인증서 없는 키 교환과 다중영역 사용에 대한 XKASS 확장

김지현*, 오희국*

*한양대학교 컴퓨터공학과

A Key Exchange Protocol without Certificate and XKASS Expansion for Multi-usages

Ji-Hyun Kim*, Hee-Kuck Oh*

*Department of Computer Science & Engineering Hanyang Univ.

요 약

XKASS(XML Key Agreement Specification)는 사용자를 통해 XKMS와 연동하여 인증서 기반으로 클라이언트를 확인하고 키 동의를 하는 프로토콜이다. 기존 XKASS에서는 공개키 암호화 기법을 사용해 키 교환을 하는 방식과 전자서명 기법을 사용해 키 교환을 하는 방식을 사용한다. 기존 XKASS의 전자서명과 공개키 방식의 암호화 해독화 방법은 인증서를 그냥 전달함으로써 사용자의 신원이 노출되는 문제점이 있고, 계산 비용이 많이 든다. 본 논문에서는 기존의 XKASS 프로토콜에서 인증서 기반 공개키 암호화 방식을 사용한 키 교환 프로토콜을 사용하지 않고 키 교환 프로토콜을 수행하여 공격자에게 사용자의 신원보호를 제공하고 계산 비용을 줄인 확장된 XKASS 방식을 제안한다. 사용자에게 제한된 priority를 주는 것으로 DoS(Denial of Service) 공격을 예방한 기존의 XKASS를 확장하여 다중 영역에 사용할 수 있도록 제안하였다.

I. 서론

최근 XML이 기존의 데이터를 보다 쉽게 표현, 교환, 저장할 수 있어서 무역, 금융, 멀티미디어, 인터넷 전자거래, 데이터 전송 및 검색 부문 등에서 광범위하게 이용됨에 따라 XML 문서에 대한 보안 문제가 대두되고 있다. 전자 문서 및 데이터를 보호하는 일은 전자거래에 있어 필수적인 사항이며, XML 보안에 대한 연구개발 또한 활발히 진행되고 있다.

XML 정보보호 기술 중 XML 키 관리 명세인 XKMS는 다양하고 복잡한 기능의 전자거래 어플리케이션에서 XML문서의 서명을 검증하거나 암호화하는 키 관리를 위한 프로토콜을 정의한다. XKMS의 특징은 일반적인 PKI(Public Key Infrastructure) 구현의 복잡성에 비해 XML의 단순성은 비즈니스 시스템간 데이터의 간편한 응용성을 제공한다.

XKASS 서비스는 클라이언트를 통해 XKMS와 연동하여 인증서 기반으로 클라이언트를 인정해 키 동의를 하고 서비스에 사용할 키를 생성해 주는 프로토콜을 정의한다. XKASS 키 교환 방식은 공개키 암호화 방식을 사용해 키 교환하는 기법과 전자서명 기법을 사용해 키 교환을 하는 방식이다. 기존의 XKASS에서는 키 교환을 할 때 인증서를 암호화하지 않고 그냥 전달하거나 서명방식으로 서명해서 XKASS에게 보내게 된다. 이렇게 드러난 사용자의 신원정보는 어딘가에 저장되어 개인의 사생활이 침해 될 수 있으며, 또한 이 정보가 악용될 수 있다. 이 논문에서는 사용자의 신원정보가 들어있는 인증서를 사용하지 않고, 프로토콜의 수행 시 계산량이 많이 드는 공개키 암호 방식을 사용하지 않으면서 XKMS의 응답표현과 Diffie-Hellman 파라미터를 해쉬한 값을 사용해 메시지의 무결성을 보장하는 키 교환 방식을 제안하였다. 제안한 프로토콜의 계산량 비교를 위해 IETF(Internet Engineering Task Force)의 IPsec(Internet Protocol Security) 공식 후보인 IKEv2(Internet Key Exchange) 키 교환 프로토콜과 JFK(Just Fast Keying) 키 교환 프로토콜,

SIGMA(SIGnature Mode of Authentication) 키 교환 프로토콜, XKASS 키 교환 프로토콜과 비교하여 제시한다.

기존의 XKASS는 DoS 공격에 대해 처음 사용자가 아닌 기존의 사용자에게는 priority를 높게 주는 방법을 사용해 DoS 공격을 예방한다. 이러한 특성을 이용해 사용자의 priority를 높일 수 있고, 사용자의 측면에서 계산량을 줄이면서 키 동의를 할 수 있도록 Diffie-Hellman 키 동의 파라미터의 특성을 이용해 XKASS를 확장하였다.

본 논문의 구성은 다음과 같다. 먼저 XKMS와 XKASS의 구조에 대해 설명하고, 비교를 위해 사용된 IKEv2, JFK, SIGMA 시스템을 설명한다. 그리고 이 논문에서 제안하는 시스템에 대해서 설명하고, 제안하는 시스템의 안전성을 분석하고, 기존의 3가지 시스템과 비교한다.

II. 관련연구

XKMS는 마이크로소프트사와 베리사인, 웹메소드 사가 2001년 4월 W3C에 제안한 XML 기반의 공개 키 관리 명세이다. XKMS명세는 현재 Draft 상태이며 최초 설계 목적은 XML 전자서명과의 연동 시 기존 PKI 시스템에 대한 복잡성을 클라이언트에 숨겨 키 관리 부담을 트러스트 서비스에 위임해 그 구현을 용이하게 하기 위함이다. XKMS는 크게 XKISS(XML Key Information Service Specification)와 XKRSS(XML Key Registration Service Specification)의 두 영역으로 구성되어있다.

두 프로토콜은 XML Schema Language,

표 1: XKMS 구성요소[3]

구 분	내 용
키 정보 서비스 (XKISS)	-XML 전자서명, XML 암호화된 데이터와 관련 키 정보 처리를 지원하기 위한 프로토콜 -식별 정보가 주어졌을 때, 필요로 하는 공개키 위치와 식별자 정보, 공개키 연결 기능 지원
키 등록 서비스 (XKRSS)	-키 쌍 소유자에 의한 키 쌍의 등록을 지원하는 프로토콜 -'Trust Services'로서 요청과 응답의 메시지 교환으로 구성

WSDL(Web Services Definition Language)에 의해 정의된 메시지 사이의 관계와 SOAP(Simple Object Access Protocol)을 채택하는 프로토콜 내에서 표현된 구조로 정의된다. 여러 응용 형태의 객체 구조에서 XKMS의 적용도 가능하다[2].

XKASS는 Phillip Hallam-Baker가 제안한 XML 기반 하에 키 교환 프로토콜 XKASS는 초기자와 응답자 사이의 요청과 요구에 의해 키 동의를 하는 효율적인 수단으로 설명할 수 있다. 하나의 요청과 응답을 통해서 서비스를 받으려는 사용자를 인증서 기반을 통해 신원을 확인하는 서비스이다. 그리고 XKASS 서비스와 요청과 응답을 통해 비밀 공유를 만들어 낸다. 그렇게 만든 비밀 공유는 XKASS 서비스에 관련된 서비스들에게 암호교환과 유용한 사용자임을 인증하기 위해 사용된다.

XKASS는 암호화 인증서를 사용한 키 동의(Key Agreement using Encryption Credentials)와 서명 인증서를 사용한 키 동의(Key Agreement using Signature Credentials)프로토콜을 제시한다[4]. 이다. 설계목적은 어느 시스템과도 연동해서 키 교환을 할 수 있도록 하고 단순성과 안전성을 높이는 것이 설계 목적이다. 초기자와 응답자 사이의 요청과 요구에 의해 키 동의를 하는 효율적인 수단으로 설명할 수 있다. 하나의 요청과 응답을 통해서 서비스를 받으려는 사용자를 인증서 기반을 통해 신원을 확인하는 서비스이다. 기존의 키 교환 방식에서는 하나의 서버에서 키 교환, 메시지 교환 등을 수행하기 때문에 서버에 대하여 계산적 부담이 많이 들었다. 하지만 XKASS는 키 교환을 하는 부분을 분리시킴으로써 한 서버에 대한 부담을 줄이는 것이 그 취지이다. 그리고 XKASS 서비스는 사용자의 요청과 응답을 통해 비밀 공유를 만들어낸다. DoS 공격에 대해서는 이 논문에서 비교한 프로토콜과는 다르게 키 교환을 수행하였던 사용자에게는 priority를 높게 주고 새로운 사용자에게는 priority를 낮게 주는 방식을 사용해 키 교환을 한다. 기존에 제시된 키 교환 프로토콜에서는 키 교환 프로토콜에 사용되는 암호 알고리즘에 관한 협상을 하여 키 교환을 수행하였지만, XKASS에서는 사용자의 제안에 의해 XKASS가 결정해서 통보해 주는 방식을 사용함으로써 프로토콜의 단순성을 높였다. XKASS는 초기자와 응답자 사이의 요청과 요구에 의해 키 동의를 하는 효율적인 수단으로 설명할 수 있다. 하나의 요청과 응답을 통해서 서비스를 받으려는 사용자를 인증서 기반을 통해 신원을 확인하는 서비스이다. 그리고 XKASS 서비스와 요청과 응답을 통해 비밀 공유를 만들어 낸다. 그렇게 만든 비밀 공유는 XKASS 서비스에 관련된 서비스들에게 암

호교환과 유용한 사용자임을 인증하기 위해 사용된다.

현재 XKASS는 암호화 인증서를 사용한 키 동의(Key Agreement using Encryption Credentials)와 서명 인증서를 사용한 키 동의(Key Agreement using Signature Credentials)프로토콜을 지원한다[4].

IKEv2(Internet Key Exchange)는 IETF(Internet Engineering Task Force)의 IPsec(Internet Protocol Security)의 공식후보로서 initial 교환과 subsequent 교환의 2phase 방식을 채택하고 있고, 프로토콜 수행 시 4개의 메시지 쌍으로 구성되고 추가적인 메시지 쌍을 교환할 수 있다. phase1에서는 IKE_SA_INIT와 IKE_AUTH로 구성되며 통신하는 두 노드에 대한 상호인증과 IKE_SA 및 CHILD_SA를 설정한다. phase2에서는 메시지를 주고받는 1번의 교환으로 구성되며 CHILD_SA를 설정하거나 프로토콜 수행 중에 발생한 오류 정보 및 이벤트 발생정보 등을 교환하고, SA(Security Associate)의 재 설정을 위해서도 사용된다. 사용자와 응답자의 통신횟수는 6회이다[5].

JFK(Just Fast Keying)는 AT&T와 IBM연구원 Columbia대학 교수 등 7명이 모여서 작성한 IKE 후보 프로토콜이다. 단순성(simplicity), 효율성(efficiency), DoS(Deniable of service)예방, Privacy의 보장, PFS(Perfect Forward Secrecy) 만족 등 안전성(security)을 설계목표로 하고 있다. 1phase 방식을 사용하고 있고 인증에는 서명 기반 방식만을 사용하고 새로운 협상방식을 사용한다. 능동 공격자로부터 응답자의 신원보호, 수동 공격자에 대해서는 초기자만 안전하다. PFS를 만족하고 DoS공격에 보낸 메시지를 그대로 보내는 방식인 stateless cookie를 응답자의 필요에 의해 세션별로 항상 교환하여 예방한다. 사용자와 응답자의 통신횟수는 4회이다[6].

SIGMA(Signature Mod of Authentication)는 이스라엘 암호학자 Hugo Krawczyk에 의해 제안이 되었다. 프로토콜의 단순화와 기능 강화(안전성 강화), 성능 향상(효율성 향상) 등을 설계목표로 하고 있는 SIGMA는 DoS 공격에 대한 예방, 능동 공격자에 대한 초기자의 신원 보호를 보장하며 PFS를 만족한다. 1phase 방식으로 SA를 설정한다. DoS 공격을 예방하기 위한 방법으로는 stateless cookie를 첨가하는 방식으로 DoS 공격이 의심될 때 응답자만이 알고 있는 cookie를 만들어 전달하고 다시 전달받는 방법을 사용해 메시지 교환을 한번 더 함으로써 예방한다. DoS 예방을 위한 이 메시지 교환 방법은 DoS 공격의 의심이 있는 메시지를 응답자가 받았을 때만 키 교환 프로토콜에 삽입되어 실행한다. 기본적인 통신 횟수는 3회이다[7].

III. 제안하는 시스템

1. 시스템 설정

제안하는 시스템은 기존의 XKASS 시스템에서 인증서를 그냥 전달하거나 서명을 해서 전달함으로 통신상 수동공격자에게 대해 사용자의 신원이 드러난다. 이렇게 드러난 신원은 사용자의 전자거래 시 사용자의 소비패턴이나 생활 패턴이 드러나게 되는 문제점이 있다. 이런 이유로 사용자의 신원은 보호되어야한다. 사용자의 신원을 보호하기 위해서 XKMS 서비스에서 지원되는 대칭키 암호화 알고리즘을 사용하여 XKMS의 응답표현으로 사용자를 인증하는 키 동의를 제안한다.

1) 데이터의 흐름

① 사용자는 WSDL 저장소에서 XKASS의 서비스 위치와 공개키 인증서, 요구되는 암호화 기법, 서비스 위치 등의 명세서를 얻는다.

② 사용자는 XKMS 서비스에 XKASS 공개키 인증서의 정당성을 인정받고, 사용자가 정당한 사용자라는 응답 표현을 제공받는다.

③ 사용자는 XKASS 서비스와 서비스에 사용될 비밀 공유키를 얻기 위해 키 동의 프로토콜을 수행하여 XKASS와 사용하는 세션키를 만들어내고 비밀 공유키를 얻는다.

④ 사용자는 XKASS 서비스와의 프로토콜에서 얻은 비밀키를 사용해 서비스를 받는다.

2) 시스템 설정에 필요한 파라미터

K_{sx}: 참여자 X와 XKMS와 알고 있는 비밀키 값

s_s, s_L: 임시적인, 긴 사용의 비밀공유 키

$-K_X, +K_X$: X의 개인키, 공개키

O: 키 취급법, 유효기간, 사용되는 알고리즘 등

L: 서버이름, 사용자 신원확인 값, 비밀키 값 등

$H(d)$: 데이터 d의 해쉬 값

$M(d,K)$: 키 K를 사용한 메시지 d의 인증코드

req: 서비스를 받기 위한 요청(요청시간, 서버이

름, 유효기간 등)

rep: 서비스 요청에 대한 응답 표현(발행시간, 서버이름, 유효기간 등)

C_X: 사용자 X의 인증서

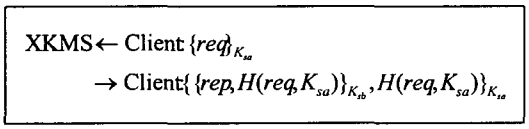
2. 인증서 없는 키 동의

기존의 인증서를 사용한 키 동의 프로토콜은 XKMS 서비스에 공개키를 생성하여 등록하는 과정을 거친 후, XKMS 트러스트 서비스에서 발급해 준 인증서를 통해서 사용자의 신원을 확인한다. 그리고 XKASS 서비스와 사용자간에 인증서 기반으로 공캐키 암호화 방법을 사용하거나 전자서명 방법을 사용해 키 교환을 한 후 서비스에서 사용할 비밀키 값을 받은 후 그 키를 사용해 서비스에 사용한다.

제안하는 시스템에서는 인증서를 사용하지 않고 계산량을 줄이기 위해 XKMS의 응답표현을 사용해 공개키 암호화 방법이나 전자서명 방법을 사용하지 않고 키 교환을 하게 된다.

단계1. 사용자는 XKASS와 키 교환에 사용할 XKMS의 응답표현을 얻기 위해 XKMS와 사용자간에 공유하고 있는 비밀 키 값으로 XKASS와의 키 교환에 사용할 응답표현을 얻기 위한 요청을 보내게 된다. XKMS는 요청에 대한 응답 표현으로 XKMS와 XKASS간에 공유하고 있는 비밀 키 값을 사용해 응답표현을 전달한다.[그림1].

그림 1 XKMS의 응답표현



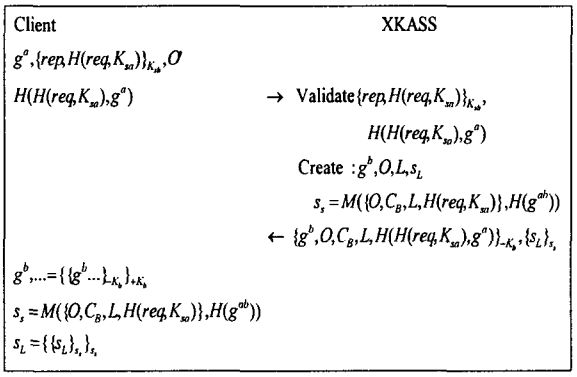
단계 2. 사용자는 XKASS에 대하여 키 동의를 받고 서비스에 사용할 키를 얻기 위한 프로토콜을 수행한다. 사용자는 자신만이 만들 수 있는 Diffie-Hellman 파라미터[10]인 $a \in Z_q$를 임의로 선택하여 g^a 값을 만들어낸다. 사용자는 XKMS에서 받은 응답표현과 응답표현을 확인 시켜주기 위한 해쉬 값, 키 사용법, 유효한 날짜, 제안하는 암호 알고리즘 등이 들어있는 O'과 XKASS에게 전달하는 메시지의 무결성을 위해 응답표현과 Diffie-Hellman 파라미터를 해쉬한 값을 만들어 보내게 된다.

단계 3. 확인과정 - XKASS는 받은 메시지를 확인하는 과정을 거친다. XKMS의 응답표현을 확인해 유효한 사용자임을 확인한다. 응답표현을 사용하여 사용자가 보내온 Diffie-Hellman 파라미터와 해쉬하여 사용자가 선택한 올바른 Diffie-Hellman 파라미터 값인지를 확인한다.

생성과정 - XKASS만이 만들 수 있는 Diffie-Hellman 파라미터인 $b \in Z_q$를 임의로 선택하여 g^b값을 만들고, 실제적으로 쓰일 암호 알고리즘 등이 들어있는 O를 만든다. 그리고 L, s_L을 생성하고 s_s값을 만들어 낸다. s_s 값에 들어가는 Diffie-Hellman 파라미터 값인 g^{ab} 값은 b 값을 알고 있는 XKASS만이 만들어 낼 수 있다. 이렇게 만들어낸 값을 XKASS의 개인키로 서명하여 전달하게 된다.

단계 4. 사용자는 XKASS가 전달한 값을 확인하고, Diffie-Hellman 파라미터의 값을 계산해 g^{ab} 값을 만들어 낸다. 그리고 XKASS와 사용자간에 사용하는 비밀 값 s_s 를 만들어내고, s_s 값을 사용해 s_L 값을 확인해 얻는다. 이렇게 함으로해서 사용자는 계산비용이 많이 드는 전자서명 계산을 XKASS가 보낸 값만을 확인하는 1번의 수행만을 하게된다.[그림2].

그림 2 제안하는 프로토콜 1



3. 다중영역 사용에 대한 키 교환

XKASS에서 DoS 공격에 대한 예방을 하기 위한 방법으로 XKASS에 처음으로 키 교환을 요청하는 새로운 사용자에게는 priority를 낮게 주고, 기존의 사용자에게는 priority를 높게 주는 방법을 사용해 DoS 공격을 예방하였다. 제한적 priority를 주는 것은 새로운 사용자에게 불리하다. 이러한 특성을 사용해 한 XKASS를 통해서 다른 XKASS에서도 키 동의를 받을 수 있도록 함으로

써 사용자는 한 XKASS에서만 키 교환을 수행 할 수 있어 자신의 priority를 높일 수 있다. 앞에서 제시한 방법을 사용해 사용자의 키 교환하는 통신 수를 줄이고자 하였다.

단계 1. 사용자는 키 교환에 사용하기 위하여 Diffie-Hellman 파라미터를 생성한다. 그리고 키 동의를 받기 위한 XKASS서비스 A, B 에 대한 인증서와 각 XKASS 서비스에 대한 목록 O', O'' 을 만든다. 이렇게 만든 값들을 사용자의 개인키로 서명을 해서 A에게 전달한다.

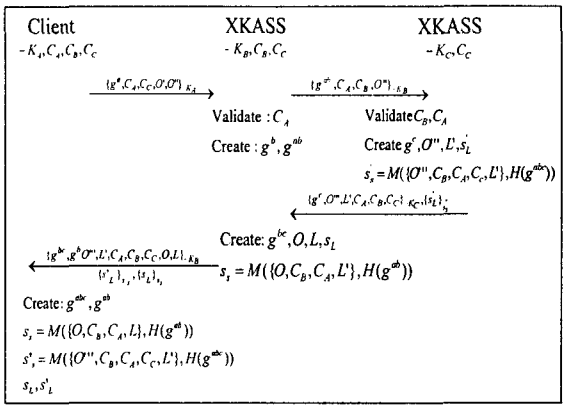
단계 2. B는 사용자에게서 받은 인증서를 이용해 유효한 사용자임을 확인하고 Diffie-Hellman 파라미터를 확인하고 자신의 Diffie-Hellman 파라미터를 만들 때 쓴 b값을 이용해 g^{ab}값을 만들어낸다. 만들어 낸 값과 C와 키 교환할 값(A에게서 전달되어온 값)을 전달해 키 교환 요청을 한다.

단계 3. C는 B가 보낸 값을 받아 인증서를 확인하여 유효한 사용자임을 확인한다. 그런 후 자신만이 알고 있는 Diffie-Hellman 파라미터 값인 (c, g^c)값과 B로부터 받은 값을 사용해 g^{abc}를 계산하고, g^{abc}를 이용해 s'_s 값을 만들어 낸다. 그런 후에 s'_s 값을 만들 때 사용한 값들을 자신의 개인키로 전자 서명하고, s'_s를 이용해 s'_L을 암호화해 B에게 전달한다.

단계 4. B는 C에게서 전달받은 값을 확인한 후, g^c 값에 자신만이 알고 있는 b값을 이용해 g^{bc} 값을 만들어낸다. B는 g^{abc}값을 만들어 낼 수 없으므로 s'_s 값을 만들어 낼 수 없다. Diffie-Hellman의 특성을 이용해 A는 B를 통해서만 g^{abc}값을 만들어 낼 수 있고, B는 g^a, g^b, g^c 값을 모두 다 알게 되지만 g^{abc}값을 만들어 낼 수 없다. B는 자신이 만든 값 g^{bc}, g^b 와 C에게서 받은 값들을 자신의 개인키로 서명을 한다. 그리고 자신과 A만이 만들 수 있는 s_s 값을 이용해 s_L 을 암호화한다. B가 생성한 값과 C에게서 받은 값들을 A에게 전달한다.

단계 5. A는 B에게서 받은 값을 확인한다. B에게서 받은 g^{bc} 값에 자신이 알고 있는 a값을 이용해 g^{abc}값을 만들고, g^b 값을 이용해 g^{ab}값을 만들어 낸다. 이렇게 만든 값을 이용해 B와의 비밀공유 값 s_s 를 만들어 내고, C와의 비밀 공유 값 s'_s를 만들어 낸 후 B와 C가 만들어서 전달한 비밀 키 값 s'_L, s_L 을 얻는다[그림3].

그림 3 제안하는 프로토콜 2



IV. 시스템의 안전성 분석

① 신원보호(identity protection) : 사용자는 인증서를 사용하지 않고 XKMS의 응답표현으로 유효한 사용자임을 밝히므로, 능동 공격자나 수동 공격자는 XKASS와 XKMS간의 비밀 공유 값을 알지 못하기 때문에 사용자의 신원을 확인 할 수 없다. 해쉬 함수의 일 방향 특성으로 능동 공격자나 수동 공격자는 XKMS의 응답표현을 알지 못한다.

② Forward Secrecy(FS) : Diffie-Hellman 파라미터의 특성에 의해 프로토콜에 참여하는 사용자의 비밀키가 노출되더라도 공격자가 이전에 도청된 세션으로부터 과거 세션 키를 계산할 수 없다.

③ Replay 공격에 안전 : 공격자가 정상적으로 사용된 메시지를 재 전송하여 프로토콜을 수행하여도 Diffie-Hellman 파라미터의 특성으로 인하여 올바른 비밀 공유값을 만들어 내기 어렵고, Diffie-Hellman 값을 새로 설정하여 정상적인 메시지의 부분을 사용한다하여도 XKASS에서 해쉬 함수를 확인하는 과정을 거치기 때문에 정상적인 해쉬 함수 값을 만들어 내지 못한다.

④ 중재자 공격(Man in the Middle Attack) : 공격자가 정상적으로 사용된 메시지를 중간에 자신이 만든 메시지와 교환해서 전달 한다하여도 공격자는 Diffie-Hellman파라미터 값을 계산해 내기 어렵고, 해쉬함수 값을 만들어내기 어려움으로 중재자 공격에 대해서 안전하다.

⑤ Denial of Service(DoS) : 사용자에게 priority를 주는 방식을 사용한다. 처음 사용자에게

는 priority를 낮게 주고 기존 사용자에게는 priority를 높게 주는 방식을 사용해 DoS 공격을 예방한다.

⑥ 효율성 : 통신의 수는 XKASS와 같지만 사용자는 공개키 암호화 방식과 전자서명 방식을 사용하지 않으므로 계산량을 줄 일 수 있다. 다중 영역에 대한 프로토콜에서는 사용자는 두 XKASS에 가서 메시지 교환을 받기 위해서는 4번의 통신량이 드는데 비해 하나의 XKASS를 통하면 2번의 메시지 교환량으로 통신 횟수를 줄일 수 있어서 효율성을 높였다.

- **시스템 비교**

아래의 표에서는 관련 연구에서 언급한 세 가지 프로토콜과 기존의 XKASS와 제안하는 시스템에 대하여 비교한 것이다.

표 2: 제안하는 프로토콜과 계산량 비교

계산량 비교	IKE		JFK		SIGMA		XKASS		제안1		제안2	
지수계산	1	1	1	1	1	1	1	1	1	1	1	1
서명계산	0	0	1	1	1	1	1	1	0	1	1	2
서명확인 계산	0	0	1	1	1	1	1	1	1	0	1	2
프로토콜수행	4+2		4		3+2		2		2		4	

V. 결 론

기존의 키 교환 프로토콜인 XKASS는 인증서 기반 공개키 암호화 방식을 사용한 키 교환 프로토콜을 사용하지 않고 키 교환 프로토콜을 수행하여 계산량을 줄이고, 사용자의 신원과 연결된 XKMS의 응답표현을 해쉬함수에 적용하여 사용함으로써 수동공격자나 능동공격자에 대하여 사용자의 신원을 보호하였다.

계산량 비교를 위해서 사용한 3가지 키 교환 프로토콜과 비교해서는 첫 번째 제안한 프로토콜은 계산량이 많이 드는 공개키 암호화 기법을 사용자에는 사용하지 않음으로 해서 계산적 효율성을 높일 수 있었고, 모바일 환경과 같은 사용자의 계산량을 줄여야하는 환경에 적합하다. 두 번째 제안한 시스템은 다른 키 교환 프로토콜에 대해서 서명계산과 서명 확인 계산량이 늘어났지만 하나의 XKASS를 통해서 다중영역의 서비스를 받을 수 있게 함으로 해서 사용자의 측면에서는 메시지 전달횟수를 줄임으로 해서 효율성을 높였다.

참고문헌

[1] XML Key Management Specification(XKMS) W3C Note 30 March 2001. http://www.w3.org/TR/2003/NOTE-xkms-20 01030.

[2] XML Key Management Specification(XKMS) Ver 2.0, W3C Working Draft 18 April 2003.

[3] 박남제, 문기영, 손승원, 송유진, 원동호, "안전한 전자거래를 위한 XML 키 관리 기술,"

정보보호학회지, pp. 72-82, June, 2003.

[4] XML Key Management Specification(XKMS) W3C Note 30 March 2001. Bindings W3C Working Draft 18 April 2003. http://www.w3.org/TR/2003/WD-xkms2-bindgs-20030418.

[5] XML Key Management Specification(XKMS 2.0) W3C Note 05 May 2003. http://www.w3.org/TR/2003/NOTE-xkms2-req-20030505.

[6] Phillip Hallam-Balker, "XML Key Agreement Service Specification(X-KASS),"

VerSign May 10th 2001.

[7] Canetti R, Krawczyk K, "Analysis of Key-Exchange protocols and their use for building secure

channels," *Proceedings of Eurocrypt*, Vol. LNCS 2045, 2001.

[8] William Aiello, Steven M. Bellovin, Matt Blaze, Ran Canetti, John Ioannidis, Angelos D, Keromytis, Omer Reingold, "Efficient, DoS-Resistant, Secure Key Exchange for Internet Protocols," *ccs'02, ACM 1-58113-612-91021*, November 18-22, 2002.

[9] Ran Canetti, Hugo Krawczyk, "Security Analysis of IKE's Signature-based Key Exchange Protocol," *Crypto 2002*, An abridged version of This paper appears in the proceedings of Crypto'2002.

[10] Whitfield Diffie and Martin E.Hellman, "New Directions in Cryptography," *IEEE Trans. on Information Theory*, Vol. 22, No. 6, PP. 644-654, 1976. bindgs-20030418