로부터 abP를 계산하는 문제가 안전함을 Diffie-Hellman 문제가 안전하다는 가정 하에서 다음과 같이 증명할 수 있다.

먼저 $(a+x)P$와 $(a+1)bP$로부터 abP를 계산하는 다항시간(polytime) 오라클 Q가 존재한다고 가정하자(반증법으로 증명하기 위해).

$$Q((a+x)P,\ (a+1)bP) = abP \tag{23}$$

여기서 aP가 알려졌다고 가정함으로써 문제를 축소(reduction)하자. 그러면 축소된 문제 $\tilde{Q}$는 다음과 같다.

$$\begin{aligned} & \tilde{Q}(aP,\ (a+x)P,\ (a+1)bP,\ P) = abP \\ \Rightarrow\ & \tilde{Q}(aP,\ xP,\ (a+1)bP,\ P) = abP \\ \Rightarrow\ & \tilde{Q}(aP,\ (a+1)bP,\ P) = abP \\ \Rightarrow\ & \tilde{Q}(aP,\ abP+bP;\ P) = abP \end{aligned} \tag{24}$$

여기서 aP를 M로 치환하여 나타내도록 하자. 그러면 다음과 같다.

$$\tilde{Q}(aP, abP+bP,\ P) = ab\,P$$

$\Rightarrow \quad \tilde{Q}(M,\ bM+ba^{-1}M,\ a^{-1}M) = bM$

$\Rightarrow \quad \tilde{Q}(M,\ bM+ba^{-1}M,\ a^{-1}M+M) = bM$

$\Rightarrow \quad \tilde{Q}(M,\ b(1+a^{-1})M,\ (1+a^{-1})M) = bM$

여기서 다시 $(1+a^{-1})M$을 N으로 치환하자. 그러면

$$\tilde{Q}(M,\ b(1+a^{-1})M,\ (1+a^{-1})M) = bM$$
$$\Rightarrow \quad \tilde{Q}((1+a^{-1})^{-1}N,\ bN,\ N) = b(1+a^{-1})^{-1}N$$

와 같이 되며, 이는 생성자가 N이고 양 쪽 개체의 개인키가 각각 b와 $(1+a^{-1})^{-1}$인 Diffie-Hellman 문제이다. 즉, $\tilde{Q}$의 존재성은 Diffie-Hellman 문제가 안전하다는 가정에 위배된다. 따라서 공격자는 abP를 계산해낼 수 없다. 따라서 Carol은 다음과 같이 K를 계산한다.

$$\begin{aligned} K &= (\hat{a}+(u-1)x)(\hat{a}-x+1)^{-1}B \\ &= (\hat{a}-x+ux)(\hat{a}-x+1)^{-1}(a+1)bP \\ &= (a+ux)(a+1)^{-1}(a+1)bP \\ &= (a+ux)bP \end{aligned} \qquad (25)$$

본 논문에서 제안하는 최종적인 EC-SRP4는 다음의 〈그림 6〉과 같다.

EC-SRP4

1. Carol은 Steve에게 사용자이름(e.g. *Carol*)을 보낸다.
2. Steve는 미리 저장된 Carol의 확인자 V와 솔트(salt) s를 찾는다. s를 Carol에게 보낸다. Carol은 s와 패스워드 Pwd를 이용하여 자신의 개인키 x를 계산한다.
3. Carol은 $\hat{a}(1 \le \hat{a} \le n-1)$을 생성한 후 $A = \hat{a}P$를 계산하여 Steve에게 보낸다.
4. Steve는 $b(1 \le b \le n-1)$을 생성한 후 $B = b(A - V + P)$를 계산하여 Carol에게 보낸다.
5. Carol과 Steve는 $K = b(a+\tilde{u}x)P$를 계산한다.
6. Carol과 Steve는 $\kappa = H_2(z)$, $\kappa' = H_3(z)$를 각각 계산한다. 여기서 z는 점 K의 x축 좌표 값이다.
7. Carol은 $M_1 = MAC_{\kappa'}(A,B)$을 Steve에게 보내고 Steve는 이를 확인한다.
8. Steve는 $M_2 = MAC_{\kappa'}(A,B,M_1)$를 계산하여 Carol에게 보내고 Carol은 이를 확인한

	Client (Carol)		Server (Steve)
1		*Carol* ⟹	(lookup s, V)
2	$x = H_1(s,Pwd)$	s ⟸	
3	$A = \hat{a}P$	A ⟹	
4		B ⟸	$B = b(A-V+P)$
5	$K = (\hat{a}+(\tilde{u}-1)x(\hat{a}-x+1)^{-1}B$ $= (a+\tilde{u}x)bP$		$K = b(A+(\tilde{u}-1)V)$ $= b(a+\tilde{u}x)P$
6	$\kappa = H_2(z)$ $\kappa' = H_3(z)$		$\kappa = H_2(z)$ $\kappa' = H_3(z)$
7	$M_1 = MAC_{\kappa'}(A,B)$	M_1 ⟹	(verify M_1)
8	(verify M_2)	M_2 ⟸	$M_2 = MAC_{\kappa'}(A,B,M_1)$

〈그림 6〉 EC-SRP4 프로토콜

다.

여기서 MAC은 앞에서 정의하였듯이, 해쉬 함수 H에 대해 $MAC_{\alpha}(\beta)=H(\alpha, \beta)$로 정의한다.

5. 4-pass EC-SRP 프로토콜

프로토콜의 성능향상에서 중요한 것 중 하나가 메시지 패스(pass)의 수를 최소한으로 줄이는 것이다. 최대한 중복될 수 있는 메시지는 함께 보냄으로써, 패스의 수를 줄이도록 한다.

Carol의 개인키를 유도해내는 과정(EC-SRP6의 1.과 2.)에서 교환된 메시지는 다른 과정(EC-SRP6의 3.~8.)에서 쓰이지 않을 뿐 아니라, 해쉬함수를 통해 x를 유도해냈기 때문에 과정 1.~2.의 메시지는 과정 3.~8.의 메시지와 독립적이라 할 수 있다. 따라서 과정 1.의 메시지는 과정 3.의 메시지와 함께, 과정 2.의 메시지는 과정 4.의 메시지와 함께 보낼 수 있으며 암호학적 보안성에는 영향이 없다.

따라서 프로토콜 진행과정 중 소요되는 총 패스 수는 4번이며, 최적화 된 EC-SRP는 〈그림 7〉과 같다.

위의 EC-SRP 프로토콜은 양방향 인증이 이루어진다. 그러나 어플리케이션에 따라서는 한 방향 인증만을 필요로하는 경우가 있으며 이러한 경우에는 과정 6.을 생략하여 3 번의 메시지 패스만으로 프로토콜이 완료된다.

Ⅳ.보안 및 성능 분석 (Security & Performance Analysis)

1. EC-SRP 프로토콜의 안전성 증명

랜덤 오라클 모델에 대해서 EC-SRP 프로토콜이 안전함을 증명한다. M. Bellare와 P. Rogaway가 AK(Authenticated Key agreement) 프로토콜과 AKC (Authenticated Key agreement with key Confirmation) 프로토콜에 대한 안전성의 정의를 내렸으며, 본 논문에서는 이 Bellare-Rogaway 모델을 기준하여 안전성을 증명한다.

안전성 증명과정에서 Diffie-Hellman 문제는 안전하며, 프로토콜 과정에서 쓰인 해쉬 함수들은 랜덤 오라클이라 가정한다. 랜덤 오라클은 블랙박스(black-box) 랜덤 함수로 제공되며 다음의 랜덤함수로 정의된다.

	Client (Carol)		Server (Steve)
1	$A=\hat{a}P$	$Carol,A \Longrightarrow$	(lookup s, V)
2	$x=H_1(s,Pwd)$	$s,B \Longleftarrow$	$B=b(A-V+P)$
3	$K=(\hat{a}+(\bar{u}-1)x(\hat{a}-x+1)^{-1}B$ $=(a+\bar{u}x)bP$		$K=b(A+(\bar{u}-1)V)$ $=b(a+\bar{u}x)P$
4	$\kappa = H_2(z)$ $\kappa' = H_3(z)$		$\kappa = H_2(z)$ $\kappa' = H_3(z)$
5	$M_1 = MAC_{\kappa}(A,B)$	$M_1 \Longrightarrow$	(verify M_1)
6	(verify M_2)	$M_2 \Longleftarrow$	$M_2 = MAC_{\kappa}(A,B,M_1)$

〈그림 7〉 4-pass EC-SRP 프로토콜

$$H(\cdot):\{0,1\}^* \rightarrow \{0,1\}^k \quad (26)$$

본 논문의 EC-SRP 프로토콜을 크게 3 부분으로 나누어 증명한다.

- 사용자의 개인키를 유도해내는 부분 (EC-SRP4의 과정 1.~2.)
- 비밀정보 K를 계산하는 부분 (EC-SRP4의 과정 3.~5.)
- 세션 키를 설정하고 키 확인(key confirmation)하는 부분 (EC-SRP4의 과정 6.~8.)

1) 사용자의 개인키를 유도해내는 부분

이 부분에서는 사용자가 자신의 패스워드와 서버로부터 전송받은 솔트에 기반하여 개인키를 유도해낸다. 여기서 사용된 솔트는 패스워드의 랜덤성을 보장하기 위해서 사용되었을 뿐이며, H_1을 랜덤 오라클이라고 가정한다면 솔트의 역할은 의미가 없다(H_1이 랜덤 오라클이라면 패스워드만으로도 개인키의 랜덤성이 보장되기 때문이다). 그러나 구현상에서 솔트를 사용하지 않은 채 해쉬함수 H_1을 사용할 경우, 패스워드의 비교적 낮은 비도로 인해 사용자들간의 개인키가 겹칠 확률이 크다. 이를 방지하기 위해 솔트를 사용하는 것이다.

비록 솔트가 공개되는 값이긴 하지만, 공격자는 사용자의 개인키를 공격자가 원하는 값이 되도록 솔트를 조작할 수 없다. 왜냐하면 공격자는 사용자의 패스워드를 알지 못하며 개인키를 유도하는 함수 H_1은 해쉬함수로 일방향(one-way) 함수이기 때문이다. 그리고 공격자가 솔트를 조작할 경우, 사용자의 개인키가 잘못된 값으로 계산되게 함으로써 인증이 성립되지 않게 할 수 있을 뿐 그 이상의 효과는 존재하지 않는다(즉, 공격자는 denial of service의 공격을 취할 수 있을 뿐이다).

따라서 본 논문에서 보안성을 증명하고자 하는 랜덤 오라클(random oracle) 모델에서는 이 부분(과정 1.~2.)을 고려대상에서 제외하는 대신 사용자가 자신의 개인키를 알고있는 것으로 가정할 수 있다.

2) 비밀정보 K를 계산하는 부분

본 논문에서 제안한 EC-SRP 프로토콜의 수학적 구조는 타원곡선(elliptic curve)에 대한 Diffie-Hellman 문제와 유사하다. Diffie-Hellman 문제는 다항시간(polytime) 공격자에 대해 안전한 것으로 알려져 있으며, 본 논문에서도 Diffie-Hellman 문제가 안전한 것으로 가정하고 EC-SRP 문제를 Diffie-Hellman 문제로 축소(reduction)시킴으로써, EC-SRP 문제의 안전함을 증명하고자 한다.

먼저 EC-SRP 문제가 안전하지 않다고 가정하자. 그러면, 공개적으로 알려지는 정보 $aP+xP$, $abP+bP$, $\bar{u}$, n, P가 주어졌을 때, 공유되는 비밀정보 $K=(a+\bar{u}x)bP$를 계산해낼 수 있는 다항시간 오라클(oracle)Q가 존재함을 의미한다.

$$Q(aP+xP,\ abP+bP,\ \bar{u},\ n,\ P)=(a+\bar{u}x)bP \quad (27)$$

여기서 $V(=xP)$와 a의 값이 알려졌다고 가정함으로써, 문제를 축소(reduction)하자. 그러면 Q의 변형된 오라클 $\tilde{Q}$의 문제는 다음과 같은 등가 관계를 형성함을 알 수 있다.

$$\begin{aligned} & \tilde{Q}(a,xP,\ abP,\ bP,\ \bar{u},\ n,\ P)=(ab+\bar{u}xb)P \\ \Rightarrow\ & \tilde{Q}(a,\ xP,\ abP,\ bP,\ \bar{u},\ n,\ P)=\bar{u}xbP \\ \Rightarrow\ & \tilde{Q}(xP,\ bP,\ n,\ P)=xbP \end{aligned} \quad (28)$$

즉 오라클 $\tilde{Q}$가 존재한다는 것은 오라클 Q의 존재에 대한 필요조건이다. 그런데, $\tilde{Q}$의 문제는 Diffie-Hellman 문제이며 이는 Diffie-Hellman 문제가 안전하다는 것에 대한 가정에

위배된다. 따라서 Diffie-Hellman 문제가 안전하다는 가정 하에서 EC-SRP 문제가 안전하다는 것이 증명되었다.

3) 세션 키를 설정하고 키 확인(key confirmation)하는 부분

이제, 과정 6.~8.에서의 키 확인(key confirmation)과 세션 키에 대한 안전성을 증명하는 부분만을 남겨 놓고 있다.

이 부분은 Simon Blake-Wilson, Don Johnson, Alfred Menezes가 랜덤 오라클 모델인 Bellare-Rogaway 모델[14]에서 안전성을 증명한 프로토콜과 동일한 형태를 가지며, 증명 또한 동일한 맥락에서 이루어질 수 있다[15]. 본 논문에서는 생략하도록 한다.

결론적으로 본 논문의 EC-SRP 프로토콜은 Bellare-Rogaway 모델 하에서 안전성이 증명되며, 이는 Diffie-Hellman 문제가 안전하고 해쉬함수가 랜덤 오라클이라는 가정을 동반한다.

2. 보안 특성(Security Attributes)

인증(authentication) 프로토콜에서 고려되는 보안 특징들과 본 논문에서의 적용성을 검토한다.

1) known-key security

: 공격자가 과거에 사용된 세션키를 알아낸다고 하더라도, 현재의 세션키를 유추할 수 없어야 한다.

세션키를 공유할 때마다 랜덤수 a와 b가 매개변수로 쓰이기 때문에 과거의 세션키는 현재의 세션키 유추에 아무런 도움을 주지 못한다.

2) (perfect) forward secrecy

: 공격자가 개인키를 알아낸다고 하더라도, 현재의 세션키를 유추할 수 없어야 한다.

공개되는 정보 $aP+xP$, $abP+bP$, $\bar{u}$, n, P와 더불어 x를 아는 상태에서 세션키를 생성한다는 것은 다음의 다항시간 오라클이 존재함을 의미한다.

$$Q(x,\ aP+xP,\ abP+bP,\ \bar{u},n,P) = (a+\bar{u}x)bP \quad (29)$$

Q는 다음의 $\tilde{Q}$와 등가임을 알 수 있다.

$$\begin{aligned}&\tilde{Q}(x,\ aP,\ abP+bP,\ \bar{u},\ n,\ P) = (abP+bP)+(\bar{u}x-1)bP\\ &\Rightarrow \tilde{Q}(x, aP,\ abP+bP,\ \bar{u},\ n,\ P) = (\bar{u}x-1)bP\\ &\Rightarrow \tilde{Q}(aP,\ abP+bP,\ n,\ P) = bP\\ &\Rightarrow \tilde{Q}(aP,\ abP+bP,\ n,\ P) = bP+P \qquad (30)\end{aligned}$$

위의 $\tilde{Q}$에서 aP를 A로 치환하여 나타내도록 하자. 그러면,

$$\tilde{Q}(aP,\ a(b+1)\ P,\ P) = (b+1)\ P$$
$$\Rightarrow \quad \tilde{Q}(A,\ (b+1)\ A,\ a^{-1}A) = a^{-1}(b+1)A$$

와 같이 되며, 이는 생성자가 A이고 양 쪽 개체의 개인키가 각각 $(b+1)$과 a^{-1}인 Diffie-Hellman 문제이다. 즉, $\tilde{Q}$의 존재성은 Diffie-Hellman 문제가 안전하다는 가정에 위배된다. 따라서 EC-SRP 프로토콜은 본 특징을 만족한다.

3) unknown key-share

: 개체 A의 개인키가 유출되지 않는 한, A가 의도하지 않은 다른 개체와 세션키가 공유될 수 없다. 즉, A가 세션키를 공유한 경우 공유한 상대방이 누구인지 확신할 수 있어야 한다.

MAC 함수가 랜덤 오라클이라는 가정에 따라, K를 구하지 않고는 인증이 성립될 수 없다. 따라서 K가 공유되었을 경우, EC-SRP 문제가 안전하다는 증명에 따라 상대방이 정당한 통신대상임을 확신할 수 있다.

4) key control

: 양쪽 개체 중 어느 한 쪽도 미리 설정된 값이 세션키가 되도록 할 수 없어야 한다.

클라이언트와 서버는 각각 자신의 랜덤수 a와 b를 발생시켜 세션키 생성에 매개변수로 쓰기 때문에 어느 한쪽에서 세션 키를 컨트롤 할 수 없다.

이들 보안 특성중 known-key security, (perfect) forward secrecy 및 unknown key-share는 Bellare-Rogaway 모델 하에서 이루어진 안전성 증명으로부터 유도될 수 있는 특성들이다[15].

3. 성능 분석(Performance Analysis)

EC-SRP 프로토콜의 성능향상에 있어서 두 가지로 나누어 생각해 볼 수 있다. 그 중 하나는 이산대수가 아닌 타원곡선 이산대수를 이용하여 프로토콜을 설계하였다는 것이다. 단순히 이점만을 고려할 때, 약 10배 이상의 성능향상이 있는 것으로 알려져 있다[7].

두 번째는 스칼라 곱셈 즉 군 연산의 회수를 줄였다는 것이다. AKC 프로토콜 수행 과정에서 발생하는 연산 중 가장 느린 연산은 군 연산이어서, 군 연산의 회수가 얼마나 적으냐는 같은 부류의 AKC 프로토콜의 수행속도를 평가하는 1 차적인 요인이 된다. EC-SRP에서 군 연산인 스칼라 곱셈은 클라이언트에서 2번, 서버에서 2.5번 이루어지며, 연산이 이루어지는 곳은 〈표 3〉과 같다.

〈표 3〉 EC-SRP 프로토콜의 군 연산(스칼라 곱셈)

클라이언트	연산	$\hat{a}P$	$(\hat{a}+(\bar{u}-1)x)(\hat{a}-x)^{-1}\ B$		계
	회수	1 회	1 회		2 회
서버	연산	$b(A-V+P)$	$(\bar{u}-1)V$	$b(A+(\bar{u}-1)V)$	계
	회수	1 회	0.5 회	1 회	2.5 회

〈표 4〉는 확인자 기반(verifier-based) 프로토콜의 군연산 회수에 대한 비교이다.

〈표 4〉 확인자 기반 프로토콜의 군 연산 회수 비교

프로토콜	클라이언트	서 버
A-EKE	4회	4회
B-SPEKE	3회	4회
SRP	3회	3회
EC-SRP	2회	2.5회

EC-SRP 프로토콜이 Thomas Wu가 제안한 SRP 프로토콜에 대해 타원곡선 연산을 이용하였다는 것과는 별도로 군의 연산 회수가 줄었음을 알 수 있다.

Ⅵ. 결 론

본 논문에서는 1997년 Thomas Wu가 제안한 SRP 프로토콜을 발전시켜 타원곡선 군 연산을 이용한 EC-SRP(Elliptic Curve Secure Remote Password) 프로토콜을 제안하였다.

EC-SRP 프로토콜은 패스워드 프로토콜로서 가질 수 있는 사전식 공격법(dictionary attack)에 대해서 안전하도록 설계되었으며, 타원곡선 알고리즘을 적용함과 동시에 군(group) 연산인 타원곡선 스칼라 곱셈(scalar multiplication)의 회수를 최소화함으로써 프로토콜 연산의 효율성을 극대화 시켰다(SRP 프로토콜의 군 연산의 회수와 비교할 때, 클라이언트의 경우 3회에서 2회로 서버의 경우 3회에서 2.5회로 각각 줄였다). 그리고 메시지 패스의 수를 4회

로 최소화함으로써 프로토콜 진행이 효율적으로 이루어지도록 하였다.

또한 AK 및 AKC 프로토콜의 안전성에 대한 공식적 정의(formal definition)를 내린 Bellare-Rogaway 랜덤 오라클 모델에서 EC-SRP 프로토콜의 안전성을 증명하였다. 즉, 단순히 여러 가지 공격의 시도에 의해 안전성을 검토하는 차원을 넘어서 안전성 자체에 대한 증명을 한 것이다.

본 논문의 EC-SRP 프로토콜은 안전하고 높은 효율성을 가지고 있으면서도, 제 3의 신뢰기관이나 부가적인 장비(*e.g.* 스마트 카드)를 필요로 하지 않는다. 따라서 본 프로토콜이 인증이나 키 동의 및 키 확인을 요하는 시스템에서 유용하게 적용되리라 기대한다.

참고 문헌

[1] A. Menezes and S.A. Vanstone, "Elliptic curve cryptosystems and their implementations", Journal of Cryptology, 1993.

[2] A. Menezes, P. van Oorschot and S. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1997.

[3] ANSI X9.42, "Agreement of Symmetric Algorithm Keys Using Diffie-Hellman", working draft, September 1997.

[4] ANSI X9.62, "The Elliptic Curve Digital Signaure Algorithm (ECDSA)", working draft, October 1997.

[5] ANSI X9.63, "Elliptic Curve KeyAgreement and Key Transport Protocols", working draft, October 1997.

[6] S. M. Bellovin and M. Merritt, " Encrypted key exchange: Password-based protocols secure against dictionary attacks", Proceedings of the 1992 IEEE Computer Society conference on Research in Security and Privacy, 1992.

[7] Certicom Corp., "Elliptic Curve Cryptosystem Tutorials and WhitePapers", http://www.certicom.ca/.

[8] IEEE P1363, "Standard Specifications For Public Key Cryptography", February 13, 1998.

[9] L. Law, A.J. Menezes, M. Qu, J. Solinas and S. Vanstone, "An Efficient Protocol for Authenticated Key Agreement", Technical report CORR 98-05, Dept. of C&O, University of Waterloo, Canada, March 1998.

[10] M. Bellare, R. Canetti and H. Krawczyk, "Keying Hash Functions for Message Authentication", Advances in Cryptology - Crypto 96 Proceedings, 1996.

[11] M. Bellare, R. Canetti and H. Krawczyk, "A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols", Proceedings of the 30th Annual Symposium on the Theory of Computing, ACM, 1998.

[12] M. Bellare, " Practice-Orented Provable-Security", Proceedings of the 1997 Information Security Workshop (ISW).

[13] M. Bellare and P. Rogaway, "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols", The proceedings of the First ACM Conference on Computer and Communications Security, ACM November 1993.

[14] M. Bellare and P. Rogaway, "Entity authentication and Key Distribution",

Advances in Cryptology: Crypto 3, 1993.

[15] S. Blake-Wilson, D. Johnson and A.J. Menezes, "Key Agreement Protocols and their Security Analysis", The Sixth IMA International Conference on Crypptography and Coding, Cirencester, England, 17-19 December 1997.

[16] T. Wu, "The Secure Remote Password Protocol", Internet Society Symposium on Network and Distributed System Security, 1998.

□ 著者紹介

이 용 기

1997년 한양대학교 전자계산학과 학사
1999년 한양대학교 전자계산학과 석사

※ 주관심분야 : 타원곡선 공개키 알고리즘, 인증 프로토콜

이 정 규

1979년 2월 한양대학교 전자공학과 학사
1986년 UCLA 전자공학과 석사
1989년 UCLA 전자공학과 박사(컴퓨터 네트워크 전공)
1979년 3월 ~ 1984년 5월 국방연구소 연구원
1989년 3월 ~ 1990년 2월 삼성전자 종합기술원 정보통신부문 수석연구원
1990년 3월 ~ 현재 한양대학교 전자계산학과 부교수
1997년 3월 ~ 현재 한양대학교 공학기술연구소 부소장

※ 주관심분야 : 무선데이타통신, 이동통신, 타원곡선 공개키 알고리즘