분할된 패스워드 기반 인증 및 키교환 프로토콜

심현정, 류종호, 염흥열

순천향대학교 정보보호학과

Split Password-Based Authenticated Key Exchange

Shim-Hyunjung, Ryu-Jongho, Youm-Heungyoul

Department of Information Security Engineering, SoonChunHyang Univ

요 약

본 논문은 신뢰 할 수 없는 네트워크를 통해서도 사용자를 인증하거나 키를 교환하는 것에 적합한 패스워드 인증 프로토콜을 제안한다. 기본 아이디어는 패스워드 검증정보의 램덤성(randomness)을 증가시키기 위하여, 패스워드를 분할한 후 이에 대한 각 패스워드 지식을 확대(amplification)하는 구조로 설계된다. 또한 서버측 파일을 암호화하여 보관함으로써 서버 파일 타협 공격에 강인하도록 구성하였다. 더불어 검증정보 및 서버의 암호화 키가 다수의 서버들에게 분산되도록 설계된 방식을 제안한다.

I 서론

패스워드는 암기하기 쉬운 간편성, 편리성으로 인하여 주로 이용되고 있는 사용자 인증 방법이다. 개방된 네트워크에서 안전하게 서버와 접속하고자 할 때는 손쉽게 사용자의 패스워드를 이용할 수 있다. 그러나 패스워드는 인간의 짧은 기억을 통해 유지됨에 따라 추측공격에 노출 될 수 있다.

1992년 Bellovin과 Merritt가 EKE(Encrypted Key Exchange)[13]로 알려진 논문을 발표한 것을 필두로 추측공격에 강하게 저항하도록 패스워드 정보에 공개키 암호 알고리즘인 DLP(Discrete Logarithm Problem) 기반의 DH(Diffie-Hellman) 알고리즘, RSA, Elliptic Curve 알고리즘 및 램덤 오라클(random oracle)모델인 일방향 해쉬함수 등을 추가하여 안전성을 향상시켜왔다.

또한 SRP[1], SNAPI[3], AuthA[5], PAK[6]에서는 패스워드-파일 타협에 저항하고 클라이언트와 서버가 서로간에 동일하지 않는 정보를 기억하는 검증자(verifier)-기반 프로토콜들이 제안하였다. 패스워드는 사용자의 기억으로만 한정되는 엔트로피(entropy)로 제한되기는 하지만 여전히 검증자만으로부터 패스워드를 유도하는 것은 계산적으로 불가능하다. 그러나 만일 서버의 파일이 타협된다면 검증자-기반 프로토콜조차 여전히 사전공격(dictionary attack)을 허용하게 된다. 이 문제에 대한 해결책으로 가장 좋은 방법으로는 [2]에서와 같이 검증자를 임계치(threshold) 기법을 통해 분산하는 것이다[7].

본 논문에서의 프로토콜 설계목표는 Bellare와 Rogaway가 AuthA에서 제시한 "패스워드 기반 키분배 방식의 설계에서 고려해야할 요구사항"을 만족시키는 새로운 프로토콜을 설계하는 것에 있으며, 또한 다음과 같은 특성을 지니도록 한다.

- 검증자 기반 인증구조로 설계하고, 서버 파일 타협 공격에 강인하도록 구성한다. 이를 위하여 AMP[7]와 같이 서버의 파일을 암호화하여 보관한다. 이것은 [7]와 [17]에서 지적한 것처럼 안전한 저장장치에 보관된 암호화 키가 만일 저성능 장치에 의하여 통제된다면 컴퓨팅의 병목현상이 야기 될 수 있다. 그러나 서버 암호화 키의 타협이 없는 경우 검증자 보관에 있어서 가장 안전한 구조이기도 하다.
- 키동의 구조는 DH 키동의를 바탕으로 구성되며, 안전성은 DLP에 기반 하여 설계한다.
- 패스워드 검증자의 램덤성(randomness)을 증가시키기 위하여, 패스워드를 분할한 후 이에 대한 각 패스워드 지식을 확대(amplification)하는 구조로 설계된다(확대 개념은 [7]를 참조). 단 분할된 패스워드 지식들의 확대는 공격자가 더 많은 정보를 분석해야 함을 의미 할뿐 패스워드 엔트로피의 증가를 의미하는 것은 아니다.
- 별도로 검증자 및 서버의 암호화 키가 다수의 서버들에게 분산되도록 설계된 방식을 제안한

다. 이 방식에 있어서 추가적인 영지식 증명이 필요하며 상세한 사항은 3장에서 설명하기로 한다.

언급된 내용을 기술하기 위하여 우선적으로 2장에서 제안된 패스워드 기반 인증 및 키동의 프로토콜 대하여 논하고 3장에서는 분산 컴퓨팅 환경에 적합하도록 서버의 비밀정보를 다수의 서버에 분배시킨 구조를 설명한다. 4장에서는 제안된 방식의 안전성 및 효율성을 분석한 후 5장에서 결론을 내린다.

II 제안된 패스워드 기반 인증 및 키동의 프로토콜

본 절에서는 두 참여자간의 패스워드 기반 인증 및 인증된 DH 키동의 프로토콜을 제안한다. 제안된 프로토콜은 수동적 도청자 및 능동적 공격자에 대한 저항성을 지니며 또한 전방향 안전성(forward secrecy)이 제공된다. 제안된 프로토콜의 설명에 앞서 우선 몇 가지 공통 파라메터에 대하여 정의를 내린다.

- p와 q는 큰 소수이고 $q|p-1$ 이다. 원시근 g는 위수가 q인 GF(p)중의 한 원소이며 유한 순환군 $G=\langle g\rangle$를 이룬다. 이와 별도로 h는 g에 의해 생성된 군의 원소이며 유한 순환 부분군 $H=\langle h\rangle$를 이루다. 단 DLP에 의해 $h\equiv g^{\Delta} \bmod p$의 이산대수 Δ는 알려지지 않게 되며 $\gcd(\phi(q),\Delta)=1$가 만족됨을 가정한다(즉 $|G|=|H|$). p, q, g, h는 참여자들 모두에게 공개된다.
- $f:\{0,1\}^{*}\rightarrow\{0,1\}^{k}/\{0\}^{k}$는 충돌회피성 해쉬함수이며 랜덤 오라클과 같이 동작된다고 가정한다(여기에서 $k\ll\log_2 q$ 이고 $q\langle p$이다). Bellare와 Rogaway[4]에 기술된 사항을 토대로 몇 가지 형식을 다음과 같이 정의한다 :

  $h_1(x)=f(00\|x\|00),\ h_2(x)=f(01\|x\|01),$

  $h_3(x)=f(01\|x\|10),\ h_4(x)=f(10\|x\|10).$
- I_C는 클라이언트의, I_S는 서버의 ID이다.
- $a^{-1} \bmod m$는 법 m에 대한 a의 곱셈역원을 의미하고 $\in_R$은 우측의 집합에서 랜덤하게 생성함을 표기한다. 기호 $\stackrel{?}{=}$는 좌우 정보의 동일성 여부를 판단하는 기호이다.

2.1 기본 아이디어

본 논문에서 제안된 패스워드 기반 인증 및 키동의 프로토콜의 기본 아이디어는 패스워드를 분할한 후 분할된 각각에 패스워드 지식을 랜덤한 높은 엔트로피를 갖는 정보와 같이 묶여 있게 함으로써, 패스워드에 대한 추측을 낮추고자 하는 것에 있다. 또한 [7]에서와 같이 서버가 유지하는 패스워드 검증자를 암호화하여 보관함으로서 만일 서버의 파일이 공격자에 의해 타협되었다 하더라도 서버의 검증자 암호화용 키가 타협되지 않는 한 패스워드 검증자를 안전하게 보관하도록 하는 것이다.

2.2절 제안된 프로토콜의 설명에 앞서 기본 아이디어에 대하여 설명한다. 클라이언트 C는 자신의 패스워드 π를 $\pi_1\|\pi_2$로 분할한 후, 각각의 패스워드 지식에 대하여 $v_1=h_1(I_C\|I_S\|\pi_1\|'1')$와 $v_2=h_1(I_C\|I_S\|\pi_2\|'2')$를 계산한다(여기에서 '1'과 '2'는 단지 분할된 패스워드의 고유 번호일 뿐이다). 서버는 이에 대한 검증자로써 $e\equiv(g^{-v_1}v_2^{-1})^{s^{-1}} \bmod p$와 $\tau\equiv g^{v_1} \bmod p$를 계산하여 프로토콜 수행 전에 기억하고 있어야 한다. $s(\in Z_q^{*})$는 패스워드 검증자를 암호화하는 서버의 비밀키이며 누설되지 않도록 안전한 장치에 보관한다. 그림 1은 제안된 아이디어를 도시한 것이다. 그림에서 최상단 부분의 괄호는 각 참여자들의 사전지식이다.

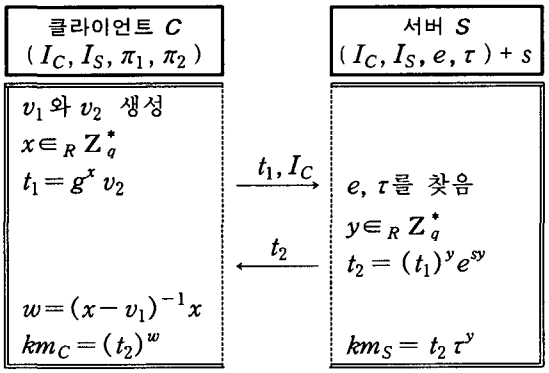

그림 1. 기본 아이디어

그림 1에서 기본 아이디어는 상호간에 정확한 정보, 즉 패스워드 π 및 이에 대한 검증자 e, τ를 사용하여 수행한다면 세션값 (km_C, km_S)은 Diffie-Hellman 키동의 기법에 바탕을 둔 g^{xy}가 된다. 그러나 이 아이디어만으로는 *explcit* 키확신

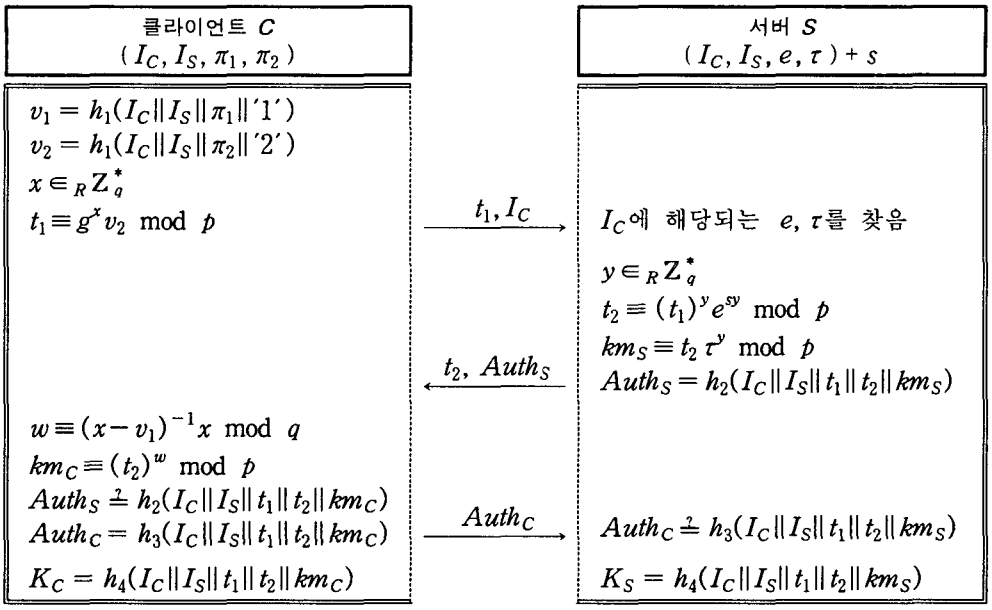

그림 2. 제안된 패스워드 기반 인증 및 키동의 프로토콜

(key confirmation)을 제공하는 *explicit* 키인증 [11]을 제공하지 못한다.

이와 같은 문제의 해결책으로 영지식 증명이 포함된 인증 프레임워크를 추가적으로 삽입하여 보안증명을 완수하면 된다. 1장에서 언급된 사항과 더불어 지금까지 제안된 프로토콜들 중에는 여러 가지 증명가능한 프로토콜들이 제안되어 왔다. 대표적인 프로토콜로서 SNAPI, EKE2[4], AuthA, PAK, AMP 등이 제안되어 왔으며 제시된 논문들은 상당히 높은 증명가능한 접근이 제시하였다. 특히 AuthA는 여러 이전 프로토콜로부터 유도된 것이지만 강한 증명가능한 방법을 제시하였다.

이에 따라 우리는 기본 아이디어에 AuthA에서 제안된 인증 프레임워크를 추가적으로 삽입하여 *explicit* 키인증을 제공하고자 한다. 다음 절에서 이에 대한 사항을 설명한다.

2.2 분할된 패스워드를 이용한 패스워드 기반 인증 및 키동의 프로토콜의 제안

제안된 프로토콜이 3-단계(3-pass)로 이루어지며 그림 2에 도시되어 있다. 기본적인 구조는 그림 1의 내용과 동일하지만, 상호인증과 상호 키확신을 위하여 클라이언트에서는 $Auth_C$의 계산이 그리고 서버에서는 $Auth_S$의 계산이 추가적으로 삽입되어 있다. $Auth_S$ 계산의 목적은 서버 S가 정확한 패스워드 검증자 (e, τ)를 알고 있고 또한 세션값 $km_S = g^{xy}$를 계산했음을 클라이언트 C에게 증명(인증 및 키확신에 대한 증명)하기 위한 것이다. 마찬가지로 $Auth_C$는 클라이언트 C가 정확한 패스워드 π를 알고 있고 또한 세션값 $km_C = g^{xy}$에 대하여 계산했음을 서버에게 증명하기 위한 것이다.

프로토콜에서 서버 S는 클라이언트 C에게서 t_1이 e^{-s} 형태로 전달되어 올 경우 프로토콜 세션을 강제적으로 종료하고 $t_1 = e^{-s}$ 이외의 값으로 재시도 하기를 요청해야 한다. 이 같은 경우 서버의 km_S는 τ^y가 되고 클라이언트의 km_C는 1 이 되기 때문이다. 즉 $km_C \neq km_S$가 된다. 이 조건은 AMP와 PAK에도 적용할 수 있다.

수동적 도청자는 프로토콜에서 어떠한 정보도 얻을 수 없다. 이에 대한 사항은 4장 보안분석에 기술할 것이다. 만일 능동적 공격자가 서버에서 s를 제외한 (e, τ)를 타협시킨 경우, 이 정보에서 패스워드 π를 유도하는 것은 DLP(s를 알아내는 것)와 동일하게 된다. 그러나 만일 강한 능동적 공격자가 서버의 s까지 타협시킨다면 사전공격을 통해 π가 노출되게 된다.

$Auth_C$와 $Auth_S$의 계산 및 검증은 지금까지 알려진 공격에 대한 증명가능한 보안성을 제공한다. 공격에 대한 안전성은 4장에서 설명할 것이다.

최종적으로 두 참여자가 상호인증과 상호 키확신에 검사를 통과하였다면 상호 동의된 세션키 (K_C, K_S)를 생성한다.

2.3 다수 분할된 패스워드를 이용한 패스워드 기반 인증 및 키동의 프로토콜의 제안

클라이언트 C는 패스워드를 n으로 분할된 형태로 확장할 수 있다. 이 방법은 패스워드 정보의 램덤성을 더욱 증가시키기 위한 것으로써, 패스워드를 분할한 후 이에 대한 각 분할 패스워드 지식을 확대하는 구조로 설계된다. 우선 자신의 패스워드 π를 $\pi_1 \| \cdots \| \pi_n$로 분할한 후, 각 패스워드 지식에 대하여 $v_i = h_1(I_C \| I_S \| \pi_i \| i)$ $(i=1, \cdots, n)$를 계산한다. 서버에서는 이에 대한 검증자로써 $e = (g^{-\prod_{i=1}^{a} v_i} \prod_{j=a+1}^{n} v_j^{-1})^{s^{-1}}$ 및 $\tau = g^{\prod_{i=1}^{a} v_i}$를 계산하여 프로토콜 수행 전에 기억하고 있어야 한다. 여기에서 $a \in Z_n^*$ $(n \geq 2)$이다.

① 클라이언트는 $v_1, \cdots, v_n$를 계산하고 $x \in_R Z_q^*$를 생성한 후, $t_1 = g^x (\prod_{j=a+1}^{n} v_j)$와 I_C를 서버에게 전달한다.

② 서버는 그림 2와 동일하게 동작하여 y, t_2, km_S, $Auth_S$를 생성한 후, t_2, $Auth_S$를 클라이언트에게 전달한다.

③ 클라이언트는 $w = (x - \prod_{i=1}^{a} v_i)^{-1} x$를 계산하고 그림 2와 같이 km_C, $Auth_C$를 생성한 후 서버에게 $Auth_C$를 전달한다.

④ 최종적으로 두 참여자 모두가 상호인증과 상호 키확신 검사를 통과하였다면 동의된 세션키 (K_C, K_S)를 생성한다.

III 비밀분산를 이용한 패스워드 기반 인증 및 키설정

앞서 언급했던 것과 같이 서버의 파일 타협에 대한 가장 좋은 해결책으로 [2]에서와 같이 검증자를 임계치 기법을 통해 분산하는 것이다[7]. 이와 관련하여 그림 2에 제안된 프로토콜에 비밀분산 기법을 적용하여 제안한다. 3.1절에서 비밀분산 기법에 관련된 몇 가지 사항을 정의하고 3.2절에서는 비밀분산 기법에 필요한 영지식 증명(ZKP: Zero-Knowledge Proof)을 기술한다. 3.3절과 3.4절에서는 분산 서버 구조에서의 패스워드 등록 및 분산된 서버와의 인증 및 키동의에 대하여 제안한다.

3.1 비밀분산 기법

멀티파티 프로토콜에서는 각 참가자의 비밀정보를 분할하여 다른 참가자에게 분배하고, 필요에 따라 원래의 정보를 복원할 필요가 있다. 이와 같은 목적에 사용되는 도구로 비밀분산 방식이다. 상세 사항[14]은 지면관계상 생략하기로 한다. 단 몇 가지 용어에 대한 정의를 내린다[14].

- Interpolation 및 Exp-interpolation
  : Lagrange 보간법 및 멱승-보간법
- Exp-VSS : Feldman's Verifiable Secret Sharing
- Joint-Exp-RSS : Joint Computationally Secure Random Secret Sharing
- Joint-Uncond-Secure-RSS : Joint Unconditionally Secure Random Secret Sharing
- Joint-Uncond-Secure-ZSS : Joint Unconditionally Secure Zero Secret Sharing

3.2 영지식 증명(ZKP)

ZKP은 특정 비밀정보를 노출시키지 않으면서도 알고 있다는 사실 그 자체를 증명하는 방법이다. 본 논문에서는 클라이언트의 비밀정보(지수값)를 서버에게 직접적으로 제공하지 않으면서도 클라이언트가 그 비밀 정보를 실제로 알고 있다는 사실만을 서버에게 확신 시켜 주는 프로토콜로 활용한다. ZKP 방법에서 자신의 비밀정보를 증명시켜주는 자를 증명자(prover)라 하고, 정확한 값인지를 검증하는 자를 검토자(verifier)라 한다. 본 절에서는 비밀분산에 사용될 2가지 비대화형(non-interactive) ZKP를 설정한다.

• 영지식 증명 (1)

[8]에서는 밑수(base)가 다른 두 값을 공개(commitment)하고 두 값의 이산대수가 동일함을 증명하는 대화형 ZKP를 제안하였다. 이를 응용하여 세 가지의 정보를 공개하고 이에 따른 이산대수가 동일함을 증명하는 것으로 확장하는 것이 가능하다. 더불어 이를 비대화형 ZKP로 바꿀 수 있다. 이에 대한 사항이 그림 3에 도시되어 있다. 우선 증명자는 $A = g^a, B = t^a, C = \tau^a$ (여기에서 $a \in Z_q$ 그리고 $A, B, C \in Z_p^*$)를 위임한 후, 밑수가 다른 세 값의 이산대수가 동일함을 비대화형

ZKP로 증명한다. 단 $s_1 = t_1 - \varphi a$ 와 $M_1 \stackrel{?}{=} g^{s_1} A^{\varphi}$ 같은 형태를 적용하여도 무방하다.

$t, \tau \in Z_p^*$는 원시근 g에 생성된 부분군에 속하며, $\log_g t$와 $\log_g \tau$를 계산하는 것은 DLP에 의해 어렵다고 가정한다. 결과적으로 증명자는 (A, B, C)의 동일한 이산대수 a를 공개하지 않고, 이 값들의 이산대수의 동일함을 증명한다.

증명자	A, B, C →	검토자
$t_1 \in_R Z_q$ $M_1 = g^{t_1}$ $M_2 = t^{t_1}$ $M_3 = \tau^{t_1}$ $\varphi = f(M_1 \| M_2 \| M_3)$ $s_1 = t_1 + \varphi a$	$M_1, M_2, M_3, \varphi, s_1$ →	
		$\varphi \stackrel{?}{=} f(M_1 \| M_2 \| M_3)$ $g^{s_1} \stackrel{?}{=} M_1 A^{\varphi}$ $t^{s_1} \stackrel{?}{=} M_2 B^{\varphi}$ $\tau^{s_1} \stackrel{?}{=} M_2 C^{\varphi}$

그림 3. 비대화형 영지식 증명 (1)

• 영지식 증명 (2)

증명자	A, B, C →	검토자
$t_1, t_2, t_3 \in_R Z_q$ $M_1 = g^{t_1} h^{t_2}$ $M_2 = g^{t_3}$ $M_3 = B^{t_1} h^{t_2}$ $\varphi = f(M_1 \| M_2 \| M_3)$ $s_1 = t_1 - \varphi a$ $s_2 = t_2 - \varphi b$ $s_3 = t_3 - \varphi c$	$M_1, M_2, M_3, \varphi, s_1, s_2, s_3$ →	
		$\varphi \stackrel{?}{=} f(M_1 \| M_2 \| M_3)$ $M_1 \stackrel{?}{=} g^{s_1} h^{s_2} A^{\varphi}$ $M_2 \stackrel{?}{=} g^{s_3} B^{\varphi}$ $M_3 \stackrel{?}{=} B^{s_1} h^{s_2} C^{\varphi}$

그림 4. 비대화형 영지식 증명 (2)

[15]에서는 위임값 A와 B의 이산대수를 공개하지 않고, 위임값 C가 A와 B에 이산대수의 곱으로 이루어졌음을 증명하였다. 또한 [10]에서는 이에 대한 비대화형 ZKP를 제안하였다. 이를 본 논문에 적합하도록 수정한 것이 그림 4에 도시되어 있다.

우선 증명자는 $A = g^a h^b$, $B = g^c$, $C = g^{ac} h^b$ $(= B^a h^b)$을 위임한 후, A와 B의 이산대수가 C에게 포함되어 있음을 검토자에게 비대화형 ZKP로 증명한다. 여기에서 $h \in Z_p^*$는 원시근 g에 생성된 부분군에 속하며, $\log_g h$를 계산하는 것은 DLP에 의해 어렵다고 가정한다.

3.3 분산 서버 구조에서의 패스워드 등록 프로토콜 : 그림 2의 제안된 프로토콜의 응용

프로토콜의 목적은 분산 구조를 갖는 서버군에서의 특정 서버 S_i와 클라이언트간의 패스워드 기반 인증 및 키동의 프로토콜을 수행하는 것에 있다. 서버군의 특정 비밀값들이 비밀분산 기법에 의해 나누어져 있기 때문에 서버 S_i는 다른 서버들의 도움을 받아야만 인증 및 키동의 프로토콜을 수행할 수 있다. 절차상에 있어 우선 (1) 서버의 비밀키 s를 생성하고, (2) 클라이언트가 v_1를 서버들에게 분산시키며, (3) 서버들은 서로 협력하여 검증자 $e = (g^{-v_1} v_2^{-1})^{s^{-1}}$를 생성한다.

(1) 서버의 비밀키 s 생성 : 각 서버들은 클라이언트의 패스워드 검증자를 안전하게 보관하기 위한 비밀값 s를 서로 협력하여 생성한다.

• Joint-Uncond-Secure-RSS[14]를 다항식 $t-1$차(degree)로 이루어지도록 한 후, Z_q^*에서 균등하게 분배된 비밀키 s를 랜덤하게 생성한다. 이 과정에는 정보이론적 보안을 제공하기 위한 보안 파라메터 ε도 역시 동시에 생성된다. 즉 $(s_1, \cdots, s_n) \xleftarrow{(t,n)} s$ 와 $(\varepsilon_1, \cdots, \varepsilon_n) \xleftarrow{(t,n)} \varepsilon$. 더불어 정보들을 검증하기 위한 검증정보 (verification information) A_a가 t개 만들어져 공개된다. 즉 $A_0, \cdots, A_{t-1}$.

(2) 검증자 v_1의 생성 및 분배 : 이 프로토콜을 수행하기 위하여 우선적으로 클라이언트가 각 서버들에게 v_1에 대한 거짓 정보를 제공하지 않는다고 가정한다.

• 클라이언트는 $v_1 = h_1(I_C \| I_S \| \pi_1 \| '1')$를 계산하고, Exp-VSS[14]를 이용하여 $t-1$차의 다항

식으로 Z_q에서 유일하게 분배되도록 한다. 즉 $(v_{1,1}, \cdots, v_{1,n}) \xleftarrow{(t,n)} v_1$. 또한 검증정보 B_α가 t개 만들어져 서버들에게 공개된다(단 $B_0 = g^{v_1} = \tau$). 여기에서 $t-1$차는 서버가 사전에 미리 정해 놓은 값이다. 이후 클라이언트는 각 서버의 공개 번호(index, $i=1, \cdots, n$)에 맞추어 각 서버 S_i에게 분배된 값 $v_{1,i}$를 안전하게 전달한다. 각 서버는 다음 식을 통해 검사한 후 이를 받아들인다.

$$g^{v_{1,i}} \stackrel{?}{=} \prod_{\alpha=0}^{t-1} B_\alpha^{i^\alpha \pmod q} \pmod p$$

(3) 검증자 $e = (g^{-v_1} v_2^{-1})^{s^{-1}} \bmod p$의 생성: 값 e를 생성하는 과정은 [14]의 공개키 생성 과정을 응용하여 만들 수 있다. 수식의 간략화를 위하여 $\sigma = (g^{-v_1} v_2^{-1})$이라고 표기한다. 이 프로토콜을 수행하기 위하여 우선적으로 클라이언트가 각 서버들에게 거짓 정보를 제공하지 않는다고 가정한다.

① 클라이언트는 $a \in_R Z_q^*$를 생성한 후, 마치 딜러(dealer)처럼 Z_q상의 $t-1$차의 다항식으로 이루어지는 Exp-VSS[14]를 이용하여, a를 각 서버들에게 분배한다.

즉 $(a_1, \cdots, a_n) \xleftarrow{(t,n)} a$. 또한 분배된 정보들을 검증하기 위한 검증정보 C_α가 t개 만들어져 서버들에게 공개된다.

- 클라이언트는 각 서버의 공개 번호(index, $i=1, \cdots, n$)에 맞추어 각 서버 S_i에게 분배된 값 a_i를 안전하게 전달한다. 각 서버는 식 $g^{a_i} \stackrel{?}{=} \prod_{\alpha=0}^{t-1} C_\alpha^{i^\alpha \pmod q} \pmod p$와 같이 검사한 후 이를 받아들인다. 이와 별도로 클라이언트는 σ^a를 서버들에게 안전하게 전달한다. 서버들은 값 σ를 알지 못하며 σ^a가 노출된다 하더라도, 패스워드에 대한 어떠한 정보도 누설되지 않는다.

② 각 서버들은 *Joint-Uncond-Secure-ZSS[14]*를 이용하여, 다항식 상수항(constant term)이 "0, zero"이고 $2t-2$차로 이루어지도록 생성한 후, Z_q에서 균등하게 분배시킨다. 더불어 검증정보 D_β가 $2t-1$개 만들어져 공개된다(즉 $D_0, \cdots, D_{2t-2}$). 단 $D_0 = 1$로 고정된다. 생성된 분배 값들은 다음과 같이 표현된다 :

$\{b_i\}_{i \in \{1, \cdots, n\}}$ 와 $\{\eta_i\}_{i \in \{1, \cdots, n\}}$.

③ 서버 S_j $(j=1, \cdots, n)$들은 $\lambda_j \equiv s_j a_j + b_j$ 와 $\lambda_j' \equiv \varepsilon_j + \eta_j \pmod q$를 계산하고, 비대화형 ZKP를 수행하기 위한 정보들 $Y_{j,1} = g^{s_j} h^{\varepsilon_j}$, $Y_{j,2} = g^{a_j}$, $Y_{j,3} = g^{s_j a_j} h^{\varepsilon_j}$를 계산하여 서버 S_i $(1 \le i \le n)$에게 $(\lambda_j, \lambda_j', Y_{j,1}, Y_{j,2}, Y_{j,3})$를 공개(broadcasting)한다. 여기에서 서버 S_i은 위 프로토콜을 통하여 이미 검증정보들 A_α, C_α, D_β를 알고 있다. 단 $\alpha = \{0, \cdots, t-1\}$ 이고 $\beta = \{0, \cdots, 2t-2\}$이다.

- 서버 S_i는 λ_j와 λ_j'의 정확성을 검사하기 위하여 $g^{\lambda_j} h^{\lambda_j'}$를 구성한 후, 이미 자신이 알고 있는 검증 정보들 $A_\alpha, C_\alpha, D_\beta$를 적절히 조합하여 이와 비교해야 한다.

  $g^{\lambda_j} h^{\lambda_j'}$를 $g^{\lambda_j} h^{\lambda_j'} = (g^{s_j a_j} h^{\varepsilon_j}) \times (g^{b_j} h^{\eta_j})$로 전개하자. 전개된 식에서 서버 S_i는 $(g^{b_j} h^{\eta_j})$ 부분을 사전에 알고 있는 검증 정보 D_β들을 $(g^{b_j} h^{\eta_j}) \stackrel{?}{=} \prod_{\beta=0}^{2t-2} D_\beta^{j^\beta}$ 와 같이 구성하여 검증할 수 있다.

- 그러나 구성된 식의 $(g^{s_j a_j} h^{\varepsilon_j})$부분에 있어서, 서버 S_i는 서버 S_j들의 비밀 분배 값들 $(s_j, a_j, \varepsilon_j)$를 모를 뿐만 아니라 서버 S_i가 지닌 검증정보들 $A_\alpha, C_\alpha, D_\beta$를 통해서도 검증이 불가능하다. 따라서 서버 S_j들은 $(g^{s_j a_j} h^{\varepsilon_j})$부분을 별도로 공개하고, 자신의 비밀 분배 값의 곱형태인 $s_j a_j$와 ε_j를 서버 S_i에게 알려주지 않으면서도 정확히 이 값의 승으로 이루어졌음에 대한 증명을 수행한다. 이와 같은 증명은 비대화형 ZKP를 통해 이루어진다. ZKP를 수행하기 위하여 서버 S_j들은 $(Y_{j,1}, Y_{j,2}, Y_{j,3})$를 계산한 후 이를 서버 S_i에게 공개한다. 결과적으로 $Y_{j,1}$과 $Y_{j,2}$의 이산대수 값을 공개하지 않고 $Y_{j,3}$가 이들의 곱으로 이루어졌음을 증명하는 것이다.

④ 서버 S_i는 $(\lambda_j, \lambda_j', Y_{j,1}, Y_{j,2}, Y_{j,3})$를 $(2t-1)$개 이상 수신한 후 다음 사항을 검증한다.

- $Y_{j,1} \stackrel{?}{=} \prod_{\alpha=0}^{t-1} A_\alpha^{j^\alpha}$ 및 $Y_{j,2} \stackrel{?}{=} \prod_{\alpha=0}^{t-1} C_\alpha^{j^\alpha}$
- $Y_{j,3} \stackrel{?}{=} g^{s_j a_j} h^{\varepsilon_j}$ (영지식 증명 (2) 수행)

• $g^{\lambda_i} h^{\lambda_i'} \stackrel{?}{=} Y_{j,3} \prod_{\beta=0}^{2t-2} D_\beta^{j^\beta}$

• 검증이 통과된다면 서버 S_i은 다음을 계산한다.

. $\mu \triangleq$ Interpolate $(\lambda_1, \cdots, \lambda_n)$ (mod q)
[$= sa$]. μ에 해당하는 다항식은 $(2t-2)$차.

. μ^{-1} (mod q) [$= s^{-1}a^{-1}$]. 단 $s, a \in \mathbf{Z}_q^*$.

. $e \triangleq (\sigma^a)^{\mu^{-1}}$ (mod p) [$= \sigma^{s^{-1}}$]
: 서버 S_i는 e를 얻는다.

3.4 분산된 서버 환경에서의 인증 및 키동의 프로토콜 : 그림 2의 제안된 프로토콜의 응용

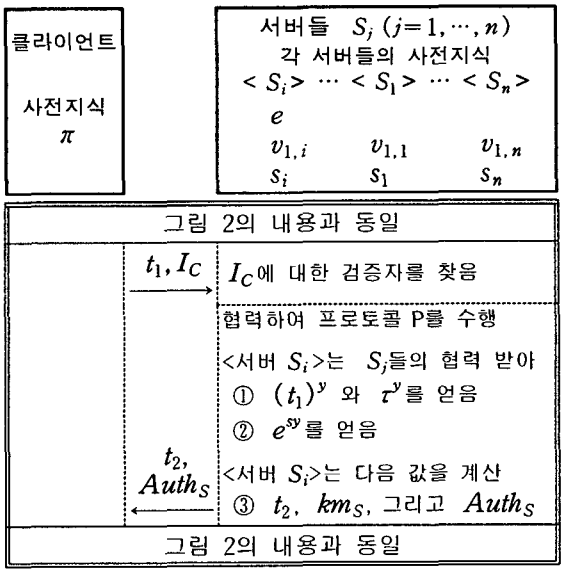

그림 5. 분산환경에서의 인증 및 키동의

다중 서버를 통한 인증의 목적은 특정 서버 S_i가 패스워드를 이용하여 클라이언트를 인증하는 것에 있다[9]. 이때 패스워드 검증자 및 서버의 비밀정보 s가 각 서버들에게 분배되어 있다. 따라서 서버들간에 서로 협력하여 인증과정을 수행하며 그림 5에 도시되어 있다.

그림 5에서의 프로토콜 P는 다음과 같이 처리된다.

① <서버 S_i>는 서버들 S_j $(j=1,\cdots,n)$의 협력을 받아 $(t_1)^y$ 와 τ^y를 계산한다. 이 과정을 정리한 것이 아래에 도시되어 있다.

S_j $(j=1,\cdots,n)$ 그리고 각 서버들의 사전지식
<서버 S_i>···<서버 S_1>···<서버 S_n>
t_1 t_1 t_1
τ τ τ

<서버 S_i>
Joint-Exp-RSS + 영지식 증명 (1) + Exp-Interpolation

	$(y_i, \cdots, y_1, \cdots, y_n)$
	$(g^{y_i}, \cdots, g^{y_1}, \cdots, g^{y_n})$
$t_1^y \xleftarrow{(t,n)}$	$(t_1^{y_i}, \cdots, t_1^{y_1}, \cdots, t_1^{y_n})$
$\tau^y \xleftarrow{(t,n)}$	$(\tau^{y_i}, \cdots, \tau^{y_1}, \cdots, \tau^{y_n})$

• 서버들은 $(t-1)$차의 다항식으로 이루어지는 Joint-Exp-RSS[14]를 이용하여 $\mathbf{Z}_q$에서 균등하게 분배된 난수 값 y를 생성한다. 즉 $\{y_1, \cdots, y_n\} \xleftarrow{(t,n)} y$ 이고 이에 대한 검증정보 E_α. 단 $\alpha = \{0, \cdots, t-1\}$.

• 각 서버들 S_j는 $(g^{y_j}, t_1^{y_j}, \tau^{y_j})$를 계산한 후 이를 <서버 S_i>에게 전달한다.

• <서버 S_i>는 $g^{y_i} \stackrel{?}{=} \prod_{\alpha=0}^{t-1} (E_\alpha)^{i^\alpha}$로 검사한 후, 영지식 증명 (1)를 이용하여 g^{y_j}와 $(t_1^{y_j}, \tau^{y_j})$의 각 이산대수 동일여부를 검사한다.

• <서버 S_i>는 $t_1^y \triangleq$ Exp-Interpolate $(t_1^{y_1}, \cdots, t_1^{y_n})$를 통하여 $(t_1)^y$를 얻는다.

• <서버 S_i>는 $\tau^y \triangleq$ Exp-Interpolate $(\tau^{y_1}, \cdots, \tau^{y_n})$를 통하여 τ^y를 얻는다.

• 보안분석 : y를 <서버 S_i>에게 노출하지 않으면서도, 값 t_1^y 와 τ^y를 계산할 수 있도록 한다.

② <서버 S_i>는 서버들 S_j $(j=1,\cdots,n)$의 협력을 받아 e^{sy}를 계산한다. 이 과정을 정리한 것이 아래에 도시되어 있다.

S_j $(j=1,\cdots,n)$ 그리고 각 서버들의 사전지식
<서버 S_i>···<서버 S_1>···<서버 S_n>
e
s_i s_1 s_n

<서버 S_i>
Joint-Uncond-Secure-ZSS
+ 영지식 증명 (2) + Interpolation

	$(y_i, \cdots, y_1, \cdots, y_n)$
$sy \xleftarrow{(2t-1,n)}$	$(\delta_i, \cdots, \delta_1, \cdots, \delta_n)$

• 서버들은 $(2t-2)$차의 다항식으로 이루어지는

Joint-Uncond-Secure-ZSS[14]를 이용하여 Z_q에서 균등하게 분배시킨다. 즉 $\{r_j\}_{j\in\{1,\cdots,n\}}$와 $\{r_j'\}_{j\in\{1,\cdots,n\}}$ 이고 검증정보 F_β.

단 $\beta=\{0,\cdots,2t-2\}$.

- 각 서버 S_j는 $\delta_j\equiv s_j y_j+r_j$ 와 $\delta_j'\equiv\varepsilon_j+r_j'$ (mod q)와 같이 계산하고(즉, 두 비밀의 곱[14]), 영지식 증명 (2)를 수행하기 위한 정보들 $V_{j,1}=g^{s_j}h^{\varepsilon_j}$, $V_{j,2}=g^{y_j}$, $V_{j,3}=g^{s_j y_j}h^{\varepsilon_j}$를 계산한 후 <서버 S_i>에게 $(\delta_j,\delta_j',V_{j,1},V_{j,2},V_{j,3})$를 전달한다.
- <서버 S_i>는 $2t-1$명 이상의 서버들로부터 각각 $(\delta_j,\delta_j',V_{j,1},V_{j,2},V_{j,3})$를 수신한 후 다음 사항을 검증한다.
  . $V_{j,1}\stackrel{?}{=}\prod_{\alpha=0}^{t-1}A_\alpha^{j^\alpha}$ 및 $V_{j,2}\stackrel{?}{=}\prod_{\alpha=0}^{t-1}E_\alpha^{j^\alpha}$
  . $V_{j,3}\stackrel{?}{=}g^{s_j y_j}h^{\varepsilon_j}$ (영지식 증명 (2)를 이용)
  . $g^{\delta_j}h^{\delta_j'}\stackrel{?}{=}V_{j,3}\prod_{\beta=0}^{2t-2}F_\beta^{j^\beta}$
  . 검사가 통과된다면 서버 S_i은 다음을 계산
    - $sy\triangleq$ Interpolate $(\delta_1,\cdots,\delta_n)$ (mod q).
      sy에 해당하는 다항식은 $(2t-2)$차.
    - e^{sy} (mod p)
- 보안분석 : $\Pr[y|g^y]\simeq\Pr[DLP]$이기 때문에 y를 계산할 수 없다. 따라서 <서버 S_i>는 sy로부터 비밀값 s를 계산할 수 없다.

③ <서버 S_i>는 다음 사항을 계산한다.

- $t_2\equiv t_1^y e^{sy}$ (mod p)
- $km_S\equiv t_2\,\tau^y$ (mod p)
- $Auth_S=h_2(I_C\|I_S\|t_1\|t_2\|km_S)$

IV 제안된 방식의 보안분석 및 효율성 분석

① 패스워드를 분할하여 사용한다. 패스워드의 분할은 패스워드 검증자의 랜덤성(randomness)을 증가시킨다. 왜냐하면 패스워드 분할하여 인증 프로토콜을 수행한다면, 그에 따른 검증자 역시 증가하게 된다. 이것은 공격자가 더 많은 정보를 분석해야 함을 의미한다. 그러나 패스워드에 대한 분할도가 높아지게 된다면, 그에 따른 클라이언트 및 서버의 계산량 역시 증가하게 된다.

② 제안된 프로토콜이 수행되는 동안 세션값에 대한 어떠한 지식도 수동적 도청자에게 노출되지 않는다.

수동적 도청(passive eavesdropping) : 수동적 도청에 대한 보안이 DLP의 어려움에 바탕을 두고 있음을 증명한다. 수동적 도청자는 참여자간에 교환되는 메시지를 도청할 수 있고, 그들 사이에서 공유된 세션키를 구하려고 시도한다. 그러나 수동적 도청자는 임의의 메시지를 바꾸거나, 삭제하거나 삽입하는 것은 불가능하다.

프로토콜에서 $km_C=km_S=g^{xy}$ (mod p)이며 편의상 km이라 정의하자.

제안된 방식의 세션값 교환 과정에서, 도청자가 프로토콜의 공통 파라메터 (p,q,g) 그리고 서버과 클라이언트 양자간의 교환정보 $(t_1=g^x v_2,\ t_2=g^{y(x-v_1)})$를 알고 있다 하더라도, 도청자가 세션값 g^{xy}를 알아내는 것은 적어도 DLP를 풀어내는 것만큼 어렵다. 이와 같은 결론을 증명하기 위하여 우선 다음과 같은 알고리즘을 정의한다[16,17,18].

- $Adv_A(\,)$: 다항식 시간 알고리즘 A에 공통 파라메터 그리고 프로토콜 수행 중에 노출되는 교환정보를 입력 값으로 하여 세션값 km을 계산하는 알고리즘이다.
- $Adv_{DLP}(\,)$: DLP를 계산하는 알고리즘에 공통 파라메터 그리고 $a\in Z_p^*$를 입력 값으로 하여 $\log_g a\in Z_q^*$를 구하는 것이다.
  즉 $Adv_{DLP}(p,q,g,a)=\log_g a$ (mod q).
- $Adv_{DHP}(\,)$: DHP(Diffie-Hellman Problem)를 계산하는 알고리즘에 공통 파라메터 그리고 $a,b\in Z_p^*$를 입력하여 $a^{\log_g b\,(\mathrm{mod}\,q)}\in Z_p^*$ 구하는 것이다.
  즉 $Adv_{DHP}(p,q,g,a,b)=a^{\log_g b}$ (mod p).
- $Adv_{DHDP}(\,)$: DHDP(Diffie-Hellman Decision Problem)[18]는 $g^{a'},g^{b'},g^{c'}\in Z_p^*$가 입력으로 주어지고 $c'\equiv a'b'$인지를 결정하는 문제이다. 만일 $c'\not\equiv a'b'$이라면 $c'\equiv a'b'+z$으로 간주할 수 있고, 결과적으로 균일한 확률 분포 갖는 G_p상에서의 g^z의 분포는 통계적으로 구별불가능(indistinguishable)하게 된다. 더불어 입력값들 $(g^{a'},g^{b'},g^{c'})$의 분포 역시 G_p^3상의 균일한 확률 분포에 통계적으로 구별 불가능하게

된다.

- DLP, DHP, 그리고 DHDP는 모두 계산적으로 등가를 이룬다. 즉 DLP의 해결이 무시할 만한 확률을 갖는다면 DHP 및 DHDP 역시 무시할 만한 확률을 지닌다[16].

주어진 알고리즘을 통해 다음과 같은 절차로 수행한다.

2장에서 해쉬함수 $f:\{0,1\}^* \rightarrow \{0,1\}^k/\{0\}^k$ ($k \ll \log_2 q$ 이고 $q<p$)로 정의했기 때문에, v_1은 $v_1:\{0,1\}^l \in Z_q^*$로 v_2는 $v_2:\{0,1\}^l \in Z_p^*$라 가정할 수 있다. 따라서 t_1은 $g^x v_2 = g^{x+z_1}$으로 또한 t_2는 $g^{y(x-v_1)} = g^{yz_2}$으로 놓을 수 있다. 여기에서 $z_1, z_2 \in Z_q^*$이다.

- 정의에 따라 $Adv_A(p,q,g,g^{x+z_1},g^{yz_2}) = g^{xy}$가 다항식 시간 안에 계산될 수 있다.
- $a' = x+z_1$, $b' = yz_2$, $c' = a'b'z_2^{-1} - yz_1$이라 정의할 경우, 위 $Adv_A()$ 알고리즘이 성립한다면 $Adv_{DHDP}(p,q,g,g^{a'},g^{b'},g^{c'})$="true" 가 된다. 또한 DHDP를 결정하는 알고리즘이 성립한다면, $\{x, y, z_1, z_2 \leftarrow Z_q^* : (g^{a'}, g^{b'}, g^{c'})\}$을 구별(즉 계산) 할 수 있음을 의미한다.
- 이 경우는 $Adv_{DLP}(p,q,g,g^{x+z_1}) = x+z_1$을 출력하고 $Adv_{DLP}(p,q,g,g^{yz_2}) = yz_2$를 출력해야 가능해야만 한다. 따라서 도청자가 위에서 정의한 알고리즘을 사용하고 주어진 절차에 따라 수행한다면 세션값 km을 구할 수 있다.

결론적으로 만일 알고리즘 $Adv_A()$가 존재한다면 $Adv_{DHDP}()$가 존재할 수 있고, $Adv_{DHDP}()$가 존재할 수 있다면 $Adv_{DLP}()$가 존재할 수 있다. 따라서 우리의 제안된 프로토콜에서 세션값 km를 구하는 것은 $Adv_{DLP}()$를 계산할 확률과 비슷하며 DLP를 해결하는 것만큼 가능하게 된다.

③ 제안된 방식은 Denning-Sacco 공격이 불가능하다[6]. 세션키가 공개되어도 패스워드는 노출되지 않을 뿐 아니라 사전공격에도 안전하다. 또한 이전 세션키를 이용하여 이후 제안된 프로토콜에 속하는 참여자로 가장할 수 없다.

이 공격은 이전 세션키를 안다고 할 때 패스워드를 알아내는 공격 방법이다. 패스워드가 아닌 t_1, t_2, 그리고 세션값 (km)을 얻을 수 있다 가정하여도, 이와 같은 정보에서 공격자가 패스워드 π 및 검증자 (e,τ)를 얻는데 아무런 도움이 되지 못한다. 더욱이 e는 서버의 비밀키 s로 암호화 $e=(g^{-v_1}v_2^{-1})^{s^{-1}}$로 되어 있어 더욱 어렵다.

패스워드를 얻기 위해서는 $\{t_1, t_2, km\}$으로부터 $\{(x, y, v_1 \leftarrow Z_q^*), (v_2 \leftarrow Z_p^*)\}$을 구별할 수 있어야 하고 이 문제는 DLP를 해결 할 수 있어야 한다. 따라서 Denning-Sacco 공격을 수행하는 알고리즘을 $Adv_{DS}()$라 할 경우에, 제안된 프로토콜에서 Denning-Sacco 공격을 수행하는 공격자의 성공 확률은 다음과 같이 이루어지게 된다.

$$\Pr[Adv_{DS}(g,p,q,t_1,t_2,km)] \simeq \Pr[Adv_{DLP}()]$$

[7,17]에서 언급한 것과 같이 이 값을 무시할만한 값이다.

④ 제안된 프로트콜은 전방향 안전성이 제공된다.

롱텀(long-term) 비밀값(패스워드)의 타협이 롱텀 비밀값 분실 이전에 생성된 세션값 (km) 분실을 의미하지 않는다면, 프로토콜은 전방향 안전성를 만족한다라고 정의한다. 공격자에게 패스워드가 주어졌다고 가정했을지라도 공격자는 v_1와 v_2만을 구할 수 있을 뿐이다. v_1와 v_2에서 세션값을 구하는 것은 $Adv_A(p,q,g,t_1,t_2) = g^{xy}$가 존재함을 의미하며, 이것은 $Adv_{DLP}(p,q,g,t_1)$와 $Adv_{DLP}(p,q,g,t_2)$가 존재함과 동일한 의미이다. 따라서 다항식 시간 알고리즘 A을 해결하는 것만큼 전방향 안전성가 훼손되게 된다.

⑤ 제안된 프로토콜은 오프라인 사전추측[11] 공격에 대한 저항성을 지닌다.

프로토콜에서 속하는 모든 패스워드에 대하여 오프라인 사전추측 공격이 불가능해야 하고, 프로토콜의 수행 중에 노출되는 정보들을 통해서도 오프라인 사전추측 공격이 불가능해야 함을 의미한다. 이 문제는 클라이언트와 서버간에 서로 주고받는 정보 t_1과 t_2에 대한 DLP를 해결해야만 패스워드에 대한 사전공격이 가능하게 된다. 따라서 $\Pr[Adv_{DLP}()]$와 비슷한 확률을 갖게 된다.

⑥ 제안된 방식은 능동적 중간자 공격(positive man in the middle attack)[11] 및 재생공격(replay attack)[11]에 강인하다.

능동적 중간자 공격은 공격자가 양쪽 개체를 합법적으로 가장하거나 혹은 클라이언트와 서버 사이에서 존재하여 두 참여자의 메시지를 가로챈 다음, 공격자와 클라이언트, 공격자와 서버 사이에 각각의 별도의 세션값을 만들어내는 공격이다. 이 공격은 가장공격(impersonation attack)과 동일하다. 공격자는 프로토콜 내의 모든 대화내용을 이용하더라도 패스워드를 모른다면 제안된 프로토콜에서의 $Auth_S \stackrel{?}{=} h_2(\ )$ 및 $Auth_C \stackrel{?}{=} h_3(\ )$ 검사를 통과하지 못한다.

재생공격은 공격자가 클라이언트의 메시지(즉 t_1)를 서버에게 재전송 하여 이미 정상적인 클라이언트에 의해 생성된 이전키(old session key)를 다시 생성하기 위함이다. 그러나 모든 통신 메시지들은 매 세션마다 균일한 확률 분포에서 랜덤하게 생성되어짐을 가정하기 때문에 이 공격에 대한 공격자의 성공 확률은 무시할 수 있다. 즉 클라이언트가 각 인증 프로토콜마다 $x \in_R Z_q^*$를 생성하고 그 사건의 확률이 모두 균일한 확률 분포 $1/\phi(q)$를 갖는다면, 공격자의 성공확률은 대략 $\Pr[Adv_{RA}(\ )] \leq 1/\phi(q)$이 된다.

⑦ 효율성 분석

효율성은 관련 다른 프로토콜들 SRP, SNAPI-X, AuthA, 그리고 PAK-X 과 비교할 수 있다. 아래 표는 AMP에 제공된 자료를 토대로 메시지 교환 횟수, 지수승 횟수, 난수생성 횟수에 대하여 비교한 것이다[7]. 서버의 지수승 연산 횟수는 효율성을 위하여 AMP와 마찬가지로 다중 병렬 멱승법(simultaneous multiple exponentiation)[12]을 취한다.

횟수	메시지 교환	지수승		난수 생성	
		클라이언트	서버	클라이언트	서버
SRP	4	3	3	1	1
SNAPI-X	5	5	4	2	3
AuthA	3	4	3	1	1
PAK-X	3	4	4	1	2
AMP	4	2	2	1	1
제안된 프로토콜	3	2	2	1	1

V 결론

본 논문에서는 패스워드 인증 키교환에서 요구되는 보안 요구사항을 만족시키는 프로토콜을 설계하였다. 제안된 프로토콜에는 분할된 패스워드를 사용하여 패스워드 검증자의 랜덤성을 증가 시켜 패스워드 추측공격이 어렵도록 하였다. 또한 비밀분산 기법을 적용하여 서버가 타협이 되어도 사전공격. 서버 가장공격이 어렵도록 하였다. 제안된 프로토콜은 기존에 제안되었던 프로토콜과 비교하여 비교적 적은 지수승을 사용하여 계산량의 부담이 적다. 그러나 분할된 패스워드에 따른 검증자의 랜덤화에 대한 증명가능한 방법을 제시하지 못하였다. 이 연구는 추후의 과제로 남겨졌다.

참고문헌

[1] T.Wu, "Secure remote password protocol," Internet Society Symposium on Network and Distributed System Security, 1998

[2] P.MacKenzie, T.Shrimpton, and M.Jakobsson, "Threshold Password-Authenticated Key Exchange," CRYPTO 2002

[3] Philip MacKenzie and Ram Swaminathan, "Secure Network Authentication with Password Identification," Presented to IEEE P1363a, August 1999

[4] M.Bellare, D.Pointcheval, and P.Rogaway, "Authenticated Key Exchange Secure Against Dictionary Attack," *Eurocrypto 2000*

[5] M.Bellare and P.Rogaway, "The AuthA protocol for password-based authenticated key exchange," 사이트 http://grouper.ieee.org/groups/1363/passwdPK/submissions.html

[6] V.Boyko, P.MacKenzie, and S.Patel, "Provably Secure Password Authenticated Key Exchange Using Diffie-Hellman," *Eurocrypt 2000*

[7] Taekyoung Kwon, "Ultimate Solution to Authentication via Memorable Password," IEEE P1363 study group

[8] D.Chaum and T.Pedersen, "Wallet databases with observer," CRYPTO92

[9] Xunhua Wang, "Intrusion-Tolerant Password Enabled PKI," *2nd annual PKI Research Workshop*, 2002

[10] Ho-Sun Yoon, Heung-Youl Youm, "A New Approach to Efficient Verifiable Secret Sharing for Threshold KCDSA" LNCS 1787, ICISC 99

[11] S.Blake-Wilson, A.Menezes, "Authenticated DH Key Agreement Protocols," LNCS 1556, SAC 98

[12] A.Menezes, P.van Oorschot, S.Vanston, "Handbook of applied cryptography," CRC Press, Inc., pp 618, 1997

[13] S.Bellovin and M.Merritt, "Encryted key exchange : password-based protocols secure against dictionary attacks," Proc. IEEE Comp. Society Symp. on Research in Security and Privacy, pp. 72-84, 1992

[14] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust Threshold DSS Signatures," LNCS 1070, EUROCRYPT96

[15] Rosario Gennaro, Michael O. Rabin, Tal Rabin, "Simplified VSS and Fast-track Multiparty Computations with Application to Threshold *Cryptography*," *In Proceedings of the 1998 ACM* Symposium

[16] Ueli Maurer, Stefan Wolf, "Diffie-Hellman, Decision Diffie-Hellman, and Discrete Logarithms," ISIT 1998, Cambrige, MA, USA, August 21-16

[17] 이정현, 김현정, 이동훈, "다중서버를 이용한 인증된 키교환 프로토콜," 정보보호학회논문지 13권 1호, 2003.2

[18] D. Boneh, "The decision Diffie-Hellman problem," Proceedings of the Third Algorithmic Number Theory Symposium, LNCS 1423, 1998.