2001 년 한국정보처리학회 춘계 학술발표논문집 제 8 권 제 1 호

Secure XML 메시지 전송 시스템 설계†

송세봉, 장의진, 고 훈, 신용태
숭실대학교 컴퓨터학과
e-mail : bdream@hpcn.ssu.ac.kr

A Design of Secure XML Message Transfer System

Sebong Song, Uijin Jang, Hoon Ko, Yongtae Shin
Dept. of Computing, Soongsil University

요 약

인터넷 기술이 비약적으로 발전하면서, 인터넷을 통해서 여러 가지 정보를 주고받고 있다. 특히 은행 업무, 쇼핑 등 인터넷을 기반으로 행하는 전자상거래가 이미 일부에서 활발히 시행되고 있다. 최근 이외의 다양한 분야에서도 새로운 응용에 XML 을 적용하고 있다. W3C 가 주도적으로 표준화하고 있는 XML 은 SGML 의 부분집합으로서 문서의 내용에 관련된 태그를 사용자가 직접 정의해 확장성을 제공하고 있다. 그러나 이러한 XML 문서의 전송 중 개인정보 유출에 대한 위협이 있을 수 있는데, 본 논문에서는 신뢰성 있는 전송을 제공하기 위해 정보교환의 새로운 패러다임으로 등장한 XML 과 멀티캐스트 기술을 기반으로 한 암호화 메시지 전송 시스템을 설계하였다.

1. 서론

인터넷 사용자의 급격한 증가로 인해 최근 인터넷 상에서의 개인정보 유출이 대단한 문제점이 되고 있다. 또한 전송되는 정보를 중간에 가로채어 정보를 유용하는 범죄도 증가하고 있다. 이에 따라 많은 인터넷 사용자들의 피해가 늘어가고 있다. 그래서 최근 각 연구소와 학교에서는 개인정보보호를 위한 연구가 활발히 진행중이다. 특히 전자상거래가 활성화 되면서 이런 개인정보 보호 및 개인정보 유출 보호에 많은 관심을 가지고 있다.

이에 본 연구에서는 인터넷 상에서 전달되는 개인의 정보 유출을 방지할 수 있는 전송 시스템을 설계하고자 한다. 여기서 차세대 전자문서 교환의 표준화로 인정 받고 있는 XML(Extensible Markup Language)문서를 이용하여 열악한 네트워크 환경을 효율적으로 이용할 수 있게 해주는 멀티캐스트를 기반으로 한 보안에 초점을 두어 연구하였다.

2 장에서는 본 연구의 기반이 되는 XML 과 그에 따르는 암호화 기법, 그리고 멀티캐스트 기반에서의 정보보호에 대해 살펴 보겠다. 3 장에서는 멀티캐스트 기반에서의 XML 암호화 기법을 이용하여 제안하는 시스템에 대해 구체적으로 논하고, 4 장에서는 결론 및 향후 연구 방향을 제시한다.

† 본 연구는 2000 년 정보통신 산업기술 개발사업 과제의 지원으로 수행 중임.

2. 관련 기술

2.1. XML

XML 은 1996 년 W3C(World Wide Web Consortium)에서 제안한 것으로, 문서를 세분화하고 그 문서들의 각 일부를 구분하는 의미론적인 태그(tag)를 정의하는 규약의 집합이다. XML 은 특정 분야에 적용되는 의미론적인 구조적 마크업 언어를 정의하는 데 있어 사용되는, 다시 말해 문법(syntax)을 정의하는 메타 마크업(meta-markup) 언어이다[1].

XML 은 기존에 사용하던 HTML(Hypertext Markup Language)의 한계를 극복하게 해주며, SGML(Standard Generalized Markup Language)의 복잡함을 해결할 수 있도록 해준다. SGML 은 다양한 기능을 갖고 있음에도 불구하고 사용이 어렵고 DTD(Document Type Declarations) 생성이나 이해가 쉽지 않았다. 그리고 마크업 언어 자체가 아니라 마크업 언어를 생성하기 위한 간접적인 표준이라는 점 등으로 인해 널리 사용되지 못했다. 반면 HTML 은 간단하고 사용하기 쉽지만 SGML 에 의해 정의된 고정적인 마크업 언어이므

로 웹이 지니고 있는 다양함이나 동적인 특성을 쉽게 반영하지 못하고 있다. 이런 문제를 극복한 것이 바로 XML 이다.

2.2. XML Encryption

인터넷 상에서 데이터를 전송할 때 대부분 IPSec(IP Security)과 SSL(Secure Sockets Layer)과 같은 표준 암호화 프로토콜을 사용한다. 그러나, XML 문서 내에서 특성 요소(element)만을 암호화하여 정해진 사용자 외에는 그 내용을 볼 수 없도록 하는 암호화 기법에는 충분하지가 못하다. 이에 대한 개발이 W3C 의 XML Encryption WG(Working Group)에서 이루어지고 있다[2].

```
<Invoice>
  <Bookorder>... </Bookorder>
  <Payment>... </Payment>
  <Cardinfo>
    <Name>Gildong Hong </Name>
    <Expiration>08/2001 </Expiration>
    <Number>1234 5678 </Number>
  </Cardinfo>
</Invoice>
```

[그림 1] 암호화 되기 전의 XML 문서 예

[그림 1]과 [그림 2]는 XML Encryption 에 대한 간단한 예를 보여주고 있다. [그림 1]은 암호화 되기 전의 XML 문서이다. 여기서 암호화를 원하는 요소는 <Cardinfo>이다.

```
<Invoice>
  <Bookorder>... </Bookorder>
  <Payment>... </Payment>
  <EncryptedData>
DDAKBgNVBAsTA1RSTDEZMBcGA1UEAxMQSGlyb3N
oaSBNYXJ1eWFtYTAeFw05OTEyMTcwMDM3MzRa
Fw0wMDAzMTYwMDM3MzRaMEQxCzAJBgNVBAYTA
kpQMQwwCgY DVQQKEwNJQk0xDDAKBgNV==
  </EncryptedData>
</Invoice>
```

[그림 2] 암호화 된 후의 XML 문서 예

Encryption 프로세싱을 마치면 [그림 2]와 같이 원하는 요소만 암호화가 된다.

```
<EncryptionInfo
xmlns="http://www.w3c.org/2000/10/xmlenc" (Id=)?>
 (EncryptionMethod (Algorithm=))    :암호화 알고리즘
 (EncryptionPropertyList)?          :메타-정보
 <ReferenceList>
  (Reference (URI=)? (Xpath=)?)+   :암호화된 데이터 참조
 </ReferenceList>
 (KeyInfo
xmlns="http://www.w3c.org/2000/09/xmldsig#")
                                    :암호화 키
</EncryptionInfo>
```

[그림 3] <EncryptionInfo>요소 구문 개요

이러한 암호화 및 복호에 필요한 암호화 알고리즘 및 키잉 정보는 <EncryptionInfo>요소에서 찾을 수 있다. [그림 3]은 <EncryptionInfo>요소의 구문을 나타낸다[3].

2.3. DES (Data Encryption Standard)

DES 알고리즘은 ANSI 에서는 DEA(Data Encryption Algorithm)로, ISO 에서는 DEA-1 으로 명명하였고, 지난 20 년간(1998 년 까지) 세계적인 표준으로 사용된 64 비트 블록암호 알고리즘이다. DES 의 구조는 데이터 암호부와 키 생성부로 구성되어 있는데 먼저 키 생성부에서 생성된 48 비트의 16 개 라운드 키는 데이터 암호부의 각 라운드로 들어가 평문 블록과 함께 치환, 대치, 키 스케줄 등을 통하여 암호문을 만들어 내고, 복호는 암호화의 역순이다[4].

2.4. 멀티캐스트 정보보호

현재 IETF(Internet Engineering Task Force)의 MSEC(Multicast Security) WG 과 IRTF(Internet Research Task Force)의 SMuG(Secure Multicast Research Group)을 중심으로 멀티캐스트 정보보호에 대한 많은 연구들이 진행중이다[5]. 또한 그룹 키를 관리하기 위한 많은 방법들이 연구되고 있다. 그룹 키 관리 알고리즘들은 크게 중앙집중 방식, 분산환경 방식으로 나눌 수 있다. 그리고 이 두 가지 방식을 복합적으로 사용하는 방식도 있다. 중앙집중 방식은 하나의 키 서버가 그룹 키를 관리한다. 중앙집중 방식은 효율적이긴 하지만 그룹에 가입한 호스트의 수가 많아질수록 서버의 오버헤드가 커지는 확장성의 문제가 있다. 분산환경 방식은 복수 개의 키 서버를 두어 그룹 키를 관리한다. 키 서버를 위한 별도의 그룹 키의 관리가 필요하므로 중앙집중 방식에 비해 복잡해지지만 뛰어난 확장성을 가지게 된다. 그러나 보안의 관점에서는 하나의 키 서버가 그룹 키를 관리하는 중앙집중 방식이 유리하다[6].

3. Secure XML 메시지 전송 시스템

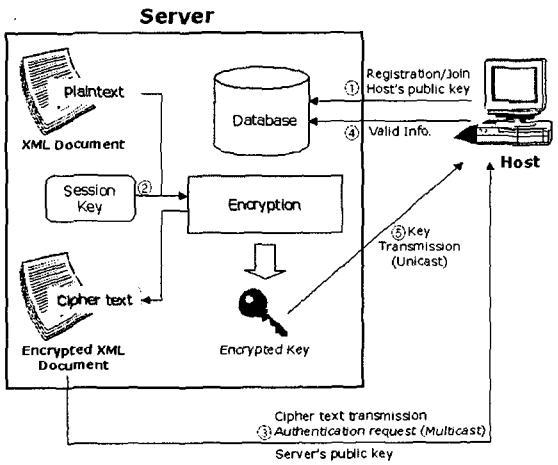
본 연구에서 암호화된 XML 메시지 전송 시 멀티캐스트를 사용한다. 이에 따르는 제한사항 2 가지는 다음과 같다.

1) Forward secrecy : 멀티캐스트 그룹을 떠나는 사용자는 그룹을 떠난 이후의 메시지를 복호 할 수 없어야 한다.
2) Backward secrecy : 멀티캐스트 그룹에 새로 가입한 사용자는 가입하기 이전의 메시지를 복호 할 수 없어야 한다.

위의 조건을 만족시킬 수 있는 방법으로 그룹에 새로운 사용자가 가입할 때마다 혹은 기존의 사용자가 탈퇴할 때마다 그룹 키를 변경시켜주는 방법이 있다. 이에 따르는 오버헤드가 있지만, 현재까지 해결책은 그 방법 외에는 없다.

3.1. 시스템 설계

[그림 4]에서는 본 시스템의 전체적인 흐름을 명시하고 있다. 우선 호스트와 서버는 각각 자신의 공개키와 비밀 키를 생성한다.



[그림 4] 시스템 구조

① 호스트는 서버에 개인의 정보를 등록하고, 생성되어 있는 그룹에 가입하게 된다. 이때 호스트는 이미 생성되어 있는 자신의 공개 키를 서버에게 전송한다.
② 서버는 세션 키를 생성한다. 세션 키를 사용하여 전송할 XML 메시지를 암호화한다.
③ 서버는 가입된 호스트에게 암호화된 메시지와 자신의 공개 키를 전송(멀티캐스트)하고, 세션 키 전송을 위해 해당 호스트에 대한 특정 정보(주민등록번호)를 요구하게 된다. 해당 호스트가 정당한 호스트인지 아닌지를 확인하기 위한 단계이다.
④ 간단한 인증을 위해 요청을 받은 호스트는 서버에게 그에 상응한 응답을 해준다.
⑤ 응답을 받은 서버는 데이터베이스에 저장되어 있는 개인정보와 비교한 뒤, 이상이 없으면 해당 호스트의 공개 키와 자신의 비밀 키를 이용해 암호화한 세션 키를 전송(유니캐스트)한다.

이렇게 키를 수신한 호스트는 서버의 공개 키와 자신의 비밀 키를 이용하여 암호화된 세션 키를 복호한다. 복호 된 세션 키를 이용하여 해당 메시지의 암호화된 요소를 복호 할 수 있게 된다.

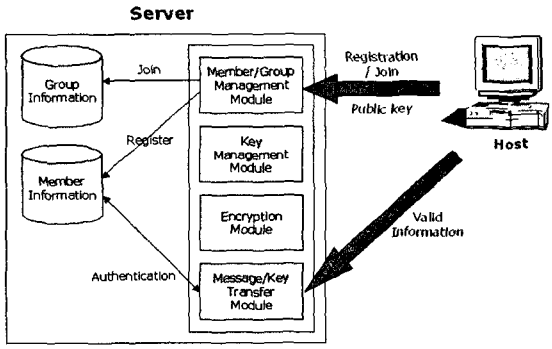
3.2. 서버의 역할

서버는 [그림 5]에서 볼 수 있듯이 크게 사용자 정보 및 그룹 관리, 암호화, 키 관리 그리고 메시지 및 키 전송의 네 가지 기능을 수행한다.

1) 사용자 정보 관리 및 그룹 관리

사용자 정보를 위한 데이터베이스는 개인정보 데이터베이스와 그룹정보 데이터베이스, 두 가지로 분류될 수 있다. 먼저 개인정보 데이터베이스에는 처음 호스트가 서버의 그룹에 등록을 요청했을 때 입력한 개인의 정보들, 즉 이름, 주민등록번호, IP 주소, 휴대폰 번호, E-mail 주소 등이 있으며, 또한 등록 과정에서 호스트의 공개키가 저장된다. 그룹정보 데이터베이스에는 그룹이 생성되었을 때의 그룹주소, 그룹이름, 그룹에 가입되어 있는 멤버들의 정보 그리고 암호화된 메시지의 복호를 위한 세션 키를 저장하고 있다. 그룹정보 데이터베이스는 차후에 서버가 클라이언트에 개인정보를 요청한 후 클라이언트로부터 전달되는 개인정보를 비교할 때에도 사용된다.

각 그룹정보 데이터베이스는 1 에서 n 까지의 그룹 호스트를 소유한다. 가입된 호스트가 0 이면 그룹은 소멸하게 된다.



[그림 5] 서버 구조

2) 암호화

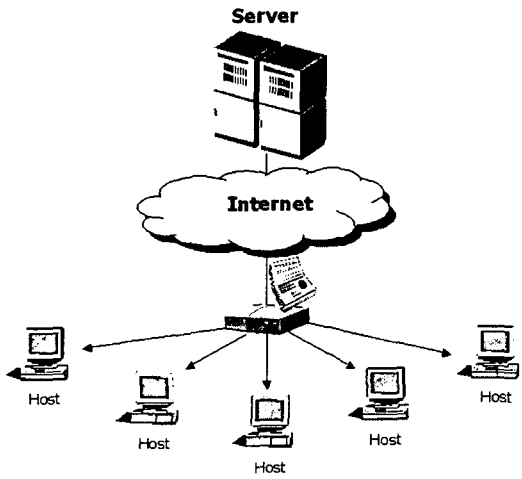
대칭 키 알고리즘에 비해 암호화 및 복호에 시간이 걸리는 비대칭 알고리즘의 특성 때문에 본 연구에서는 메시지의 암호화에 대칭 키 알고리즘인 DES 를 사용한다. 세션 키를 사용하여 XML 메시지의 해당 요소를 암호화하고, 암호화에 사용한 세션 키는 키 교환을 위해 비대칭 알고리즘인 RSA 를 사용해 암호화 된다. 여기서 사용하는 암호화 알고리즘들은 요구에 따라 바뀔 수 있다.

3) 키 관리

서버는 그룹의 생성과 함께 메시지를 암호화 하기 위한 세션 키를 생성한다. 이 세션 키는 해당 세션 동안 지속적으로 사용되며, 서버는 새로운 호스트가 그룹에 가입하거나, 기존에 가입되어 있는 호스트가 탈퇴할 경우 rekey operation 을 수행한다. 이로써 앞서 언급한 forward/backward secrecy 를 만족시킬 수 있다. 세션 키가 새로 생성되었을 경우 기존에 가입되어 있는 호스트들에게는 데이터베이스에 저장되어 있는 공개키를 참조하여 암호화 된 세션 키를 전송한다.

4) 멀티캐스트 메시지 전송

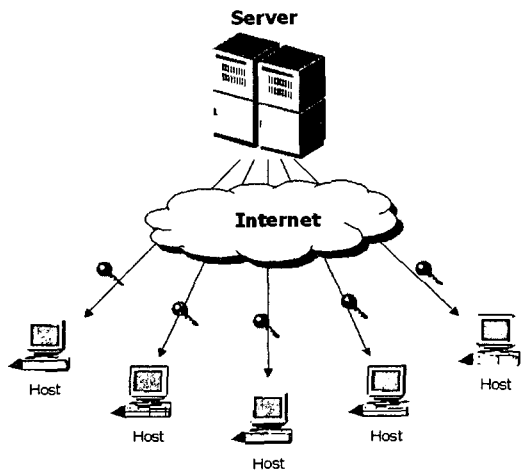
호스트가 가입되어 있는 그룹의 멀티캐스트 주소로 암호화된 XML 메시지를 멀티캐스트 한다[그림 7]. 복호를 위한 키는 멀티캐스트를 하지 않고 유니캐스트 한다.



[그림 7] 메시지 전송 (멀티캐스트)

5) 키 전송

XML 메시지를 암호화할 때 사용한 세션 키는 비대칭 알고리즘을 사용하여 암호화 한다. 이렇게 암호화된 세션 키는 각각의 호스트에게 유니캐스트 하게 된다. 이러한 키 전송은 2 가지 종류로 나눠진다. 하나는 서버의 요청된 정보에 먼저 알맞게 응답한 호스트에게 키를 전송하는 Anyone 방법, 그리고 한 호스트라도 메시지를 수신할 수 없다면, 다른 호스트들도 메시지를 읽지 못하는 All or Nothing 방법이 있다.



[그림 8] 키 전송 (유니캐스트)

5-1) Anyone

서버는 개인정보 요구에 알맞은 응답을 한 호스트에게 복호를 위한 키를 전송하게 된다. 서버가 호스트에게 암호화된 메시지를 전송하면서 동시에 호스트의 개인정보를 요구하게 된다. 호스트가 요구된 정보에 올바른 정보를 보내면, 서버를 이 호스트에게 복호를 위한 키를 전송하게 된다. 호스트는 이 키를 이용해서 암호화된 문서를 복호 하여 해당 문서를 읽을 수 있다[그림 8]. 먼저 올바른 정보를 호스트부터 서버는 차례대로 키를 전송한다. 올바른 정보를 보내지 못한 호스트는 복호 키를 수신할 수 없다.

5-2) All or Nothing

Anyone 과 마찬가지로 서버는 호스트에게 개인정보를 요청하게 되는데, Anyone 과 다른점은 같은 암호화 메시지를 받은 호스트 중에서 어떤 한 호스트가 잘못된 정보를 보내면, 올바른 정보를 보낸 다른 호스트들도 복호 키를 받을 수 없다.

4. 분석 및 향후 연구 방향

동일한 데이터를 여러 호스트들에게 전송할 때 멀티캐스트를 사용함으로써 네트워크의 대역 낭비를 방지할 수 있다는 장점이 있는 반면 여전히 개인 정보가 유출될 수 있다는 문제가 있다. 이러한 문제점 해결을 위해 제시한 시스템에서는 XML 문서 구조 중 특정 요소만을 암호화하는 기법을 통해 중요한 데이터를 안전하게 여러 호스트들에게 전송할 수 있다. 복호를 위한 키는 간단한 확인 절차에 의해 확인된 호스트에게만 전송되므로, 중요한 정보의 유출을 방지할 수 있다.

또한 현재 멀티캐스트 정보보호를 위해 다양한 기법들이 제안되고 있다[6]. 향후에는 메시지 암호화 뿐 아니라 멀티캐스트 정보보호에 있어 중요 논제가 되고 있는 그룹 인증 및 사용자 인증과 확장성을 가지는 키 관리 기법을 이용한 연구가 이루어져야 할 것이다.

참고문헌

[1] Elliotte Rusty Harold, "XML Bible," IDG BOOKS, 1999.
[2] XML Encryption Working Group, http://www.w3c.org/Encryption/2001/
[3] Takeshi Imamura, "Proposal: Syntax for keying Information & Encryption Algorithm," W3C XML Encryption Workshop, 2000.
[4] TradeSign, http://www.tradesign.net/
[5] IETF Secure Multicast Group (SMuG), http://www.securemulticast.org/
[6] 한근희, "멀티캐스트의 정보보호," 정보처리 제 7 권 제 2 호, 2000.
[7] Matthew J. Moyer, "A Survey of Security Issues in Multicast Communications," IEEE Network, 1999.