韓國 컴퓨터情報學會 論文誌
第12卷 第4號, 2007. 9.

2007-12-4-2-2

안전한 하이브리드 인증 메시지 프로토콜

양 형 규*, 최 종 호**

The Secure Hybrid Authentication message protocol

Yang Hyung Kyu*, Choi Jong Ho**

요 약

안전한 채널은 가로채기와 같은 악의적인 공격에 대한 보안(방어 능력)을 제공한다. 반면에 인증 시스템은 클라이언트와 e-서버가 위조 공격(fabrication)으로부터 안전하도록 설계되어야 한다. 본 논문에서는 위에서 설명한 보안 기능을 제공하는 하이브리드 인증 프로토콜을 제안한다. 이 프로토콜은 SSL 인증에서 직면하고 있는 시간지연 문제를 해결함으로서 인증 시스템의 성능을 향상시킨다. 또한, 제안한 인증 시스템은 위•변조 공격으로부터 안전한 클라이언트와 서버를 구축할 수 있는 방식이다. 변경된 3-방향 인증 방식을 사용해서 타임 서버가 필요 없으며, 역시 타임스템프도 필요 없다.

Abstract

Asecure channel provides protection against interception, while an authentication system is created to protect the client and the server from fabrication attacks. This paper proposes a hybrid authentication algorithm, which fixes the lapses problem encountered inthe SSL authentication. Also, the proposed hybrid authentication system has been created to protect the client and the server from modification and fabrication attacks. By using a modified three-way authentication there is no need for a timeserver, thus timestamps are not needed.

▸ Keyword : OTP(One-Time Password), 인증(Authentication), 위조(Fabrication), 공개키(Public Key), 개인키(Privite Key)

• 제1저자 : 양형규
• 접수일 : 2007. 7.10, 심사일 : 2007. 7.25, 심사완료일 : 2007. 8.23
* 강남대학교 컴퓨터미디어정보공학부 교수
** 강남대학교 전자시스템정보공학부 교수

Ⅰ. 서 론

클라이언트와 서버사이에 SSL을 이용하여 설치된 안전한 채널은 불법수정과 가로채기 같은 공격으로부터 안전하도록 설계되었다. 반면에, 인증 시스템은 클라이언트와 서버를 위조 공격으로부터 보호하기 위해 요구된다. 일반적으로 하이브리드 인증 프로토콜은 이와 같은 공격으로부터 안전하도록 인증 기능을 제공할 수 있어야 한다[1,2].

클라이언트와 서버사이에 동작하고 있는 1-방향, 2-방향, 그리고 3-방향 인증 프로토콜에 대한 개요는 다음과 같다.[3,4].

1) 1-방향 인증 프로토콜은 다음과 같은 요소와 기능을 포함한다. (단, A는 사용자이고 B는 서버이다)
   - A의 객체(Identity)와 A에 의해서 생성된 인증 토큰(authentication token)
   - B의 객체와 B에게 전송될 인증 토큰
   - 전송된 인증 토큰에 대한 integrity와 originality(한 번 이상 사용되지 않음)
2) 2-방향 인증 프로토콜은 1-방향 인증 프로토콜의 요소와 기능을 포함하고 다음과 같이 설계된다.
   - A에 대한 응답으로 생성된 인증 토큰은 B가 생성
   - A가 B에게 보낸 인증 토큰은 originality
   - 토큰의 상호 보안
3) 3-방향 인증 프로토콜은 1-방향와 2-방향 인증 프로토콜의 특성을 포함하고 다음과 같이 설계된다.
   - A와 B가 상호간 한번이상의 통신을 하더라도 타임스템프를 사용하지 않아도 2-방향과 같은 인증 기능 제공

3-방향 인증 프로토콜은 타임 서버를 사용하지 않더라도 A와 B에 대한 인증을 제공한다. 성능 향상 관점에서 볼 때, 이것은 서버가 많은 연결들을 처리 할 수 있도록 해준다(범수균외 2인, 2004). 즉, 서버가 클라이언트와 동기를 맞출 필요가 없어진다. 또한, 이러한 특성은 3-방향 인증 프로토콜에게 재전송 공격으로부터의 안전성을 제공해준다. 왜냐하면 3-방향 인증 프로토콜은 이러한 공격을 막기 위해서 시간 동기화 작업이 필요 없기 때문이다[5].

Ⅱ. 기존의 3-방향 인증 프로토콜

서버가 클라이언트에 대한 신원 보증 즉 인증을 하기 위한 방법이 Winslett, M. 등(2001)에서 제시되었다. Winslett, M. 등은 개인들이 안전 보조 장치와 디지털 인증서에 대한 사용을 권고한다. 개인식별을 위해서 공개 키와 같은 유일한 일반적인 주체를 사용하는 대신에, Winslett, M. 등은 서버의 보안 정책에 의해서 전송되는 일반적인 주체의 사용을 제안하였다. 개인식별로는 주민등록증 혹은 운전면허증 등이 사용되며 이러한 인증서는 제 삼자가 오프라인으로 제공한다. 이러한 방법은 서비스 객체에 대한 직접적인 접근 같은 문제점을 내포한다. 따라서 보안 담당자는 다음과 같은 복잡한 정책들을 수립한다.

1) 서버는 자신의 보안 정책을 클라이언트와 공유할 수 있는 수단을 필요로 한다.
2) 클라이언트는 보안 정책을 이해할 수 있어야 한다.
3) 보안 정책에 따른 인증서가 요구된다.

한국 유저가 미국에 있는 서버에 대한 접근을 시도 할 때 어떤 일이 발생할까? 그들은 물리적으로 서버에 대한 신분 인증서를 어떻게 얻을 수 있을까? 또한 서버는 인증서 소유자가 정말로 원래의 소유자인지를 검증할 수 있는 방법이 없다. 하지만 이러한 문제들은 하이브리드 인증메세지 프로토콜을 사용해서 해결 할 수 있다[6].

다음은 기존의 3-방향 인증 방식에 대한 정형적인 표현 방법이다.(단, A : 클라이언트, B: 서버) 첫 번째, A는 난수 rA를 생성한다. 이것은 재전송 공격을 검출하고 위조를 방지하기 위해서 사용한다. A는 B에게 다음과 같은 메시지를 전송한다[7].

단계 1 : A → B, B {t_A, r_A, Ap, sgnData}

단, B{} = B의 공개키를 사용해서 {}의 내용을 암호화

t_A = 타임 스템프

r_A = 난수

Ap = A의 공개키

SgnData = 전자서명

B는 메시지 수신 후 자신의 비밀 키를 사용해서 다음과 같이 수행한다.

1) Ap를 얻는다.
2) SgnData를 검증하고 메시지에 대한 무결성을 확인한다.
3) B는 의도된 수신자인지를 검사한다.
4) rA가 재전송된 것인지를 검사한다.

만약 위의 검사가 모두 사실이라면 B는 난수 rB를 생성한다. B는 A에게 다음과 같은 메시지를 전송한다.

단계 2 : B → A, A {t_B, r_B, Bp, r_A, sgnData}

단, t_B : 타임스템프

B는 메시지 포함된 sgnData를 사용해서 데이터 무결성을 제공한다. A는 B가 송신한 메시지를 수신 후 다음과 같이 수행한다.

1) SgnData를 검증하고 메시지에 대한 무결성을 확인한다.
2) A는 의도된 수신자인지를 검사한다.
3) rB가 재전송된 것인지를 검사한다.
4) rA가 원래 보낸 rA와 같은지를 검사한다.

만약 위의 검사가 모두 통과되면 A는 마지막 메시지를 송신한다.

단계 3 : A → B, B {r_B, Bp}

B가 메시지를 수신 후, B는 rBrk 재전송 된 것인지 검사하고 A가 자신이 보낸 올바른 공개 키를 소유했는지 검사한다. 결과적으로 완전 3-방향 authentication 은 다음과 같다.

A → B, B {0, r_A, A, sgnData}

단, sgnData := E_{Ap}〔HASH(r_A||A)〕

B → A, A {0, r_B, B, r_A, sgnData}

단, sgnData := E_{Ap}〔HASH(r_B||B)〕

A → B, B{r_B, B}

III. 제안한 3-방향 인증 알고리즘

제안한 3-방향 인증 모델은 오직 클라이언트와 인증을 돕기 위한 타임서버의 사용을 배제하는 서버에 의존한다. 즉, 표준 3-방향 인증에서 사용하는 타임스템프는 사용하지 않는다. 이것은 논문 ITU-T X.509.(1988), William Stallings.(2000) 그리고 I'Anson C. (1990)에서와 같이 몇몇 문제를 내포한다[4,8].

A는 클라이언트고 B는 서버 그리고 C는 제 삼자라고 가정하자(이전에 A와 B는 인증된 상태). 공격은 C가 처음으로 도청한 메시지를 B에게 보낼 때 시작된다.

단계 1 : C → B: B{tA, rA, Ap}

단, B{} = B의 공개키를 사용해서 {}의 내용을 암호화

t_A = 타임 스템프

r_A = 난수

Ap = A의 공개키

B는 자신이 A와 통신하고 있다고 생각하지만 실제로 C와 통신하고 있는 상태로 응답한다.

단계 2 : B → C: A{tA, r'A, Bp, rA}

단, r'_A = 난수, Bp = B의 공개키

C는 동시에 A가 위의 메시지를 사용해서 인증을 시작하도록 한다. 결과로 A는 C에게 다음 메시지를 전송한다.

단계 3 : A → C: B{tA, r'A, Ap}

C는 B에 의해서 C에게 제공된 같은 nounce를 사용해서 다음과 같이 응답한다.

단계 4 : C → A: C{tA, r'B, Bp, r'A}

A는 다음과 같이 응답한다.

단계 5 : A → C: B{r'B}

이것은 C가 B에게 자신이 A와 통신하거 있다는 것을 확인시키기 위해서 필요한 메세제이다. 따라서 C는 B에게 위 메시지를 다시 보낸다.

단계 6 : C → B: B{r'B}

그림 1. 하이브리드 인증 프로토콜 구조, 검사, 변수저장
Fig. 1 Scheme, detection, and variable storage of hybrid authentication protocol

단계	자료 구조	검사 항목	저장 변수
1	MSG 1 → r_a 100 바이트 랜덤 수 → 클라이언트 공개키 → sgnData	None	r_a 클라이언트 공개키
2	MSG 2 → r_b 100 바이트 랜덤 수 → 서버 공개키 → r_a 100 바이트 랜덤 수 → sgnData	sgnData	r_a r_b 클라이언트 공개키 서버 공개키
3	MSG 3 → r_a 100 바이트 랜덤 수 → 서버 공개키	송신한 r_a to r_a sgnData	r_b 서버 공개키
4		송신한 r_b to r_b 송신한 서버 공개키 to 서버 공개키	

위의 경우처럼, 만약 타임서버가 없고 표준 3-방향 인증이 사용된다면, 서버는 중간자공격에 취약하다.

아직 이것은 클라이언트와 서버가 서로 타임서버를 사용하지 않고 인증 할 수 있도록 고칠 수 있다. 마지막 메시지를 다음과 같이 수정하면 된다. 즉, A가 마지막 메시지를 보낼 때 메시지가 B의 공개키를 포함도록(A → B :{rB, Bp})한다. 이것은 B가 위의 메시지가 A로부터 받은 메시지이고 B는 사실상 A를 인증했다는 사실을 명백히 해준다.

Ⅳ. 제안한 하이브리드 인증 프로토콜

안전한 전자상거래 프레임에는 두 개의 인증 형태가 존재한다. 사용자 레벨 인증과 기계 레벨 인증이다. 사용자 레벨 인증은 일회용 패스워드(OTP)를 사용한다[9]. 반면에 기계 레벨 인증은 하이브리드 인증 프로토콜을 사용한다. OTP 방식은 클라이언트가 서버에게 사용자를 인증하도록 하고 재사용 공격을 방지하도록 해준다[10,11]. OTP가 수정된 3-방향 인증 알고리즘을 사용해서 안전한 링크를 생성하는 것이 가능토록 해준다(그림 4 참조). 따라서 OTP와 하이브리드 인증 프로토콜은 SSL의 인증 방식이 가지고 있는 문제점을 서로 보충해준다[12].

하이브리드 인증 시스템의 구현은 세 부분으로 나눌 수 있다. 이러한 구분은 인증 과정을 용이하게 한다.

1) 메시지를 포함하는 자료 구조
2) 클라이언트 메시지 교환
3) 서버 메시지 교환

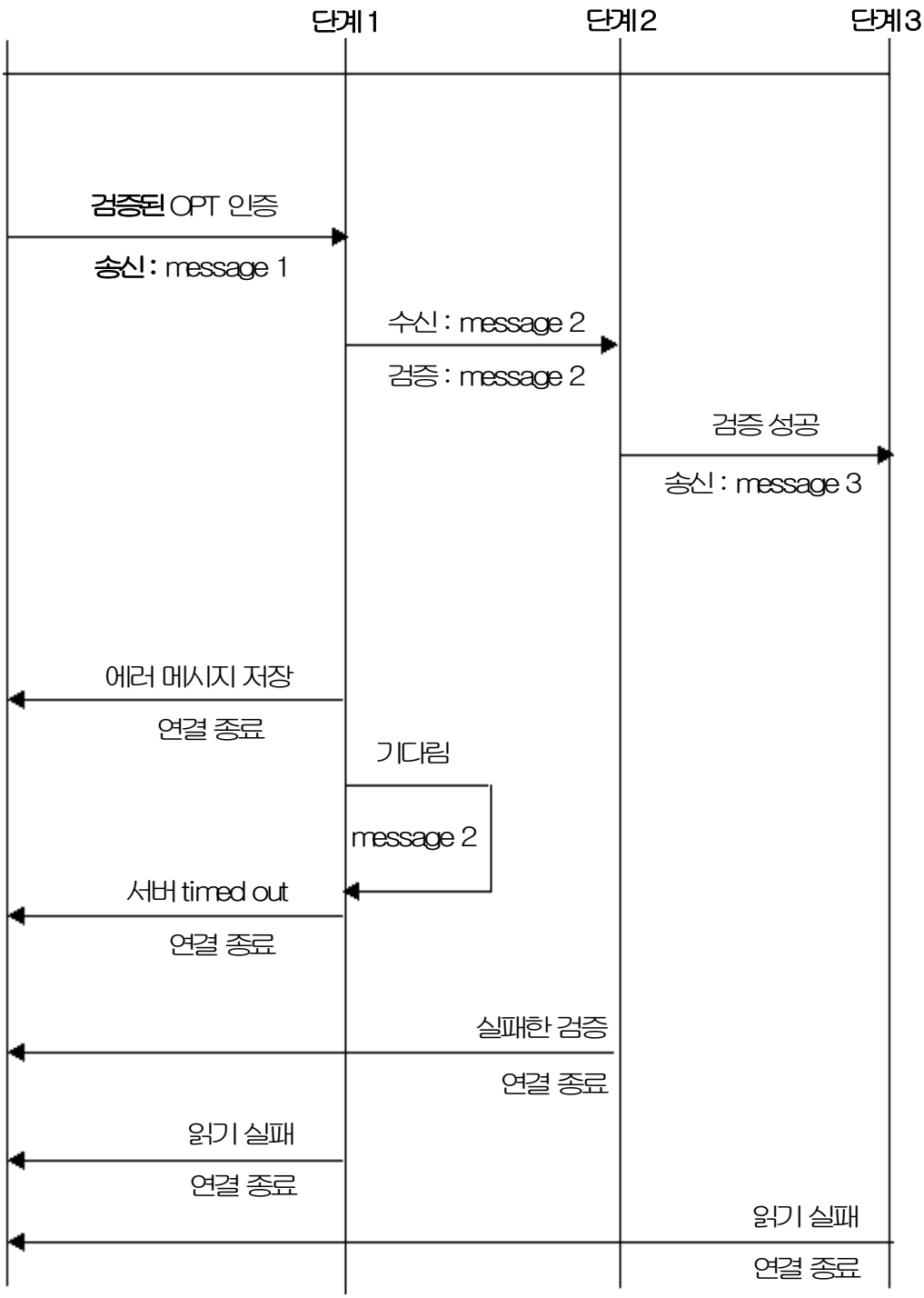

그림 2. 클라이언트 메시지 교환
Fig. 2 Exchange of client message

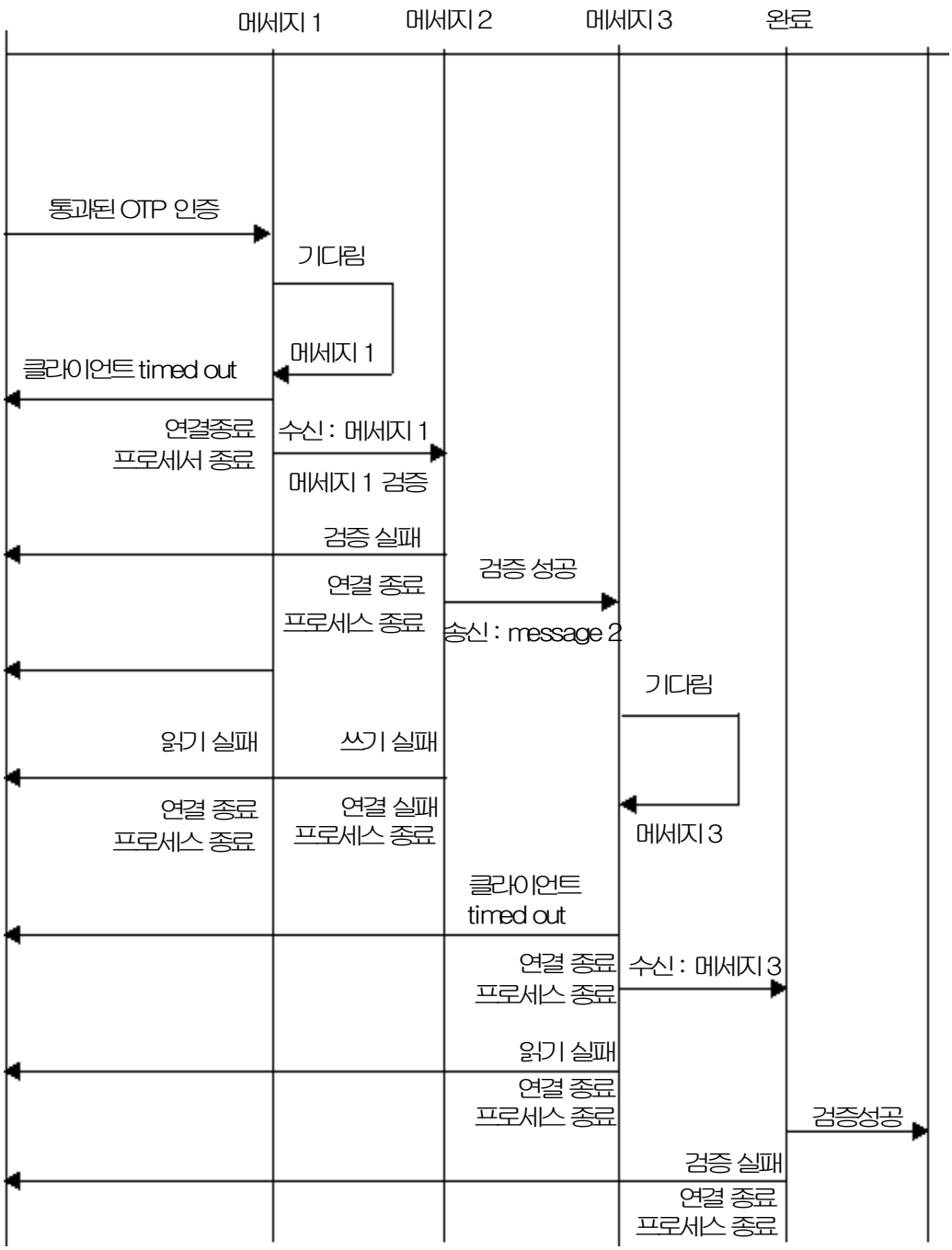

그림 3. 서버 메시지 교환
Fig. 3 Exchange of server message

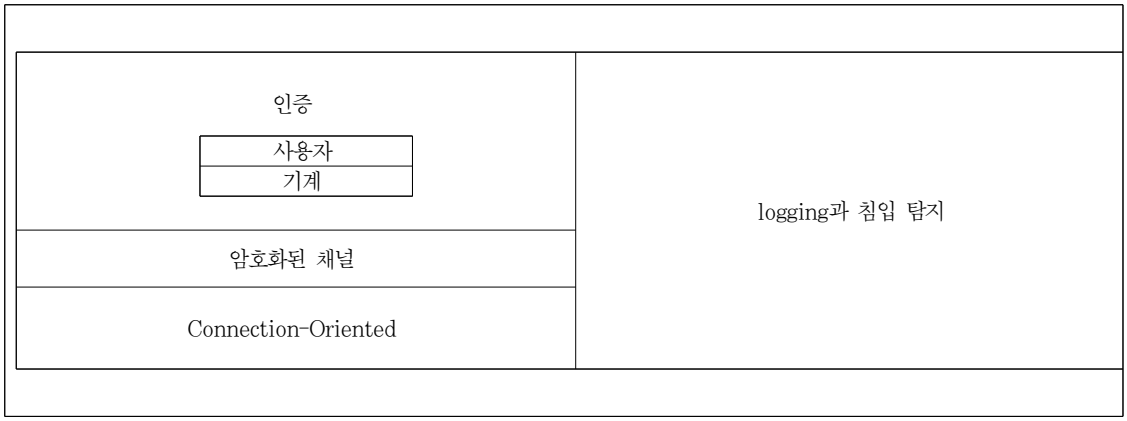

그림 4. 서버 프레임워크
Fig. 4 Framework of server

4.1 자료 구조

다음은 단계와 클라이언트와 서버가 서로 인증할 때 생성되는 자료 구조에 대한 설명이다.

단계 1 : 클라이언트는 100 바이트의 랜덤 수(ra)를 생성하고 나중에 MSG2와 비교하기 위해 저장한다. 클라이언트는 사용자 공개키를 읽고 그것을 저장한다. 그리고 전자 서명 데이터(sgnData)는 ra와 공개키를 사용해서 생성된다. 이것은 제 삼자가 공개키와 송신된 랜덤 수에 대한 변경을 방지한다. 클라이언트는 정보를 서버 공개키를 사용해서 암호화하고 개인키를 소유하고 있는 서버는 암호문을 복호화 한다.

단계 2 : 서버는 메시지 MSG1을 복호화하고 전자 서명을 생성하기 위해서 보내준 클라이언트의 공개키를 사용한다. 이것은 MSG1에 포함된 메시지와 비교한다. 만약 전자 서명이 검증된다면 랜덤 수(ra)는 후에 사용을 위해 저장된다. 서버는 100 바이트의 랜덤 수(rb)를 생성하고, 자신의 공개키와 랜덤 수(rb)을 전자 서명한다. 메시지 MSG2는 클라이언트가 보낸 랜덤 수(ra)와 서버의 공개 키, 서버의 랜덤 수(rb) 그리고 전자 서명(sgnData)으로 구성된다. 클라이언트의 랜덤수(ra)는 전자 서명에 포함 될 필요는 없다. 왜냐하면 클라이언트는 랜덤 수(rb)가 변경되었는지를 확인하기 위해서 랜덤 수(rb)을 복사하기 때문이다. 이러한 자료는 클라이언트의 공개키로 암호화 되어 클라이언트에게 보낸 진다.

단계 3 : 서버로부터 MSG2를 수신한 상태에서 클라이언트는 메시지를 복화화해서 클라이언트가 보낸 랜덤 수(ra)와 비교한다. 만약 같다면, 클라이언트는 서버의 랜덤 수(rb)와 공개키를 가지고 서버가 보낸 것과 비교하기 위해서 전자 서명을 수행한다. 전자 서명이 같다면, 메시지에 대한 변경이 없다고 판단한다. 그리고 클라이언트는 서버가 보낸 정보를 가지고 마지막 메시지를 생성한다. 클라이언트는 마지막 메시지를 생성하기 위해서 서버의 랜덤 수(rb)와 공개키를 사용한다. 그리고 생성된 메시지를 서버의 개인키로 암화화 한다.

단계 4 : 서버가 클라이언트로부터 마지막 메시지를 수신한 상태에서 서버는 메시지를 복호화하고, 서버가 클라이언트에 보낸 랜덤 수(rb)가 클라이언트가 다시 보낸 것과 같은지를 검사한다. 이와 같은 검사가 공개키에 대해서도 이루어진다. 만약 랜덤 수(rb)와 공개키가 같다면, 클라이언트와 서버는 인증된다.

4.2 클라이언트 메시지 교환

프로토콜은 클라이언트가 서버와 상호 통신하도록 만들어 졌다. 다음은 클라이언트가 서버에게 자신을 인증하기 위한 단계를 자세히 설명한 것이다(그림 2 참조).

단계 1 : 클라이언트가 사용자의 OTP를 사용해서 인증된 상태에서, 클라이언트는 첫 번째 메시지 (B{0, rA, A, sgnData})을 송신한다. 만약 어떤 이유로 서버가 메시지 전송 시간동안 다운이 된다면, 서버는 에러 메시지를 출력하고 클라이언트는 연결을 끊는다.

단계 2 : 만약 클라이언트가 서버로부터 응답을 기다리고 서버가 늦게 응답한다면 클라이언트는 시간 종료하고 연결을 끊는다. 두 번째 메시지가 서버로부터 수신됐다면, 서버는 인증됐으며, 만약 실패했다면 연결을 종료한다.

단계 3 : 메시지가 검증된 상태에서 클라이언트는 마지막 메시지 (B{rB,B})을 송신한다. 만약 에러가 발생한다면 클라이언트는 연결을 종결한다. 마지막 메시지가 성공적으로 송신 됐다면, 클라이언트는 곧바로 서버가 클라이언트를 인증했는지를 인식한다.

클라이언트는 서버가 위에 언급된 것과 다른 어떤 의심스러운 행동을 한다면 서버에게 경고 없이 곧바로 연결을 종료한다. 이것은 만약 서버가 제 삼자에 의해서 공격당했다면 제 삼자가 클라이언트가 연결을 종료한 이유에 대한 정보를 얻지 못하도록 하기 위함이다. 그리고 만약 제 삼자가 통신 내용을 도청했더라도, 연결 종료는 제 삼자에게 클라이언트와 서버가 에러 발생시 상호간 연결 방법과 관련한 어떠한 정보도 얻을 수 없도록 한다.

4.3 서버 메시지 교환

프로토콜은 서버가 클라이언트와 상호 통신하도록 만들어 졌다. 다음은 서버가 클라이언트에게 자신을 인증하기 위한 단계를 자세히 설명한 것이다(그림 3 참조).

단계 1 : 서버는 클라이언트가 첫 번째 메시지 (B{0, rA, A, sgnData})를 송신할 때까지 기다린다. 만약 클라이언트가 메시지를 너무 늦게 보낸다면, 서버는 연결을 종료시키고 프로세서를 종료시킨다. 만약 클라이언트가 메시지를 받지 못하면, 연결을 종료하고 프로세서 역시 종료시킨다. 만약 서버가 첫 번째 메시지를 수신한다면, 서버는 수신한 파라메터로부터 sgnData을 생성하고 클라이언트가 보내준 sgnData와 자신이 생성한 sgnData와 비교한다. 비교가 실패하면 연결을 종료하고 프로세서 역시 종료된다.

단계 2 : 만약 메시지가 검증됐다면, 서버는 두 번째 메시지 (A{0, rB, B, rA, sgnData})를 생성하고 이것을 클라이언트에게 송신한다.

단계 3 : 서버는 클라이언트가 세 번째와 마지막 메시지를 송신할 때까지 기다린다. 만약 클라이언트가 늦게 송신한다면, 서버는 연결을 종료하고 프로세서를 죽인다. 만약 서버가 보낸 메시지가 검증에 실패했다면, 서버는 읽기 실패를 출력하고 연결을 폐쇄하고(이것은 이미 클라이언트에 의해서 실행됐다) 그리고 프로세서를 죽인다.

단계 4 : 그렇지 않다면, 만약 서버가 마지막 메시지 (B{rB,B})를 클라이언트로부터 받은 후 서버가 이것을 검증한다면, 클라이언트는 서버에 의해서 인증되었다는 것을 의미한다. 만약 서버가 세 번째 메시지를 검증하지 못했다면, 연결은 종료되고 프로세서 역시 종료된다.

만약 이러한 시점에서 서버가 연결을 종료한다면 클라이언트는 자신이 시간 종료됐거나 서버에게 출력할 수 없는 에러가 발생됐다는 사실을 발견할 것이다.

V. 결 론

제안한 인증 시스템은 위•변조 공격으로부터 안전한 클라이언트와 서버를 구축할 수 있는 방식이다. 변경된 3-방향 인증 방식을 사용해서 타임 서버가 필요 없으며, 역시 타임스템프도 필요 없다. 이것은 클라이언트와 서버 만 요구되기 때문에 소수의 안전한 컴퓨터만 사용하면 되는 이점을 제공한다. 또한 클럭 동기화가 필요 없기 때문에 서버는 더욱더 많은 링크를 연결시킬 수 있다.

본 논문에서 우리는 사용자 공개키를 사용함으로서 OTP 시스템과 하이브리드 인증 프로토콜간 안전한 링크를 생성할 수 있는 하이브리드 인증 프로토콜을 제안한다. 재안한 하이브리드 인증 프로토콜은 다음과 같은 네 가지의 특성을 제공한다.

1) 위·변조 공격은 클라이언트가 서버의 공개키를 서버에게 다시 송신하기 때문에 성공 할 수 없다.
2) 이것은 하이브리드 인증 프로토콜이 하위 레벨 인증 기능을 수행토록 하며, 또한 OTP 시스템이 사용자 인증 기능을 수행할 수 있도록 한다.
3) 하이브리드 인증 프로토콜은 SSL 인증 문제를 해결하기 위해서 framework 상에 존재 할 수 있다.
4) 하이브리드 인증 프로토콜은 타임 서버와 같은 다른 참가자의 사용 없이 클라이언트에게 서버의 인증을 서버에게 클라이언트의 인증을 가능토록 하는 기능을 제공한다.

참고문헌

〔1〕 정우열, 이선근, "Twofish 암호 알고리즘의 성능향상을 위한 개선된 MDS 블록 설계", 한국컴퓨터정보학회논문지, 제10권 제5호, pp. 109-114, 2005.

〔2〕 박대우, 서정만, "TCP/IP 공격에 대한 보안방법 연구", 한국컴퓨터정보학회논문지, 제10권 제5호, pp. 217-225, 2005.

〔3〕 오수현, "Okamoto-Tanaka 키 분배 프로토콜의 안전성 분석", Journal of the Korea Data Analysis Society, Vol.7, No.2, pp. 677-683, 2005.

〔4,〕 William Stallings, Network Security Essentials: Applications and Standards, Upper Saddle River, NJ: Prentice Hall, 2000.

〔5〕 안영화, 양형규, "보안모듈을 이용한 키분배/인증 프로토콜 구현 및 분석", Journal of the Korea Data Analysis Society, Vol.7, No.1, pp.271-282, 2005.

〔6〕 FIPS PUB 180-1, SECURE HASH STANDARD, National Institute Of Standards and Technology, 2001.

〔7〕 Rivest R., "The MD5 Message-Digest Algorithm", Internet RFC 1321, 2001.

〔8〕 ITU-T X.509, Information Technology - Open Systems Interconnection - The Directory: Authentication Framework.CCITT Recommendation X.509, 1998.

〔9〕 Haller N., Metz C. , Nesser, M.Straw P. A, One-Time Password System. RFC2289, 1998.

〔10〕 Bart Preneel and Antoon Bosselaers, The Cryptographic Hash Function RIPEMD-160, CryptoBytes Vol. 3 No. 2, 1997.

〔11〕 The RIPEMD-160 page, Antoon Bosselaers. http://www.esat.kuleuven.ac.be/~bosselae/ripemd160.html

〔12〕 Marianne Winslett, Neil Ching, Vicki Jones, Igor Slepchin, "Assuring Security and privacy for Digital Libray Transactions on the Web: Client and Server Security Policies", 2001.

저 자 소 개

양 형 규

1995년 2월 : 성균관대학교 정보공학 박사

1995년 ~ 현재 : 강남대학교 컴퓨터미디어정보공학부 교수

관심분야 : 정보보호, 네트워크 보안 등

최 종 호

1987년 2월 : 중앙대학교 전자공학 박사

2002년 ~ 현재 : 강남대학교 전자시스템정보공학부 교수

관심분야 : 정보통신정책, 영상정보통신, 컴퓨터시각, 제스처인식