타원곡선을 이용한 AMP 프로토콜
(Elliptic Curve AMP Protocol)

안 창 섭 † 허 신 ††
(Chang Sup Ahn) (Shin Heu)

요 약 패스워드를 이용한 인증 및 키교환 알고리즘은 뛰어난 편의성의 장점을 지니지만 사람이 기억할 수 있는 패스워드는 한계가 있어서 엔트로피(entropy)가 낮다.

패스워드의 편의성을 유지하면서 이러한 단점을 극복하기 위해 낮은 엔트로피의 패스워드를 이용하여 안전한 인증 및 키교환을 수행하는 AMP(Authentication and key agreement via Memorable Password) 프로토콜이 제안되었다. AMP 프로토콜은 이산대수문제(Discrete Logarithm Problem)에 기반한 Diffie-Hellman을 이용하여 프로토콜을 완성하였다.

그러나 본 논문에서는 AMP를 더욱 효율적으로 수행하기 위해 타원곡선 암호화를 AMP에 적용한다. 즉, 이산대수문제 대신에 타원곡선이산대수문제(Elliptic Curve Discrete Logarithm Problem)에 기반한 EC-AMP(Elliptic Curve-AMP) 프로토콜을 제안하고 구현을 통해 높은 성능을 입증한다. EC-AMP는 AMP와 마찬가지로 랜덤 오라클(random oracle) 모델에서 여러 가지 공격에 대해 안전하므로 인증 및 키교환이 필요한 네트워크 환경에 패스워드를 이용함으로 얻을 수 있는 편의성과 타원곡선이산대수문제가 제공하는 안전성을 동시에 보장할 수 있다.

키워드 : 인증, 패스워드, 타원곡선

Abstract Authentication and Key Agreement using password provide convenience and amenity, but what human can remember has extremely low entropy.

To overcome its defects, AMP(Authentication and key agreement via Memorable Password) which performs authentication and key agreement securely via low entropy password are presented. AMP uses Diffie-Hellman problem that depends on discrete logarithm problem.

Otherwise, this thesis applies elliptic curve cryptosystem to AMP for further efficiency. That is, this thesis presents EC-AMP(Elliptic Curve-AMP) protocol based on elliptic curve discrete logarithm problem instead of discrete logarithm problem, and shows its high performance through the implementation. EC-AMP secures against various attacks in the random oracle model just as AMP. Thus, we may supply EC-AMP to the network environment that requires authentication and key agreement to get both convenience and security from elliptic curve discrete logarithm problem.

Key words : Authentication, Password, Elliptic Curve

1. 서 론

사용자 인증(user authentication) 방법은 인증하고자 하는 대상에 따라 다음과 같이 구분할 수 있다.

① 생체인식: 사용자가 누구인가? (음성판별, 지문인식, 홍채인식)

② 토큰검사: 사용자가 가지고 있는가? (스마트 카드, OTP 토큰)

③ 패스워드: 사용자가 알고 있는가? (패스워드, PIN)

위 각 방법은 개별적으로 사용될 수 있고, 두 가지 이상의 방법을 복합적으로 사용하여 더욱 높은 보안성을 제공할 수도 있다.

생체인식 방법은 위조, 위장 등의 위험성은 적으나 별도의 부가 장비로 인한 비용이 소요되고, 특정 방법에 따라서는 사용자가 이용하기 거북스러워하는 경우도 있을 수 있다. 토큰검사 방법 또한 부가 장비로 인한 비용

† 정 회 원 : (주)정소프트 부설 연구소
velocity@jungsoft.com
†† 종신회원 : 한양대학교 전자계산학과
shinheu@cse.hanyang.ac.kr
논문접수 : 2002년 3월 2일
심사완료 : 2002년 9월 25일

소모와 항상 휴대하고 다녀야 하는 불편함이 있고 분실 및 도난의 위험도 가지고 있다.

패스워드를 이용하는 방법은 부가 장비가 필요 없고 가장 간편하다는 장점을 가지고 있으나 인간의 기억력에 의존하는 방법이기 때문에 낮은 엔트로피(entropy)를 가지고 있다. 또한 클라이언트와 서버가 연결되어있는 네트워크는 도청이 가능하기 때문에 네트워크 상에서 전송되는 패스워드나 해시(hash)된 값을 가로채서 공격하는 사전공격(dictionary attack)이 가능하다. 패스워드를 보관·관리하는 서버측 패스워드 파일이 유출될 경우에는 공격자가 합법적인 클라이언트나 서버로 위장하는 위장공격(impersonation attack), 사전공격이 가능하게 된다는 문제점을 가지고 있다.

패스워드 인증이 가지는 이러한 취약점 해결 방안으로 AMP(Authentication and key agreement via Memorable Password) 프로토콜이 제안되었다[1]. AMP는 확인자(verifier)를 이용하고 이산대수문제(Discrete Logarithm)를 기반으로 하는 Diffie-Hellman[2,3] 방식의 프로토콜이며 랜덤 오라클 모델에서 증명 가능한 접근(provable approach)으로 안전한 인증 및 키교환을 수행한다[1].

그러나 본 논문에서 제안하는 EC-AMP(Elliptic Curve-AMP) 프로토콜은 이산대수문제대신에 타원곡선 이산대수문제에 기반하여 상호인증 및 키교환을 수행한다. 이로 인해 보다 높은 보안성과 효율성을 가질 수 있으며[4,5], 랜덤 오라클 모델(random oracle model)에서 여러 가지 공격에 대해 안정성을 제공한다.

본 논문은 2장에서 기존 연구에 대해 알아보고, 3장에서는 EC-AMP 프로토콜을 설계하며 프로토콜의 보안성과 효율성에 대해서 다룬다. 4장에서 구현 및 수행 성능에 관해 서술하고, 5장에서 결론을 맺는다.

2. 관련 연구

2.1 패스워드 기반 인증 프로토콜

가장 간단하면서 현재 널리 쓰이고 있는 인증 방법은 Unix 시스템에서의 인증 방법이다. 그러나 Unix 시스템에서 사용되는 패스워드 인증 시스템은 평문수준(plain text-equivalent)의 인증 시스템이다. 사용자가 입력하는 패스워드는 단방향 함수로 처리하여 /etc/passwd 파일에 저장한다. 만약 이 패스워드 파일이 공격자에게 유출된다면 오프라인 사전공격에 의해 사용자의 패스워드가 노출되기 쉬우므로 평문과 동일한 수준의 보안성을 가지게 된다[1,7].

보안적 취약점을 보완하고자 패스워드 기반의 challenge-response 방법을 사용할 수 있다. challenge-response는 다음과 같은 방법으로 설명된다[1,6].

1. A는 B에게 랜덤 값 x를 보낸다.
2. B는 A에게 랜덤 값 y를 보낸다. (challenge)
3. A는 미리 알고 있던 패스워드와 두 개의 랜덤 값을 사용하여 계산을 수행한 뒤 이를 B에게 보낸다. (response)
4. B는 A와 동일한 계산을 하여 response가 맞는지 검사한다.

challenge-response 방법은 인증 할 때마다 challenge 값이 바뀌기 때문에 중간에서 가로챈 값이 계속적으로 사용될 수 없다. 그러나 패스워드를 이용하여 response를 생성 할 경우 challenge, response 값들을 가로챈 뒤 사전공격을 통해 패스워드를 추측할 수 있으므로 패스워드 기반의 challenge-response 방법도 평문수준의 보안성을 가진다[7].

그 후 제안된 EKE(Encrypted Key Exchange)[6] 프로토콜은 대칭키 암호화와 공개키 암호화 기법을 조합한 패스워드 기반의 인증 및 키교환 프로토콜이다[6]. EKE는 중간에 가로챈 정보로부터 패스워드를 유추해 내기 어렵게 하여 사전공격으로부터 패스워드가 유출되는 것을 막았다. 뿐만 아니라 현재의 세션키를 알아내더라도 미래의 인증 과정에서 유도되는 세션키는 알 수 없다는 forward secrecy를 제공한다.

그러나 클라이언트와 서버측 모두 패스워드 혹은 해시된 패스워드 값을 사용하기 때문에, 이것은 평문수준의 보안성을 가진다는 것을 의미하며, 서버측의 패스워드 파일이 유출될 경우에 안정성이 깨진다는 것을 알 수 있다. 이러한 구조를 대칭 셋업이라 하며, EKE의 변형 혹은 일종인 DH-EKE, SPEKE도 이러한 문제점을 가지고 있다.

A-EKE(Augmented EKE)는 최초의 확인자 기반(verifier based) 프로토콜이다[7]. 클라이언트는 패스워드를 알고 있는 반면 서버에서는 확인자만을 가지고 인증을 하게된다. 그렇기 때문에 확인자가 유출되더라도 사용자의 패스워드를 직접 알아 낼 수 없다. 이러한 구조를 비대칭 셋업이라 한다. 그러나 A-EKE는 EKE가 제공하는 forward secrecy를 제공하지 못한다.

이러한 EKE 계열 프로토콜의 단점에도 불구하고 EKE, DH-EKE, A-EKE 등과 같은 프로토콜로 인해 패스워드 인증 프로토콜 연구가 활성화 되게 되었고 [1,3,6], SRP[7], GXY 등의 프로토콜이 제안되었다[1,7]. 그러나 이들 대부분의 프로토콜이 안전함을 증명하기에 불충분하거나 패스워드를 임시적인 방법으로 보호하기 때문에, 이들 중 몇몇 프로토콜들은 이미 깨졌거나 해독

중이다. 반면에 OKE를 시작으로 SNAPI, AMP, PAK 등은 증명 가능한 접근을 제시함으로써 발전된 패스워드 인증 프로토콜을 제시하였다[1,8,9].

A-EKE, SRP, GXY 등은 확인자 기반 프로토콜이다. 그러나 확인자 기반 프로토콜이라도 서버의 확인자가 유출될 경우 사전공격이나 서버위장을 통해서 불법 접근이 가능하게 된다.

RSA 등의 공개키 암호화를 사용하여 사용자 인증을 하게되면 또 다른 문제점을 야기한다. 공개키 암호화 구조를 이용하려면 서버와 클라이언트는 엄청난 오버헤드를 유발하는 키생성을 해야하고 각각 자신의 개인키를 "안전하게" 보관하여야 한다. 뿐만 아니라 상대방의 공개키가 변조되지 않았음을 보장받아야 한다. 이는 곧 또 다른 여러 가지 문제를 불러일으키므로 패스워드 기반 인증으로부터 얻을 수 있는 "간편함"이라는 성질에 위배된다. 이러한 관점에서 본다면 EKE 계열의 프로토콜은 비효율적인 프로토콜이라고 할 수 있다.

2.2 타원곡선 암호 시스템

본 논문은 기존의 AMP 프로토콜에 타원곡선 암호 시스템을 적용시켜 그 효율을 높여 EC-AMP를 제안한다. EC-AMP에 사용되는 타원곡선 암호 시스템에 대해 간략히 살펴본다.

2.2.1 타원곡선의 정의

p를 3이상의 소수라고 했을 때, 유한체(finite filed) F_p에서의 타원곡선 E는 다음과 같이 정의된다[10,11].

$$y^2 = x^3 + ax + b \quad (a, b \in F_p, 4a^3 + 27b^2 \neq 0 \bmod p) \quad (1)$$

집합 $E(F_p)$는 위 식 (1)을 만족하는 유한체 F_p내의 수의 쌍 (x, y)로 구성되고, O는 무한의 점(point at infinity)으로 정의한다.

예를 들어 $p=23$이고 E: $y^2 = x^3 + x + 4$라 하면,

$4a^3 + 27b^2 = 4 + 432 = 436 \equiv 22 (\bmod 23)$이므로 적당한 타원곡선이 된다. 이때, 타원곡선 E위의 모든 점은 그림 1과 같다.

(0,2)	(0,21)	(1,11)	(1,12)	(4,7)	(4,16)	(7,3)
(7,20)	(8,8)	(8,15)	(9,11)	(9,12)	(10,5)	(10,18)
(11,9)	(11,14)	(13,11)	(13,12)	(14,15)	(14,18)	(15,6)
(15,17)	(17,9)	(17,14)	(18,9)	(18,14)	(22,5)	(22,19)

그림 1 타원곡선 E: $y^2 = x^3 + x + 4$위의 모든 점

2.2.2 타원곡선에서의 연산

1) 두 점의 덧셈

타원곡선 위의 점 2개를 더했을 때, 또 다른 타원곡선 위의 점이 덧셈에 대한 결과가 된다.

타원곡선 위의 서로 다른 두 점을 $P=(x_1, y_1)$, $Q=(x_2, y_2)$라 할 때, 두 점의 합을 $R=(x_3, y_3)$라하고, 다음과 같이 구한다.

1. P와 Q를 지나는 직선을 그린다.
2. 이 직선과 타원 곡선과 만나는 제 3의 점을 R'라 한다.
3. R'를 x축으로 대칭이동 시킨 점을 R이라 한다.

기하학적인 두 점의 덧셈은 그림 2에 나타나 있다.

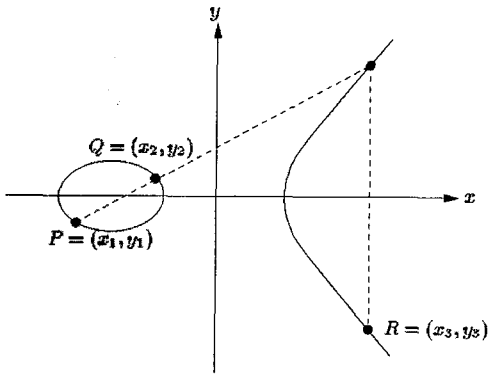


그림 2 타원곡선 위의 두 점의 덧셈

2) 한 점의 2배

타원곡선 위의 점 $P=(x_1, y_1)$를 2배 한 점을 $R=(x_3, y_3)$라하고, 다음과 같이 구한다.

1. P의 접선을 긋는다.
2. 접선과 타원곡선이 만나는 점을 R'라 한다.
3. R'를 x축으로 대칭이동 시킨 점을 R이라 한다.

기하학적인 2배 과정은 그림 3에서 볼 수 있다.

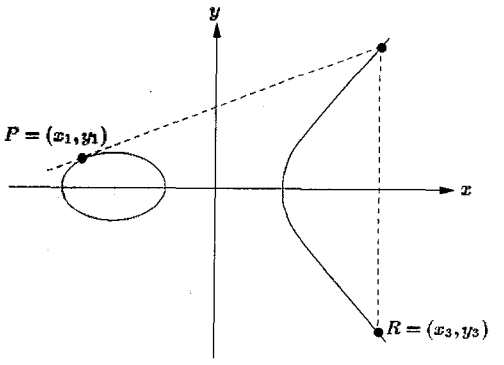


그림 3 타원곡선 위의 한 점의 2배

3) 기본 수식

기하학적인 방법으로부터 다음의 수식들을 유도할 수 있다[10,11].

1. $P+O=O+P=P,\ \forall P\in E(F_p)$
2. $P=(x,y)$일 때, $-P=(x,-y)$
3. $P+(-P)=O$
4. $P=(x_1,x_2), Q=(x_2,y_2)$일 때, $P+Q=(x_3,y_3)$,
$x_3=(\frac{y_2-y_1}{x_2-x_1})^2-x_1-x_2,\ y_3=(\frac{y_2-y_1}{x_2-x_1})(x_1-x_3)-y_1$
5. $P=(x_1,y_1)$일 때, $2P=(x_3,y_3)$,
$x_3=(\frac{3{x_1}^2+a}{2y_1})^2-2x_1,\ y_3=(\frac{3{x_1}^2+a}{2y_1})(x_1-x_3)-y_1$

예를 들어 $P=(4,7)$, $Q=(13,11)$일 때, $P+Q=(15,6)$이다.

2.2.3 타원곡선의 기본 성질

유한체 F_q 위의 타원 곡선 E가 가지는 점의 개수는 $\#E(F_q)=q+1-t$ (단, $|t|\le 2\sqrt{q}$)로 나타내고, 이를 E의 위수(order)라 한다.

q가 소수일 경우, E는 순환(cyclic)하고 E 위의 한 점 $E(F_q)=\{kP:\ k\in Z\}$인 생성자(generator) P가 존재한다.

2.2.4 타원곡선이산대수문제와 이산대수문제의 수학적 비교

표 1 타원곡선이산대수문제와 이산대수문제

	이산대수문제	타원곡선이산대수문제
군	Z_p^*	E 위의 점 (x,y)와 O
연산	Z_p^*내의 곱셈	E 위에서 덧셈
정의	g와 p가 주어지고 $g^x \bmod p$에서 x를 찾는 문제	E와 그 위의 점 P가 주어지고 $Q=xP$에서 x를 찾는 문제

두 문제의 보안성과 성능에 대한 언급은 5장에서 찾아볼 수 있다.

2.3 AMP

2000년에 권태경이 AMP(Authentication and Key Agreement via Memorable Password)[1]를 제안하였다. AMP는 엔트로피가 낮은 패스워드를 이용하여 안전한 인증 및 키교환을 위해 패스워드를 증폭하고 시간 유동적인 파라미터를 추가했다. 패스워드 증폭의 기본개념과 AMP^n 프로토콜에 대해 간략히 살펴본다.

2.3.1 패스워드 증폭

사전공격에 취약한 낮은 엔트로피의 패스워드를 높은 엔트로피의 인자를 추가하는 것이 AMP에서 제안하는 패스워드 증폭의 개념이다.

패스워드 π에 시간 유동적인(높은 엔트로피) x를 선택해서 $\pi+x \bmod q$를 패스워드 대신에 사용한다. $\pi^x \bmod q$를 사용하게 되면 F_q에서의 π의 위수가 상대적으로 높지 않을 수 있기 때문에 $\pi^x \bmod q$를 패스워드 증폭의 방법으로 사용하는 것은 좋지 못한 방법이라고 할 수 있겠다. 여기서 사용되는 x는 항상 비밀하게 유지해야 하고 프로토콜이 실행할 때마다 새로 생성되어야 한다. 낮은 엔트로피의 사용자 패스워드 π는 추측이 가능하지만 $\pi+x \bmod q$는 추측이 불가능하다[1].

이 방법은 패스워드에 대한 정보를 누출시키지 않기 때문에 영지식 증명(zero-knowledge proof)의 성질과 유사하다.

2.3.2 AMP^n

AMP와 이를 간략화한 AMP^n(AMP-naked)은 challenge-response 방식으로 설계된다. 즉, 초기값(접속요청 등)을 Alice가 Bob에게 전송하고, Bob이 challenge를 Alice에게 보낸 뒤, Alice가 challenge에 대한 response를 보냄으로써 프로토콜이 완성된다[1,3,6]. 그림 4에 나타나 있다.

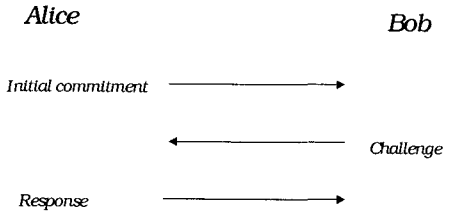


그림 4 challenge-response

AMP^n 프로토콜은 다음과 같다.

우선 Alice와 Bob은 곱셈군(multiplicative group) Z^*_p에서 소수-위수 서브그룹(prime-order subgroup)을 생성하는 생성자 g와 g의 위수 q를 공유한다.

Alice는 난수 x를 선택하여 g^x를 Bob에게 보내면, Bob은 난수 y와 Alice의 패스워드 π로부터 생성된 g^π를 이용하여 $(g^x g^\pi)^y=g^{(x+\pi)y}$를 계산하여 Alice에게 보낸다. Alice는 $(g^{(x+\pi)y})^{(x+\pi)^{-1}x}=g^{xy}$를 계산하고, Bob은 $(g^x)^y=g^{xy}$를 얻음으로 g^{xy}를 비밀리에 공유하게 된다. 여기서 사용된 지수는 Z_q위의 수가 된다.

3. EC-AMP

기존의 AMP프로토콜에 타원곡선암호를 적용한 EC-

AMP를 설계 및 정의하고, 이의 수행성능과 안정성에 대해 기술한다.

3.1 파라미터 정의

EC-AMP 프로토콜은 Alice(클라이언트)와 Bob(서버) 간의 프로토콜로 정의하며, Eve(공격자)는 수동공격 및 능동공격을 시도한다. π는 패스워드, τ는 임의의 솔트(salt)를 의미한다. $\{0,1\}^*$, $\{0,1\}^\infty$는 각각 한정된 길이의 이진수와 무한길이의 이진수를 나타낸다.

$h(): \{0,1\}^* \mapsto \{0,1\}^k$는 충돌 없는 단방향 해시함수라 가정한다. 즉, EC-AMP 프로토콜에 사용된 모든 해시함수는 랜덤 오라클 성질을 가진다고 가정한다.

EC-AMP 프로토콜은 유한체 F_p(p는 소수)에서 타원곡선 E의 한 점 P가 주어지고 $Q=xP$일 때 x를 찾기가 계산상 불가능(*computationally infeasible*)하다는 타원곡선이산대수문제에 기반한다. 여기서 타원곡선 E는

$$y^2 = x^3 + ax + b,$$
$$(a, b \in F_p, 4a^3 + 27b^2 \neq 0 \bmod p) \quad (1)$$

로 정의하고, E위의 점 P, Q의 덧셈연산 $P(x_1, y_1) + Q(x_2, y_2) = (x_3, y_3)$는 다음과 같이 정의한다.

$$\lambda = \begin{cases} \dfrac{3x_1^2 + a}{2y_1}, & \text{if } P = Q \\ \dfrac{y_2 - y_1}{x_2 - x_1}, & \textit{otherwise} \end{cases} \quad (2)$$

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = \lambda(x_1 - x_3) - y_1 \end{cases}$$

모든 연산은 유한체 F_p에서의 연산이므로 이후로 $\bmod p$는 생략한다.

앞에서 살펴본 바와 같이 AMP 프로토콜은 곱셈군 Z_p^*와 이것의 소수-위수 서브그룹 Z_q를 정의한다. Diffie-Hellman 문제에 기반하기 때문에 지수연산을 하는데, AMP에 사용되는 연산은 서브그룹 연산이기 때문에 지수는 Z_q위의 수이다. 이와 유사하게 EC-AMP에서는 타원곡선 E 위의 점 P의 위수를 q라 할 때, 점 P의 계수는 F_q위의 연산을 하게 된다.

3.2 패스워드 증폭

EC-AMP 프로토콜의 기본개념은 AMP와 마찬가지로 낮은 엔트로피의 패스워드를 높은 엔트로피로 증폭시키고 시간 유동적인 파라미터를 추가하는 것이다. 또한 패스워드에 대한 정보를 누출시키지 않는 영지식 증명(zero-knowledge proof)과 유사한 성질을 가지기 때문에 사전공격이 불가능하다.

패스워드 π에 임의(높은 엔트로피)의 x를 선택해서 $\pi + x \bmod q$를 패스워드 대신에 사용한다. 여기서 사용되는 x는 ANSI X9.17 등의 의사난수생성기를 이용하여 생성하고, 항상 비밀하게 유지해야 하며, 프로토콜이 실행할 때마다 새로 생성되어야 한다.

3.3 EC-AMP^n 프로토콜

3.3.1 개념

Alice는 패스워드 π를 알고 있고, Bob은 πP를 가지고 있다고 가정하자. 증폭된 패스워드 프로토콜은 challenge-response 방식으로 설계된다. 즉, 초기값을 Alice가 Bob에게 전송하고, Bob이 challenge를 Alice에게 보낸 뒤, Alice가 response를 보냄으로써 프로토콜이 완성된다.

Alice는 높은 엔트로피를 가지는 난수 x를 선택하여 xP를 Bob에게 보내고, Bob은 높은 엔트로피의 난수 y를 선택하여 $(xP + \pi P)y = (x + \pi)yP$를 계산한 뒤 Alice에게 보낸다. 이 값이 challenge역할을 하게 된다. 타원곡선이산대수문제에 의하여 x, y, π에 대한 정보는 외부에 노출되지 않는다.

한편, Alice는 x, π를 알기 때문에 $w = (x + \pi)^{-1} \bmod q$를 계산할 수 있으며, x, π를 알지 못하는 다른 이는 이 값을 계산할 수 없다.

$$(x + \pi)yP \cdot w = ((x + \pi)(x + \pi)^{-1} y \bmod q)P = yP \quad (3)$$

위 식 (3)의 성질을 이용하면 w를 알고 있는 Alice만이 Bob이 안전하게 가지고 있는 yP를 생성해 낼 수 있다. 여기서 타원곡선 E위의 한 점 P의 위수는 q이므로, P에 곱해지는 계수는 유한체 F_q 위의 연산이다.

Alice와 Bob은 이로써 xyP라는 둘만이 알고있는 값을 계산해 내게 되고, 이것이 곧 response가 된다. 실제로는 xyP가 세션키가 되고, response로는 해시된 $h(xyP)$를 사용하게 된다.

Bob이 πP를 안전하게 보관하는 것이 중요하다. 만약에 πP이 유출될 경우에는 위장공격이 가능하게 된다.

3.3.2 설계

각 데이터를 전송하는 Alice와 Bob사이의 통신채널은 안전하지 않은 채널이라 가정하고, 3.3.1에서의 내용을 기반으로 EC-AMP^n 프로토콜을 설계하면 다음과 같다.

1. Alice는 F_q위의 엔트로피가 높고, 랜덤한 값 x를 선택하여 자신의 id와 xP를 Bob에게 전송한다.
2. Bob은 해당 id의 확인자 πP를 불러낸 뒤 $\theta_2 = (x + \pi)yP$를 Alice에게 보낸다. 그 후, θ_1에 y를 곱해서 $\beta = xyP$를 계산한다.
3. Alice는 $w = (x + \pi)^{-1} x \bmod q$와 θ_2를 곱해서 $\alpha = xyP$ 를 유도해 낸다.

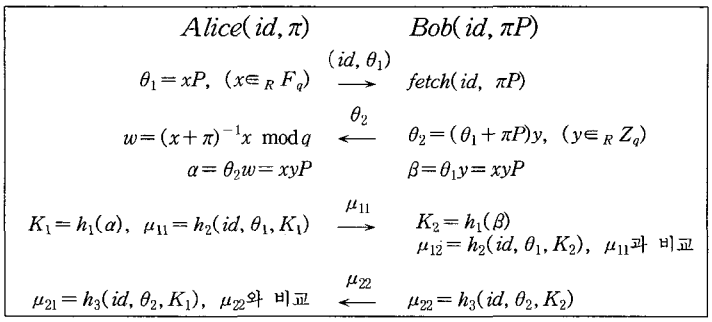


그림 5 EC-AMP^n

4. Alice와 Bob은 각각 α, β로부터 해시함수를 통해 μ_{11}, μ_{12}를 생성한다.
5. Alice는 μ_{11}을 Bob에게 보내고, Bob은 자신이 생성한 μ_{12}와 μ_{11}를 비교하여 일치하지 않으면 프로토콜을 중지하고, 일치하면 다음으로 진행한다.
6. Alice와 Bob은 각각 μ_{21}, μ_{22}를 계산한다.
7. Bob은 μ_{22}를 Alice에게 보내고, Alice는 자신이 생성한 μ_{21}과 일치하는지 검사한다. 일치하지 않으면 프로토콜을 중단하고, 일치하면 상호인증이 완료된다. 세션키를 xyP로 하며 종료한다.

EC-AMP^{n}을 그림으로 나타내면 그림 5와 같다.

3.3.3 취약점

EC-AMP^n(EC-AMP-naked)은 3.3.1에서 서술한 바와 같이 πP가 유출되었을 경우 클라이언트 위장공격에 취약하다.

예를 들어, Eve가 πP를 알고 있다면 다음의 과정이 가능하다.

1. Eve는 $x(\pi P)$를 Bob에게 보낸다.
2. Bob은 $\theta_2 = (x+1)y\,\pi P$, $\beta = xy\pi P$를 계산한다.
3. x를 알고있는 Eve는 θ_2로부터 $xy\pi P$를 계산할 수 있다.
4. 이후의 과정은 문제없이 수행될 수 있다.

이는 서버의 파일이 유출되어도 안전성을 보장하는 비대칭 셋업의 보안성을 보장하지 못함을 의미한다. 위와 같은 취약점을 극복하기 위해서 서버측의 확인자(패스워드 파일) 또한 증폭하는 방법을 쓸 수 있다.

3.4 EC-AMP 프로토콜

3.4.1 증폭된 패스워드 파일

앞서 3.3.3에서 서술한 것과 같이 서버측의 패스워드 확인자가 누출되었을 경우에는 위장공격이 가능하다. 이러한 경우를 막기 위해 서버측 패스워드 파일도 클라이언트의 패스워드를 증폭하듯이 증폭시킨다.

서버는 πP대신에 임의의 τ와 서버측 패스워드 ζ로 구성된 확인자를 가지며, ζ는 보안을 위해 스마트카드 등의 보안성 있는 저장장치에 보관할 수 있다.

증폭된 패스워드 파일 ν는 다음과 같은 형태를 지닌다.

$$\nu = (\zeta+\tau)^{-1}uP \quad (\text{단, } u = h(id,\pi),\ \zeta \in_R F_q,\ \tau:\{0,1\}^k) \tag{4}$$

EC-AMP 프로토콜을 준비과정인 셋업과정과, 그 후의 실행과정으로 구분하고, 각 데이터를 전송하는 Alice와 Bob사이의 통신채널은 안전하지 않은 채널이라 가정한다.

3.4.2 프로토콜 셋업

1. 전역 파라미터; Alice와 Bob은 타원곡선 E와 그 위의 점 P, p, q를 공유한다.
2. 등록; 안전한 방법을 통해서 Alice는 선택한 패스워드를 Bob에게 전달한다. 예를 들어, 방문등록을 통해서 $u = h_1(id,\pi)$를 Bob에게 전달한다.
3. 서버저장; Bob은 임의의 τ, $(\tau \in_R \{0,1\}^k)$를 선택한 뒤 다음을 확인자로 저장한다.

$$(id, \tau, \nu = (\zeta+\tau)^{-1}uP) \tag{4}$$

3.4.3 프로토콜 실행

1. Alice는 $\theta_1 = xP$, $(x \in_R F_q)$을 생성해 Bob에게 (id, θ_1)을 보낸다.
2. Bob은 해당 사용자의 τ와 ν를 읽어온 뒤 θ_2를 생성하여 Alice에게 보낸다.

$$\theta_2 = (\theta_1)y + (\zeta+\tau)y\nu,\ (y \in_R F_q) \tag{5}$$

3. Alice는 θ_2를 기다리는 동안 다음을 계산한다.

$$u = h_1(id,\pi),\ \chi = (x+u)^{-1} \bmod q \tag{6}$$

θ_2를 받으면,

$$e = h_2(\theta_1, \theta_2, id, Alice, Bob),\quad w = \chi(x+e) \bmod q,\ \alpha = \theta_2 w \tag{7}$$

을 계산한다. 결과적으로,

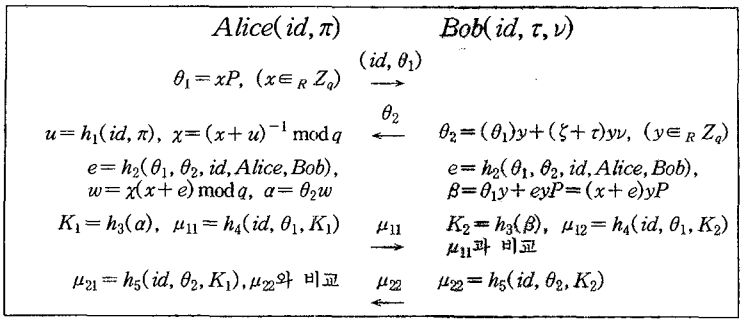


그림 6 EC-AMP

$$\begin{aligned} \alpha &= \theta_2 w = (x+u)yPw \\ &= (x+u)yP(x+u)^{-1}(x+e) \\ &= (x+e)yP \end{aligned} \tag{8}$$

을 계산하게 된다. 그 후에

$$K_1 = h_3(\alpha),\ \mu_{11} = h_4(id, \theta_1, K_1) \tag{9}$$

을 계산하여 Bob에게 μ_{11}를 보낸다.

4. μ_{11}를 기다리는 동안 Bob은

$$\begin{aligned} e &= h_2(\theta_1, \theta_2, id, Alice, Bob), \\ \beta &= \theta_1 y + eyP = (x+e)yP \end{aligned} \tag{10}$$

을 계산하고,

$$K_2 = h_3(\beta),\ \mu_{12} = h_4(id, \theta_1, K_2) \tag{11}$$

을 계산하여 도착된 μ_{11}와 μ_{12}를 비교하여 둘이 일치하면,

$$\mu_{22} = h_5(id, \theta_2, K_2) \tag{12}$$

을 생성하여 Alice에게 보낸다. 만약 일치하지 않으면 모든 과정을 중단한다.

5. Alice는 μ_{22}를 기다리며

$$\mu_{21} = h_5(id, \theta_2, K_1) \tag{13}$$

을 계산하고 수신된 μ_{22}와 일치하게 되면 상호인증이 완료된다. 만약 일치하지 않으면 모든 과정을 중단한다.

이 과정은 그림 6과 같다. 결과적으로 상호인증을 통해 비밀키 $\alpha = \beta = (x+e)yP$를 서로 공유하게 된다.

3.5 보안성 및 효율성

3.5.1 보안성

EC-AMP 프로토콜에 사용된 모든 해시함수는 랜덤 오라클 성질을 가지며 타원곡선이산대수문제를 풀기는 계산적으로 불가능하다고 가정한다. 이러한 가정 하에서 다음의 특성들을 만족한다.

1. Perfect forward secrecy를 제공한다;
   패스워드(π 혹은 u)가 유출된 경우라도 타원곡선이산대수문제를 풀 수 없기 때문에 이전에 사용된 세션키 (α 혹은 β)를 얻을 수 없다.
2. Denning-Sacco 공격에 안전하다;
   이전에 사용된 세션키가 유출되어도 타원곡선이산대수문제를 풀 수 없기 때문에 사용자의 패스워드를 알아낼 수 없다.
3. Replay 공격에 안전하다;
   θ_1, θ_2, μ_{11}, μ_{22} 등에는 해당 세션에만 해당되는 일시적인 파라미터가 들어있는데 이러한 파라미터를 찾는 것은 타원곡선이산대수문제에 의하여 알아낼 수 없다.
4. 온라인 추측공격에 안전하다;
   온라인 상에서 패스워드를 추측하여 시도하는 공격은 횟수를 제한하므로 막을 수 있다.
5. 확인자 유출;
   EC-AMP^n 프로토콜을 제외한 EC-AMP 프로토콜은 패스워드파일이 유출될 경우라도 ζ를 모르면 사전공격이나, 서버위장을 할 수 없다.

3.5.2 효율성

이미 널리 쓰이는 RSA(인수분해문제)나 DSA(이산대수문제)같은 알고리즘은 현재 허용되는 보안기준(10^{12} MIPS year)를 만족하기 위해 1024-bit 크기의 유한체를 사용하여야 하는 반면에 ECC(Elliptic Curve Cryptography)는 160-bit 크기의 유한체로도 충분하다[4]. 더 나아가 보다 높은 보안성이 필요할 경우에 RSA, DSA는 2차 함수적인 많은 비트 수를 필요로 하는 것에 비해 ECC는 약간의 추가 비트 수로도 훨씬 높은 보안성을 만족한다[4]. 예를 들어, 2048-bit 크기의 유한체를 사용하는 RSA, DSA와 192-bit 크기의 유한체의 ECC는 동일한 보안성을 제공한다. 다음 그림 7을 보면 쉽게 비교가 가능하다.

수행 성능 측면에서 ECC와 기존 RSA, DSA를 비교해 본다면, RSA, DSA는 유한체 상의 곱셈과 모듈러

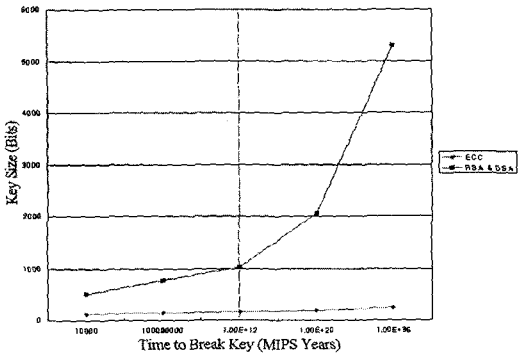


그림 7 ECC와 RSA, DSA의 보안성

연산의 연속으로 이루어져 있고, ECC의 경우는 타원곡선 위에 존재하는 점의 덧셈의 연속으로 이루어져 있다. 타원곡선 위의 점의 덧셈은 2장에서도 설명한 것과 같이 여러 개의 정수 덧셈과 정수 곱셈으로 이루어져 있다. 따라서 동일한 크기의 유한체에서 연산을 할 경우에 RSA, DSA가 ECC에 비해서 더욱 높은 수행 성능을 갖는다. 그러나 그림 7에서 볼 수 있듯이 ECC가 필요로 하는 유한체의 크기가 훨씬 작다. 때문에 동일한 보안성을 가지는 유한체의 크기끼리 비교를 한다면 ECC가 RSA, DSA에 비해 효율적이다[4]. 예를 들어, 160-bit의 ECDSA[11]와 1024-bit의 DSA를 비교했을 때, ECC가 대략 2~6배정도 빠르면서 비슷한 정도의 보안성을 가진다[4,5]. 결과적으로 Diffie-Hellman 기반의 AMP 프로토콜보다 ECC 기반의 EC-AMP 프로토콜이 더욱 빠른 수행시간을 보장한다.

그리고 AMP^{n}과 AMP 프로토콜은 Alice와 Bob사이에 총 4개의 패킷이 전송된다. 그 중에서 μ_{11}, μ_{22}는 해시된 값의 결과이기 때문에, SHA-1이나 HAS-160을 해시함수로 사용한다면, μ_{11}, μ_{22}의 크기는 160-bit가 된다. 그러나, θ_1, θ_2는 F_p의 숫자이므로 대략 1024-bit 크기를 가진다. EC-AMP 계열에서의 μ_{11}, μ_{22}는 AMP 계열과 그 크기가 같지만 EC-AMP 계열에서의 θ_1, θ_2는 타원곡선 위의 한 점이 된다. (x, y)의 쌍으로 나타내어지기 때문에 이는 사용되는 유한체 크기의 약 2배 정도의 크기를 가질 것이다. 384-bit 혹은 320-bit의 크기이다.

총 4개-그 중 큰 데이터는 θ_1과 θ_2뿐이다-의 데이터만이 전송되므로 통신부하측면에서 살펴본다면 표 2에서 볼 수 있듯이 A-EKE, B-SPEKE, PAK-X 등보다 효율적이고, SRP, GXY 등과 비슷하다[1]. 표 2안에 굵은 글씨로 써진 것은 비교하는 프로토콜 중에 해당 부분에서 성능이 가장 좋다는 것을 의미한다.

위 프로토콜들의 공통으로 계산복잡도가 가장 높은 부분은 지수곱셈 혹은 타원곡선연산이다. EC-AMP와 AMP는 클라이언트측, 서버측의 복잡한 연산($\theta_1, \theta_2, \alpha$ or β)이 각 2회로 가장 적다. 또한 두 작업이 병렬로 겹치는 부분이 한 군대 존재하기 때문에 병렬처리적인 입장에서 본다면 총 3단계의 큰 연산으로 프로토콜이 완성된다. 따라서 계산복잡도 측면에서는 위 프로토콜 중에서 가장 효율적이라고 할 수 있다.

4. 구현 및 수행 성능

본 논문에서는 EC-AMP와 AMP 프로토콜의 키를 생성하는 부분까지만 구현하였다. 왜냐하면 해시값을 주고받음으로 상호인증을 완성하는 부분은 해시된 값을 전송하기 때문에 두 프로토콜의 성능이 동일하기 때문이다.

그리고 실험적인 목적으로 클라이언트와 서버를 하나의 호스트에서 실행하였기 때문에 실제 발생하는 네트

표 2 AMP계열과 다른 프로토콜의 비교

	Protocol Steps	Large Blocks	Exponentiations			Multiplication On EC			Random Numbers	
			Client	Server	Parallel	Client	Server	Parallel	Client	Server
A-EKE	7	3	4	4	6				**1**	**1**
B-SPEKE	4	3	3	4	6				**1**	2
SRP	4	**2**	3	**2**	4				**1**	**1**
GXY	4	**2**	4	3	5				**1**	**1**
SNAPI-X	5	5	5	4	7				2	3
AuthA	5/3	**2**	4	3	6				**1**	**1**
PAK-X	5/3	3	4	4	8				**1**	2
AMP	4	**2**	**2**	**2**	**3**				**1**	**1**
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	**1**	**1**

워크의 지연시간 등을 고려하지 않았다. 또한, 중간 결과를 텍스트로 출력하는 점도 속도에 영향을 미치는 요소이다.

4.1 환경

본 논문의 구현은 Java 언어를 사용하였다. 구현을 위해 사용한 환경은 다음과 같다.

표 3 구현 환경

	Intel Pentium 3 700MHz
	Windows 2000 Professional
	Java
	JDK 1.3.1_02

4.2 소수, 타원곡선

AMP와 EC-AMP와의 수행성능 비교를 위해 각각에 사용되는 소수와 타원곡선, 생성자 등을 정의한다.

AMP 프로토콜 구현에 사용된 소수는 SKIP(Simple Key management for Internet Protocols)에서 제안한 방법으로 생성된 안전한 소수(safe prime)를 사용했다[12]. SKIP에서 제안한 소수 p는 $(p-1)/2$도 소수임을 만족하므로 p의 소수-위수 서브그룹 q는 $(p-1)/2$이다.

표 4 AMP를 위한 1024-bit 소수

	171718397966129586011229151993178480901904202533705695869569760169920539808075437788747086722975900425740754301098468647941395164593810074170462799608062493021989285837416815548721035874378548121236050948528229416139585571568998066586304075565145536350296006867635076744949977849997684222020336013226588207303
	858591989830647930056145759965892404509521012668528479347848800849602699040377188943735433614879502128703771505492343239706975822969050370852313998040312465109946429187084077743605179371892740606180254742641147080697927857844990332931520377825727681751480034338175383724749889249988842111010168006613294103651

표 4는 1024-bit 크기의 소수 p와 p의 소수-위수 서브그룹인 q를 보여준다.

마찬가지로 표 5는 2048-bit 크기일 경우의 p와 q를 각각 보여준다.

표 5 AMP를 위한 2048-bit 소수

	31087337795061487877547416545715496334920954980132212151448781444321393445568157959166911302972918628838917381559539620290244963511997037011253946065678925033455872043721454426215650798450188675325621498188688302603627388365642425546473761584899398546726625631228589029183157123265299738241899897560139599077166257814263354432724020387267456594044458497157226037520021564951601668256091905149808373739011153824316842260356584928931097012930709279713696588076097146536216639697002502410139891180002231258705541413293860269631209702305813614701588402302998104362562812340366960005570331931340105075488237470969553357627
	15543668897530743938773708278577481674604774900661060757243907221606967227840789795834556514864593144194586907779698101451224817559985185056269730328394625167279360218607272131078253992250943376628107490943441513018136941828212127732368807924496992733633128156142945145915785616326498691209499487800697995385831289071316772163620101936337282970222292485786130187600107824758008341280459525749041866950557691215842113017829246446554850646535463985684829403804857326810831984850125120506994559000111562935277070664693013481560485115290680735079420115149905218128140617018340800278516596567005253774411873548477667881 3

EC-AMP 프로토콜 구현에 사용된 타원곡선 E와 유한체의 위수 p 등은 NIST(National Institute of Standards and Technology)에서 1999년 6월에 제안한 것을 사용했다[13].

192-bit의 크기를 가지는 P-192는 다음 표 6와 같다.

표 6 EC-AMP를 위한 192-bit 타원곡선

	6277101735386680763835789423207666416083908700390324961279
	-3
	64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1
	188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012
	07192b95ffc8da78631011ed6b24cdd573f977a11e794811
	6277101735386680763835789423176059013767194773182842284081

a, b는 타원곡선 E를 $y^2=x^3+ax+b$라 했을 때의 계수이고, xG, yG는 각각 E의 생성자의 x좌표와 y좌표의 값을 나타낸다. n은 생성자 G의 위수이다.

4.3 실행 결과

4.3.1 AMP의 실행 결과

AMP의 경우 1024-bit와 2048-bit 크기의 유한체를 가지고 각각 구현을 하였다.

그림 8은 1024-bit 크기의 유한체를 가지고 있는 AMP 클라이언트의 실행 결과이다.

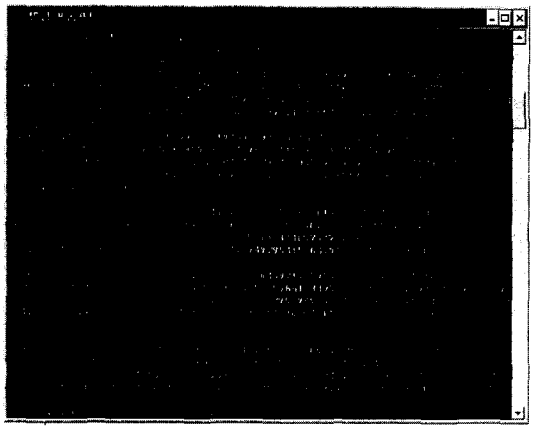


그림 8 AMP(1024-bit)의 클라이언트 결과

그림 9는 AMP 서버의 실행 결과이다.

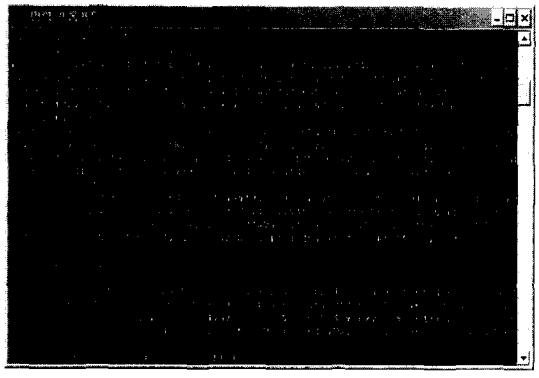


그림 9 AMP(1024-bit)의 서버 결과

4.3.2 EC-AMP의 실행 결과

그림 10은 192-bit 크기의 EC-AMP 클라이언트의 실행 결과를, 그림 11은 EC-AMP 서버의 실행 결과를 보여준다.

그림 10 EC-AMP(192-bit)의 클라이언트 결과

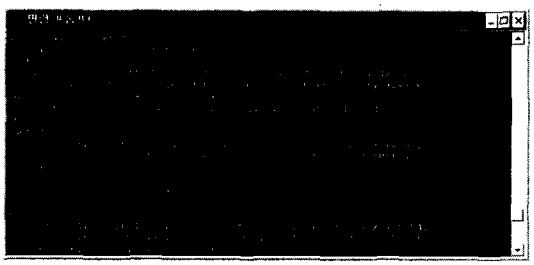
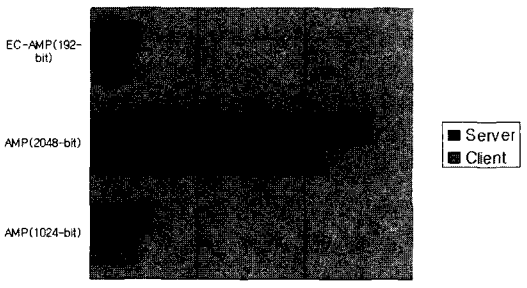


그림 11 EC-AMP(192-bit)의 서버 결과

4.4 수행 성능 및 기대 효과

각 프로토콜마다 10회씩 실행하여 측정한 평균값을 사용했다. 프로토콜을 수행하는데 걸리는 시간은 그림 12에 나타나 있다.



	AMP(1024-bit)	AMP(2048-bit)	EC-AMP(192-bit)
Server	1120.7	5239.4	907.4
Client	870.1	4354.2	773.2

그림 12 EC-AMP와 AMP의 수행 성능

위 그림 12의 결과를 보면 192-bit의 EC-AMP가 제일 빠르고 1024-bit의 AMP, 2048-bit AMP가 그 뒤를 따르고 있음을 알 수 있다. 192-bit의 EC-AMP와 1024-bit의 AMP의 수행시간 차이가 그리 크지 않으나, 192-bit의 EC-AMP와 동일한 보안성을 제공하는

2048-bit AMP의 수행시간 차이는 커짐을 알 수 있다. 따라서 높은 보안성이 필요할수록 EC-AMP가 더욱 유리할 것이다.

또한, 클라이언트와 서버간 전송되는 θ_1, θ_2의 크기는 그림 13에서 볼 수 있듯이 AMP의 경우(1024-bit 혹은 2048-bit)에 비해서 EC-AMP(320-bit 혹은 384-bit)가 $\frac{1}{3} \sim \frac{1}{5}$가량 작다. 많은 서버, 클라이언트가 프로토콜을 수행할 경우, EC-AMP가 네트워크 부하 측면에서 매우 효율적이라고 볼 수 있다.

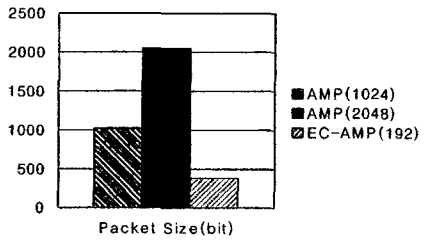


그림 13 EC-AMP와 AMP의 패킷 크기

5. 결론 및 향후 과제

본 논문에서는 이산대수문제를 사용하는 AMP 프로토콜을 향상시킨 타원곡선이산대수문제에 기반한 EC-AMP 프로토콜을 제안하였다. 기존에 하드웨어를 이용하거나 생체인식 등의 여러 가지 강력한 인증 방법들이 있으나 편의성이나 비용면에서 취약하다. 그렇지만 EC-AMP는 패스워드를 이용한 편리하고도 안전한 인증방법을 제공한다. 이를 통해서 서버/클라이언트, VPN 등의 분산환경에서의 인증, 키교환에 보안성과 편의성을 동시에 더할 수 있을 것이다.

실험을 통하여 EC-AMP가 같은 보안성을 갖는 AMP보다 대략 5배정도 빠른 것을 확인하였으며, 통신 데이터 길이를 5분의 1 이상으로 줄임으로써, 최근 높은 보안성의 필요가 강조되는 환경에 계산효율과 통신효율을 크게 높일 수 있을 것으로 예상된다.

본 논문에서 제공된 EC-AMP, EC-AMP^{n}뿐만 아니라 상호인증 부분을 생략하여 더욱 간략한 프로토콜을 설계하거나, AMP$^+$, AMP^{++} 등의 프로토콜에도 타원곡선을 적용하여 다양한 프로토콜을 설계할 수 있을 것이다.

효율적인 타원곡선 덧셈 알고리즘, F_{2^n} 유한체를 사용한 타원곡선, 안전한 타원곡선 방정식 등을 이용하여 보다 높은 수행속도와 안전성을 가지는 프로토콜에 대한 연구가 계속되어야 하겠다. 그러나 패스워드를 이용한 인증, 키교환 방법이 편의성과 보안성을 갖기 위해서는 근본적으로 사용자의 패스워드를 잘 관리하여야 한다. 패스워드에 대한 사용자의 올바른 인식과 표준화를 통해 이를 극복하여야 할 것이다.

참 고 문 헌

[1] Taekyoung Kwon, "Authentication and Key Agreement via Memorable Password," *IACR ePrint*, 2000.

[2] Alfred J. Menezes, Paul C.van Oorschot and Scott A. Van stone, *Handbook of Applied Cryptography*, pp.49-125, CRC Press, 1997.

[3] Bruce Schneier, *Applied Cryptography*, pp.513-522, Wiley, 1995.

[4] Certicom Corp., "Remarks on the security of the Elliptic curve cryptosystem," http://www.certicom.com, 2000.

[5] Julio Lopez and Ricardo Dahab, "Performance of Elliptic Curve Cryptosystems," *TR-IC-00-08*, http://www.dcc.unicamp.br/ic-main/publications-e.html, 2000.

[6] S.Bellovin and M.Merritt, "Encrypted Key Exchange: password-based protocols secure against dictionary atteks," *Proceeding of the 1992 IEEE Computer Society conference on Research in Security and Privacy*, pp.72-84, 1992.

[7] T.Wu, "Secure Remote Password protocol," *Internet Society Symposium on Network and Distributed System Security*, 1998.

[8] Victor Boyko, Philip MacKenzie and Sarvar Patel, "Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman," *Eurocrypt2000*, 2000.

[9] Phil MacKenzie, "More Efficient Password Authenticated Key Exchange," *RSA2001*, 2001.

[10] Gareth Jones, "Cryptography and Elliptic curves," *Project report of Univ. of Southampton*, 1999.

[11] Don Johnson and Alfred Menezes, "The Elliptic Curve Digital Signature Algorithm," *TR-CORR 99-34, Dept. of C&O, University of Waterloo*, 1999.

[12] SKIP, "Assigned Numbers for SKIP Protocols", http://www.skip.org, 1998.

[13] NIST, "Recommended Elliptic Curves for federal government use," 1999.

안 창 섭
2000년 2월 한양대학교 전자, 컴퓨터 공학부 학사. 2002년 2월 한양대학교 컴퓨터공학과 석사. 2002년 1월 ~ 현재 (주) 정소프트 부설 연구소 연구원. 관심분야는 암호학, 인증, 공개키기반구조

허 신
정보과학회논문지 : 시스템 및 이론
제 29 권 제 9 호 참조