情報保護學會論文誌
第19卷 第6號, 2009. 12

저전력 장비에 적합한 효율적인 RSA 기반의 PAKE 프로토콜

이 세 원,[1†] 윤 택 영,[1] 박 영 호,[2] 홍 석 희[1‡]
[1]고려대학교 정보경영공학전문대학원, [2]세종사이버대학교

Efficient RSA-Based PAKE Procotol for Low-Power Devices

Se Won Lee,[1†] Taek-Young Youn,[1] Yung Ho Park,[2] Seokhie Hong[1‡]
[1]Graduate School of Information Management and Security, Korea University,
[2]School of Computer Engineering, Sejong Cyber University

요 약

패스워드 기반의 키 교환(PAKE) 프로토콜은 난수를 공유하거나 PKI가 구축되어 있지 않은 환경에서 공유한 패스워드를 사용하여 세션키를 공유할 수 있게 함으로써 안전한 통신을 제공하는 암호학적 도구이다. RSA를 사용하여 효율적으로 설계하는 것이 쉽지 않기 때문에 대다수의 PAKE 프로토콜들은 Diffie-Hellman 키 교환을 기반으로 설계되어 왔다. 본 논문에서는 RSA 암호화의 효율성을 활용하여 비대칭 통신환경에 적합한 효율적인 RSA-PAKE 프로토콜을 제안한다. 제안하는 RSA-PAKE 프로토콜이 이론적인 계산량과 파라메타를 바탕으로 한 실험을 통하여 얼마나 효율적인지 판단한다. 제안하는 RSA-PAKE 프로토콜에서 비대칭 통신환경의 저전력 장비는 계산적으로 기존 프로토콜 중에서 안전하고 가장 효율적인 CEKEP보다 약 84% 효율적인 비용으로 키 교환을 수행할 수 있다. 특히, 일정 부분의 연산을 키 교환 과정이 진행되기 이전에 수행함으로써 키 교환 과정의 효율성을 극대화 할 수 있다. 제안하는 RSA-PAKE 프로토콜의 안전성은 RSA 문제를 기반으로 렌덤 오라클 모델에서 증명한다.

ABSTRACT

Password-Authenticated Key Exchange (PAKE) Protocol is a useful tool for secure communication conducted over open networks without sharing a common secret key or assuming the existence of the public key infrastructure (PKI). It seems difficult to design efficient PAKE protocols using RSA, and thus many PAKE protocols are designed based on the Diffie-Hellman key exchange (DH-PAKE). Therefore it is important to design an efficient PAKE based on RSA function since the function is suitable for designing a PAKE protocol for imbalanced communication environment. In this paper, we propose a computationally-efficient key exchange protocol based on the RSA function that is suitable for low-power devices in imbalanced environment. Our protocol is more efficient than previous RSA-PAKE protocols, required theoretical computation and experiment time in the same environment. Our protocol can provide that it is more 84% efficiency key exchange than secure and the most efficient RSA-PAKE protocol CEPEK. We can improve the performance of our protocol by computing some costly operations in offline step. We prove the security of our protocol under firmly formalized security model in the random oracle model.

Keywords: key exchange, password, authentication, RSA

접수일(2009년 6월 12일), 수정일(2009년 9월 5일),
게재확정일(2009년 9월 28일)

† 주저자, eseone@korea.ac.kr
‡ 교신저자, hsh@cist.korea.ac.kr

I. 서 론

패스워드 기반 암호기법은 기억 가능한 엔트로피가 낮은 정보를 패스워드로 사용하기 때문에 패스워드 추측 공격에 안전하게 설계되어야 한다. 일반적인 패스워드 추측 공격은 온라인 패스워드 추측 공격과 오프라인 패스워드 추측 공격이 있다. 공격자는 패스워드를 추측하고, 추측한 패스워드를 사용하여 프로토콜에 직접 참여하여 서버의 반응을 통해 추측한 패스워드가 맞는지 확인하는 온라인 패스워드 공격은 패스워드 기반 프로토콜의 경우 피할 수 없는 공격이지만, 실패 횟수를 제한하는 방법 등으로 쉽게 막을 수 있으므로 위협적인 공격으로는 간주하지 않는다. 네트워크상에 전송되는 통신 메시지를 수집하고 수집된 데이터를 통해 추측한 패스워드로 생성되었는지 검증함으로써 추측한 패스워드가 올바른지 아닌지 확인하는 오프라인 패스워드 추측 공격은 수집된 데이터를 사용하여 오프라인에서 모든 패스워드에 대한 확인이 가능하기 때문에 패스워드 기반 프로토콜에 대한 위협적인 공격이 된다.

RSA-PAKE 프로토콜은 다른 PAKE들과 달리 e-잉여 공격이라는 형태의 공격에 대한 안전성을 고려해야 한다. RSA-PAKE 프로토콜은 매 세션마다 새로운 RSA 변수 (n,e)를 생성하여 사용하기 때문에 인증서와 같이 공개키를 인증할 수 있는 정보가 없는 상태로 (n,e)를 사용한다. 이와 같은 구성상 특성에 의해 공격자는 전송되는 공개키를 $e|\phi(n)$를 만족하도록 변조함으로써 공격을 시도할 수 있다. 이러한 공격을 e-잉여 공격이라고 하고 RSA-PAKE 프로토콜에 큰 위협이 된다. e-잉여 공격에 안전하기 위해 전송되는 공개키의 신뢰성을 확인하기 위해 크게 두 가지 방법이 사용된다. 하나는 challenge/respones 방식의 프로토콜을 수행하여 공개키의 적합성을 검증하는 방법이고, 다른 하나는 특정 조건을 만족하는 공개키를 사용하여 e-잉여 공격에 대한 안전성을 보장하는 방법이다. challenge/respones 방식을 기반으로 설계된 최초의 프로토콜은 Zhu 등에 의해 제안되었다[15]. Wong 등은 challenge/respones 과정의 효율성을 개선한 프로토콜을 제시하였으나 안전성이 증명되지 않았다[12]. Catalaro 등은 최초로 안전성이 증명 가능한 프로토콜을 제안되었다[5]. 최근 challenge/ respones 방식을 사용하는 RSA-PAKE 프로토콜에 대한 일반적인 안전성 증명이 Shir 등에 의해 제안되었다[11]. 특정 조건을 만족하는 공개키를 사용함으로써 e-잉여 공격에 대한 안전성을 보장하는 최초의 RSA-PAKE 프로토콜인 SNAPI 프로토콜은 Mckenzie 등에 의해 제안되었다[8]. SNAPI 프로토콜은 큰 소수를 공개하도수로 사용함으로써 e-잉여 공격에 대한 안전성을 획득하였다. 그러나, SNAPI 프로토콜은 RSA 암호화 과정에서 큰 도수를 사용하기 때문에 연산 효율성이 낮다. 이점을 보안하기 위해 Zharg 등은 공개키로 작은 소수를 사용할 수 있는 안전성 증명이 가능한 RSA-PAKE 프로토콜인 PEKEP과 CEKEP이 제안되었다[14]. PEKEP와 CEKEP은 3, 5와 같이 작은 소수를 사용할 경우 소수 생성 비용과 소수 판정 비용이 매우 적지만 다수의 RSA 암호화를 수행하기 때문에 전체 연산 효율성이 높지 않다. 매우 효율적인 프로토콜인 RSA-EPAKE 프로토콜이 Park 등에 의해 제안되었으나[10] 최근 Youn 등에 의해 취약점이 발견되어 안전하지 않음이 밝혀졌다[13].

본 논문에서는 매우 효율적인 키 교환을 제공하는 RSA-PAKE 프로토콜을 제안한다. 제안하는 RSA-PAKE 프로토콜은 SNAPI 프로토콜, PEKEP, CEKEP와 RSA-EPAKE 프로토콜보다 매우 효율적인 키 교환을 제공한다. 비대칭 환경에서의 두 통신 주체는 서로 다른 능력을 가진 주체이며, 시스템이 잘 갖추어지고 연산 능력이 큰 매체와 시스템 사양도 비교적 낮고 연산 능력이 적은 매체이다. 저전력 장비란 시스템 사양이 비교적 낮고 연산 능력이 적은 매체이며, 예를 들면 모바일기기가 있다. 즉, 연산 능력이 작은 저전력 장비를 사용하는 사용자의 경우 매우 효율적으로 키 교환을 수행할 수 있도록 구성되어 있어 두 통신 주체가 상이한 통신 능력을 보유하고 통신에 참여하는 비대칭 환경에 매우 적합하게 사용될 수 있다. 제안하는 RSA-PAKE 프로토콜에서 적은 연산 능력을 가진 통신 주체의 키 교환 비용은 기존 제안된 프로토콜 중에서 가장 효율적인 프로토콜인 CEKEP보다 약 84% 이상 효율성이 개선되었다. 특히, 일정 부분 연산을 키 교환 프로토콜의 수행 이전에 미리 계산함으로써 실시간 키 교환 비용을 절감할 수 있다. 제안하는 RSA-PAKE 프로토콜의 안전성은 RSA 문제를 기반으로 렌덤 오라클 모델에서 증명하였다.

II. 안전성 모델 및 정의

제안하는 RSA-PAKE 프로토콜의 안전성을 증명하기 위해[1]에 제시된 안전성 모델을 살펴본다. 본 논문에서는 안전성 모델에 대한 간략한 설명을 제공하므

로 안전성 모델에 대한 자세한 설명은 [1]를 참고한다.

서버와 클라이언트를 각각 A와 B, 공격자는 E로 정의한다. D는 패스워드의 집합과 pw는 패스워드로 정의할 때, 두 통신 주체 A와 B는 키 교환을 수행하기 위해 $pw \in D$를 사전에 공유한 것으로 가정한다. Π_A^i는 주체 A의 프로토콜 i번째 세션에 대한 인스턴스로 정의한다. 각 인스턴스마다 세션 아이디 sid, 파트너 아이디 pid, 그리고 세션키 sk가 할당된다. 각 인스턴스는 sid와 pid를 갖고 프로토콜의 모든 과정을 수행하고, 프로토콜의 모든 과정이 올바르게 수행된 경우 sk가 할당된다.

공격모델 (Adversarial model): 안정성의 개념을 정의하기 위해 E의 능력에 대한 공격 모델을 설명한다. E는 수동적인 공격자로서 통신 메시지를 관찰하거나 능동적인 공격자로서 사용자들 사이의 메시지를 없애거나 공격자의 의도대로 메시지를 바꿀 수 있는 등의 행동을 할 수 있다. E의 공격 능력은 다음과 같이 오라클 질의의 형태로 정의할 수 있다.

- **Send**(A, i, M): E가 인스턴스 Π_A^i에 메시지 M을 전송하면, Π_A^i는 프로토콜을 수행하여 나온 응답을 E에게 전송한다. E는 M을 변조하거나 다른 인스턴스의 M을 전송할 수 있다.
- **Execute**(A,i,B,j): A와 B에 대한 인스턴스 Π_A^i와 Π_B^j는 이전에 프로토콜을 수행하지 않은 상태이다. E는 인스턴스 Π_A^i와 Π_B^j가 정상적인 프로토콜을 진행하는 동안 통신되는 모든 M을 얻을 수 있다.
- **Reveal**(A,i): E는 인스턴스 Π_A^i의 sk_A^i를 가지고 있다.
- **Test**(A,i): 인스턴스 Π_A^i에서 랜덤한 한 비트 b를 생성하고, 출력값으로 E에게 sk_A^i가 주어지면 b는 1로 결정되고, 랜덤한 sk가 주어지면 0으로 결정된다. 이 질의는 프로토콜이 진행되는 동안 단 한번 사용할 수 있다.
- **Oracle**(M): E는 공간 O에서 확률적으로 랜덤하게 선택하는 함수 H에 접근할 수 있는 오라클을 갖고 있다. O는 랜덤오라클 모델 또는 표준 모델에서 작동되는가에 따라 결정된다.

Partnering과 Freshness: 서로 다른 A와 B에 대하여 인스턴스 Π_A^i와 Π_B^j를 가정하자. 두 인스턴스가 같은 sid를 갖고서 프로토콜이 정당하게 진행되고 서로 상대방을 의미하는 pid를 갖는 경우, 즉 $sid_A^i = sid_B^j$와 $pid_A^i = pid_B^j$를 만족하면 인스턴스 Π_A^i와 Π_B^j는 Partner라고 정의한다. 인스턴스 Π_A^i가 Freshness라는 의미는 올바른 모든 검증과정을 통과한 인스턴스이고, 인스턴스 Π_A^i와 상대방에 대해 Reveal 질의가 이루어지지 않은 경우를 의미한다.

정확성(Correctness): 인스턴스 Π_A^i와 Π_B^j가 $sid_A^i = sid_B^j$와 $pid_A^i = pid_B^j$을 만족하는 Partner이고, 서로 정상적임을 확인한 후 두 인스턴스는 같은 세션키 $sk_A^i = sk_B^j$를 갖고 있는 상태를 의미한다.

E가 새로운 인스턴스에 Test 질의를 하고 받은 메시지가 랜덤값인지 아니면 정상적인 프로토콜에서 나온 세션키인지 판단하게 해주는 Test 질의에서의 출력값인 b를 올바르게 추측하는 사건을 Succ라고 정의한다. E의 공격능력을 다음과 같이 정의한다: $Adv_E^{ake} = 2\Pr(\mathrm{Succ}) - 1$. 다항식 시간 내의 E가 온라인 패스워드 추측 공격이 제한 없이 수행할 수 있다면 Adv_{E}^{ake}는 1에 가까워진다. 온라인 패스워드 추측공격의 경우 E가 추측한 패스워드로 프로토콜에 참여하여 추측한 패스워드가 맞는지 틀린지 알 수 있다.

정의 1. 패스워드 기반 키 교환 프로토콜은 모든 다항식 시간동안 공격을 수행하는 공격자 E가 매우 작은 (negligible) ε에 대해 $Adv_{\mathrm{E}}^{ake} \le Q_{se}/|D| + \varepsilon$ 의 공격 능력을 가지는 경우 안전하다고 정의한다. 이때, $|D|$는 D의 크기이고, Q_{se}는 Send 질의 횟수를 나타낸다.

III. 제안하는 프로토콜

본 장에서는 효율적인 RSA-PAKE 프로토콜을 제안한다. 두 사용자를 A, B라고 한다. Z_n은 n보다 적은 양의 정수 집합이고, $Z_n^* = \{x \in Z_n : \gcd(x, n) = 1\}$이다. 제안하는 RSA-PAKE 프로토콜에서는 해쉬함수 $H: \{0,1\}^* \rightarrow \{0,1\}^{k_1}$, $H_1, H_2, H_3 : \{0,1\}^* \rightarrow \{0,1\}^{k_2}$를 사용한다. A와 B는 세션키를 생성하기 위해 다음과 같이 프로토콜을 수행한다.

1. A는 $R_A \in_R \{0,1\}^k$와 ℓ-비트 모듈러 n을 생성하고, B에게 (R_A, n, A)을 전송한다.
2. B는 n이 ℓ-비트 홀수가 아니면 프로토콜을 종료한

[표 1] 제안하는 RSA-PAKE 프로토콜

A	password : pw	B
Generate RSA modulus $n=pq\in[2^{l-1},2^l-1]$ and $R_A\in\{0,1\}^k$		
	R_A, n, A →	
		If n isn't ℓ-bit odd number, reject. Generate m-bit prime e. $R_B\in\{0,1\}^k$, $R\in_R Z_n^*$ $\alpha=H(pw,R_A,R_B,A,B,e,n)$ If $\gcd(\alpha,n)\neq 1$, $z\in Z_n^*$ else $z=\alpha R^e \bmod n$.
	← e, R_B, z, B	
If e isn't m-bit prime, reject. $\alpha=H(pw,R_A,R_B,A,B,e,n)$ If $\gcd(e,\phi(n))\neq 1$ or $\gcd(\alpha,n)\neq 1$, $b\in_R Z_n^*$ else $d=e^{-1} \bmod \phi(n)$ and $b=(z\alpha^{-1})^d \bmod n$ $\beta=H_1(b,R_A,R_B,A,B,e,n)$		
	β →	
		$\beta=^? H_1(R,R_A,R_B,A,B,e,n)$ If not, reject $\gamma=H_2(R,R_A,R_B,A,B,e,n)$ $sk=H_3(R,R_A,R_B,A,B,e,n)$
	← γ	
$\gamma=^? H_2(b,R_A,R_B,A,B,e,n)$ If not, reject $sk=H_3(b,R_A,R_B,A,B,e,n)$		

다. 정당한 n일 경우 m-비트 소수 e, $R\in_R\{0,1\}^k$, $R_B\in Z_n^*$ 를 생성하고, $\alpha=H(pw,R_A,R_B,A,B,e,n)$를 계산한다. $\gcd(\alpha,n)\neq 1$ 이면 난수 $z\in Z_n^*$를 선택하고, 식이 성립하면 $z=\alpha R^e \bmod n$를 계산하여 A에게 (e,R_B,z,B)를 전송한다.

3. e가 m-비트 소수가 아닌 경우 A는 프로토콜을 종료한다. e가 m-비트 소수일 경우 A는 $\alpha=H(pw,R_A,R_B,A,B,e,n)$를 계산한다. $\gcd(\alpha,n)=1$ 또는 $\gcd(e,\phi(n))=1$가 성립하지 않으면 난수 $b\in Z_n^*$를 생성하고, 두 식이 성립하면 $d=e^{-1} \bmod \phi(n)$와 $b=(z\alpha^{-1})^d \bmod n$를 계산한다. $\beta=H_1(b,R_A,R_B,A,B,e,n)$를 계산하고 B에게 β를 전송한다.
4. B는 $\beta=H_1(R,R_A,R_B,A,B,e,n)$이 아니면 프로토콜을 종료한다. 해당 조건식이 만족하면 $\gamma=H_2(R,R_A,R_B,A,B,e,n)$와 $sk=H_3(R,R_A,R_B,A,B,e,n)$를 계산하고 A에게 γ를 전송한다.
5. A는 $\gamma=H_2(b,R_A,R_B,A,B,e,n)$이 아닌 경우 프로토콜을 종료한다. 식이 성립하면 세션키 $sk=H_3(b,R_A,R_B,A,B,e,n)$를 계산하고 프로토콜을 종료한다.

IV. 안전성 분석

본 장에서는 2장에서 살펴본 안전성 모델을 바탕으로 본 논문에서 제안한 RSA-PAKE의 안전성을 증명한다. Bellare와 Rogaway의 랜덤오라클 모델을 기반으로 기술한다[1]. 이 모델에서는 새로운 질의마다 랜덤값를 주는 오라클인 해쉬 함수가 사용된다. 이때 해쉬 함수에서 사용된 질의와 응답은 리스트로 저장되어 지고, 만약 이전의 질의로 다시 질의 되는 경우 랜덤값이 아닌 이전에 리스트에 저장된 값을 준다. RSA 문제를 푸는 것이 어렵다는 것을 가정하고, 증명과정에서는 t시간 동안 내에 RSA 문제가 풀 수 있는 확률이 아주 작다는 표현으로 $Adv^{rsa}(t)$를 사용한다. 제안하는 RSA-PAKE의 안전성은 Hybrid Argument 로 구성되어 있으며 Zhang 등의 논문에서 제시된 안전성 분석 방법과 거의 동일하다[14]. 결론적으로 제안하는 RSA-PAKE 프로토콜은 정리 1에서 보이는 것과 같이 증명 가능한 안전성을 제공한다.

정리 1을 증명하기 위한 Hybrid Argument는 5개의 Experiment와 각 Experiment사이에서 공격자가 가지는 능력의 차이를 보이기 위한 보조정리 4개, 그리고 마지막 Experiment에서 공격자의 공격능력을 증명하기 위한 보조정리 1개로 구성된다. 첫 번째 Experiment는 실제 공격과 동일한 환경을 제공하는 실험이고 마지막 Experiment는 공격자가 negligible한 공격 능력을 가지는 환경으로 설계되어 있다.

Hybrid Experiment P_0 : E는 다양한 오라클을 사용하여 공격할 수 있다.(ex. Send, Execute, Reveal, Test 질의) E는 4개의 독립적인 랜덤 오라클 H_1, H_2, H_3, H을 사용할 수 있다. 각 랜덤 오라클마다 입력 값에 대한 출력 값을 주고 리스트에 저장한다. 새로운 입력 값이면 출력 값을 랜덤값으로 주고, 이전과 동일한 입력 값일 경우 리스트에 저장되어 있는 출력 값을 준다. 여기서 $Adv(E)=Adv(E,P_0)$이고, 본 논문에서는 E가 다른 인스턴스들을 이용하여

$Adv(E,P_0) \le Q_{se}/|D| + \varepsilon$을 만족시키도록 하는 것이 목적이다.

Hybrid Experiment P_1 : Execute 오라클은 인스턴스의 세션키를 생성하기 위해 랜덤값을 선택할 수 있다. 두 인스턴스 Π_A^i와 Π_B^j에서 **Execute**(A,i,B,j) 오라클이 사용되면, 세션키 sk_A^i와 sk_B^j는 랜덤 오라클 H_3의 출력 값이 아닌 $\{0,1\}^k$에서 선택한 랜덤값으로 결정된다.

보조정리 1. 랜덤 오라클 사용한 횟수를 Q_h, Execute 오라클 사용한 횟수를 Q_{ex}로 정의할 때 모든 다항식 시간 내의 E는 다음 식을 만족한다:

$$|Adv(E,P_1) - Adv(E,P_0)| \le Q_{ex} Adv^{rsa}(O(t)) + Q_{ex} Q_h/\phi(n)$$

증명. E가 P_1과 P_0를 구별가능하다는 것은 RSA를 풀 수 있다는 것과 같음을 보여줌으로써 Zhang[14]의 논문의 보조정리 1과 유사한 방법으로 보조정리1를 증명할 수 있다. 자세한 증명은 부록 A에서 기술한다.□

다른 보조정리를 정의하고 증명하기 위해 Send 오라클을 5가지로 세분화하여 정의하고 각 오라클이 공격 알고리즘 E에게 미치는 영향을 확인하도록 한다.

- **Send$_0$**(A,i) : 인스턴스 Π_A^i는 ℓ-비트 RSA 모듈러 n과 $R_A \in \{0,1\}^k$를 생성한다. (R_A, n, A)를 E에게 보낸다.
- **Send$_1$**(B,j,A,n,R_A) : 인스턴스 Π_B^j는 n이 ℓ-비트인 홀수인지 확인하고, 아니면 프로토콜을 종료한다. n이 ℓ-비트인 홀수인지 확인한 후 m-비트 소수 e, $R_B \in \{0,1\}^k$와 $R \in Z_n^*$을 생성한다. 그리고 $\alpha = H(pw, R_A, R_B, A, B, e, n)$를 계산한다. 만약 $\gcd(\alpha, n) = 1$을 만족하지 않으면 $z \in Z_n^*$를 생성하고, 만족할 경우 $z = \alpha R^e \bmod n$를 계산하여 E에게 (e, R_B, z, B)를 보낸다.
- **Send$_2$**(A,i,e,R_B,z) : 인스턴스 Π_A^i는 e가 m-비트 소수인지 확인하고, 아니면 Π_A^i는 종료한다. 인스턴스 Π_A^i는 $\alpha = H(pw, R_A, R_B, A, B, e, n)$를 계산한다. 그리고 $\gcd(e, \phi(n)) \neq 1$ 또는 $\gcd(\alpha, n) \neq 1$인 경우 $b \in Z_n$를 생성하고, 아닌 경우 $d = e^{-1} \bmod \phi(n)$를 계산하여 $b = (z\alpha^{-1})^d \bmod n$를 복호화한다. b와 통신 정보를 사용하여 $\beta = H(b, R_A, R_B, A, B, e, n)$를 E에게 보낸다.
- **Send$_3$**(B,j,β) : 인스턴스 Π_B^j는 프로토콜 진행시 (R, R_A, R_B, e, n)를 알고 있다. 인스턴스 Π_B^j는 **Send$_1$**(B,j,A,n,R_A)에서 (R_A, n)를 받고, (R, R_B, e)을 생성했다. $H_1 = (R, R_A, R_B, A, B, e, n)$를 계산하여 β와 다르면 프로토콜을 종료하고, 같다면 $(R, R_A, R_B, A, B, e, n)$을 입력값으로 랜덤 오라클 H_2와 H_3을 이용하여 출력값을 갖는다. H_3의 출력 값은 sk_B^j로, H_2의 출력 값은 공격자에게 보낸다.
- **Send$_4$**(A,i,γ) : 인스턴스 Π_A^i는 프로토콜 진행시 (b, R_A, R_B, e, n)를 알고 있다. **Send$_0$**(A,i) 오라클에서 (R_A, n, A)와 **Send$_2$**(A,i,e,R_B,z) 오라클에서 (b, R_B, e)를 알 수 있다. $H_2(b, R_A, R_B, A, B, e, n)$를 계산하여 γ와 다르면 프로토콜을 종료하고, 같은 값이라면 $H_3(b, R_A, R_B, A, B, e, n)$를 sk_A^j로 갖는다.

메시지가 인스턴스에 의해 생성된 경우 "오라클로 생성되었다"(oracle-generated)고 정의한다. 그 외의 경우 "공격이 개입되어 생성되었다"(adversarially- generated)라고 정의한다. 인스턴스 Π_A^i에 의해 메시지가 생성된 경우 "Π_A^i-오라클로 생성되었다" (Π_A^i- oracle-generated)라고 정의한다.

Hybrid Experiment P_2 : 인스턴스 Π_B^j는 **Send$_1$** 오라클에서 Π_A^i-오라클로 생성된 메시지 (R_A, n, A)를 고려한다. 두 인스턴스 Π_A^i와 Π_B^j가 올바른 모든 검증과정을 통과한 메시지를 받는다면 같은 $\{0,1\}^k \in sk$를 갖는다. Π_B^j는 검증과정을 통과하여 정당한 메시지라고 판단하고 Π_A^i는 그렇지 않은 경우, Π_B^j만이 랜덤한 세션키를 갖고 Π_A^i는 세션키를 갖지 못한다.

보조정리 2. Send 오라클 사용한 횟수를 Q_{se}로 정의할 때 모든 다항식 시간 내의 E는 다음 식을 만족한다:

$$|Adv(\mathrm{E},\mathrm{P}_2) - Adv(\mathrm{E},\mathrm{P}_1)| \le Q_{se} Adv^{rsa}(O(t)) + Q_{se}/2^{k_2}$$

증명. 보조정리 2는 Zhang[14]의 논문의 보조정리 2와 유사한 방법으로 증명할 수 있다. 자세한 증명

은 부록 B에서 기술한다. □

Hybrid Experiment P_3 : 인스턴스 Π_B^j는 **Send$_1$** 오라클에서 Π_A^i-오라클로 생성된 메시지 (R_A, n, A)와 인스턴스 Π_A^i는 $Send_2$ 오라클에서 Π_B^j-오라클로 생성된 메시지 (e, R_B, z, B)를 고려한다. 두 인스턴스 Π_A^i와 Π_B^j가 메시지에 모든 검증과정을 통과하면 서로 같은 $\{0,1\}^k \in sk$를 갖는다.

보조정리 3. Send 오라클 사용한 횟수를 Q_{se}로 정의할 때 모든 다항식 시간 내의 E는 다음 식을 만족한다: $Adv(E,P_3) = Adv(E,P_2)$.

Send$_1$ 오라클에서 인스턴스 Π_A^i는 Π_B^j-오라클로 생성된 메시지 (e, R_B, z, B)를 받고서 인스턴스 Π_B^j는 공격이 개입된 메시지 (R_A, n, A)를 받는다고 생각하자. 이 때 메시지 (e, R_B, z, B)는 메시지 (R_A, n, A)의 영향을 받기 때문에 (e, R_B, z, B)도 공격자에 의해 생성된 메시지라고 볼 수 있다. 따라서 보조정리 3은 증명 없이 확인이 가능하다.

Hybrid Experiment P_4 : 인스턴스 Π_A^i(또는 Π_B^j)는 **Send$_2$**(또는 **Send$_1$**) 오라클에서 공격이 개입되어 생성된 메시지를 받는다. 이 때 Π_A^i(또는 Π_B^j)가 올바른 모든 검증 과정을 통과한 메시지라고 확인되면, 이 실험은 중지되고 공격자는 성공이라고 판단한다. 이 경우 공격자의 공격능력을 증가시킨다.

보조정리 4. Send 오라클 사용 횟수를 Q_{se}로 정의할 때 모든 다항식 시간 내의 공격자 E는 다음 식을 만족한다:

$$Adv(\mathrm{E},\mathrm{P}_3) \le Adv(\mathrm{E},\mathrm{P}_4)$$

P_3에서는 Execute와 Send 오라클로부터 정당한 인스턴스들에 의해 생성된 랜덤한 세션키를 가지고 있다. P_4는 공격이 개입되어 생성된 메시지를 이용하여 세션키의 교환 가능한 조건을 만족시키기 위해 P_3보다 E의 공격능력이 더 크다. 따라서 보조정리 4은 증명 없이 확인이 가능하다.

보조정리 5. 모든 다항식 시간 내의 공격자 E는 **Send$_1$**, **Send$_2$** 질의를 통해 $Q_{se} \le |D|$를 만족하는 Q_{se}를 수행한다. 이 때 공격자 E는 다음 식을 만족한다.

$$Adv(\mathrm{E},\mathrm{P}_4) \le \frac{\mathrm{Q_{se}} \cdot \ell}{\mathrm{e} \cdot 2^{\mathrm{m}-2}} + \frac{\mathrm{Q_{se}}}{|D|} + \frac{\mathrm{Q_{se}}}{2^{\mathrm{k_2}-1}} + 2Q_{se}Adv^{rsa}(O(t)) + \frac{2Q_{se}Q_h}{\phi(n)}$$

증명) 자세한 증명은 부록 C에서 기술한다.

앞에 제시된 보조정리 5에 제시된 Hybrid Argument는 5개의 Experiment와 각 Experiment사이에서 공격자가 가지는 능력의 차이를 이용하여 정리 1을 증명한다.

정리 1. 모든 다항식 시간동안 E는 다른 인스턴스들과 $Q_{se} \le |D|$를 만족하는 Send 질의를 수행한다. 이 때 제안하는 RSA-PAKE를 공격하는 E의 공격능력은 다음과 같다.

$$Adv_E^{ake} \le \frac{Q_{se}}{|D|} + (3Q_{se} + Q_{ex})Adv^{rsa}(O(t)) + \frac{\ell}{e \cdot 2^{m-2}} + \frac{Q_{se}}{2^{k_2-1}} + \frac{(2Q_{se} + Q_{ex})Q_h}{\phi(n)}$$

단, Q_{ex}과 Q_{se}는 각각 Execute와 Send 질의 횟수이고, Q_h는 랜덤 오라클을 사용한 횟수이다.

증명. 보조정리 1에서 5까지 나온 결과를 종합해 보면 P_0에서 E의 공격능력은 다음과 같다:

$$\begin{aligned} Adv(\mathrm{E},\mathrm{P}_0) &\le |\mathrm{Adv}(\mathrm{E},\mathrm{P}_1) - \mathrm{Adv}(\mathrm{E},\mathrm{P}_0)| \\ &\quad + |Adv(\mathrm{E},\mathrm{P}_2) - \mathrm{Adv}(\mathrm{E},\mathrm{P}_1)| \\ &\quad + \cdots + |Adv(\mathrm{E},\mathrm{P}_4) - \mathrm{Adv}(\mathrm{E},\mathrm{P}_3)| \\ &\le \frac{Q_{se}}{|D|} + (3Q_{se} + Q_{ex})Adv^{rsa}(O(t)) \\ &\quad + \frac{Q_{se} \cdot \ell}{e \cdot 2^{m-2}} + \frac{Q_{se}}{2^{k_2-1}} + \frac{(2Q_{se} + Q_{ex})Q_h}{\phi(n)} \\ &\le \frac{Q_{se}}{|D|} + (3Q_{se} + Q_{ex})Adv^{rsa}(O(t)) \\ &\quad + O\left(\frac{(2Q_{se} + Q_{ex})Q_h}{\phi(n)}\right) \end{aligned}$$

앞서 제시한 RSA 가정에 의해 $Adv^{rsa}(O(t))$는 매우 작은 값이다. 또한, 프로토콜에서 사용되는 변수 ℓ-비트 n과 m-비트 소수 e를 충분히 큰 값으로 선택함으로써 $\frac{Q_{se}}{2^{k_2-1}}$, $\frac{(2Q_{se} + Q_{ex})Q_h}{\phi(n)}$, $\frac{\ell}{e \cdot 2^{m-2}}$을 무시할 수 있는 크기의 값으로 설정할 수 있다. 따라서 충분히 큰 변수를 사용하면 제안하는 RSAPAKE 프로토콜은 렌덤 오라클 모델하에서 증명 가능한 안전성을 보장한다. □

V. 효율성분석

본 장에서는 제안한 RSA-PAKE 프로토콜의 e-잉여 공격에 대한 안전성을 분석하고, 안전성이 요구되는 RSA 공개키 변수 e의 최소 크기를 제시한다. RSA 공개키 e의 크기를 기준으로 기존의 RSA- PAKE 프로토콜들과 연산량을 비교하여 효율성에 대해 논의한다.

5.1. 변수 크기 분석

소수의 분포에 대한 수학적 사실에 근거하여 본 논문에서 제안한 RSA-PAKE 프로토콜의 e-잉여 공격에 대한 안전성을 분석 및 증명한다.

정리 2. (Prime Number Theorem[4]) 임의의 양의정수 x보다 작은 소수의 개수를 $\pi(x)$라고 정의한다. $\pi(x)$는 충분히 큰 x에 대해 다음과 같이 추정할 수 있다:

$$\pi(x) \approx \lim_{x\to\infty} \frac{x}{\ln x}.$$

잘 알려진 정리인 정리 2를 이용하여 본 논문에서 제안하는 RSA-PAKE 프로토콜에서 사용하는 소수의 크기를 정하기 위한 기준이 되는 정리 3를 증명한다.

정리 3. n은 ℓ-비트인 RSA 모듈러이고, e는 m-비트 소수이다. E가 e에 대한 정보를 알지 못하고 $e|\phi(n)$을 만족하는 n을 생성할 수 있는 확률은 다음과 같다:

$$\Pr[\gcd(e,\phi(n)) \neq 1] \leq \frac{\ell}{2^{m-1}}.$$

증명. 정리 2에 의해 m-비트 소수의 개수는 다음과 같다:

$$\begin{aligned}\pi(2^m)-\pi(2^{m-1}) &= \frac{2^m}{m\cdot \ln 2} - \frac{2^{m-1}}{(m-1)\cdot \ln 2} \\ &= \frac{(m-2)2^{m-1}}{m(m-1)\ln 2}\end{aligned}$$

$m \geq 5$인 경우 $(m-1)\ln 2 \leq (m-2)$이므로, 미리 m-비트 소수 한 개를 임의로 선택하고 이 소수를 맞출 수 있는 확률은 다음과 같이 계산 된다:

$$\frac{1}{\pi(2^m)-\pi(2^{m-1})} = \frac{(m-1)\ln 2}{m-2}\cdot\frac{m}{2^{m-1}} < \frac{2m}{2^m}$$

즉, E가 m-비트 소수 한 개를 추측하여 고정된 미지의 m-비트 소수를 맞출 수 있는 확률은 최대 $2m/2^m$이다. E가 n을 선택하는 경우 $\phi(n)$의 인수로 m-비트의 인수를 최대 $k=\lfloor \ell/m \rfloor$개 포함하도록 생성할 수 있다. $\phi(n)$를 구성하는 k개의 m-비트 인수를 $p_i (i=1,2,\cdots,k)$로 표기하자. E가 모르는 고정된 m-비트 소수 e에 대하여 $e|\phi(n)$을 만족시킬 수 있는 확률은 어떤 i에 대해 $p_i = e$를 만족하는 확률과 같다. 그러므로 $\Pr[\gcd(e,\phi(n)) \neq 1]$를 다음과 같이 정리할 수 있다.

$$\begin{aligned}\Pr[\gcd(e,\phi(n)) \neq 1] &= \Pr[p_1 = e] + \Pr[p_2 = e] \\ &\quad +\cdots+\Pr[p_k = e] \\ &= \frac{2m}{2^m}+\frac{2m}{2^m}+\cdots+\frac{2m}{2^m} \\ &= \frac{2km}{2^m} \leq \frac{2\ell}{2^m}\end{aligned}$$

결론적으로 E가 n을 선택하여 $e|\phi(n)$이 만족할 수 있는 확률은 최대 $2\ell/2^m$이다. □

본 논문에서 제안하는 RSA-PAKE 프로토콜은 A가 e를 선택하지 않고 B가 e를 선택한다. 따라서, E가 e-잉여 공격을 성공하기 위해서는 B가 선택할 e를 예측하여 자신이 선택하는 n이 $e|\phi(n)$을 만족하도록 구성해야 한다. 정리 3에 의하여 E의 e-잉여 공격의 성공 확률이 최대 $2\ell/2^m$임을 알 수 있다. E가 e-잉여 공격을 성공할 확률에 대해서 $2\ell/2^m < 1/|D|$를 만족하도록 ℓ과 m을 설정하면, 패스워드를 추측하는 확률보다 낮게 할 수 있다. 즉, E가 e-잉여 공격을 시도하여도 온라인 패스워드 공격보다 높은 확률로 공격에 성공할 수 없다.

5.2. 연산량 비교 및 분석

기존에 제안된 RSA-PAKE 프로토콜인 SNAPI, PEKEP, CEKEP, RSA-EPAKE 프로토콜들의 효율성과 본 논문에서 제안한 프로토콜의 효율성을 비교한다. [표 2]는 Windows XP, 3.40GHz Pentium(R) D CPU, 2 GB RAM인 프로세서를 이용하여 Microsoft Visual C++ 6.0 프로그램을 통해서 실험 수행한 결과이다. 동일한 조건에서의 비교를 위해 RSA 모듈러 n의 크기가 1024-비트인 경우를 고려한다. 또한, 지수연산은 기본적인 square-and- mltiply 알고리즘을 사용하고 소수 생성 및 판정은 Miller-Rabin 소수 판정법을 사용한다. 각 프로토콜에

[표 2] 제안하는 RSA-PAKE 프로토콜과 이전 RSA-PAKE 프로토콜들의 서버와 클라이언트 연산량 비용 비교
(걸린 시간 단위 : $1/10^3$초, 시행횟수 : 100,000번, S=서버, C=클라이언트)

		연산량		연산량 합 (총 걸린 시간)
		소수생성 및 판정비용 (걸린시간)	암호화 및 복호화 비용 : 모듈러 n에 대한 지수승 (걸린시간)	
SNAPI	S	n과 1025-비트 소수 e생성 (760.58+4343.06)	1024-비트 지수승 1번 (48.78)	(5152.42)
	C	1025-비트 소수 e 판정 (1542.75)	1025-비트 지수승 1번 (49.06)	(1591.81)
PEKEP	S	n생성 (760.58)	1024-비트 지수승 2번 (97.56)	(858.14)
	C	- (0)	1024-비트보다 큰 지수승 1번 (188.31)	(188.31)
CEKEP	S	n생성 (760.58)	1024-비트 지수승 3번 (146.34)	(906.92)
	C	- (0)	약 80-비트 지수승 2번 (29.20)	(29.20)
EPAKE	S	n과 96-비트 소수 e생성 (760.58+4.88)	1024-비트 지수승 1번 (48.78)	(814.24)
	C	96-비트 소수 e판정 (2.41)	96-비트 지수승 1번 (5.41)	(7.82)
제안하는 RSA-PAKE	S	n생성 비용과 52-비트 소수e 판정 (760.58+0.55)	1024-비트 지수승 1번 (48.78)	(809.91)
	C	52-비트 소수 e생성 (1.23)	52-비트 지수승 1번 (3.37)	(4.60)

서 요구되어지는 세부적인 사항을 살펴보자. SNAPI 프로토콜에서 서버는 n과 1024-비트보다 큰 소수 e를 생성하고, 클라이언트는 e의 소수 판별 비용과 RSA 암호화인 e에 대한 지수승 연산 비용이 요구된다. PEKEP와 CEKEP에서 서버는 n과 소수 e를 생성한다. 여기서 $e=3$, $\varepsilon=2^{-80}$인 경우를 고려한다. PEKEP에서 클라이언트는 RSA 암호화 과정에서 1024-비트보다 큰 지수에 대한 지수승이 요구 된다. CEKEP에서는 전체적인 라운드 수가 증가하고, 클라이언트의 RSA 암호화 연산 횟수가 늘어난다. 그러나 CEKEP의 총 RSA 암호화 연산 비용은 PEKEP와 비교시 상당히 적다. CEKEP에서 클라이언트는 약 80-비트 지수승 연산이 2번 요구된다. RSA-EPAKE 프로토콜에서 서버는 n과 최소 96-비트 소수 e를 생성한다. 클라이언트는 96-비트 소수 판정 비용과 RSA 암호화 연산 비용으로 96-비트 지수승 연산이 소요된다. 정리 3에 의해서 제안한 프로토콜에서는 서버가 n을 생성하고, 클라이언트에서 최소 52-비트 소수 e를 생성한다. 소수를 생성하는데 필요한 알고리즘인 Miller-Rabin 소수 판정법에서 각 소수에 대해 실행되는 수행 횟수를 살펴보자. Miller-Rabin 소수 판정법을 사용할 경우 소수의 크기 및 어떠한 확률 범위 안에 그 수가 소수라는 가능성에 따라 테스트 횟수가 다르게 설정되어 있다. 기준은 [4]에서 제시된 것을 따른다. k는 비트수, t는 테스트 횟수이며, $p_{k,t}$는 테스트를 통과한 값이 진정한 소수일 확률로 정의된다. 각 해당 k비트마다 t와 k에 해당하는 $p_{k,t}$ 식을 선택하여 계산하여 적용하였다. 이러한 설정에서 $t \geq k/4, k \geq 21$을 만족하는 비교적 작은 k-비트에 대해 $p_{k,t} < 1/7 \cdot k^{15/4} \cdot 2^{-k/2-2t}$이 성립한다[4]. 제안하는 RSA-PAKE 프로토콜에서는 $p_{k,t} \leq 2^{-40}$을 만족하면 되므로 52-비트일 경우 t는 17번이다. 이전 프로토콜들은 $p_{k,t} \leq 2^{-80}$을 만족하는 소수를 사용해야 하므로 이를 실험에 반영하였다. SNAPI 프로토콜에서는 1025-비트 소수의 경우 t는 3번이고, RSA-EPAKE 프로토콜에서는 96-비트 소수의 경우 t는 27번이다. 이와 같이 Miller- Rabin 소수 판정법을 사용하여 소수의 신뢰성 수치와 각 비트수에 요구되는 t를 구해서 평준하게 실험하였다.

결과적으로 제안하는 RSA-PAKE 프로토콜은 기존의 RSA-PAKE 프로토콜들에 비해 효율적인 키

교환을 제공한다. 또한, e를 미리 계산하여 실제 키 교환 과정에서 52-비트 지수에 대한 지수승 한 번만 수행하면 되므로, 매우 빠르게 키 교환 과정을 진행할 수 있다.

Ⅵ. 결 론

본 논문에서는 매우 효율적인 RSA-PAKE 프로토콜을 제안하였다. 기존의 대부분의 PAKE프로토콜들은 RSA를 사용하여 효율적으로 설계하는 것이 쉽지 않기 때문에 Diffie-Hellman 키 교환을 기반으로 설계되어 왔다. 따라서 효율적으로 키 교환을 제공하는 안전한 RSA-PAKE 프로토콜을 설계하는 것은 매우 의미 있는 연구이다. 비대칭 통신환경에 적합한 효율적으로 제안된 RSA-PAKE 프로토콜은 기존에 제안된 PAKE 프로토콜보다 매우 효율적인 키 교환을 제공한다. CEKEP의 가장 효율적이라 고려되는 $e=3$, $\varepsilon=2^{-80}$인 경우와 비교했을 때, 표2에서 제시했듯이 제안하는 RSA-PAKE 프로토콜의 클라이언트 연산량은 CEKEP보다 약 84%의 효율성이 개선되었음을 알 수 있다. 특히, 소수의 생성을 오프라인 단계에서 미리 수행하는 경우 실제로 키 교환이 수행되는 과정에서의 연산량을 줄일 수 있다. 이와 같은 경우 CEKEP보다 약 88% 개선된 효율성을 보인다.

참 고 문 헌

[1] M. Bellare, D. Pointcheval, and P. Rogaway, “Authenticated key exchange secure against dictionary attack,” Advances in Cryptology - EUROCRYPT 2000 Proceedings, LNCS 1807, pp. 139-155, 2000.

[2] S. Bellovin and M. Merritt, “Encrypted key exchange : Password-based protocols secure against dictionary attacks,” Proc. of the IEEE Symposium on Research in Security and Privacy, pp. 72-84, May 1992.

[3] S. Bellovin and M. Merritt, “Augmented encrypted key exchange : A passwordbased protocol secure against dictionary attacks and password file compromise,” Proc. of the 1st ACM Conference on Computer and Communications Security, ACM, pp. 244-250, Nov. 1993.

[4] D.M. Burton, Elementary number theory, 6th Ed., McGraw-Hill Higher Education, 2007.

[5] D. Catalano, D. Pointcheval, and T. Pornin, “Trapdoor Hard-to-Invert Group Isomorphisms and Their Application to Password-Based Authentication,” IACR 2007, pp. 115-149, Jan. 2007.

[6] R. Gennaro and Y. Lindell, “A framework for password-based authenticated key exchange,” Ad-vances in Cryptology - EUROCRYPT 2003 Proceedings, LNCS 2656, pp. 524-542, 2003.

[7] S. Lucks, “Open key exchange: How to defeat dictionary attacks without encrypting public keys,” Proc. of Security Protocol Workshop, LNCS 1361, pp. 79-90, 1997.

[8] P. MacKenzie, S. Patel, and R. Swaminathan, “Password-authenticated key exchange based on RSA,” Advances in Cryptology, ASIACRYPT 2000 Proceedings, LNCS 1976, pp. 599-613, 2000.

[9] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, Handbook of Applied Cryptography, 4th Ed., CRC Press, Oct. 1996.

[10] S.J. Park, J.H. Nam, S.J. Kim, and D.H. Won, “Efficient Password-Authenticated Key Exchange Based on RSA,” CT-RSA, LNCS 4377, pp. 309 - 323, 2007.

[11] S.H. Shin, K. Kobara, and H. Imai, “RSA-based Password-Authenticated Key Exchange,Revisited,” IEICE TRANSACTIONS on Information and Systems, vol. E91-D, no. 5, pp. 1424-1438, May 2008.

[12] D.S. Wong, A.H. Chan, and F. Zhu, “More Effcient Password Authenticated Key Exchange Based on RSA,” In Proc. of INDOCRYPT 2003, LNCS 2904, pp. 375-387, 2003.

[13] T.Y. Youn, Y.H. Park, C.H. Kim, and J. Lim, “Weakness in a RSA-based password authenticated key exchange protocol,” Inf. Process. Lett, vol. 108, no. 6, pp. 339-342, Nov. 2008.

[14] M. Zhang, “New approaches to password authenticated key exchange based on RSA,” Proc. of Asiacrypt, LNCS 3329, pp. 230-244, 2004.

[15] F. Zhu, D.S. Wong, A.H. Chan, and R. Ye, “Password Authenticated Key Exchange Based

on RSA for Imbalanced Wireless Networks," In Proc. of ISC 2002, LNCS 2433, pp. 150-161, 2002.

<著者紹介>

이 세 원 (Se Won Lee) 학생회원
2006년 2월: 숭실대학교 수학과 학사
2006년 9월 ~ 현재: 고려대학교 정보경영공학전문대학원 석사 과정
<관심분야> 암호이론, 정보보호이론, 암호프로토콜

윤 택 영 (Taek-Young Youn) 학생회원
2003년 2월: 고려대학교 수학과 이학학사
2005년 2월: 고려대학교 정보보호대학원 정보보호학과 공학석사
2005년 3월 ~ 현재: 고려대학교 정보경영공학전문대학원 박사 과정
<관심분야> 암호이론, 정보보호이론, 암호프로토콜, 부채널공격

박 영 호 (Young-Ho Park) 정회원
1990년: 고려대학교 수학과 이학사
1993년: 고려대학교 수학과 이학석사
1997년: 고려대학교 수학과 이학박사
2006년 2월~현재: 세종 사이버 대학교 부교수
<관심분야> 정수론, 공개키 암호, 암호 프로토콜, 부채널 공격

홍 석 희 (Seokhie Hong) 종신회원
1995년: 고려대학교 수학과 학사
1997년: 고려대학교 수학과 석사
2001년: 고려대학교 수학과 박사
1999년 8월~2004년 2월: (주)시큐리티 테크놀로지스 선임연구원
2003년 3월~2004년 2월: 고려대학교 시간강사
2004년 4월~2005년 2월: K.U. Leuven 박사후연구원
2005년 3월~현재: 고려대학교 정보경영전문대학원 부교수
<관심분야> 대칭키 암호 알고리즘, 공개키 암호 알고리즘, 포렌식

부 록

A. 보조정리 1에 대한 증명

보조정리 1. 랜덤 오라클 사용한 횟수를 Q_h, Execute 오라클 사용한 횟수를 Q_{ex}로 정의할 때 모든 다항식 시간 내의 E는 다음 식을 만족한다.

$$|Adv(E,P_1)-Adv(E,P_0)| \le Q_{ex}Adv^{rsa}(O(t)) + Q_{ex}Q_h/\phi(n)$$

증명. E가 P_1과 P_0를 구별가능하다는 것은 RSA를 풀 수 있다는 것과 같음을 보여줌으로써 보조정리1을 증명할 것이다. 같은 랜덤값을 사용하는 동일한 Execute 질의가 사용되는 경우는 확률상 매우 작으므로 배제한다.

P_0에서 R을 포함한 정보를 H_3오라클을 통해 세션키가 생성되고, P_1에서는 R에 대한 정보 없이 H_3오라클을 사용하지 않고서 랜덤값으로 세션키가 생성된다. 여기서 알 수 있듯이 R을 알고 있다면 P_0와 P_1를 구별할 수 있다. 다음은 R을 구할 수 있는 확률을 구하기 위한 과정으로 게임1과 게임2를 정의한다.

게임1. E가 인스턴스 Π_A^i와 Π_B^j 사이에서 랜덤 오라클 H_1과 H_2를 사용한 올바른 메시지를 취득할 수 있을 경우이다.

1) Π_A^i는 $R_A \in \{0,1\}^k$와 RSA 모듈러 n을 Π_B^j에게 준다.
2) Π_B^j는 m-비트 소수 e, $R_B \in \{0,1\}^k$와 $R \in Z_n^*$를 생성한다. $\alpha = H(pw, R_A, R_B, A, B, e, n)$를 계산한다. $\gcd(\alpha, n) \neq 1$ 이면 난수 $z \in Z_n^*$를 선택하고, 식이 성립하면 $z = \alpha R^e \bmod n$를 계산하여 Π_A^i에게 (e, R_B, z, B)를 준다.
3) Π_A^i는 $b = (z\alpha^{-1})^d \bmod n$와 $\beta = H_1(b, R_A, R_B, A, B, e, n)$를 계산하여 Π_B^j에게 β를 준다.
4) Π_B^j는 Π_A^i에게 β를 받고, 정당한 Π_A^i라고 알 수 있다. 그리고 Π_B^j는 $\gamma = H_2(R, R_A, R_B, A, B, e, n)$를 계산하여 Π_A^i에게 준다.
5) Π_A^i는 γ를 받고, 검증과정을 통과하면 정당한 Π_B^j라고 알 수 있다. 게임은 여기서 종료되고 E는 추측한 R을 출력한다.

게임2. Π_A^i와 Π_B^j가 랜덤 오라클 H_1과 H_2를 사용하지 않고 랜덤값을 사용하는 경우 :

1) Π_A^i는 $R_A \in \{0,1\}^k$와 RSA 모듈러 n을 Π_B^j에게 준다.
2) Π_B^j는 m-비트 소수 e, $R_B \in \{0,1\}^k$와 $R \in Z_n^*$를 생성한다. $\alpha = H(pw, R_A, R_B, A, B, e, n)$를 계산한다. $\gcd(\alpha, n) \neq 1$ 이면 난수 $z \in Z_n^*$를 선택하고, 식이 성립하면 $z = \alpha R^e \bmod n$를 계산하여 Π_A^i에게 (e, R_B, z, B)를 준다.
3) Π_A^i는 (z, e, R_B)를 받고, $\beta \in \{0,1\}^k$를 생성하여 Π_B^j에게 준다.
4) Π_B^j는 Π_A^i에게 β를 받고, 정당한 Π_A^i라고 알 수 있다. Π_B^j는 $\gamma \in \{0,1\}^k$를 생성하여 Π_A^i에게 준다.
5) Π_A^i는 γ를 받고, 검증과정을 통과하면 정당한 Π_B^j라고 알 수 있다. 게임은 여기서 종료되고 E는 추측한 R을 출력한다.

게임1과 게임2에서 올바른 R을 추측하는 E의 확률을 각각 $\Pr(R \to G_1)$, $\Pr(R \to G_2)$라고 하자. 또한 $\Pr(R)$는 R를 추측하는 확률이라고 정의한다. $\Pr(R)$는 게임1에서 R를 추측하는 확률과 같다. 게임1에서는 E가 R의 추측 값으로 Z_n^*에서 랜덤값 x를 선택할 수 있고, H_1 또는 H_2오라클에 $(x, R_A, R_B, A, B, e, n)$를 입력하여 받은 값과 β 또는 γ와 비교하여 x가 정당한 값인지 판단할 수 있다. E는 H_1 또는 H_2오라클에 $(R, R_A, R_B, A, B, e, n)$를 입력하여 값을 얻을 수 있는 사건을 $TH_{1,2}$라고 하고, H_1 또는 H_2오라클을 사용하지 않는 사건을 $FH_{1,2}$이라 하자.

$$\begin{aligned}\Pr(R \to G_1) &= \Pr(R \to G_1 | TH_{1,2})\Pr(TH_{1,2}) \\ &\quad + \Pr(R \to G_1 | FH_{1,2})\Pr(FH_{1,2}) \\ &\le \Pr(TH_{1,2}) + \Pr(R \to G_1 | FH_{1,2}) \\ &= \Pr(TH_{1,2}) + \Pr(R \to G_2)\end{aligned}$$

게임 1에서 $FH_{1,2}$의 경우 β와 γ는 k-비트 랜덤값과 구별하지 못하므로 게임2에서 R를 추측하는 확률과 같다. $\Pr(R \to G_2)$를 계산해 보자. RSA 공개키 (n,e)와 n보다 작은 랜덤한 자연수 c가 주어질 경우, c를 복호화하는 효율적인 알고리즘 C를 구성하자. 알고리즘 C는 게임2와 2번째 단계에서 Π_B^j가 $z = \alpha c \bmod n$를 계산하는 단계만 다르고, 게임2의 나머지 과정은 동일하게 진행된다. 게임2에서 알고리즘 C에서의 출력 값이 E에게 주어진다. 게임2에서 E가 받은 출력 값 x가 정당한

값이라면 $\alpha x^e = z \bmod n$를 만족한다. 이 경우 x는 c의 복호화 값이므로 게임2에서 R를 찾은 확률은 다음과 같이 정리된다.

$$\Pr(R \rightarrow G_2) \le Adv_C^{rsa}(O(t)) \le Adv^{rsa}(O(t))$$

H_1 또는 H_2 오라클을 이용한 횟수를 Q_h라고 정의할 때, $\Pr(TH_{1,2}) = Q_h/\phi(n)$을 만족하고, R을 추측할 수 있는 확률은 $\Pr(R) \le Adv^{rsa}(O(t)) + Q_h/\phi(n)$이 된다. E는 다른 인스턴스에 대해 Excute 질의가 이루어지기 때문에, 보조정리1이 증명된다. □

B. 보조정리 2에 대한 증명

보조정리 2. Send 오라클 사용한 횟수를 Q_{se}로 정의할 때 모든 다항식 시간 내의 E는 다음 식을 만족한다.

$$|Adv(\mathrm{E},\mathrm{P}_2) - Adv(\mathrm{E},\mathrm{P}_1)| \le Q_{se} Adv^{rsa}(O(t)) + Q_{se}/2^{k_2}$$

증명. 인스턴스 Π_B^j는 Π_A^i-오라클로 생성된 메시지 (R_A, n, A)를 받아서 메시지 (e, R_B, z, B)를 생성하고, E에게 생성한 메시지를 준다고 가정하자. 정당한 n을 사용하므로 $\gcd(n,\alpha) \neq 1$인 확률은 배제한다. z는 랜덤 오라클 H에 $(pw, R_A, R_B, A, B, e, n)$를 질의하여 나온 α가 사용되어 계산되어진다고 가정한다. 인스턴스 Π_A^i가 n을 생성했기 때문에 E는 $\phi(n)$을 알지 못하고 개인키 d도 알 수 없다. 보조정리1에서 보였듯이 E가 랜덤값 R를 알 수 있는 확률은 $Adv^{rsa}(O(t))$보다 작고, R에 상관없이 β를 단순히 맞추는 방법의 확률은 $1/2^{k_2}$이다. 인스턴스 Π_B^j는 Π_A^i에서 받은 β의 검증과정을 수행하고, γ를 계산하여 Π_A^i에게 준다. 인스턴스 Π_A^i는 γ의 검증과정이 올바르게 수행된다면 같은 세션키를 갖게 된다. 그러므로 인스턴스 Π_A^i와 Π_B^j가 세션키를 갖는 것은 E가 R을 올바르게 추측한 것과 같으며, E는 다른 인스턴스에 대해 Send 질의가 이루어지기 때문에, P_2와 P_1에서 E의 공격능력의 차이는 $Q_{se}(Adv^{rsa}(O(t)) + 1/2^{k_2})$보다 작게 된다. □

C. 보조정리 5에 대한 증명

보조정리 5. 모든 다항식 시간 내의 공격자 E는 **Send₁**, **Send₂** 질의를 통해 $Q_{se} \le |D|$를 만족하는 Q_{se}를 수행한다. 이 때 공격자 E는 다음 식을 만족한다.

$$Adv(\mathrm{E},\mathrm{P}_4) \le \frac{\mathrm{Q_{se}} \cdot \ell}{\mathrm{e} \cdot 2^{\mathrm{m}-2}} + \frac{\mathrm{Q_{se}}}{|D|} + \frac{\mathrm{Q_{se}}}{2^{\mathrm{k}_2-1}} + 2Q_{se} Adv^{rsa}(O(t)) + \frac{2Q_{se}Q_h}{\phi(n)}$$

증명. P_4에서 E가 **Send₁**, **Send₂** 오라클을 사용 횟수를 각각 Q_{send_1}, Q_{send_2}라고 정의할 때, 각각의 오라클에서 공격이 개입되어 생성된 메시지를 이용한 방법으로 볼 수 있다.

Case 1. Send₂ 오라클에서 공격이 개입되어 생성된 메시지 (e, R_B, z, B)를 받은 인스턴스 Π_A^i를 고려하자. $\gcd(\alpha, \phi(n)) \neq 1$이거나 $\gcd(\alpha, n) \neq 1$인 경우 Π_A^i는 z를 랜덤값으로 주기 때문에 같은 세션키의 교환이 이루어지지 않는다. 그러므로 $\gcd(\alpha, n) = 1$과 $\gcd(\alpha, \phi(n)) = 1$인 경우를 고려한다. 인스턴스 Π_A^i는 $d = e^{-1} \bmod \phi(n)$와 $b = (z\alpha^{-1})^d \bmod n$를 계산할 수 있다. 그러나 E는 $\phi(n)$을 알 수 없으므로 d와 b를 계산할 수 없다. 그 다음 인스턴스 Π_A^i는 β를 계산하여 E에게 준다. E는 γ를 구하여 인스턴스 Π_A^i에게 주어야한다. 여기서 b에 대해 모르고 E가 γ를 생성하는 확률은 단지 2^{-k_2}이다. 만약 b를 알고서 γ를 계산 가능한 두 가지 경우에 대해 알아보자. 먼저 생각할 수 있는 방법은 E가 선택한 랜덤값 $z \in Z_n^*$에서 b를 구할 수 있는 확률로 보조정리1에서의 a를 구하는 확률과 유사하다.

$$\Pr(b) \le Adv^{rsa}(O(t)) + Q_h/\phi(n)$$

다른 방법으로는 E가 z를 생성하여 b를 구하는 방법이다. E는 패스워드의 추측 값으로 $pw_i \in D$를 선택하여 랜덤 오라클 H을 이용하여 $\alpha_i = H(pw_i, R_A, R_B, A, B, e, n)$를 얻는다. 랜덤값 R_i를 생성하여 $z = \alpha_i R_i^e \bmod n$를 계산한다. 랜덤 오라클 H을 이용하여 얻은 $\alpha = (pw, R_A, R_B, A, B, e, n)$가 α_i와 같다면 b도 R_i와 같게 된다. E가 패스워드를 잘못 추측했을 때에는 잘못된 α의 경우로 b를 알 수 없다. 이 방법으로 b를 구할 수 있는 확률은 다음과 같다: $\Pr(b) \le \Pr[\alpha_i = \alpha] \le 1/|D|$. 두 경우를 종합해보면 다음과 같다:

$$\Pr(b) \le 1/|D| + Adv^{rsa}(O(t)) + Q_h/\phi(n).$$

Send₂을 사용한 횟수를 Q_{send_2}라고 할 때, Case 1에서는 다음과 같은 확률이 만족한다.

$$\Pr[Succ of case1] \le Q_{send_2}(\Pr(b)+\frac{1}{2^{k_2}}) \le \frac{Q_{send_2}}{|D|}+Q_{send_2}Adv^{rsa}(O(t)) + \frac{Q_{send_2}Q_h}{\phi(n)}+\frac{Q_{send_2}}{2^{k_2}}$$

Case 2. Send$_1$ 오라클에서 공격이 개입되어 생성된 메시지 (R_A, n, A)를 받은 인스턴스 Π_B^j를 고려하자. 인스턴스 Π_B^j은 n이 ℓ-비트 홀수인지 확인하고 만족하지 않으면 프로토콜을 종료한다. 그리고 m-비트 소수 e를 생성하고, $\{0,1\}^k \in R_B$와 $R \in Z_n^*$를 생성한다. 그 다음 랜덤오라클 H를 이용하여 $\alpha = H(pw, R_A, R_B, A, B, e, n)$를 얻는다. $z = \alpha R^e \bmod n$를 계산하여 (e, R_B, z, B)를 E에게 준다. E는 β를 생성하여 Π_B^j에게 주어야한다. β를 생성하기 위한 방법을 알아보자. 다른 정보 없이 β를 생성하는 방법과 랜덤오라클 H_2를 이용하기 위해 필요한 정보인 R을 찾아서 β를 구하는 방법으로 나눌 수 있다. β를 단순히 맞추는 방법의 확률은 $1/2^{k_2}$이다.

β를 생성하기 위해 R을 구하는 확률에 대해 알아보자. $\gcd(\alpha, n) = 1$인 사건을 A라고 하고, $\gcd(e, \phi(n)) = 1$인 사건을 B라고 정의한다.

$$\begin{aligned}\Pr(R) &= \Pr(R|A\wedge B)\Pr(A\wedge B) + \Pr(R|A\wedge\neg B)\Pr(A\wedge\neg B) + \Pr(R|\neg A)\Pr(\neg A)\\ &\le \Pr(R|A\wedge B) + \Pr(R|A\wedge\neg B)\Pr(A\wedge\neg B)+\Pr(R|\neg A)\\ &\le 1/|D| + 1/e\cdot l/2^{m-1} + 1/\phi(n)\end{aligned}$$

$\gcd(\alpha, n) \neq 1$인 경우 프로토콜에 의해 인스턴스 Π_B^j가 R와 관계없는 랜덤한 z를 주고, E는 R을 정하여 β를 생성해서 Π_B^j에게 준다. 랜덤값 R를 맞추는 것과 같으므로 $\Pr(R|\neg A) = 1/\phi(n)$이 된다. ℓ-비트인 n과 m-비트 소수 e에 대해서 $\Pr(A) \approx 1$로 가정하면 $\gcd(e, \phi(n)) \neq 1$인 사건이 일어날 확률은 정리 3에 의해서 $\ell/2^{m-1}$보다 작게 된다. 즉, $\Pr(A\wedge\neg B) \le \ell/2^{m-1}$이 된다. $\Pr(R|A\wedge\neg B)$은 $\gcd(e, \phi(n)) \neq 1$인 사건이 일어날 경우 패스워드를 알 수 있기 때문에 $x^e \equiv 1 \pmod n$을 만족하는 확률과 같다. 방정식 근의 개수는 e개이므로 $\Pr(R|A\wedge\neg B) \le 1/e$이 된다.

또한 $\Pr(A\wedge B) \approx 1$임을 쉽게 알 수 있고, 이때 R를 찾는 확률은 패스워드를 찾는 확률과 같아지므로 $\Pr(R|A\wedge B) \le 1/|D|$을 만족한다. 단, $|D| \approx 2^{40}$이다.

$$\Pr[\text{Succ of case2}] \le Q_{send_1}(\Pr(R)+\frac{1}{2^{k_2}}) \le Q_{send_1}(\frac{1}{|D|}+\frac{1}{e}\cdot\frac{\ell}{2^{m-1}}+\frac{1}{\phi(n)}+\frac{1}{2^{k_2}})$$

두 경우에서 나온 결과를 종합하면 다음과 같다. 여기서 $Q_{se} = Q_{send_1} + Q_{send_2}$이다.

$$\Pr[\text{Succ}] = \Pr[\text{Succ of case1}] + \Pr[\text{Succ of case2}] \le \frac{Q_{se}\cdot\ell}{e\cdot 2^{m-1}}+\frac{Q_{se}}{|D|}+\frac{Q_{se}}{2^{k_2}}+Q_{se}Adv^{rsa}(O(t))+\frac{Q_{se}Q_h}{\phi(n)}$$

그리고 $Q_{se}/|D| \le 1$을 만족하므로 P_4에서 공격자의 공격능력은 다음과 같이 정리된다.

$$Adv(E, P_4) = 2\Pr[\text{Succ}] - 1 \le \frac{Q_{se}\cdot\ell}{e\cdot 2^{m-2}}+\frac{Q_{se}}{|D|}+\frac{Q_{se}}{2^{k_2-1}} + 2Q_{se}Adv^{rsa}(O(t))+\frac{2Q_{se}Q_h}{\phi(n)}$$

□