분산 디렉토리 시스템에서의 인증에 관한 연구

최성민*, 이인숙**, 장청룡**, 원동호*
* 성균관대학교 정보공학과
** 한국통신 연구개발단

A Study on Authentication for Distributed Directory System

Sung-Min Choi*, In-Sook Lee**, Chung-Ryong Jang**, Dong-Ho Won*
* SUNG KYUN KWAN UNIV.
Department of Information Engineering
** KOREA TELECOM RESEARCH CENTER

요 약

본 논문에서는 정보를 효율적으로 저장, 관리, 전송할 수 있는 기능을 담당하는 분산 디렉토리 시스템(X.500 시리즈)의 모델과 MHS와의 연계 방안을 알아 보고, 이 시스템에서의 인증 방식 중의 하나인 상세 인증(strong authentication)의 문제점과 그 해결 방안을 제시하였다. 또한, X.509 권고안에서는 디지탈 서명을 생성하는 구체적인 방법으로 특정 성질을 갖는 공개키 암호 방식인 RSA를 이용하였으나, 본 논문에서는 ElGamal 방식을 분산 디렉토리 시스템에 적용시킴으로써 RSA 방식이 아닌 다른 방식도 X.509에 적용 가능하다는 것을 보였다.

1. 서 론

컴퓨터 네트워크는 컴퓨터 기술과 통신 기술의 집합체로 오늘날 고도의 정보화 사회에서 요구되는 각종 서비스를 제공하고 있으며, 통신 매체를 통한 정보의 교환 및 공유와 관련하여 정보의 양적 증가와 부대 가치의 증가라는 측면에서 괄목할 만한 발전을 이루었다. 이러한 정보 산업의 발전과 더불어 정확성, 효율성이 요구됨에 따라 통신 네트워크가 다양해지고, 처리, 전송, 축적, 보관해야 할 정보량이 급증하고 있

다.[1] 이러한 방대한 정보의 효율적인 처리를 위해 사용자, 장치(device), 응용(application) 등과 같은 통신 엔티티에 대한 총괄적인 이름 영역(global name space)을 설정하고 그것을 분산시킨 분산 디렉토리 시스템의 연구가 활발히 진행되고 있다.[2,3,4,5] 그러나, 이러한 긍정적인 측면외에 분산 디렉토리 시스템에서 정보 내용의 변경과 불법적인 유출, 그리고 미확인 발신자 및 수신자 등에 의해 항상 위험을 받음으로써 정보의 보안성이 요구되고 있는 실정이다. 따라서 안전한 통신 환경을 구축하기 위하여 불법적인 사용자의 액세스를 막는 일과 서로 상대방의 정당한 사용자임을 확인시키는 인증에 관한 일이 우선적으로 행해져야 한다.

본 논문에서는 분산 디렉토리 시스템의 모델과 인증에 관한 내용을 살펴보며 MHS와의 연계 방안을 알아 보고, 기존의 인증 방식의 문제점과 해결 방안을 제시하였다. 또한, X.509 권고안에서 디지탈 서명을 생성하는 구체적인 방법으로 특정 성질을 갖는 공개키 암호 방식인 RSA를 이용한 것을 ElGamal 방식을 분산 디렉토리 시스템에 적용시킴으로써 RSA 방식이 아닌 다른 방식도 X.509에 적용 가능하다는 것을 보였다.[6,7,8]

2. 분산 디렉토리 시스템 모델

분산 디렉토리 시스템은 OSI 네트워크 상에서 사용자, 장치, 응용 등과 같은 통신 엔터티에 관한 총괄적인 이름 영역을 설정하고 분산시켜 관리하는 시스템으로서 통신 서비스에 관련되는 모든 정보를 수집, 자료화하여 효율적인 기능을 담당함으로써 컴퓨터 통신의 수준 높은 서비스 기능과 신뢰성을 향상시킬 수 있는 중요한 시스템이다.

디렉토리 시스템 모델에는 기능 모델, 기구 모델, 정보 모델, 보안 모델 등이 있다.

1) 기능 모델(functional model)

디렉토리 시스템은 디렉토리 사용자(user), DUA(Directory User Agent), DSA(Directory System Agent)의 집합으로 이루어졌으며, 하나의 DSA로 구성된 중앙 집중형(centralized) 시스템과 둘 이상의 DSA로 이루어진 분산(distributed) 시스템으로 나뉘어진다. DSA는 디렉토리의 일부분인 OSI 응용 프로세스로, DUA 및 또는 다른 DSA에게 DIB(Directory Information Base)에 대한 액세스를 제공한다.

디렉토리의 기능 모델은 그림 1과 같이 하나 또는 그 이상의 DSA가 사용자의 요구를 수행하기 위하여 국부 데이타 베이스(local data base)에 있는 정보를 사용하거나 다른 DSA와 상호 작용을 하는 형태를 나타낸 것이다.

기능 모델의 운용으로서는 참조(referral), 연쇄(chainning), 다중(multicasting), 혼성 모드(mixed mode) 등이 있다.

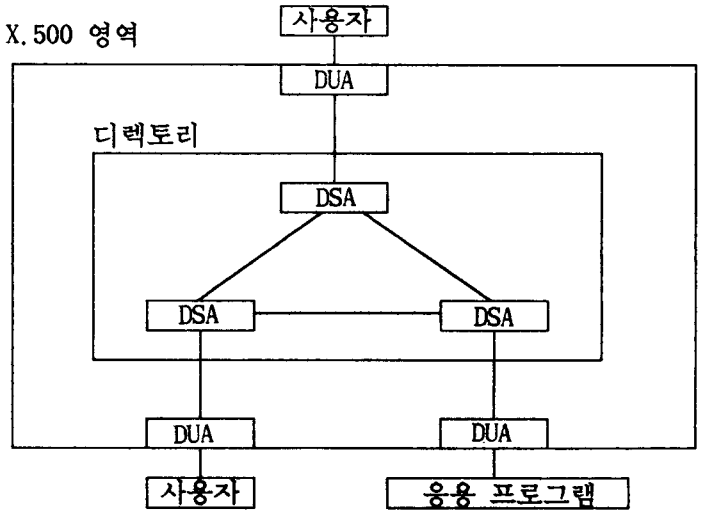

그림 1. 디렉토리의 기능 모델

2) 기구 모델(organizational model)

하나 이상의 DSA와 0개 또는 그 이상의 DUA 집합은 DMD(Directory Management Domain)라는 단일 기구에 의해 관리되어진다. DMD가 DUA를 관리한다는 것은 해당 DUA에 대하여 DMD가 유지보수 또는 소유 등의 서비스에 대한 모든 것을 책임진다는 것을 의미한다.

DMD는 기구의 서비스 목적에 따라서, 공공의 목적으로 서비스를 수행하는 기구인 ADDMD(ADministration DMD)와 기구의 사설 네트워크를 통하여 서비스를 수행하는 기구인 PRDMD(PRivate DMD)로 나뉘어진다.

3) 정보 모델(information model)

정보 모델은 디렉토리 내의 정보의 구성 형태를 나타내는 것이다. 디렉토리는 모든 정보를 식별 가능한 모든 개체(entity)로 정의되는 오브젝트 단위로 저장한다. 디렉토리가 액세스를 제공하고, 판독하거나 처리할 수 있는 모든 정보의 집합을 DIB (Directory Information Base)라 한다.

DIB는 각각의 오브젝트에 관한 정보를 포함하는 엔트리로 구성되어 있고 이 엔트리들의 내부 저장 구조는 트리 구조를 가지며 이를 DIT(Directory Information Tree) 구조라 한다.

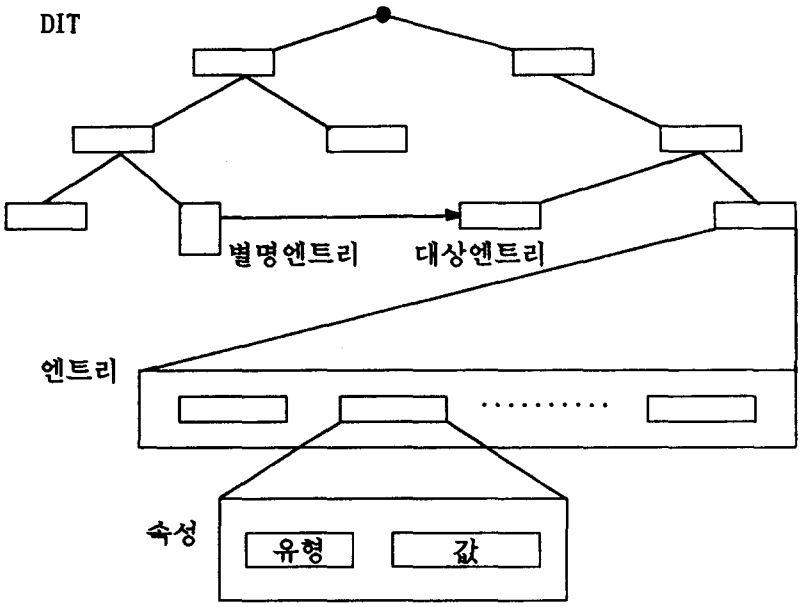

그림 2. 디렉토리의 정보모델

4) 보안 모델(security model)

디렉토리 시스템은 불법의 액세스로부터 디렉토리 시스템을 보호하고 디렉토리 사용자들에 대한 권리를 제어하기 위하여 디렉토리 정보 보안 정책(security policy)에 따라 각각의 디렉토리 사용자에게 액세스 권리를 정하고 디렉토리 정보를 보호해야 할 의무가 있다.

디렉토리 시스템이 채택한 보안 정책으로서는 각 디렉토리 사용자에게 액세스 권한을 지정하고 그 권한을 제어 및 관리하는 권한 부여(authorization)와 DSA와 디렉토리 사용자들 사이의 정당성을 확인하는 인증(authentication)이 있다. 본 논문에서는 4가지 모델 중 보안 모델에 중점을 두고자 한다.

2.1 MHS와 디렉토리의 연계 방안

X.400 계열의 권고안인 MHS(Message Handling System)는 디렉토리 서비스의 사용을 권장하고 있으나, 어떤 형태로 서비스를 사용한다는 내용은 없고 단지 어떤 목적으로 사용한다는 것과 디렉토리에 저장할 정보의 형태 등에 대해서만 기술되어 있다. 기능 모델의 측면에서 디렉토리 시스템은 DUA와 DSA로만 구성되어 있고, 그 중에서도 실제 서비스를 제공하는 것은 DSA로 비교적 단순한 형태를 가지고 있다. 이에 비해 MHS에서는 여러 종류의 오브젝트가 존재하며, 그 구성이 디렉토리에 비해 복잡하다. 여기서 어떤 오브젝트가 디렉토리에 접근하여 서비스를 사용할 것인가를 결정하는데 이것을

결정하기 위해서는 우선 MHS의 동작 방식을 알아야 하며, 특히 메시지의 전달시에 이름과 주소에 관계되는 행동이 큰 의미를 가진다. MHS에서는 AU(Access Unit)나 MS(Message Store)가 메시지를 MTA(Message Transfer Agent)에게 전달할 때 상대방의 O/R 이름(Originator/Recipient Name)을 주게되며, MTA는 이 O/R 이름에서 O/R 주소가 없는 경우 디렉토리 이름을 가지고 디렉토리로부터 O/R 주소를 알아내도록 한다. 따라서 MTA는 디렉토리 서비스를 이용하는 기본적인 사용자가 되고, UA도 그 특성상 디렉토리 서비스의 사용자가 될 수 있다.

이러한 사항들을 토대로 MHS와 분산 디렉토리 시스템의 연계 방안을 그림 3에 도시하였다.

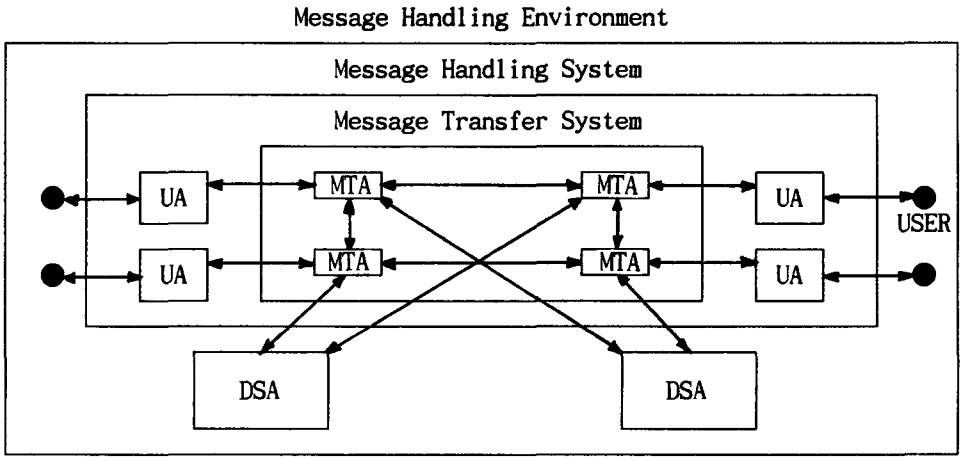

그림 3. MHS와 분산 디렉토리시스템의 연계방안

3. 분산 디렉토리 시스템에서의 인증

CCITT 권고안 X.509는 분산 디렉토리 시스템 상에서 인증과 공개키 획득 과정을 주요 내용으로 하고 있다. 인증에는 사용자의 고유명과 패스워드를 이용한 단순 인증(simple authentication)과 공개키와 비밀키를 이용하여 암호화 및 복호화하는 공개키 암호 방식을 적용시킨 상세 인증(strong authentication)이 있다. 본 장에서는 분산 디렉토리 시스템 상에서의 공개키 획득 과정을 알아 보고 상세 인증의 문제점과 해결 방안을 제시하였다.

3.1 사용자의 공개키 획득 과정

사용자가 인증 절차를 신뢰할 수 있게 하려면 우선 사용자가 신뢰하는 출처로부터

다른 사용자의 공개키를 얻어내야만 한다. 이러한 출처를 CA(Certificate Authority)라 하며, 이는 인증을 행하는 공개키를 확인하기 위하여 공개키 알고리즘을 사용한다. 그러므로, 사용자들이 인증을 행하기 전에 각 CA들을 신뢰하기 위해서는 순방 인증 경로(forward certificate path)에 의하여 상대방의 공개키를 얻고 역방 인증 경로(reverse certificate path)에 의하여 자신의 공개키를 통신하려는 상대방이 획득하면 CA들을 신뢰할 수 있다.

CA는 사용자의 고유명 및 공개키 등 정보의 집합체에 서명 또는 자신의 비밀키로 암호화 함으로써 사용자에 대한 인증을 생성한다.

고유명 A를 가진 사용자 인증은 CA에 의해 생성되며 다음과 같은 형태를 갖는다.

$CA\langle\langle A\rangle\rangle = CA\{SN, AI, CA, A, Ap, T_A\}$

단, SN: 인증의 일련 번호

AI: 서명하는데 사용된 알고리즘 식별자

CA: 인증 권한자

A_P: A의 공개키

T_A: 인증의 유효기간

인증에 기재된 서명은 CAP 자료를 가진 임의의 사용자에 의해 그 유효성이 검증될 수 있다. 다음의 ASN.1 데이터 유형은 인증을 나타내는데 사용될 수 있다.

```
Certificate ::=          SIGNED SEQUENCE{
    version                   [0]Version DEFAULT 1988,
    serial Number              SerialNumber,
    signature                 Algorithmidentifier
    issuer                    Name
    validity                  Validity,
    subject                    Name,
    subjectPublicKeyInfo     SubjectPublicKeyInfo}
Version       ::=            INTEGER {1988(0)}
SerialNumber ::=             INTEGER
Validity      ::=
SEQUENCE{
    notBefore    UTCTime,
    notAfter     UTCTime}
SubjectPublicKeyInfo ::=
    SEQUENCE{
       algorithm        AlgorithmIdentifier
       subjectKey       BIT STRING}
AlgorithmIdentifier ::=
    SEQUENCE{
```

algorithm OBJECT IDENTIFIER
parameters ANY DEFINED By algorithm
OPTIONAL}

사용자 A가 다른 사용자 B의 공개키를 얻고자 하는 경우에 CA<<B>>의 공개키를 얻으면 이 과정은 완료된다. A가 CA<<B>>의 공개키를 얻기 위해서는 각 인증 권한자의 디렉토리 엔트리에 있는 정보를 획득해야 한다. 이러한 인증의 종류는 타 인증 권한에 의해 생성되는 순방 인증과 인증 권한자 자신에 의해 생성되는 역방 인증 등이 있다. 이러한 인증의 존재는 사용자로 하여금 한 점에서 다른 점까지의 인증 경로를 구성할 수 있게 한다.

인증들은 "UserCertificate", "CACertificate", "CrossCertificatePair" 유형의 속성으로써 디렉토리 엔트리내에 저장된다. 이들 속성 유형들은 디렉토리에 알려져있다. 이 속성들은 동일한 프로토콜을 다른 속성으로써 사용할 때 그 효능을 나타낼 수 있다. 이러한 속성 유형의 규격은 다음과 같다.

```
UserCertificate ::=          ATTRIBUTE
    WITH ATTRIBUTE_SYNTAX Certificate
CACertificate ::=          ATTRIBUTE
    WITH ATTRIBUTE_SYNTAX Certificate
CrossCertificatePair ::=          ATTRIBUTE
    WITH ATTRIBUTE_SYNTAX CertificatePair
CertificatePair ::=
    SEQUENCE{
        forward [0] Certificate OPTIONAL
        reverse [1] Certificate OPTIONAL
        -- at least one must be present --}
```

그림 4는 DIT 단편의 가상 예를 나타낸 것으로, 사용자의 CA들이 계층적으로 형성되어 있고 사용자가 자신의 인증 권한자의 공개키와 자신의 공개키와 비밀키를 알고 있다고 가정한다. A는 B에 대한 인증 경로를 설정하기 위해 디렉토리로부터 다음과 같은 인증 정보를 얻을 수 있다.

X<<W>>, W<<V>>, V<<Y>>, Y<<Z>>, Z<<B>>

A가 이러한 인증 정보를 얻었다면, Bp를 알기 위해 순차적으로 인증 경로를 풀어야 한다.

Bp = Xp· X<<W>>, W<<V>>, V<<Y>>, Y<<Z>>, Z<<B>>

일반적으로 A는 B에서 A로의 복귀 인증 경로를 설정하기 위해 디렉토리로부터 다음과 같은 인증 정보를 얻어야 한다.

Z<<Y>>, Y<<V>>, V<<W>>, W<<X>>, X<<A>>

B는 A로부터 이러한 인증 정보들을 수신했을 때, Ap를 얻기 위하여 순차적으로 복귀 인증 경로를 풀어야 한다.

Ap = Zp·Z<<Y>>, Y<<V>>, V<<W>>, W<<X>>, X<<A>>

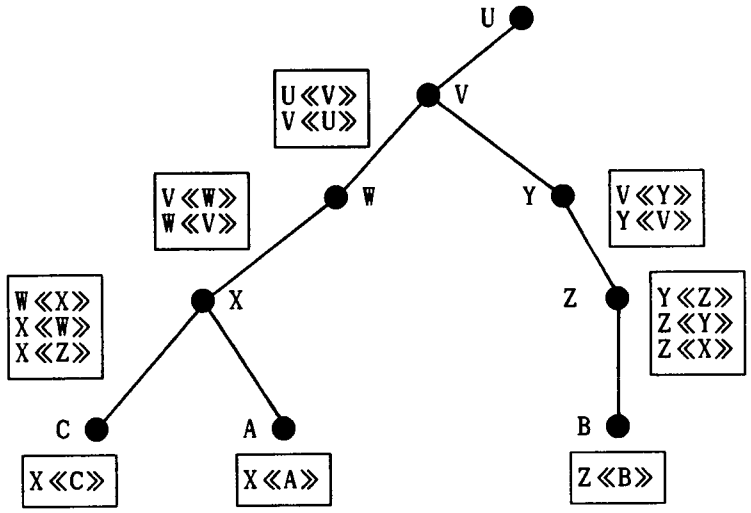

그림 4. 공개키 획득 과정

3.2 상세 인증

공개키와 비밀키를 가지고 암호 및 복호하는 공개키 암호 방식을 이용하는 인증 방식으로 단방향 인증, 양방향 인증, 3방향 인증 등이 있다.

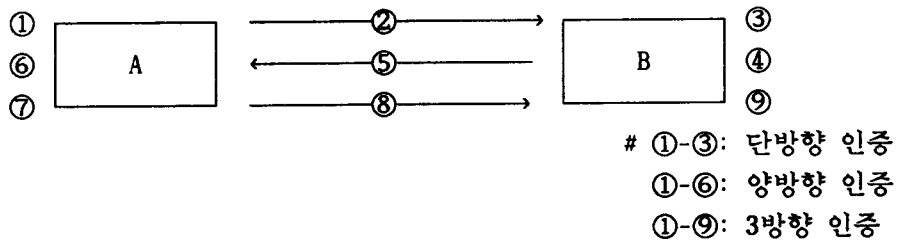

그림 5. 상세 인증

순서 ① A는 랜덤 번호 r_A를 생성한다.

단, r_A: replay attack을 검출하고, forgery를 방지하기 위하여 사용

순서 ② A는 다음과 같은 메시지를 B에게 전송한다.

B → A, A{t_A, r_A, B, sgnData, B_P[encData]}

단, B → A: A에서 B까지의 확인 경로

t_A: time stamp로써 토큰의 생성시간(선택적)과 만료시간으로 구성

B: B의 고유명

sgnData: 데이터 발신 인증이 디지탈 서명에 의해 제공

encData: 비밀키 등으로 사용될 비밀 정보

순서 ③ B는 다음과 같은 동작을 행한다.

- A의 인증 토큰이 만료되었는 지를 점검하기 위하여 B→A로부터 A_p를 얻음
- 서명을 확인하고 서명된 정보의 무결성 검증
- B 자신이 요구된 수신자인가를 점검
- 시간 스탬프가 현재인지 점검
- 랜덤 번호 r_A가 재시도 되었는지 점검

순서 ④ B는 r^B를 생성한다.

순서 ⑤ B는 B{t_B, r_B, A, r_A, sgnData, A_p[encData]}를 A에게 전송한다.

순서 ⑥ A는 다음과 같은 동작을 행한다.

- 서명을 확인한 후 서명된 정보의 무결성을 확인
- A 자신이 요구된 수신자인지 점검
- 시간 스탬프가 현재인지와 r_A가 재시도 되었는지 점검

①-⑥: 위의 과정과 동일 (시간 스탬프 t_A, t_B가 0일 수도 있고, 점검될 필요가 없음)

순서 ⑦ 수신된 r_A가 송신한 r_A와 동일한 지를 비교한다.

순서 ⑧ A는 A{r_B}를 B에게 전송한다.

순서 ⑨ B는 서명을 점검한 후, 서명된 정보의 무결성 점검한다.
수신된 r_B가 자신이 보낸 r_B와 동일한 지를 비교한다.

3.3 토큰 형태의 문제점과 개선 방안

A가 B에게 전송할 토큰의 일반적인 형태는 다음과 같다.

A{t_A, r_A, B, r_B, sgnData, B_p[encData]}

침해자 C가 A와 B사이의 토큰을 도청하더라도, C는 B_p[encData]를 복호할 수 없으므로 A가 B에게 전송하는 메시지의 내용은 알 수는 없다. 그러나, A가 B에게 전송한 토큰중 sgnData, B_p[encData]는 그대로 사용할 수 있다. 즉, 다음과 같이 새로운 토큰을 구성하여 B에게 전송할 수 있다.

C{t_C, r_C, B, r_B, sgnData, B_p[encData]}

C로부터 전송된 토큰을 수신한 B는 토큰이 C로부터 전송되었다는 것을 확인할 수 있

으므로 B_P[encData]를 복호한 메시지 역시 C로부터 전송되었다고 생각한다. 그러므로, B는 A와 통신했을 때의 응답과 동일한 응답을 C에게 한다. 따라서, C는 B가 자신에게 전송한 응답을 참조하여 A가 B에게 보냈던 메시지 내용이 무엇인 가를 추측할 수 있다.

이러한 문제점은 A가 B에게 보내는 토큰의 형태를 다음과 같이 개선하면, 해결 가능하다.

$A\{t_A, r_A, B, r_B, sgnData, B_P[encData \| A]\}$

즉, $B_P[\cdot]$에 encData와 A의 고유명을 concatenation함으로써 B가 B_P[encData]를 복호하였을 때, A의 고유명이 확인되므로 위에서 언급한 문제점은 해결된다.

또한, 토큰 형태를 다음과 같이 개선하여도 위의 문제점을 해결할 수 있다.

$A\{t_A, r_A, B, r_B, sgnData, B_P[encData], A_S[h(L)]\}$

단, $L = t_A \| r_A \| B \| r_B \| sgnData \| encData$

즉, C는 A가 전송했던 토큰을 획득하더라도 B_P[encData]를 복호할 수 없기 때문에 L을 구성할 수 없으므로 위에서 언급한 문제점은 해결된다.

3.4 3방향 인증 프로토콜의 문제점과 개선 방안

X.509의 9.4절에서 설명된 3방향 인증은 결점이 있고 완전한 대등 실체 인증(peer-entity authentication)을 제공하지 못한다. X.509의 권고안에서 보면, 시간 스탬프 t_A, t_B가 0으로 될 수도 있다는 것과 이러한 시간 스탬프를 점검할 필요가 없다고 하는 것이 상세 인증 절차의 안전성에 있어 문제점을 야기시킬 수 있다.

만약 A와 B가 위의 프로토콜을 0으로 세트하여 사용하고 악의의 참가자 C가 위의 3개의 메시지를 도청했다고 가정할 경우, C는 다음과 같은 과정을 통하여 B에게 A인 것처럼 위장할 수 있다.

① C가 A가 전에 사용했던 메시지를 B에게 전송함으로써 통신을 시작한다.

$C \rightarrow B$: $A\{0, r_A, B\}$

② B가 새로운 r_B'로 C에게 전송한다.

$B \rightarrow C$: $B\{0, r_B', A, r_A\}$

(B는 A에게 토큰을 전송한다고 생각하지만 실제로는 C에게 전송됨)

※ C는 A로 하여금 자신(C)과 인증을 행하도록 공작을 한다.

①' $A \rightarrow C$: $A\{0, r_A', C\}$

②' $C \rightarrow A$: $C\{0, r_B', A, r_A'\}$

(B가 C에게 준 r_B'를 사용)

③' A→C: A{r_B'}

③ C는 A와의 통신에서 얻은 A{r_B'}를 B에게 전송한다.

C→B: A{r_B'}

위와 같은 문제점은 X.509의 9.4절을 다음과 같이 수정하면 개선할 수 있다.

A{r_B} ⇒ A{r_B, B}

즉, 단계 ③에서 C는 A{r_B',C}을 얻으므로 C가 B에게 A인 것처럼 위장하는 프로토콜에 적합한 A{r_B',B}를 얻지 못한다. 그러므로 C는 B에게 A인 것처럼 더 이상 위장할 수 없다.

4. ElGamal 방식의 적용 방안

CCITT 권고안 X.509에서 제시된 인증은 공개키 암호 방식과 관용 암호 방식을 허용하고 있으나, X.509의 8.1절에서 정의된 디지탈 서명을 얻는 기법으로 매우 특별한 성질을 갖는 공개키 암호 시스템을 요구하고 있다. 실제로, 이것은 X.509의 부기 C에서 설명된 것처럼 RSA 공개키 암호 시스템의 사용을 요구하며, 다른 더 바람직한 디지탈 서명을 얻는 기법의 사용을 배제하고 있다. 그러므로, 본 장에서는 RSA 방식이 아닌 다른 방식을 이용할 수도 있다는 것을 실제로 보이기 위하여 ElGamal 방식[9]을 이용하여 분산 디렉토리 시스템에 적용하였다.

4.1. 사용자의 공개키 획득 과정

사용자가 신뢰하는 CA로부터 서로의 공개키를 획득할 때, 기존의 RSA 방식의 적용시 공개키 획득 과정은 DIT에서 인증 정보를 얻고 이것을 순차적으로 풀어서 상대방의 공개키를 획득하였다. 그러나 ElGamal 암호 방식의 적용시에는 인접한 인증 권한자들의 세션키를 이용하여 사용자의 공개 정보를 획득하여 통신하려는 사용자에게 주어야 하며, 또한 경로 확인을 위하여 공개 정보의 내용이 각 인증 권한자의 고유명을 포함하여야 한다. 그림 4의 계층적 구조의 예에서 사용자 A가 사용자 B의 공개키를 얻고자 하는 경우에 K_{AX}(SN,AI,X,W,V,Y,Z,y_B,T_B)를 얻으면 이 과정은 완료된다.(단, K_{AX}는 A와 X의 세션키이고, X,W,V,Y,Z는 계층적으로 배열된 인증 권한자의 고유명)

4.2. 상세 인증 절차

(시스템의 사전 준비 과정)

사용자 A, B의 비밀키: x_A, x_B

공개 정보: GF(p), 원시 원소 g, 사용자 A, B의 공개키: $y_A=g^{x_A}$, $y_B=g^{x_B}$

ElGamal 방식을 이용한 상세 인증 절차는 다음과 같다.

순서 1 A는 $k_A \in Z^*_{p-1}$를 선택하여 $r_A = g^{k_A} \pmod p$를 생성한다.

순서 2-1 A는 서명하고자 하는 메시지 m_A에 대하여 $x_A \cdot r_A + k_A \cdot s_A = m_A \pmod{p-1}$ 계산하여 s_A를 구한다.

2-2 $M=(t_A, B, m_A, r_A, s_A)$를 다음과 같이 암호화하여 전송한다.

키: $K = y_B^{k_A} \pmod{p-1}$

암호문: $C=(C_1, C_2)$

$C_1=g^{k_A} \pmod p$

$C_2=K \cdot M \bmod p$

순서 3 B는 다음과 같은 동작을 행한다.

- B는 B→A 에서 A의 공개키 y_A를 얻는다.
- 암호문 C를 자신의 비밀키를 이용하여 복호한 후 서명을 확인하고 서명된 정보의 무결성을 검증한다.

  식 $g^{m_A} = y_A^{r_A} \cdot r_A^{s_A} \pmod{p-1}$가 만족하는 지 확인한다.
- B 자신이 요구된 수신자인 지를 점검한다.
- 시간 스탬프가 현재인지와 랜덤 번호 r_A가 재시도 되었는지 점검한다.

순서 4 B는 $k_B \in Z^*_{p-1}$를 선택하여 $r_B = g^{k_B} \pmod p$를 생성한다.

순서 5-1 A는 서명하고자 하는 메시지 m_B에 대하여 $x_B \cdot r_B + k_B \cdot s_B = m_B \pmod{p-1}$ 계산하여 s_B를 구한다.

5-2 $M=(t_B, A, r_A, m_B, r_B, s_B)$를 다음과 같이 암호화하여 전송한다.

키: $K = y_A^{k_B} \pmod{p-1}$

암호문: $C=(C_1, C_2)$

$C_1=g^{k_B} \pmod p$

$C_2=K \cdot M \bmod p$

순서 6 A는 다음과 같은 동작을 행한다.

· 암호문 C를 자신의 비밀키를 이용하여 복호한 후 서명을 확인하고 서명된 정보의 무결성을 검증한다.

식 $g^{m_B} = y_B^{r_B} \cdot r_B^{s_B} \pmod{p-1}$가 만족하는 지 확인한다.

· A 자신이 요구된 수신자인 지를 점검한다.

· 시간 스탬프가 현재인지와 랜덤 번호 r_B가 재시도 되었는지 점검한다.

순서 7 수신된 r_A가 송신한 r_A와 동일한 지를 비교한다.

순서 8 A는 (r_B, B)를 위와 같이 암호문의 형태로 전송한다.

순서 9 B는 송수신된 r_B를 비교한다.

5. 결론

본 논문에서는 CCITT 권고안 X.509의 상세 인증 부분증 토큰 형태의 문제점과 시간 스탬프가 0으로 될 수도 있고, 점검될 필요가 없다고 함으로써 야기되는 문제점을 지적하고 해결 방안을 모색하였다. 또한, 권고안에서의 특정 방식(RSA)을 이용하는 것을 배제하고, 각종 인증 방식을 사용할 수 있도록 확장하기 위하여 본 논문에서는 ElGamal을 이용한 인증 방식을 분산 디렉토리 시스템에 적용하였다. 디렉토리 시스템의 구현시 이러한 인증 방식은 불법의 액세스로부터 시스템을 보호하고 사용자들에 대한 권리를 제어함으로써 안전한 통신 환경을 제공하는데 중요한 역할을 하리라 사료된다.

향후 연구 방향으로는 보다 강력한 인증 서비스를 제공할 수 있도록 확률적 암호 시스템, ZKIP 등 분산 디렉토리 시스템에 적합한 방식을 모델링하여야 하며, MHS, EDI 시스템 구축시 제공되어야 하는 각종 부가가치 서비스를 제공할 수 있는 방안을 강구해야 하겠다.

참고문헌

[1] 한국전자통신연구소, "현대암호학", 1991.

[2] C.Fritzner, L.Nilsen, A.Skomedal, "Protecting Security Information in Distributed Systems", IEEE Computer Society Symposium on Research in Security and Privacy, pp.245-254, 1991.

[3] Colin I'Anson, Chris Mitchell, "Security Defects in CCITT Recommendation X.509-The Directory Authentication Framework", pp.30-34, 1990.

[4] C.J.Mitchell, M.Walker and P.D.C.Rush, "CCITT/ISO Standards for Secure Message Handling", IEEE Journal on Selected Areas in Communications, pp.517-524, 1987.

[5] 장 청룡, 안 금혁, 원 동호, "개방형 통신망에서의 실체 인증", 한국통신학회 하계종합학술발표회 논문집, Vol.11, No.1, pp.134-137, 1992.7.

[6] CCITT, Recommendation X.501, "The Directory-Models", Dec.1988.

[7] CCITT, Recommendation X.509, "The Directory-Authentication Framework", Dec. 1988.

[8] CCITT, Recommendation X.400, "Message Handling-System and service overview", 1988.

[9] T.ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Trans. Inform. Theory 31, pp.469-472, 1985.