상호 신분인증 프로토콜의 분석 및 설계

임 채 훈 . 이 필 중

포항공과대학 전자전기공학과

Analysis and Design of Mutual Authentication Schemes from Zero-Knowledge Technique

Chae Hoon Lim . Pil Joong Lee

Dept. Electronic and Electrical Engneering, POSTECH

요약문 본 논문에서는 영지식 기술로부터 얻어진 신분인증 방식을 두 사용자 쌍방에 그대로 적용시켜 얻은 상호 신분인증 프로토콜은 oracle session attack 하에서 결코 안전하지 않음을 보인다. 그 예로서 참고문헌 [1] 에 제시된 프로토콜은 신분인증 과정 뿐만 아니라 키분배 과정 역시 안전하지 않으며 또한 참고문헌 [2] 의 프로토콜 역시 상호 신분인증 과정은 전혀 안전하지 않음을 보인다. 본 논문에서는 이러한 공격이 성공할 수 있었던 몇가지 요인들을 분석하여 안전한 상호 신분인증 프로토콜을 설계하는 체계적인 방법을 제시하고 이를 바탕으로 기존의 신분인증 방식을 키분배가 가능한 안전한 상호 신분인증 방식으로 재구성한 몇가지 설계 예들을 제시하기로 한다.

I. 서 론

컴퓨터 통신의 발달과 함께 전송 혹은 저장중인 정보의 불법 변경이나 비밀 유지 등의 정보보안에 관한 관심이 고조되고 이에 대한 대응책으로 암호 프로토콜에 관한 연구가 활발히 진행되고 있다. 비밀보장 (secrecy) 서비스를 위해서는 관용 암호 알고리즘이 널리 이용되며 이에 필요한 세션키의 분배가 중요한 관건이 된다. 메세지의 무결성 (integrity) 보장이나 부인방지 (nonrepudiation) 기능 등의 보안 서비스에서 근간이 되는 대표적인 공개키 알고리즘으로는 디지

탈 서명법을 들 수 있다. 미국에서는 이미 1977년도에 DES가 연방표준안으로 채택되어 미국 정부는 물론이고 각종 기업체들에서도 널리 이용되어 왔으며 또한 최근에는 해쉬함수 (Secure Hash Standard : SHS) 와 디지탈 서명의 표준안 (Digital Signature Standard : DSS) 이 제안되어 검토중에 있는 등 암호 프로토콜의 실용화는 이미 우리 주위에서 급속도로 진행되고 있다[3] [4] [5].

암호통신의 기본이 되는 가장 중요한 프로토콜의 하나로 신분인증 프로토콜을 들 수 있다. 중요한 건물이나 자원(정보)의 접근통제를 위한 용도로 뿐만 아니라 컴퓨터 통신시의 원하는 상대방과의 정확한 연결을 확인하는 것은 필수적인 만큼 사용자 인증을 위한 용도로도 신분인증 프로토콜은 향후 정보화사회의 컴퓨터망에서 제공될 보안서비스의 기본적인 프로토콜의 하나가 될 것이다. 사람의 신분을 인증하는 방법은 크게 다음의 세가지 방법으로 나눌 수 있다. 즉 그 사람만이 가지고 있는 소유물 (what one only has : possessions) 에 의한 방법, 그 사람만이 알고 있는 지식 (what one only knows : knowledges) 에 의한 방법, 그리고 그 사람 고유의 신체적 특징 (what one only is : physical characteristics) 에 의한 방법 [6] 등이 그것이다. 이들 방법은 각기 나름대로의 장단점을 가지고 있다. 예를들면 소유물에 의한 방법은 분실이나 도난의 위험이 있으며 지식에 의한 방법은 잊어버릴 염려가 있을 뿐더러 사람의 기억에 의존하는 만큼 그 지식이 너무 길거나 기억하기 어려워서는 곤란하다는 문제점이 있다. 또한 신체적 특징을 이용하는 방법은 그 절차가 복잡할 뿐더러 통신망을 통해 전송할 수가 없으므로 원거리 인증 (remote verification) 에는 사용하기 어렵다는 문제점 등이 존재한다. 따라서 신분인증이 사용되는 환경에 따라서 이들 방법을 적절히 조합하여 서로의 단점을 보완할 필요가 있다. 신분인증의 도구로 널리 연구되고 또한 이미 실용화 단계에 있는 대표적인 방법으로 스마트카드 (smart card) 를 이용하는 방법을 들 수 있는데 여기서도 스마트카드라는 소유물과 그 소유자가 기억해야 하는 PIN (Personal Identification Number) 번호, 그리고 카드내에 저장된 그 사용자 고유의 비밀키 등 여러가지가 결합되어서야 제 기능을 할 수가 있는 것이다.

우선 신분인증 프로토콜은 그것이 사용되는 환경에 따라서 그 방법이나 보안상의 요구조건 등에 있어 상당한 차이가 있음을 주목할 필요가 있다. 근접 인증 (local verification) 의 경우는 증명자 (인증받고자 하는 사람) 와 인증자 (혹은 단말기) 가 근접한 상태에서 인증과정이 이루어지는 만큼 소유물이나 지식, 신체적 특징 등을 임의로 결합하여 사용할 수 있으며 또한 인증실패시에는 카드압류 및 경고음 등에 의해 침입자가 현장에서 체포될 수 있으므로 비교적 낮은 안전도로도 충분한 경우가 많다. 그러나 일반적인 컴퓨터망에서의 사용자 원거리 인증시에는 일단 사용 가능한 방법이 그 사용자 고유의 비밀지식을 확인하는 방법 뿐이며 또한 공격자는 반복 시행에 의한 요행을 바라고 인증과정을 되풀이 할 수 있으므로 보다 높은 안전도가 요구

된다. 접근 대상에 따라 다르겠지만 대체로 근접인증의 경우에는 10^{-3} - 10^{-5} 정도, 원거리 인증의 경우에는 10^{-6} - 10^{-9} 정도의 오류율이면 적절하다는 것이 일반적인 견해이다.

근접 인증과 원거리 인증 사이의 차이를 좀더 자세히 살펴보자. 암호학에서 연구되는 대부분의 신분인증 프로토콜은 그 사용자만이 알고 있는 비밀키에 의해 사용자를 인증하는 방법을 택하고 있다. 그러나 근접 인증만을 생각한다면 적절한 가정하에 이러한 비밀키에 의한 인증보다도 훨씬 간단한 신체적 특징을 이용하는 방법이 있을 수 있다[7]. 즉 각 사용자의 신체적 특징은 유일하게 기술될 수 있고 이런 기술자(descriptor)로부터 충분한 정확도를 가지고 각 사용자를 식별하는 것이 가능하며 또한 신뢰할 수 있는 센타가 존재하여 안전한 디지탈 서명을 이용, 각 사용자의 기술자를 디지탈 서명하여 그에게 전해 주는 것이다. 그러면 각 사용자는 자신을 인증해야 할 필요가 있을 때는 단지 이 서명을 상대방에게 제시하면 상대방은 그 서명이 센타의 서명임을 확인하고 또한 그 사용자의 기술자에 기술된 신체적 특징으로부터 그를 확인할 수 있을 것이기 때문이다. 실제로 Sandia National Laboratory 에서는 RSA 서명을 이용하여 이러한 방법으로 신분을 인증하는 방법을 구현하여 원자로 (Zero Power Plutonium Reactor at Idaho Falls) 의 접근통제를 위해 사용하였다는 사실이 보고되었으며 이것이 공개키 알고리즘을 실제로 이용한 최초의 예로 알려져 있다[8][9]. 물론 이 방법이 간단하기는 하지만 자동화하기가 쉽지 않으며 또한 그 안전성이 사용되는 서명에 직결되는 만큼 안전한 (existentially unforgeable under an adaptive chosen-message attack [10]) 서명을 사용해야 하나 지금까지 안전한 것으로 알려진 서명들은 모두 비실용적이라는 문제점을 지적할 수 있다. 더우기 이러한 방법은 증명자와 인증자가 서로 다른 장소에 있는 원거리 인증에는 사용될 수 없음은 명백하므로 일반적인 신분인증 방법으로 생각할 수는 없지만 근접 인증과 원거리 인증의 차이점을 보여 주는 단적인 예라 하겠다.

일반적인 신분인증 프로토콜의 발전에 원동력이 된 것은 영지식 증명 (zero-knowledge proof) 에 관한 GMR 의 논문 [11] 이며 이를 응용하여 신분인증 프로토콜을 개발, 이 분야의 새로운 지평을 연 사람은 Fiat 와 Shamir [12] 였다. 이들이 1986 년 최초의 실용적인 신분인증 프로토콜로서 Fiat-Shamir 방식을 발표한 이래 많은 프로토콜들이 제안되어 왔으며 이들은 크게 영지식 프로토콜 (zero-knowledge protocol) 과 3-move 프로토콜로 나눌 수 있다. 영지식 프로토콜로는 (Feige-) Fiat-Shamir 방식 [12][13], Beth 방식 [14], 그리고 Ohta-Okamoto 방식 [15] 등이 있으며 이들 역시 다음에 설명하는 3-move 프로토콜로 쉽게 바꿀 수 있다. 3-move 프로토콜은 Guillou-Quisquater 방식 [16] 이나 Schnorr 방식 [17] 등과 같이 이들을 반복 실행함으로써 영지식이 되는 프로토콜로 이들외에도 Brickell-McCurley 방식 [18], Okamoto 방식 [19] 등이 있다. 영지식 프로

토콜들의 병렬 실행버전 (반복 횟수를 1로 두고 각 단계의 전송정보를 한꺼번에 주고 받는 방법) 역시 3-move 이나 이들은 원래의 순차적인 반복 실행시와 같은 계산량 및 통신량을 필요로 하는 반면 영지식의 성질을 잃게 되므로 이들은 3-move 프로토콜에 포함시키지 않기로 한다.

3-move 프로토콜은 통신량이나 계산량에 있어서 영지식 프로토콜에 비해 훨씬 효율적일 뿐만 아니라 쉽게 디지탈 서명으로 바꿀 수 있다는 잇점이 있다. 또한 Feige-Fiat-Shamir 방식, Ohta-Okamoto 방식, Brickell-McCurley 방식, 그리고 Okamoto 방식 등은 3-move 프로토콜로서 안전성이 입증된 방식들이며 이는 곧 이들 프로토콜에서 비록 증명자의 비밀키에 대한 부분적인 정보가 누출된다 하더라도 이들 정보는 공격자가 증명자를 사칭하는 데는 아무런 도움이 되지 않는다는 것을 의미하므로 실제적인 의미에서 안전성에 있어서도 문제가 전혀 없다고 할 수 있다. 따라서 통신량, 계산량, 안전성 등 모든 면들을 고려할때 3-move 프로토콜이 보다 효율적이고 실용적인 프로토콜이라 할 수 있다.

본 논문에서는 기존의 3-move 신분인증 프로토콜들로부터 안전한 3-move 상호 신분인증 방식 (3-move mutual authentication scheme) 을 설계하는 체계적인 방법을 제시한다. 특히 기존의 신분인증 프로토콜을 두 사용자 쌍방에 그대로 적용함으로써 얻어지는 상호 신분인증 프로토콜은 결코 안전하지 않음을 보이고 그 예로 기존에 제시된 두 가지 프로토콜 [1] [2] 에 대한 공격법을 제시한다. 다음으로 이와같은 공격법이 성공적으로 적용될 수 있었던 요인들을 분석하여 이를 바탕으로 기존의 영지식 기술로부터 생성된 3-move 신분인증 방식을 안전한 3-move 상호 신분인증 방식으로 변형시켜 보기로 한다. 또한 이 신분인증 과정에서 교환되는 랜덤정보들을 이용하면 후속되는 비밀통신을 위해 필요한 세션키를 분배할 수 있음도 보인다.

II. 신분인증 프로토콜에 대한 공격법

서론에서도 언급했듯이 신분인증 프로토콜이 사용되는 환경은 크게 근접 인증 (local verification) 과 원거리 인증 (remote verification) 으로 나눌 수 있다. 근접 인증의 경우는 사용자와 건물의 입구 등에 설치된 단말기와의 신분확인 과정에서처럼 대부분 단말기가 사용자를 확인하는 일방적인 프로토콜로도 충분할 것이나 원거리에서의 터미날 접속이나 상대방 확인 등에서는 서로간의 상호 신분인증이 요구되는 것은 당연하다 [20]. 컴퓨터망이 일반화되기 이전에는 컴퓨터에 대한 접근통제용으로 패스워드와 같은 일방향의 신분인증 방식만으로도 충분하였으나 이제는 컴퓨터망의 급속한 확장과 복잡한 연결 등으로 사용자나 컴퓨터 모두가 상호 신분인증을 필요로 한다. 예를들어 원거리에서 주 컴퓨터의 중요한 정보를 꺼내 보거나 새로운 정보를 입력하려고 하는 경우 주 컴퓨터는 단말기을 이용하는 사용자의 신분을 확인하고 해당 정보에

대한 접근권한이 있는 지의 여부를 확인해야 할 것이며 또한 사용자 역시 접속된 컴퓨터가 자신이 연결하고자 하는 컴퓨터인지를 확인하는 것은 필수적이다. 특히 컴퓨터망을 통해 비밀리에 중요한 거래를 하고자 하는 두 사용자의 경우는 서로간의 신분을 확인함과 동시에 이후의 비밀통신에 사용될 세션키를 분배할 수 있는 프로토콜을 사용하는 것이 보다 경제적일 것이다.

서론에서도 언급했듯이 참고문헌 [15] [18] [19] 의 3-move 신분인증 방식 들은 안전성이 증명된 방식들이다. 여기서 신분인증 프로토콜이 안전하다고 함은 대략적으로 말해서 공격자가 정직한 증명자 A 와의 (polynomially many times 만큼) 신분인증 프로토콜을 시행하여 그 과정에서 얻어낸 어떤 정보를 이용하더라도 그가 제 3 자에게 자신을 증명자 A로 사칭할 수 없다는 것을 의미한다 [13] [19]. 따라서 안전성이 증명되었다 함은 공격자가 어떤 방법으로 이들 프로토콜에 관여하던 간에 그 과정에서 얻어낸 정보들로 그가 증명자를 사칭는 것은 불가능하다는 것이 입증된 것이므로 이 신분인증 프로토콜이 어떤 경우에도 안전할 것으로 생각되지만 이것은 잘못된 생각이다. 즉 공격자가 증명자를 사칭하는데 필요한 정보을 얻기위해 증명자를 real-time oracle 로 이용하는 공격하에서는 이러한 안전성 증명은 아무런 효과가 없기 때문이다. 공격자가 사용자 B 에게 자신을 사용자 A 로 가장하고자 할 때 프로토콜의 안전성에 의하면 그가 사용자 A 의 비밀키를 모르는 한 결코 사용자 B 의 질문에 제대로 응답할 수 없으므로 이때 공격자는 사용자 A 와 또 다른 세션을 시작하여 그가 필요로 하는 대답을 사용자 A 와의 세션에서 얻어내는 것이다.

이러한 oracle session attack 의 가장 간단한 예가 한 사용자의 질의-응답을 단순히 중간에서 중계해 주는 경우로 Desmedt 등이 언급한 mafia fraud (혹은 middle person attack) 나 terrorist fraud 등이 여기에 해당한다 [21][7]. 우선 mafia fraud 를 간단히 살펴보자. 이 공격이 mafia fraud 라는 이름이 붙은 것은 두명의 마피아 단원들이 결탁하여 한 사용자를 희생양으로 삼아 중간에서 막대한 이득을 남길 수 있다는 시나리오에서 기인한다. 여기에는 4 명의 사용자가 관련되며 무대는 마피아 소유의 레스토랑과 제 3 의 장소인 보석상이다. 즉 마피아 소유의 레스토랑 손님인 사용자 A, 레스토랑 주인인 사용자 B, 그와 결탁하여 보석상에서 보석상 주인을 속이려는 마피아 단원의 하나인 사용자 C, 그리고 보석상 주인인 사용자 D 가 관련되어 있으며 사용자 B 와 C 는 서로 안전한 무선링크 (radio link) 로 연결되어 비밀리에 대화를 주고 받을 수 있으나 사용자 A 와 D 는 이를 모른다. 그리고 각 사용자는 신분인증 프로토콜이 구현된 스마트카드를 이용하여 신분을 인증하고 금액을 결재한다. 이제 레스토랑의 손님인 사용자 A 가 사용자 B 에게 음식값을 지불하려고 자신의 스마트카드에 자신만의 PIN 번호를 입력시켜 활성화시킨 후

사용자 B 의 단말기 (interrogating device) 에 넣는 순간 사용자 B 는 사용자 C 에게 무선링크로 연락을 취한다. 그러면 사용자 C 는 보석상에서 적당히 값나가는 보석을 집어 주인에게 대금을 지불하기 위해 마찬가지로 자신의 스마트카드를 사용자 D 의 단말기에 넣는다. 이제 사용자 C (의 스마트카드) 는 무선링크를 통하여 사용자 D 로부터의 질문을 사용자 B 에게로 중계하고 사용자 B 는 마찬가지로 이를 자신의 질문인양 사용자 A 에게 중계하며 그 응답 (사용자 C 가 사용자 A 로 가장하는데 필요한 응답) 을 그대로 사용자 C 에게 중계한다. 결국 인증과정은 사용자 A 와 D 사이에서 일어나며 사용자 B 와 C 는 중간에서 이들 사이의 전송정보들을 서로에게 그대로 중계함으로써 자연 보석값은 아무것도 모르는 사용자 A 가 지불하는 격이 된다

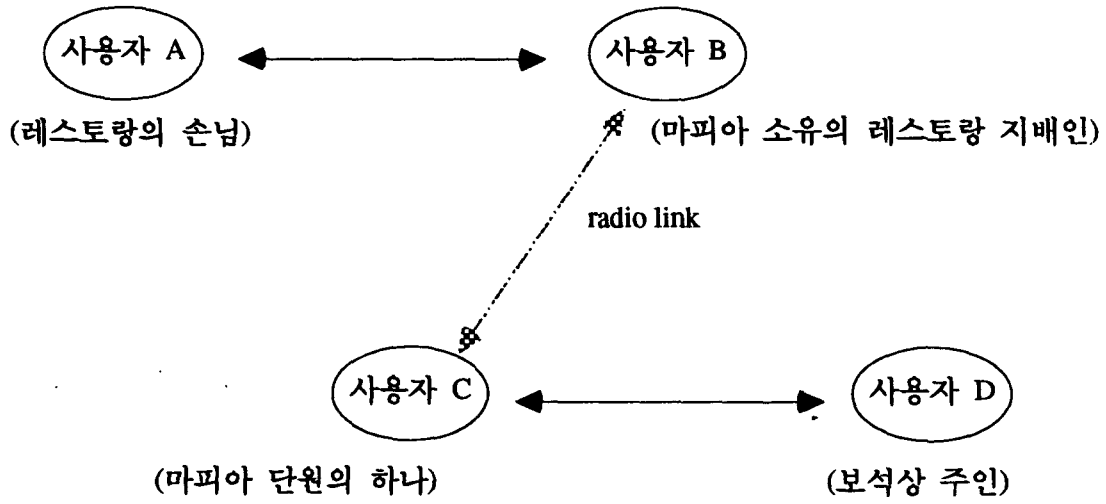

그림 1. mafia fraud 시나리오의 구성도

이 공격의 한 변형으로 보다 극단적인 예는 신분인증 프로토콜이 비자 프로토콜로 이용되는 경우로 terrorist fraud 가 바로 그것이다. 여기에는 세명의 사용자가 관련된다. 즉 어떤 나라 (α-land) 에 입국금지자 명단에 올라있는 테러리스트 (사용자 B) 가 α-land 에 입국하기 위해 세관 (사용자 C) 에서 신원조회를 받으려 하고 α-land 내에는 테러리스트의 일행인 합법적인 시민인 사용자 A 가 있어 역시 무선링크를 통해 테러리스트가 자신으로 가장해 입국하도록 도와주려고 한다. 그러면 사용자 B 와 C 사이의 모든 질의-응답은 사실은 무선링크를 통해 합법적인 시민인 사용자 A 와 C 사이의 대화가 될 것이므로 테러리스트는 무사히 α-land 로 입국하게 될 것이다. 이 공격법도 위의 mafia fraud 에서와 유사하나 mafia fraud 에서는 인증자가 제 3 자를 증명자로 사칭할 수 있도록 증명자와의 모든 대화를 중간에서 중계해 주는 경우이고 (증명자는 이용당함) terrorist fraud 에서는 증명자가 고의로 그의 ID 를 빌려주어 제 3 자가 그를 사칭할 수

있도록 도와 주는 경우이므로 약간의 차이가 있다. 결국은 모두가 증명자를 사칭하는데 필요한 정보를 얻기 위해 그를 real-time oracle 로 이용하는 경우이나 전자는 증명자를 속이고 그를 사칭하는 경우이며 후자는 증명자가 스스로 oracle 의 역할을 떠맡는 경우라 할 수 있다.

위와같은 위협들은 신분인증 프로토콜이 실제로 스마트카드로 구현되었을 때 매우 치명적인 결과를 초래할 수 있으므로 이를 해결하는 방법들이 반드시 강구되어야 할 것이다. 우선 이러한 공격들이 가능한 것은 스마트카드가 단말기와 대화하는 동안 이들이 외부와 통신을 할 수 있다는 사실에서 출발하므로 가장 간단히 생각할 수 있는 해결법은 스마트카드와 단말기를 인증기간 동안 외부와 완전히 격리시키는 것이다 [7] [22]. 그러나 이는 프로토콜상으로 해결하는 방법이 아니며 이를 위해 보다 복잡한 격리시설이 필요하므로 특수한 경우를 제외하고는 일반적인 해결법으로는 적합치 않은 것으로 보인다. 다음으로 제시된 해결법이 두 사용자 사이에 일정한 시간간격 t 를 미리 정해 두고 각 사용자는 상대방으로부터 응답을 받은 후 자신의 local clock 상으로 정확히 t 초만에 자신의 응답을 전송하는 방법이다 [23]. 이는 게임이론에서 chess grandmaster problem 을 해결하는 방법을 신분인증 프로토콜의 중계에 의한 공격을 막는 방법에 응용한 것으로 보통의 수정시계기술의 정확도만으로도 이들 공격을 쉽게 검출해 낼 수 있는 것으로 알려져 있다.

위에서 살펴본 mafia fraud 나 terrorist fraud 는 한 사용자와의 세션 전체를 하나의 oracle 로 이용하는 oracle session attack 의 일종으로 생각할 수 있으며 이들은 일반적인 컴퓨터 통신에서도 그대로 적용될 수 있으므로 상호 신분인증 프로토콜에서도 이들을 막는 방법이 필요할 것이다. 여기서 주목할 것은 위에서 언급한 mafia fraud 나 terrorist fraud 를 막는 방법들이 모두 스마트카드를 이용한 근접 인증의 경우에만 가능하다는 것이다. 일반적인 컴퓨터망을 통한 원거리 인증의 경우 단말기를 외부와 격리시킨다는 것은 어불성설이며 Beth-Desmedt 의 방법 역시 상대방의 위치를 확인할 수 없으며 또한 통신지연이 일정치 않은 유선 통신망에서는 아무런 의미가 없기 때문이다. 이처럼 일반적인 원거리 인증의 경우에는 이러한 공격들을 검출하는 것은 쉽지 않을 것으로 생각된다.

그러나 II 장에서 제시하겠지만 단말기와 단말기 사이의 일반적인 통신환경 하에서도 증명자가 자신이 원하는 상대방과만 통신을 시작한다는 가정하에서는 (즉 상대방이 전송한 ID 및 공개키 증명서를 확인하여 그 ID 가 자신이 원하는 사용자의 ID 일때만 프로토콜을 시작하며 다른 사용자의 요청에 의해서는 절대로 먼저 프로토콜을 시작하지 않는다는 가정) mafia fraud 는 쉽게 막을 수 있다. 그러나 증명자가 기꺼이 제 3 자로 하여금 자신을 사칭할 수 있도록 모든 도움을 제공하는 terrorist fraud 유형의 공격은 일반적인 원거리 인증에서는 항상 가능하므로 이

를 인증 프로토콜에서 공격의 한 유형으로 분류하는 것은 의미가 없을 것으로 보인다. 즉 이는 사용자 A가 사용자 B와의 인증과정을 거친 후 사용자 C에게 자신의 신분으로 통신을 할 수 있도록 자리를 비켜주는 것과 다를 바가 없기 때문이다.

다음에는 oracle session attack의 다른 한 예로 증명자를 사칭하는 데 필요한 일부의 정보만을 그 증명자와의 세션으로부터 얻어내는 경우를 예로 들어본다. 많은 학자들에 의해 지적되었듯이 [24]-[26] 1988년 버전 CCITT X.509의 Authentication Framework [27]에서 제시된 3-way strong authentication scheme (공개키 암호법을 이용한 상호 신분인증 방식)은 이러한 oracle session attack하에서 안전하지 않음을 쉽게 보일 수 있다 (이러한 결함은 그후 ISO의 표준안 [28] (DIS 버전)에서는 교정되었으나 이 ISO의 신분인증 프로토콜 중 관용 암호법을 이용한 3-way authentication scheme (DP 버전) [29]은 역시 oracle session attack에 의해 쉽게 깨어진다 [30]). X.509의 3-way strong authentication scheme은 간단히 다음과 같이 구성된다 (표기를 간단히 하기 위해 원래의 방식에서 신분인증에 꼭 필요한 필드만을 표시했음). 여기서 기호 $A\{X\}$는 메세지 X에 대한 사용자 A의 서명으로 메세지 X와 X의 해쉬값에 대한 A의 서명을 연결시킨 전체를 가리킨다.

i) 사용자 A는 랜덤수 R_A를 선택하여 상대방의 이름 B와 함께 자신의 비밀키로 서명한 $A\{R_A, B\}$를 사용자 B에게 전송한다.

ii) 사용자 B는 사용자 A로부터 받은 $A\{R_A, B\}$로부터 A의 공개키로 이 서명을 인증하여 무결성을 확인하고 그것이 사용자 A로부터 자신에게로 온 것임을 확인한다. 사용자 B는 랜덤수 R_B를 선택하여 $B\{R_B, A, R_A\}$를 사용자 A에게 전송한다.

iii) 사용자 A는 B의 공개키로 서명 $B\{R_B, A, R_A\}$를 인증하여 무결성을 확인하고 그것이 사용자 B로부터 자신에게로 온 것이며 랜덤수 R_A가 i) 단계에서 자신이 보낸 랜덤수와 동일한지를 조사하여 과거 전송정보의 재전송이 아님을 확인한다. 조건이 만족되면 $A\{R_B\}$를 사용자 B에게 전송한다.

iv) 사용자 B는 서명 $A\{R_B\}$를 확인하고 랜덤수 R_B가 자신이 ii) 단계에서 보낸 랜덤수와 동일한지를 조사한다.

이 프로토콜에 대한 oracle session attack은 그림 2와 같이 진행된다. 여기서 공격자인 사용자 C는 사용자 B에게 자신을 사용자 A로 가장하려고 하며 또한 공격자는 과거에 사용자 A와 B사이의 통신을 관측하여 A가 B에게 보낸 메세지 $A\{R_A', B\}$를 가지고 있다고 가정한다. 이제 사용자 C는 사용자 A로 하여금 자신에게 통신을 시작하도록 하거나 혹은 사용자 A가 자신에게 통신을 시작하려고 할 때 동시에 자신은 사용자 B와의 통신을 시작한다.

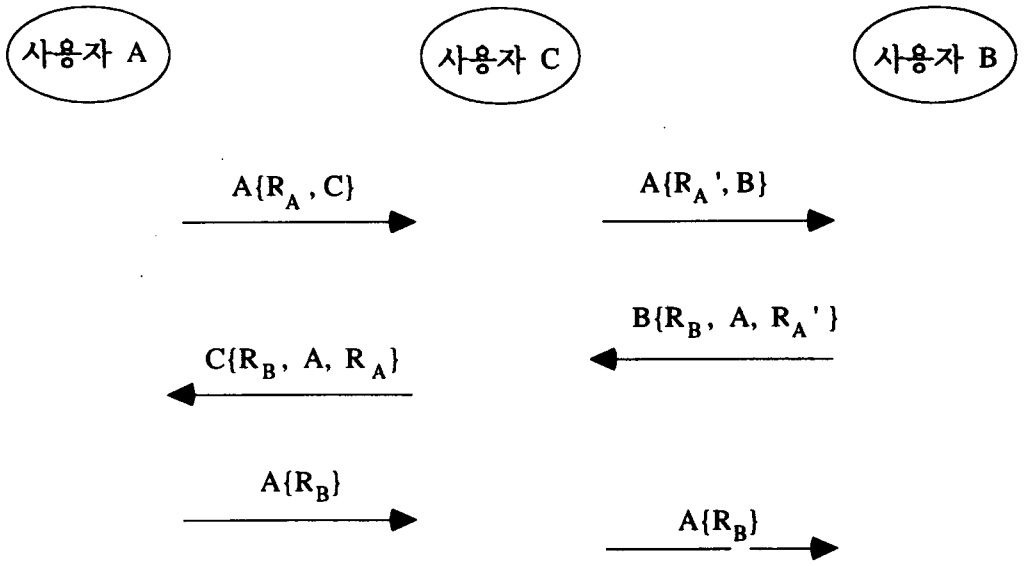

그림 2. X.509 의 3-way authentication scheme 에 대한 oracle session attack

위의 공격에서 사용자 C 가 사용자 B 에게 자신을 사용자 A 로 가장하기 위해서는 사용자 B 로부터의 전송 $B\{R_B, A, R_A'\}$ 에 대한 응답으로 R_B 에 대한 사용자 A 의 서명이 필요하나 그는 이 서명을 스스로 생성할 수 없으므로 사용자 A 와의 새로운 세션을 시작한다. 이 세션에서 사용자 A 로부터의 전송 $A\{R_A, C\}$ 에 대한 응답으로 자신의 랜덤수를 사용자 B 로부터 받은 R_B 로 두면 사용자 A 는 이 랜덤수에 대한 그의 서명 $A\{R_B\}$ 를 전송할 것이며 이것이 그가 사용자 B 에게 자신을 사용자 A 로 가장하는데 필요한 마지막 응답이므로 그는 무사히 사용자 B 를 속일 수 있게 된다. 이러한 공격을 막는 방법은 매우 간단하다. 즉 프로토콜의 마지막 전송에서도 상대방의 이름을 서명에 포함시키는 것이다. 위의 공격에서 만일 사용자 A 가 $A\{R_B\}$ 대신에 $A\{R_B, C\}$ 를 전송한다면 이는 공격자가 사용자 B 에게 전송해야 할 $A\{R_B, B\}$ 와는 다르므로 공격자에게 아무런 소용이 없게 되기 때문이다.

이상에서 공격자가 특정 사용자를 사칭하기 위해 그 사용자와의 real-time oracle session 을 통해 그가 필요로 하는 전송정보를 얻어내는 oracle session attack 에 대해 살펴보았다. 안전한 신분인증 프로토콜이라 하더라도 이러한 real-time oracle session attack 에 대해서는 결코 안전할 수 없으며 특히 이 공격은 상호 신분인증 프로토콜의 경우에 매우 쉽게 적용되는 경우가 많으므로 프로토콜의 설계시에 이를 막도록 세심한 주의를 기울여야 한다.

III. 기존방식의 분석 및 대응방안

상호 신분인증 프로토콜의 가장 간단한 예는 두 사용자 쌍방에 기존의 신분인증 프로토콜을 그대로 적용하여 병렬화하는 것이 되겠지만 이 장에서는 이러한 프로토콜은 결코 안전하지 않음을 보이기로 한다. 그 예로서 참고문헌 [1] 과 [2] 에 제시된 프로토콜을 oracle session attack 으로 분석하여 이들이 전혀 안전하지 않음을 보인다. 제시되는 공격법은 임의의 신분인증 프로토콜을 상호 신분인증 프로토콜로 변형시켰을 때도 쉽게 적용될 수 있으므로 이러한 공격이 성공하게 된 원인을 분석해 보고 이에 대한 대응방안들을 제시하여 IV 장에서 제시하는 안전한 상호 신분인증 프로토콜을 설계하는 기본 지침으로 삼고자 한다..

참고문헌 [1] 에 제시된 프로토콜은 Fiat-Shamir 의 영지식 프로토콜 [12] 을 두 사용자 쌍방에 그대로 적용한 신분인증 과정과 그후의 세션키 분배를 위한 여분의 전송으로 구성된다. 이 프로토콜은 아래에 제시되는 것처럼 신분인증 과정뿐만 아니라 키분배 과정까지도 완전한 공격이 가능하다. 우선 시스템 준비 (system setup) 및 사용자 등록단계에서 신뢰성있는 센타는 RSA 형법 N 을 선택, 공개하고 시스템에 가입하고자 하는 각 사용자 A 에게 그의 신원을 확인한 후 비밀키로 $S_{Aj}=SQRT(V_{Aj}^{-1})$, $V_{Aj}=f(I_A, j)$, $j = 1, \ldots, k$ 를 계산하여 비밀리에 전해 준다. 여기서 f 는 일방함수이며 임의의 수에 대해 합성수 N 을 법으로 하는 제곱근이 항상 존재하는 것은 아니나 편의상 k 개의 제곱근을 $S_{Aj}, j = 1, \ldots, k$ 로 두기로 한다. 참고문헌 [1] 의 프로토콜은 다음과 같다. 프로토콜은 완전 대칭이므로 한 사용자 A 에 대해서만 기술한다.

i) 사용자 A 는 B 의 ID 로부터 $V_{Bi}=f(I_B, i)$, $i = 1, \ldots, k$ 를 계산한다. 그리고 랜덤수 $R_A \in Z_N$ 을 선택하여 $X_A \equiv R_A^2 \bmod N$ 을 계산하여 사용자 B 에게 전송한다.

ii) 사용자 A 는 k 개의 랜덤 비트 $E_{Bi}, i = 1, \ldots, k$ 를 사용자 B 에게 전송한다.

iii) 사용자 A 는 $Y_A \equiv R_A \prod_{E_{Ai}=1} S_{Ai} \bmod N$ 을 계산, 사용자 B 에게 전송한다.

iv) 사용자 A 는 사용자 B 로부터 받은 X_B, Y_B 를 이용, $X_B \equiv Y_B^2 \prod_{E_{Bi}=1} V_{Bi} \bmod N$ 이 성립하는지를 검사한다. 만일 이 조건이 만족되지 않으면 프로토콜을 중단한다.

v) 사용자 A 는 랜덤수 $Z_A \in Z_N$ 을 선택하여 $T_A \equiv (R_A X_B Y_B^{-1} + S_A) Z_A \bmod N$ 을 계산, 사용자 B 에게 전송한다. 여기서 S_A 는 $S_A \equiv \prod_{E_{Ai}=1} S_{Ai} \bmod N$ 을 나타낸다.

vi) 사용자 A 는 사용자 B 와의 세션키로 $K_A \equiv T_B S_A Z_A \bmod N$ 을 계산한다.

이제 위의 프로토콜에 대한 oracle session attack 을 제시한다 (그림 3 참조). 여기서 공격자인 사용자 C 는 사용자 B 에게 자신을 사용자 A 로 증명하려고 하며 그는 동시에 사용자 A, B 와의 프로토콜을 시작한다. 사용자 A 와의 프로토콜에서는 사용자 A 가 먼저 전송을 시작하기를

기다리며 사용자 B 에게로의 전송은 사용자 A 로부터의 전송을 받은 후 바로 시작한다. 각 사용자들 사이에 ID 를 교환하여 V 값들을 계산하는 사전단계는 생략하기로 한다. 기호 $\equiv_N$ 는 mod N 으로 계산됨을 의미한다.

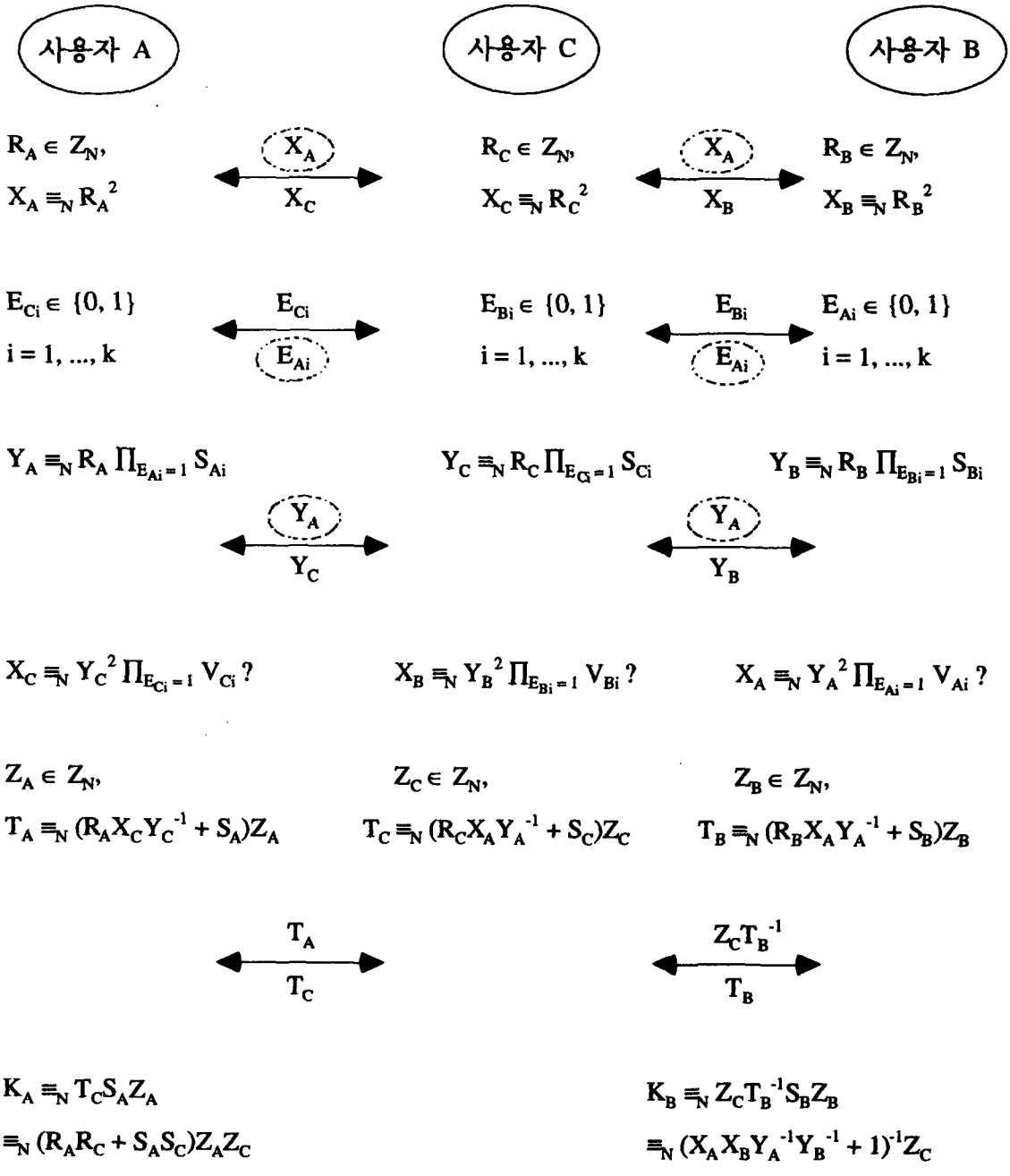

그림 3. 참고문헌 [1] 의 프로토콜에 대한 공격

우선 인증과정에서의 공격을 간략히 살펴보자. 우선 사용자 C 가 사용자 A 혹은 사용자 B

로부터의 전송을 그대로 중계해 주는 부분(점선으로 동그라미 친 부분)을 주의해서 보면 이들은 모두 사용자 C가 자신을 사용자 A로 가장하는데 필요한 최종 응답인 Y_A를 얻기 위한 것임을 알 수 있다. 즉 사용자 A로부터의 최종 응답인 Y_A가 사용자 B에게 자신을 사용자 A로 가장할 수 있는 응답이 되도록 하기 위해 공격자는 1단계에서 사용자 A로부터의 witness를 그대로 사용자 B에게 중계하고 또한 2단계에서는 사용자 B로부터의 challenge를 사용자 A에게 그대로 중계한 것이다. 결국 사용자 B가 사용자 C로부터 받는 witness와 response는 모두 사용자 A로부터 온 것이므로 사용자 A가 프로토콜을 따르는 한 사용자 B는 항상 상대방을 사용자 A로 인증하게 된다. 사용자 C 역시 사용자 B가 계산하는 것과 동일한 인증조건 검사를 하여 사용자 A를 인증할 것이므로 이 조건검사가 통과된다는 것은 곧 그가 사용자 B에게 자신을 사용자 A로 성공적으로 가장하였음을 의미하게 된다.

다음에는 키분배 과정에서의 공격을 살펴보자. 사용자 C가 인증과정에서 사용자 B를 성공적으로 속였다고 해서 반드시 키분배 과정에서도 그와 세션키를 공유하게 되는 것은 아니지만 위의 프로토콜에서는 이것이 가능하다. 즉 사용자 C는 사용자 B로부터 T_B를 받은 후 이에 대한 역원을 사용자 B에게 전송하면 사용자 B가 계산하게 되는 세션키가 공개정보만으로 계산될 수 있는 숫자가 되기 때문이다. 여기서 T_B^{-1} 대신에 $Z_C T_B^{-1}$를 전송한 것은 단지 사용자 B가 상대방으로부터 받은 것이 T_B의 역원인지를 모르도록 이를 랜덤화시킨 것에 불과하다. 결국 사용자 B가 사용자 A와의 세션키로 알고 계산하는 것은 다음과 같이 사용자 C도 계산할 수 있는 값이 된다. 즉 $K_B \equiv Z_C T_B^{-1} S_B Z_B \equiv (R_B X_A Y_A^{-1} + S_B)^{-1} Z_B^{-1} Z_B S_B Z_C \equiv (R_B X_A Y_A^{-1} S_B^{-1} + 1)^{-1} Z_C \bmod N$이고 마지막 식에서 $R_B S_B^{-1} \equiv X_B Y_B^{-1} \bmod N$이 성립하므로 이 K_B는 공개정보와 사용자 C가 선택한 랜덤수 Z_C만으로 구성된다. 물론 사용자 C는 사용자 A와는 정상적으로 프로토콜을 따랐으므로 그와의 신분인증 및 키분배 과정은 사용자 A가 프로토콜을 따르는 한 성공적으로 이루어질 것이다.

위와같은 공격은 어떤 신분인증 프로토콜을 이용하여 상호 신분인증 프로토콜을 구성하더라도 그들이 두 사용자 쌍방에 대칭적으로 적용되는 한 항상 가능할 것이라는 것은 쉽게 알 수 있다. 위와 동일한 방법으로 참고문헌 [2]의 프로토콜에 대해 이 공격을 적용해 보자. 이 프로토콜은 Beth의 영지식 프로토콜 [14] 에서 $m = 1, h = 1$인 3-move 버전을 두 사용자 쌍방에 적용시키고 마지막 단계에서 세션키 공유를 위해 한번씩의 전송을 더 하며 또한 이때 생성된 세션키를 인증하기 위해 또 한번의 전송이 필요하게 된다. 우선 Beth의 방식을 간략히 살펴보자. Beth 방식의 시스템 준비단계는 센타(SKIA : Secure Key Issuing Authority)에서 사용자의 개

인정보에 대한 ElGamal 의 서명을 생성하여 이를 그 사용자의 비밀키로 분배한다. 즉 SKIA 는 유한체 (finite field) GF(q) (q 는 소수이거나 소수의 멱승) 와 이 유한체상의 원시원소 α, m 개의 비밀키 $x_1, \ldots, x_m$, 이에 대응하는 공개키 $y_i = \alpha^{x_i}$ (GF(q) 상에서의 연산), $i = 1, \ldots, m$, 그리고 일방함수 f 를 선택하여 자신의 비밀키를 제외한 모든 값들을 공개한다. 이제 SKIA 는 시스템에 가입하고자 하는 각 사용자 A 에 대해 그의 신원을 확인한 후 그의 개인정보 $name_A$ 로부터 일방함수에 의해 $ID_{Ai} = f(name_A, i)$, $i = 1, \ldots, m$ 을 계산하고 이에 대한 ElGamal 의 서명을 생성한다. 즉 랜덤수 $k_A \in Z_{q-1}$ 를 선택, $r_A = \alpha^{k_A}$ 를 계산하고 $x_i r_A + k_A s_{Ai} \equiv ID_{Ai} \bmod q-1$ 를 만족하는 m 개의 s_{Ai} 를 구해 r_A 와 s_{Ai} 들을 저장한 카드를 각 사용자 A 에게 발행한다.

사용자 A 가 사용자 B 에게 신원을 인증하는 과정은 다음과 같다. 우선 프로토콜의 시작단계에서 사용자 A 는 $name_A$ 와 r_A 를 사용자 B 에게 전송하고 사용자 B 는 $name_A$ 로부터 ID_{Ai} 를 계산한다. 그리고 다음의 프로토콜을 h 회 ($i = 1, \ldots, h$) 반복하여 모든 과정을 통과하면 사용자 B 는 상대방이 사용자 A 임을 믿는다.

i) 사용자 A 는 랜덤수 $t_{Ai} \in Z_{q-1}$ 를 선택, $z_{Ai} = r_A^{-t_{Ai}}$ 를 계산하여 사용자 B 에게 전송한다.

ii) 사용자 B 는 랜덤수 $b_{Aij} \in Z_{q-1}$, $j = 1, \ldots, m$ 를 선택, 사용자 A 에게 전송한다 (Beth 의 원래 방식에서는 b_{Aij} 를 증명상의 이유로 Z_{q-1} 보다 훨씬 작은 스페이스에서 선택하였으나 실제 구현상 Z_{q-1} 상에서 선택하는 것이 바람직하며 이렇게 하더라도 안전도가 낮아지지는 않을 것으로 생각된다.)

iii) 사용자 A 는 $u_{Ai} \equiv t_{Ai} + \sum b_{Aij} s_{Aj} \bmod q-1$ 을 계산, 사용자 B 에게 전송한다.

iv) 사용자 B 는 $z_{Ai} r_A^{u_{Ai}} \prod y_j^{r_A b_{Aij}} = \alpha^{v_i}$, $v_i \equiv \sum b_{Aij} ID_{Aj} \bmod q-1$ 가 성립하는 지를 조사한다. 이 조건이 만족되지 않으면 프로토콜을 중단한다.

Bauspieβ-Knobloch 의 프로토콜은 Beth 방식에서 m=1 (즉 SKIA 및 각 사용자 A 의 비밀키는 하나씩임) 이고 반복횟수 h = 1 인 3-move 프로토콜을 두 사용자 쌍방에 적용한 상호 신분인증 과정과 이를 통과했을 때의 키분배 및 확인과정으로 구성된다. 상호 신분인증이 끝난 후의 키분배 및 확인과정은 그림 4와 같다. 키분배 과정에서 z_A, z_B 는 신분인증 과정에서 상대방이 전송한 것임이 확인된 수이지만 e_A, e_B 는 신분인증 후에 교환한 랜덤수이므로 이것이 상대방으로부터 전송된 수라는 보장이 없다. 즉 사용자 A 는 c_{A1} 이 사용자 B 와의 공유키라는 것은 확신할 수 있지만 c_{A2} 대해서는 그렇지가 못하다. 그리고 세션키의 계산이 대칭적이지 못하여 사용자 A 가 z_B 를 이용하여 계산한 값은 사용자 B 가 e_A 를 이용하여 계산한 값과 같게 되므로 이 한번의 전송만으로는 두 사용자가 인증된 세션키를 공유할 수 없다. 그래서 마지막 단계와 같은 세션키 확인과정이 필요한 것이다. 자세한 내용은 참고문헌 [2] 를 참조하기 바란다.

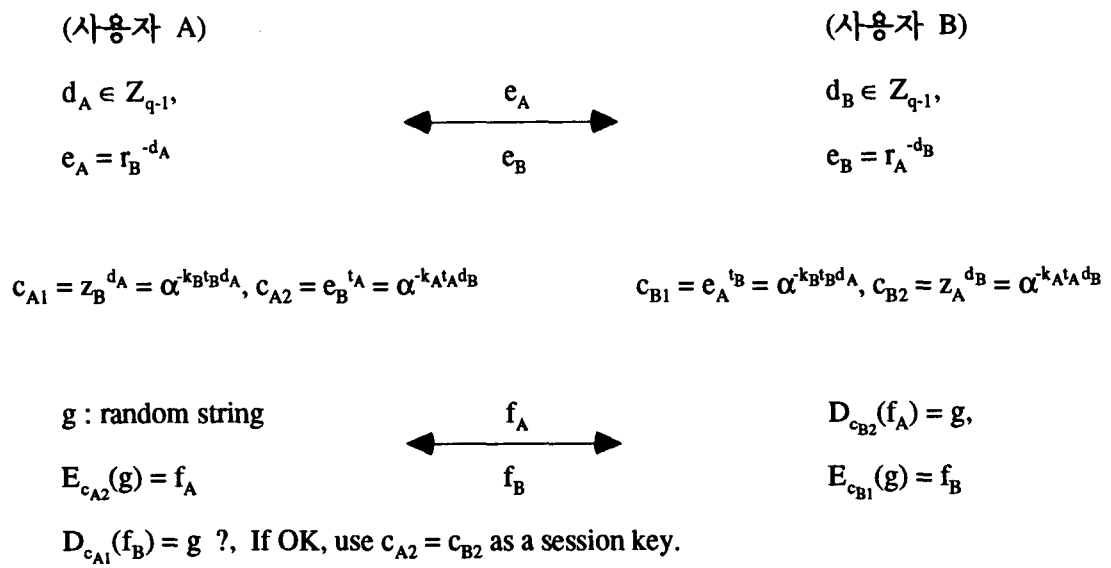

그림 4. Bauspieβ-Knobloch 의 프로토콜의 키분배 과정

참고문헌 [2] 에서 Bauspieβ-Knobloch 가 제안한 프로토콜에 대한 공격은 그림 5 에 도시된 것과 같이 진행된다. 이 공격에서도 앞의 경우와 마찬가지로 공격자인 사용자 C 가 자신을 사용자 B 에게 사용자 A 로 가장하기 위해 두 사용자 A, B 와 동시에 프로토콜을 시작하며 각 사용자들은 프로토콜의 시작단계에서 자신들의 $name_i$ 및 r_i 를 서로 교환하여 상대방의 ID_{ij} 들을 계산하였다고 가정한다. 첫 3 단계의 인증과정은 앞의 공격법에서와 마찬가지로 공격자가 사용자 B 를 속이는데 필요한 정보를 얻기 위해 A, B 와의 대화에서 필요한 부분들을 각각 양쪽으로 중계하여 (점선 동그라미 친 부분) 결국 사용자 B 의 인증조건 검사를 통과하게 된다. 또한 그림에서 알 수 있듯이 공격자는 세션키의 확인과정까지 무사히 통과할 수가 있다. 그러나 참고문헌 [1] 에 대한 공격에서와는 달리 여기서는 최종적으로 사용자 B 와의 공유키를 얻는 것은 불가능하다. 이는 세션키 계산이 Diffie-Hellman 의 프로토콜을 바탕으로 하기 때문이다. 그러나 이러한 복잡한 과정을 거쳐 세션키를 분배하는 것이 상대방을 확인하고 인증된 세션키를 얻기 위한 것임을 고려할 때 전체 프로토콜의 인증과정이 안전하지 않다는 것은 결국 이 프로토콜이 1 라운드의 Diffie-Hellman 프로토콜과 다를바가 없으므로 신분인증 프로토콜을 결합한 의미가 전혀 없게 된다.

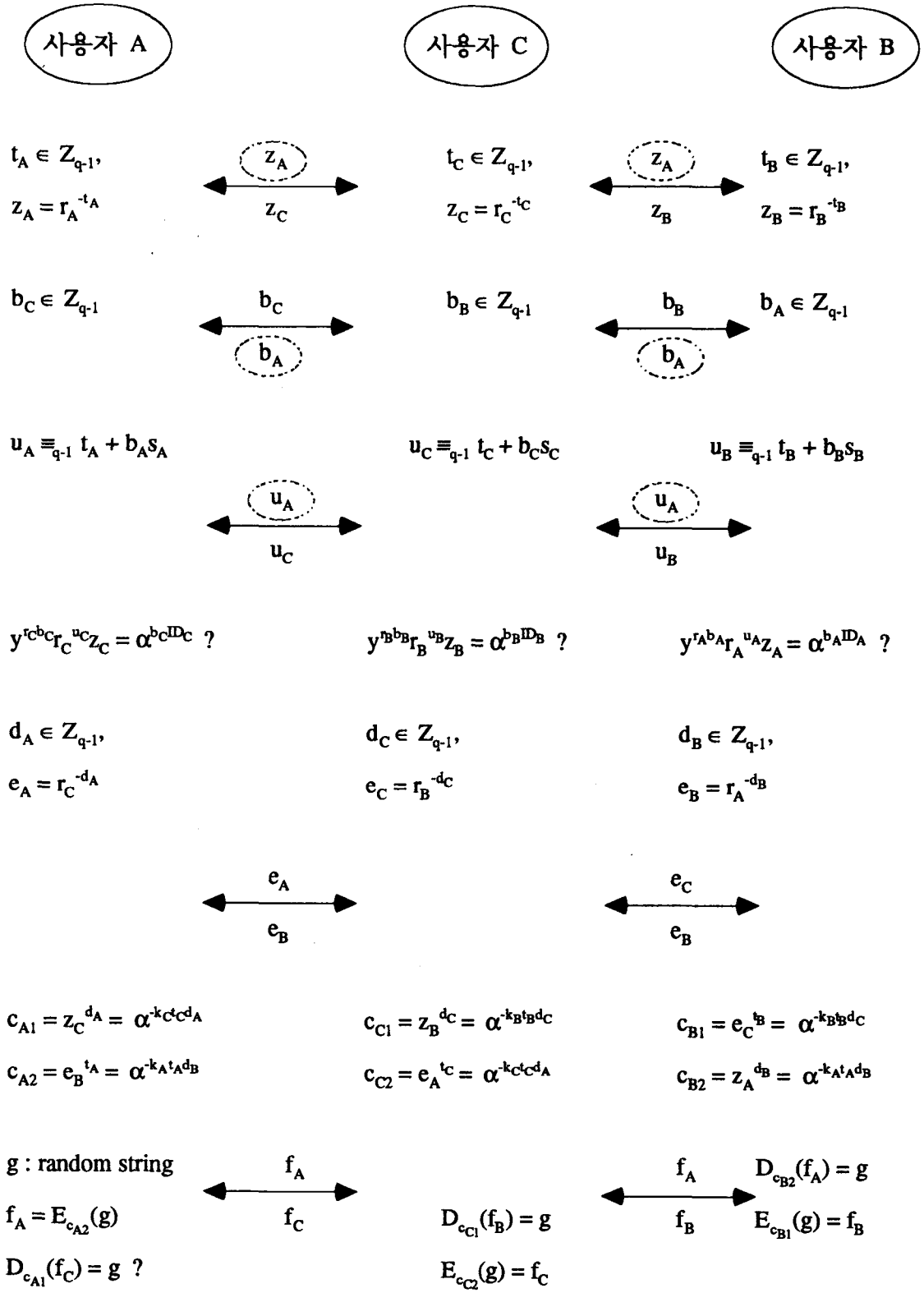

그림 5. 참고문헌 [2] 의 프로토콜에 대한 공격

이상에서 상호 신분인증과 동시에 세션키를 분배하기 위해 제안되었던 기존의 두 프로토콜에 대한 oracle session attack을 통해 이들이 전혀 안전하지 않음을 보였다. 두 프로토콜의 공격에서 쉽게 유추할 수 있듯이 이와같은 공격이 가능한 가장 큰 요인은 상호 신분인증 과정의 1 단계 및 2 단계에서 교환되는 witness와 challenge 값들이 프로토콜에 관련된 두 사용자와 전혀 무관하다는 사실에 기인한다. 즉 공격자가 원하는 정보(3 단계에서 사용자 A로부터의 응답)를 얻기 위해서는 1 단계에서 사용자 A로부터의 witness를 그대로 사용자 B에게 중계하고 2 단계에서는 사용자 B로부터의 challenge를 그대로 사용자 A에게 중계해 주어야 하므로 만일 이들 두 값중에 어느 하나라도 통신하고 있는 상대방과 연관을 시켜 준다면 위에서 제시한 공격은 막을 수 있을 것이다. 그 방법으로 다음의 두가지를 생각할 수 있다.

첫째 방법으로 1 단계에서 witness를 전송할 때 이를 상대방의 ID와 연관을 시켜 당사자를 제외하고는 누구도 이로부터 ID를 분리해 낼 수 없도록 하는 것이다. 이는 다음과 같은 간단한 방법으로 해결할 수 있다. Fiat-Shamir 방식을 예로들어 보자. 우선 센타는 법으로 사용되는 합성수 N보다 충분히 작은 랜덤한 소수 q를 공통의 파라미터로 추가로 공개한다. 이제 사용자 A는 1 단계의 전송을 $T_A \equiv R_A^2 \bmod N$, $X_A \equiv T_A \oplus ID_B \bmod q$ 와 같이 계산하여 이 X_A를 사용자 B에게 전송하고 사용자 B 역시 동일한 방법으로 사용자 A의 ID를 결합시킨 X_B를 전송한다(혹은 공개된 해쉬함수 h를 이용하여 $X_A = h(T_A, ID_B)$와 같이 계산하는 것도 좋은 방법이다). 그러면 3 단계에서 사용자 B의 인증조건 검사는 다음과 같이 이루어진다. 먼저 $\underline{T}_A \equiv Y_A^2 \prod_{E_{Ai}=1} V_{Ai} \bmod N$을 계산하고 자신의 ID를 이용, $\underline{X}_A \equiv \underline{T}_A \oplus ID_B \bmod q$를 계산하여 이것이 1 단계에서 사용자 A로부터 받은 X_A와 같은지를 조사하는 것이다. 이러한 간단한 변형만으로도 앞에서 제시한 공격을 막을 수 있음은 쉽게 알 수 있다. 사용자 A가 1 단계에서 사용자 C에게 전송하는 witnesss는 $X_A \equiv T_A \oplus ID_C \bmod q$, $T_A \equiv R_A^2 \bmod N$가 될 것이고 사용자 B가 마지막 3 단계에서 검사하는 인증조건은 $\underline{X}_A \equiv \underline{T}_A \oplus ID_B \bmod q$, $\underline{T}_A \equiv Y_A^2 \prod_{E_{Ai}=1} V_{Ai} \bmod N$가 될 것이기 때문이다.

두번째로 생각할 수 있는 것은 2 단계의 challenge 값으로 대화를 하는 두 사용자를 묶는 방법이다. 즉 challenge 값을 각 사용자가 독립적으로 랜덤하게 선택하는 대신에 1 단계에서 교환된 witness와 ID의 함수로 계산하는 것이다. 예를들어 Fiat-Shamir 방식에서 공개된 해쉬함수를 이용하여 $E = h(X_A, ID_A) \oplus h(X_B, ID_B) \in [1, 2^k)$와 같이 공통의 challenge 값을 계산할 수 있다(물론 이는 어느 신분인증 프로토콜을 이용하더라도 적용될 수 있다). 일단 전송할 witness X_A가 결정된 후라야 challenge 값을 알 수 있으며 역으로 challenge 값을 결정한 후 전송할 witness를 계산하는 것은 해쉬함수의 일방성에 위배되므로 이처럼 공통의 challenge 값을 계산하는 방법은 각 사용자가 랜덤하게 선택하던 원래의 방법과 마찬가지의 효과를 가져온다. 더우기 이 경우

는 2 단계의 전송이 필요없으므로 일석이조의 효과를 얻을 수 있다. 주의할 것은 여기서도 각 사용자의 ID 가 공통의 challenge 값을 계산하는 과정에 관련되어야 한다는 것이다. 그렇지 않고 만일 $E = h(X_A) \oplus h(X_B)$ 와 같이 challenge 값을 계산한다면 사용자 C 가 사용자 B 로부터 받은 witness 도 자신의 것인양 그대로 사용자 A 에게로 증계하는 경우 결국 두 사용자 A 와 B 는 같은 challenge 값을 계산할 것이고 따라서 사용자 C 가 필요로 하는 정보를 얻을 수 있을 것이기 때문이다.

이상에서 제시한 두가지 방법에 의하면 앞에서 제시한 두 프로토콜에 대한 oracle session attack 은 쉽게 막을 수 있다. 그러나 이번에는 다음과 같은 oracle session attack 을 생각해 보자. 즉 앞에서 제시한 공격법에서 만일 사용자 C 가 사용자 A 와의 대화에서도 자신을 사용자 B 로 가장하여 프로토콜을 진행할 수 있다면 사용자 C 는 두 사용자 A 와 B 의 중간에서 사용자 A 와의 대화에서는 사용자 B 로 가장하고 사용자 B 와의 대화에서는 사용자 A 로 가장하여 두 사용자들로부터 받은 정보들을 단지 중계만 해주면 되므로 결국 두 사용자를 모두 속일 수 있게 된다. 이러한 공격에서는 위에서 제시한 두가지의 대응방안도 아무런 효과가 없을 것이다. 이는 다름아닌 II 장에서 설명한 mafia fraud 의 특수한 경우임을 쉽게 알 수 있다. 즉 mafia fraud 의 시나리오에서 상호 신분인증 프로토콜이 사용된다고 가정하면 그림 1 에서 B 는 결국 A 에게 자신을 D 로 증명하고 C 는 D 에게 자신을 A 로 증명하는 셈이 되며 B 와 C 를 동일한 사용자로 두면 바로 여기서 고려하는 공격형태가 될 것이다.

II 장에서 언급했듯이 일반적인 컴퓨터 통신하의 원거리 인증에서 이러한 공격을 검출하는 일반적인 방법을 찾는 것은 매우 어려울 것으로 보인다. 우선 이러한 공격이 가능하려면 사용자 C 가 사용자 B 를 가장하여 사용자 A 에게 통신을 요청했을 때 사용자 A 가 먼저 자신의 witness 를 전송해야 한다. 즉 이 공격 시나리오에서 프로토콜상 전송정보의 흐름은 A - (C) - B - (C) - A 의 순서로 순환되어야 하므로 공격자는 어떤 방법으로든 (자신이 가장하고자 하는 대상인) 사용자 A 가 자신을 사용자 B 로 간주하고 프로토콜을 시작하도록 해야 한다. 따라서 만일 사용자 A 가 상대방으로부터 통신요청을 받았을 때는 자신이 먼저 witness 를 전송할 것이 아니라 통신을 요청한 상대방이 먼저 witness 를 전송하기를 기다린다면 위에서 제시한 두 가지 방법들로도 이 공격을 막을 수 있다. 사용자 A 가 먼저 프로토콜을 시작할 때는 항상 자신이 통신하고자 하는 상대방의 ID 를 결합한 전송정보를 보낼 것이므로 위와같은 공격의 위협은 사라지기 때문이다. 결국 이러한 공격은 프로토콜 자체로 막으려고 하기보다는 프로토콜의 진행절차에 제한을 둠으로써 훨씬 쉽게 막을 수가 있다. 즉 프로토콜은 항상 통신을 요청한 쪽에서 전송을 먼저 시작해야 하고 상대방은 통신 요청자로부터 witness 를 받은 후에야 자신의 witness

를 전송해야 한다는 규칙을 두는 것이다. 이는 완전한 익명의 사용자들로 구성된 컴퓨터 통신망에서 자신이 불이익을 당하지 않기 위해서는 각 사용자 스스로가 당연히 지켜야 할 규칙이며 또한 이것이 프로토콜의 진행에 별다른 장애가 되지는 않으므로 결코 지나친 제한은 아니라고 생각된다. 특히 본 논문에서 중점을 두는 3-move 상호 신분인증 프로토콜은 비대칭형 프로토콜로서 통신을 시작하는 사용자와 이에 응답하는 사용자가 확연히 구분될 뿐더러 통신을 시작한 사용자로부터 전송된 witness를 받은 후에라야 응답을 할 수가 있으므로 묵시적으로 위의 규칙을 바탕으로 하는 프로토콜이라 할 수 있다. 따라서 이후의 모든 프로토콜에서는 이러한 가정하에 mafia fraud 유형의 공격은 안전성 분석에서 배제하기로 한다.

IV. 안전한 상호 신분인증 프로토콜의 설계

이 장에서는 III 장에서 제시한 공격방법 및 이에 대한 대응방안 등을 바탕으로 기존의 3-move 신분인증 방식을 안전한 3-move 상호 신분인증 방식으로 변형시켜 보기로 한다. 이러한 프로토콜에 키분배 기능을 결합시키는 것은 비교적 쉬운 일이며 이는 제시되는 프로토콜들이 상호 신분인증을 위한 용도로 뿐만 아니라 키분배 프로토콜로도 이용될 수 있음을 의미하므로 많은 응용에서 매우 유용하게 사용될 수 있을 것이다. 우선 영지식 기술에 바탕을 둔 3-move 신분인증 방식의 일반적인 모델로부터 앞의 분석들을 바탕으로 안전한 3-move 상호 신분인증 방식의 일반모델을 제시하기로 한다.

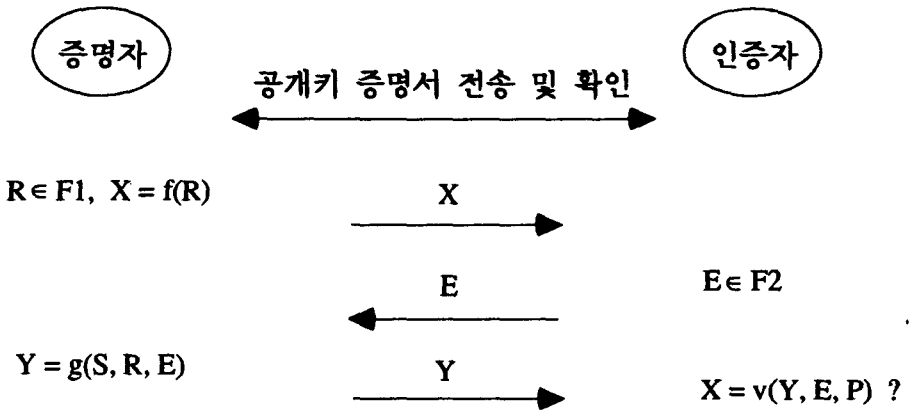

그림 6. 3-move identification scheme 의 일반모델

서론에서 언급한 모든 3-move 신분인증 방식들은 다음과 같은 일반적인 모델의 구체적인

프로토콜에 해당한다 (그림 6 참조). 즉 1 단계에서는 증명자가 정해진 스페이스 F1 에서 선택한 랜덤수 R 에 이산대수나 소인수분해 문제 등에 바탕을 둔 일방함수 f를 적용시켜 그 결과인 X = f(R) 을 랜덤수에 대한 witness 로 상대방에게 전송하고, 2 단계에서는 이를 받은 인증자가 지정된 스페이스 F2 상에서 선택한 랜덤수 E 를 challenge 로 증명자에게 전송하며 마지막 3 단계는 증명자가 1 단계에서 사용한 랜덤수 R 과 자신의 비밀키 S, 그리고 2 단계에서 받은 E 등을 결합하여 response Y = g(S, R, E) 를 계산, 인증자에게 전송하는 것으로 구성된다. 그러면 인증자는 인증조건 x = v(Y, E, P) 가 성립하는 지를 조사하여 상대방을 확인할 수 있다. 여기서 P 는 증명자의 비밀키 S 에 대응하는 공개키로 P = f(S) 로 주어진다.

이제 이러한 3-move 신분인증 방식이 Feige-Fiat-Shamir 의 안전성 정의 [13] 에 따라 안전하다고 가정하고 (예 : 참고문헌 [15][18][19] 의 방식들) 이를 두 사용자 쌍방에 그대로 적용시킨 경우를 생각해 보자. II, III 장에서 상세히 다루었듯이 이 경우는 oracle session attack 하에서 쉽게 깨어진다. 그러나 일단 적법한 사용자를 real-time oracle 로 이용하지 않는 한 이 프로토콜은 두 사용자 모두에 대해 어떤 공격하에서도 안전하다는 것이 안전성 증명에 의해 보장된다. 이는 곧 안전한 신분인증 프로토콜을 두 사용자 쌍방에 적용시킨 상호 신분인증 프로토콜은 이러한 oracle session attack 하에서만 안전하다면 어떤 공격에 대해서도 안전하다는 것을 의미한다.

그러면 이제 III 장의 끝부분에서 제시한 oracle sesssion attack 을 막을 수 있는 방법을 적용, 안전한 3-move 상호 신분인증 방식의 일반모델을 생각해 보자. 먼저 서로의 challenge 값을 교환하는 2 단계는 III 장에서 제시한 것처럼 1 단계에서 교환된 witness 들의 함수로 공통의 challenge 값을 계산함으로써 전체 프로토콜을 2 라운드로 줄일 수 있다. 그리고 oracle session attack 을 막기 위해 1 단계의 witness 는 반드시 상대방의 ID 를 결합시켜 계산한다. 다음에는 통신에 응답하는 사용자가 자신의 witness 와 response 를 동시에 전송하도록 함으로써 이 2 라운드의 프로토콜을 비대칭형의 3-move 프로토콜로 쉽게 재구성할 수 있다. 이를 다음의 그림 7 에 도시하였다. 여기서 f, g, h, v 등은 프로토콜에서 정의되어야 할 함수들로서 f, g, v 는 원래의 3-move 신분인증 방식에서 witness 와 response 를 계산하는 함수와 인증조건 (verification condition) 을 검사하는 함수를 각각 나타내며 h 는 안전변수에 따라 정해진 영역에서 출력값을 갖는 일방성 해쉬함수를 나타낸다.

다음에는 위의 모델에 따라 기존의 안전한 3-move 신분인증 방식을 이용하여 상호 신분인증 및 키분배 프로토콜의 실례를 들어본다. 본 논문에서는 Okamoto [19] 가 제안한 이산대수문제에 바탕을 둔 방식과 Ohta-Okamoto [15] 가 제안한 소인수분해문제에 바탕을 둔 방식을 이용한 두가지 상호 신분인증 프로토콜을 제시하기로 한다. 후술하는 방식들에서 알 수 있겠지만

Okamoto 등은 Schnorr 방식이나 GQ 방식에서 안전성 증명이 어려웠던 것은 $Y \equiv g^X \bmod p$ 나 $I \equiv S^v \bmod n$ 과 같은 합동식이 주어진 영역에서 단 하나의 해만을 갖기 때문이라는 사실에 주목하여 이러한 합동식들이 원하는 수만큼의 해를 갖도록 함으로써 그들의 프로토콜이 안전하다는 것을 증명할 수 있었다.

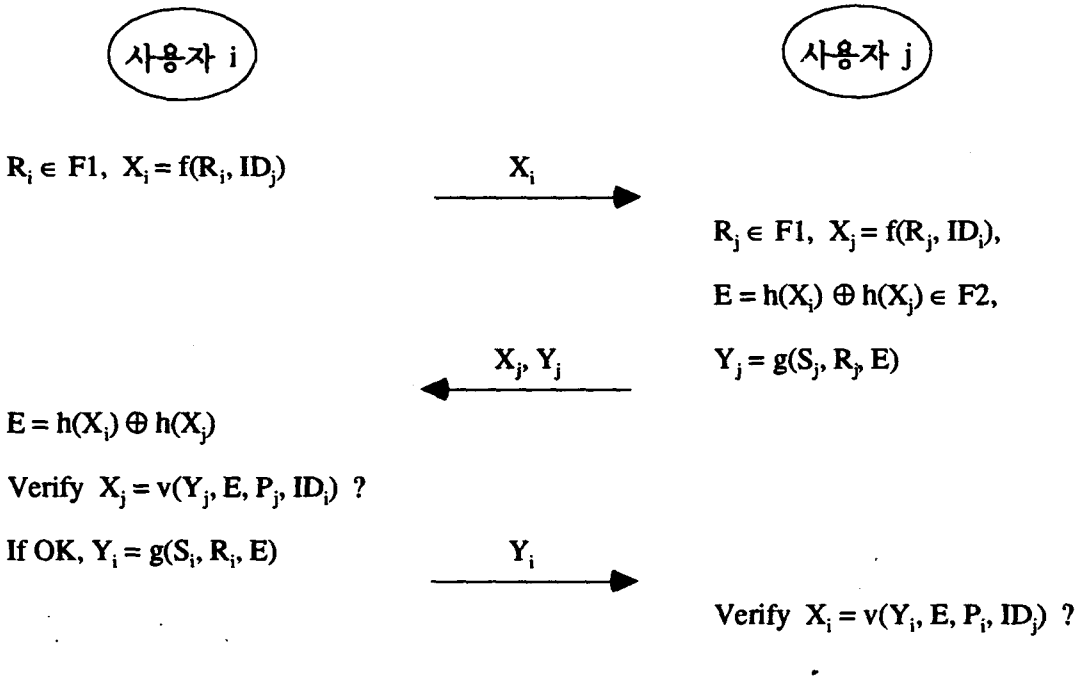

그림 7. 3-move mutual authentication scheme 의 일반모델

먼저 Okamoto 의 방식을 간략히 살펴보자. Okamoto 는 참고문헌 [19] 에서 몇 가지의 프로토콜을 제시하였으나 여기서는 이산대수문제를 이용한 그의 첫번째 프로토콜을 이용한다. 먼저 시스템 준비단계에서 신뢰할 수 있는 센타는 큰 소수 p 로서 p-1 이 140 비트 이상의 소수 q 를 약수로 갖도록 소수 p 를 선택하여 p, q 를 공개한다. 또한 안전변수로 $t\ (\geq 20)$ 를 공개하고 GF(p) 상에서 위수 (order) 가 q 인 기본원소 g_1 을 선택, $g_2 \equiv g_1^{\alpha} \bmod p$ 를 계산하여 g_1, g_2 는 공개하고 α 는 누구도 알지 못하게 안전하게 폐기한다. 이제 시스템에 가입하고자 하는 각 사용자 i 는 자신의 비밀키로 $S_{i1}, S_{i2} \in Z_q$ 를 선택하여 공개키 $P_i \equiv g_1^{-S_{i1}} g_2^{-S_{i2}} \bmod p$ 를 계산, 공개한다. 사용자 i 가 j 에게 신분을 인증하는 과정은 다음과 같다. 공개키 및 공개키 증명서는 이미 전송되어 확인되었다고 가정한다.

i) 사용자 i 는 Z_q 상에서 두 랜덤수 R_{i1}, R_{i2} 를 선택하여 $X_i \equiv g_1^{R_{i1}} g_2^{R_{i2}} \bmod p$ 를 계산, 사용자 j 에게 전송한다.

ii) 사용자 j는 랜덤수 $E_i \in [0, 2^t)$를 선택, i에게 전송한다.

iii) 사용자 i는 $Y_{i1} \equiv R_{i1} + S_{i1}E_i \bmod q$, $Y_{i2} \equiv R_{i2} + S_{i2}E_i \bmod q$를 계산, j에게 전송한다.

iv) 사용자 j는 $X_i \equiv g_1^{Y_{i1}} g_2^{Y_{i2}} P_i^E \bmod p$가 성립하는 지를 조사하여 i를 인증한다.

위의 프로토콜을 깨는 것이 이산대수문제만큼 어렵다는 사실을 증명할 수 있었던 것은 합동식 $P_i \equiv g_1^{-S_{i1}} g_2^{-S_{i2}} \bmod p$가 주어진 P_i에 대해 q개의 해 (S_{i1}, S_{i2})가 존재하며 무한한 계산력을 가진 공격자라 하더라도 사용자 i가 전송한 (X_i, Y_{i1}, Y_{i2})로부터 그가 어떤 (S_{i1}, S_{i2})를 사용하였는지를 알 수 없다는 사실에 기인한다. 자세한 증명과정은 참고문헌 [19]를 참조하기 바란다.

이제 위의 프로토콜을 키분배가 가능한 3-move 상호 신분인증 방식으로 변형시켜 보자. 이를 위해 시스템 파라미터와 각 사용자의 공개키/비밀키 생성을 약간 바꾸기로 한다. 즉 센타는 시스템 파라미터로 p, q, $g = g_1$, t만을 공개하고 각 사용자 i가 자신의 비밀키의 일부로 $\alpha_i \in Z_q$를 선택하여 $g_i \equiv g^{\alpha_i} \bmod p$를 계산, 공개한다. 다음에 설명되는 프로토콜에서 알 수 있듯이 이처럼 α_i를 각 사용자가 선택하게 한 것은 인증과정 후의 세션키 분배를 위한 것이다. 따라서 그의 공개키는 $P_i \equiv g^{-S_{i1}} g_i^{-S_{i2}} \equiv g^{-(S_{i1} + \alpha_i S_{i2})} \bmod p$와 같이 계산되고 여기서 $S_i \equiv S_{i1} + \alpha_i S_{i2} \bmod q$로 둔다면 결국 $P_i \equiv g^{-S_i} \bmod p$의 형태가 된다 (사용자 i의 공개키 : $\{g_i, P_i\}$, 비밀키 : $\{\alpha_i, S_{i1}, S_{i2}\}$). 또한 t비트 이하의 출력을 내는 일방성 해쉬함수 h는 모든 사용자들이 이용할 수 있다고 가정한다. 두 사용자 i와 j 사이의 3-move 상호 신분인증 프로토콜은 다음과 같다 (그림 8 참조).

[3-move mutual authentication scheme 1]

i) 사용자 i는 Z_q 상에서 두 랜덤수 R_{i1}, R_{i2}를 선택, $T_i \equiv g^{R_{i1}} g_i^{R_{i2}} \bmod p$를 계산한 후 자신이 통신하고자 하는 상대방인 사용자 j의 ID를 이용, $X_i \equiv T_i \oplus ID_j \bmod q$를 계산하여 이 X_i를 사용자 j에게 전송한다.

ii) 사용자 j 역시 사용자 i와 마찬가지로 $X_j \equiv T_j \oplus ID_i \bmod q$, $T_j \equiv g^{R_{j1}} g_j^{R_{j2}} \bmod p$를 계산한다. 그리고 공통의 challenge 값으로 $E = h(X_i) \oplus h(X_j) \in [0, 2^t)$를 구해 $Y_{j1} \equiv R_{j1} + S_{j1}E \bmod q$, $Y_{j2} \equiv R_{j2} + S_{j2}E \bmod q$를 계산, (X_j, Y_{j1}, Y_{j2})를 사용자 i에게 전송한다.

iii) 사용자 i는 $E = h(X_i) \oplus h(X_j)$, $\underline{T}_j \equiv g^{Y_{j1}} g_j^{Y_{j2}} P_j^E \bmod p$를 계산하여 $X_j \equiv \underline{T}_j \oplus ID_i \bmod q$가 성립하는 지를 조사한다. 이 조건이 만족되면 자신의 응답 $Y_{i1} \equiv R_{i1} + S_{i1}E \bmod q$, $Y_{i2} \equiv R_{i2} + S_{i2}E \bmod q$를 계산, (Y_{i1}, Y_{i2})를 사용자 j에게 전송한다. 그리고 $K \equiv (T_j^{R_i} \bmod p) \bmod q$, $R_i \equiv R_{i1} + \alpha_i R_{i2} \bmod q$를 사용자 j와의 세션키로 계산한다. 만일 조건이 만족되지 않으면 프로토콜을 중단한다.

iv) 사용자 j는 $\underline{T}_i \equiv g^{Y_{i1}} g_i^{Y_{i2}} P_i^E \bmod p$를 계산, $X_i \equiv \underline{T}_i \oplus ID_j \bmod q$가 성립하는 지를 조사한다. 조건이 만족되면 사용자 i와의 세션키로 $K \equiv (T_i^{R_j} \bmod p) \bmod q$, $R_j \equiv R_{j1} + \alpha_j R_{j2} \bmod q$를 계산하고 그렇지 않으면 연결을 끊는다.

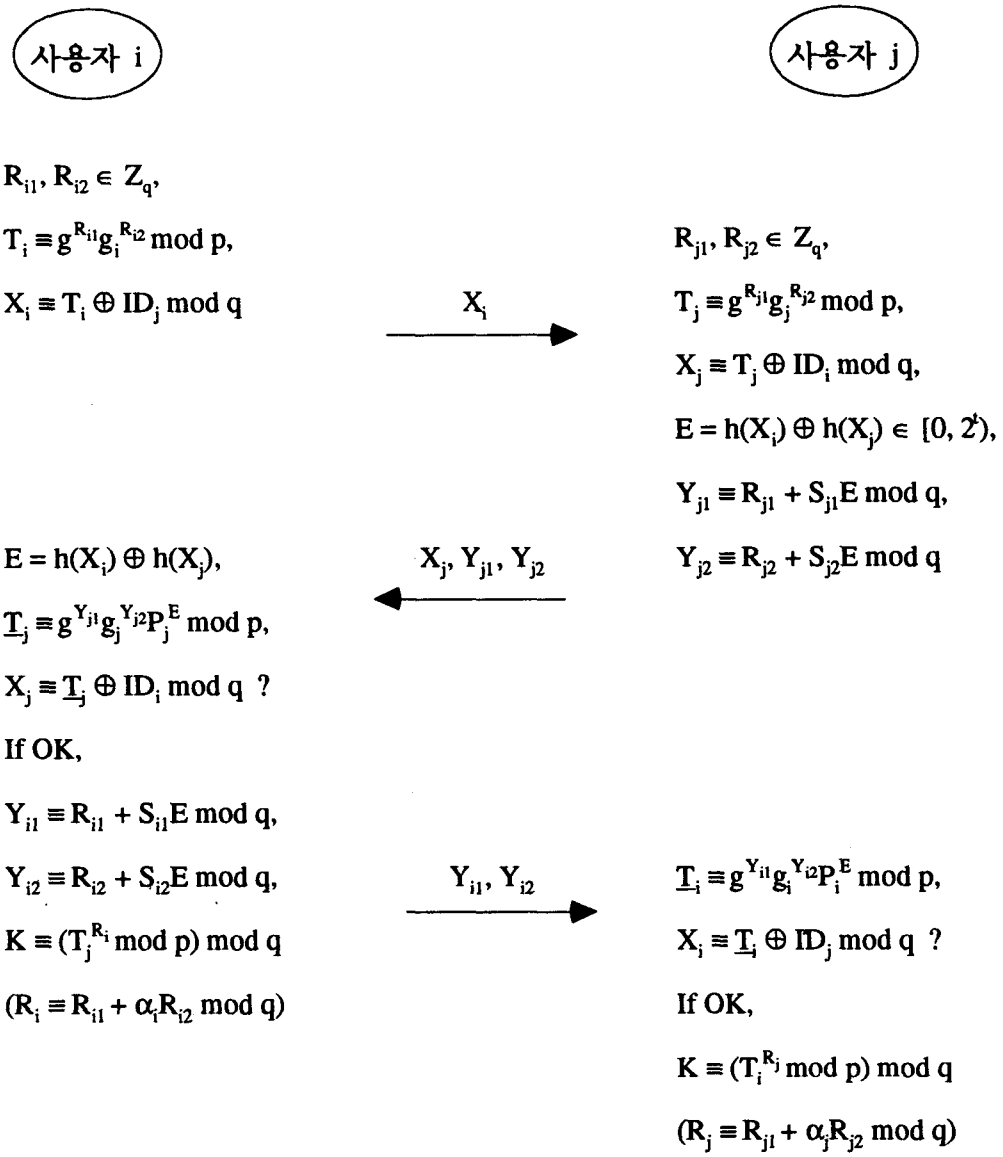

그림 8. Okamoto 방식을 이용한 상호 신분인증 및 키분배

위의 프로토콜에서 두 사용자간의 상호인증이 성공적으로 끝나면 $T \equiv T_j^{R_i} \equiv T_i^{R_j} \equiv g^{R_i R_j}$ 가 성립하므로 $K \equiv T \bmod q$ 를 두 사용자간의 세션키로 사용할 수 있다. 여기서 지수부는 $R_i R_j \equiv (R_{i1} + \alpha_i R_{i2})(R_{j1} + \alpha_j R_{j2}) \equiv (Y_{i1} + \alpha_i Y_{i2} - E(S_{i1} + \alpha_i S_{i2}))(Y_{j1} + \alpha_j Y_{j2} - E(S_{j1} + \alpha_j S_{j2})) \equiv (Y_{i1} + \alpha_i Y_{i2} - ES_i)(Y_{j1} + \alpha_j Y_{j2} - ES_j) \equiv Y_{i1} Y_{j1} + Y_{i1} Y_{j2} \alpha_j + Y_{i2} Y_{j1} \alpha_i - EY_{i1} S_j - EY_{j1} S_i + Y_{i2} Y_{j2} \alpha_i \alpha_j - EY_{j2} \alpha_j S_i - EY_{i2} \alpha_i S_j + E^2 S_i S_j \bmod q$ 와 같이 각 세션마다 변하는 랜덤수를 완전히 없애고 고정된 비밀키와 알려진 공개정보들만으로 전개할 수 있으므로 만일 일정수 (4 개) 의 T 가 알려진다면 ($g^{\alpha_i \alpha_j}, g^{\alpha_j s_i}, g^{\alpha_i s_j}, g^{s_i s_j} \bmod p$ 를 알 수 있으므로)

이들로부터 다른 세션의 세션키는 쉽게 계산될 수 있다 (known-key attack). 따라서 T 자체를 세션키로 사용하는 것은 안전하지 않으므로 $K \equiv T \bmod q$ 같이 T 를 q 로 나눈 나머지를 세션키로 사용하였다 (여기서 K 는 140 비트 이상의 수이므로 사용되는 관용 암호법의 키 길이에 따라 이로부터 64 혹은 128 비트 등과 같이 필요한 길이만큼 적절히 추출하여 사용하면 될 것이다). 상호 신분인증 과정은 III 장의 후반부에서부터 IV 장 전반부의 일반모델에 이르기까지의 설명과정으로부터 그 안전성에 관한 논리를 쉽게 유추할 수 있으므로 여기서는 더 이상 언급하지 않는다.

다음에는 Ohta-Okamoto 방식 [15] 을 이용하여 소인수분해문제에 바탕을 둔 프로토콜을 제시한다. 이 방식은 GQ 방식과 마찬가지로 FS 방식을 확장시킨 방식이나 그들은 공개 지수부 (public exponent) 를 적절히 선택함으로써 이를 깨는 것이 합성수를 소인수분해하는 것만큼 어렵다는 것을 증명하였다. 센타는 두 큰 소수 p_1, p_2 의 곱으로 된 합성수 n, $GCD(L, p_1-1) = L$ 이고 $GCD(L, p_2-1) = 1$ 을 만족하는 t (≥ 20) 비트 길이의 소수 L, n 보다 충분히 작은 랜덤한 소수 q, Z_n^* 에서 충분히 큰 위수를 갖는 원소 g, 그리고 L 보다 작은 출력을 내는 일방성 해쉬함수 h 등을 시스템 파라미터로 공개한다(여기서 q, g, h 는 필요에 의해 원래의 방식에서 추가로 공개한 파라미터임). 이때 L 을 선택하는 방법은 여러가지가 있으나 여기서는 가장 간단한 방법을 택하였다. 자세한 내용은 참고문헌 [15] 를 참조하기 바란다. 시스템에 가입하고자 하는 각 사용자 i 는 자신의 비밀키 $S_i \in Z_n$ 를 선택하고 $P_i \equiv S_i^{-L} \bmod n$ 을 그의 공개키로 공개한다. 여기서 $GCD(L, p_1-1) = L$ 이고 $GCD(L, p_2-1) = 1$ 이므로 P_i 가 주어지면 $P_i \equiv S_i^{-L} \bmod n$ 을 만족하는 해 S_i 는 정확히 L 개가 존재하게 되며 이것이 이 방식의 안전성 증명을 가능하게 하는 요인이다. 그리고 이 방식은 설령 n 의 인수를 알고 있는 센타라 하더라도 주어진 P_i 에 대해 $P_i \equiv S_i^{-L} \bmod n$ 을 만족하는 해 S_i 를 구하는 것은 쉽지 않으므로 GQ 방식과는 달리 identity-based scheme 으로 만들 수는 없다.

Ohta-Okamoto 의 신분인증 프로토콜은 잘 알려진 GQ 방식과 동일하므로 생략하기로 하고 두 사용자 i, j 사이의 상호 신분인증 프로토콜만을 기술한다 (그림 9 참조).

[3-move mutual authentication scheme 2]

i) 사용자 i 는 $\rho_i \in Z_n$ 을 랜덤하게 선택, $R_i \equiv g^{\rho_i} \bmod n$, $T_i \equiv R_i^L \bmod n$ 을 계산한 후 $X_i \equiv T_i \oplus ID_j \bmod q$ 를 사용자 j 에게 전송한다.

ii) 사용자 j 역시 $X_j \equiv T_j \oplus ID_i \bmod q$, $T_j \equiv R_j^L \bmod n$, $R_j \equiv g^{\rho_j} \bmod n$ 을 계산한다. 그리고 공개된 해쉬함수 h 를 이용, 공통의 challenge 값으로 $E = h(X_i) \oplus h(X_j) \in Z_L$ 를 계산한 후 $Y_j \equiv R_j S_j^E \bmod$

n을 계산하여 (X_j, Y_j)를 사용자 i에게 전송한다.

iii) 사용자 i는 상대방로부터 받은 (X_j, Y_j)로부터 $E = h(X_i) \oplus h(X_j)$, $Y_j^L P_j^E \equiv \underline{T}_j \bmod n$을 계산한 후 $X_j \equiv \underline{T}_j \oplus ID_i \bmod q$가 성립하는 지를 조사하여 상대방이 자신이 통신하고자 하는 사용자 j가 맞는 지를 확인한다. 이 조건이 만족되면 자신의 응답 $Y_i \equiv R_i S_i^E \bmod n$을 전송하고 사용자 j와의 세션키로 $K \equiv (T_j^{\rho_i} \bmod n) \bmod q$를 계산한다. 조건이 만족되지 않으면 프로토콜을 중단한다.

iv) 사용자 j는 $Y_i^L P_i^E \equiv \underline{T}_i \bmod n$을 계산하여 $X_i \equiv \underline{T}_i \oplus ID_j \bmod q$가 성립하는 지를 조사한다. 조건이 만족되면 사용자 i와의 세션키로 $K \equiv (T_i^{\rho_j} \bmod n) \bmod q$를 계산하고 그렇지 않으면 프로토콜을 중단한다.

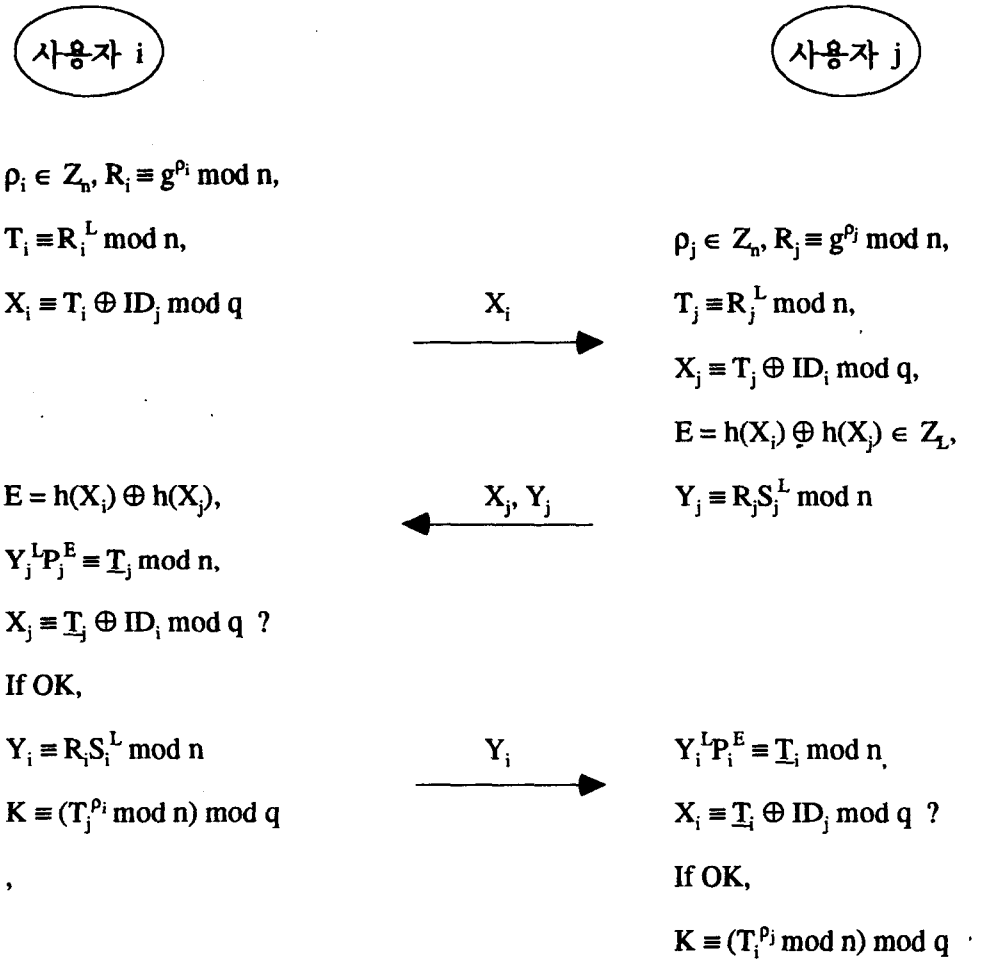

그림 9. Ohta-Okamoto 방식을 이용한 상호 신분인증 및 키분배

프로토콜이 성공적으로 끝나면 두 사용자는 세션키로 $K \equiv (g^{L\rho_i\rho_j} \bmod n) \bmod q$를 공유하게 됨을 알 수 있다. 원래의 방식과 달리 R_i를 랜덤하게 선택하는 대신 g에 의해 발생되는 곱셈군

의 원소로 계산한 것은 세션키 분배를 위한 것이며 g의 위수가 충분히 크다면 이는 R_i를 랜덤하게 선택하는 것과 동일한 효과를 얻을 수 있을 것이므로 문제가 되지 않는다(참고문헌 [31] 참조; 참고문헌 [31]의 Key Distribution Scheme 2.2.3 과 비교).

이상에서 기존의 3-move 신분인증 방식을 이용하여 안전한 3-move 상호 신분인증 방식을 설계하는 일반적인 모델을 제시하고 그 구체적인 예를 제시하였다. 물론 3-move 상호 신분인증 방식은 여기서 제시한 모델 이외에도 다양한 방법으로 설계가 가능하겠지만 이러한 접근법의 잇점은 우선 안전성이 증명된 신분인증 프로토콜을 이용함으로써 가능한 공격법을 real-time oracle session attack으로 제한시킬 수 있고 따라서 이러한 공격만 막을 수 있다면 쉽게 안전한 상호 신분인증 방식을 설계할 수 있다는 점이다.

V. 결 론

본 논문에서는 기존의 신분인증 프로토콜을 두 사용자 쌍방에 병렬로 적용시켜 얻은 상호 신분인증 프로토콜은 적법한 사용자를 real-time oracle로 이용하는 oracle session attack 하에서는 결코 안전하지 않음을 보이고 그 원인을 분석하여 이를 막을 수 있는 방안들을 제시하였다. 그리고 3-move 신분인증 방식을 이용하여 안전한 3-move 상호 신분인증 방식을 설계하는 체계적인 방법 및 일반적인 모델을 제시하고 그 구체적인 설계 예들도 제시하였다. 또한 상호 신분인증 과정에서 교환된 랜덤정보들을 이용하면 쉽게 세션키를 분배할 수 있음도 보였다. 이와같이 안전한 상호 신분인증 기능과 키분배 기능을 동시에 수행할 수 있는 프로토콜은 향후 컴퓨터 통신망에서의 다양한 보안 서비스들에서 매우 유용하게 사용될 수 있을 것이다.

[참고문헌]

[1] 이윤호, 양형규, 장청룡, 원동호, "영지식 증명을 이용한 키분배 방식에 관한 연구," *'91 정보보호학술발표논문집*.

[2] F.Bauspieb and H.K.Knobloch, "How to keep authenticity alive in a computer network," *Proc. Eurocrypt'89*.

[3] A Proposed Federal Information Processing Standard for Secure Hash Standard, *Federal Register Announcement*, Jan. 31, 1992, pp.3747-3749.

[4] A Proposed Federal Information Processing Standard for Digital Signature Standard, *Federal Register*

Announcement, Aug. 30, 1991, pp.42980-41982.

[5] M.E.Smid and D.K.Branstad, "Reponse to comments on the NIST proposed digital signature standard," *DRAFT COPY, presented at Crypto'92.*

[6] 조주연, 이필중, "개체의 특징에 의한 신분인증에 관한 연구," *통신정보보호학회지* (예정)

[7] S.Bengio, G.Brassard, Y.G.Desmedt, C.Goutier, and J.J.Quisquater, "Secure implementation of identification systems," *J. Cryptology*, Vol.4, No.3, 1991, pp.175-183.

[8] G.J.Simmons, "A system for verifying user identity and authorization at the point-of-sale or access," *Cryptologia*, 8(1), 1984, pp.1-21.

[9] P.D.Merillat, "Secure stand-alone positive personal identity verification system (SSA-PPIV)," *Technical Report SAND79-0070, Sandia National Laboratories*, Mar. 1979.

[10] S.Goldwasser, S.Micali, and R.Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM J. Comput.*, 17(2), 1988, pp.77-94.

[11] S.Goldwasser, S.Micali, and C.Rackoff, "The knowledge complexity of interactive proofs," *SIAM J. Comput.*, 18, 1, 1989, pp.186-208.

[12] A.Fiat and A.Shamir, "How to prove yourself : Practical solution to identification and signature problems," *Proc. Crypto'86.*

[13] U.Feige, A.Fiat, and A.Shamir, "Zero-knowledge proofs of identity," *Proc. STOC*, 1987.

[14] T.Beth, "Efficient zero-knowledge identification scheme for smart cards," *Proc. Eurocrypt'88.*

[15] K.Ohta and T.Okamoto, "A modification of the Fiat-Shamir scheme," *Proc. Crypto'88.*

[16] L.S.Guillou and J.J.Quisquater, "A practical zero-knowledge protocol fitted to security microprocessors minimizing both transmission and memory," *Proc. Eurocrypt'88.*

[17] C.P.Schnorr, "Efficient signature generation by smart cards," *J. Cryptology*, Vol.4, No.3, 1991, pp.161-174.

[18] E.F.Brickell and K.S.McCurley, "An interactive identification scheme based on discrete logarithms and factoring," *J. Cryptology*, Vol.5, No.1, 1992, pp.29-39.

[19] T.Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," *Proc. Crypto'92.*

[20] P.J.Lee, "Secure user access control for public networks," *Proc. Auscrypt'90.*

[21] Y.Desmedt, C.Goutier, and S.Bengio, "Special uses and abuses of the Fiat-Shamir passport protocol," Proc. Crypto'87.

[22] S.Bengio, G.Brassard, Y.G.Desmedt, C.Goutier, and J.J.Quisquater, "Aspects and importance of secure implementations of identification systems," *Manuscript M209, Philips Research Lab.*, Brussels, May, 1987.

[23] T.Beth and Y.Desmedt, "Identification tokens - or : Solving the chess grandmaster problem," *Proc. Crypto'90.*

[24] M.Burrows, M.Abadi, and R.M.Needham, "A logic of authentication," *ACM Operating Systems Review*, Vol.23, No.5, 1989, pp.1-13.

[25] A.Harding and S.Brady, "Review of authetication techniques for security standards," *Proc. Int'l Sym. Message Handling Systems and Application Layer Communication protocol*, Zurich, Switzerland, Oct., 1990.

[26] C.I'Anson and C.Mitchell, "Security defects in CCITT Recommendation X.509," *Computer Commun. Review*, Vol.20, No.2, 1990, pp.30-34.

[27] C.C.I.T.T. Recommendation X.509, The Directory - Authentication Framework, *C.C.I.T.T.*, Dec., 1988.

[28] Information Technology - Security techniques - Entity authentication mechanism - Part 3 : Entity authentication using a public key algorithm, *ISO/IEC DIS 9798-3.*

[29] Information Technology - Security techniques - Entity authentication mechanism - Part 2 : Entity authentication using symmetric techniques, *ISO/IEC DP 9798-2*, 1990.

[30] R.Bird, P.Janson, S.Kutten, R.Molva, M.Yung, A.Herzberg, and I.Gopal, "Systematic design of efficient provably secure two-way authentication protocols," *Proc. Crypto'91.*

[31] T.Okamoto and K.Ohta, "How to utilize the randomness of zero-knowledge protocols," *Proc. Crypto'90.*