# SEED 알고리즘과 XML Encryption 적용 방안

차무홍*, 신동규*, 신동일*, 김주한**, 이재승**

세종대학교 컴퓨터공학과*, 한국전자통신연구원**

# A Plan of SEED Algorithm apply to XML Encryption

Moo-hong Cha*, Dong-kyu Shin*, Dong-il Shin*, Joo-han Kim**, Jae-seung Lee**

Department of Computer Engineering Sejong Univ*, ETRI**

## 요 약

ebXML, Web Services와 같은 XML을 기반으로 한 프레임워크 기술들이 발전하고 IT환경이 복잡해지면서 정보보안 기술의 발달과 그 필요성이 더욱 많이 요구되고 있다. UN공식 표준 언어인 XML 문서를 보호하기 위한 보안 메커니즘 가운데 정보의 기밀성을 보장하기 위하 XML 암호화에 대한 관심은 더욱 커지고 있는 상황이지만 XML 암호화 표준 명세에는 국내 표준 암호화 알고리즘인 SEED 암호화 알고리즘은 포함 되어 있지 않다. 따라서 ebXML, Web Services등을 국내 표준으로 비준 할 때에 국내 표준 암호화의 적용 방안이 필요하므로 본 논문은 국제 표준인 XML 암호화 명세에 국내 표준 암호화 알고리즘인 SEED암호화 알고리즘의 적용방안을 연구한다.

## I. 서론

EAM(Extranet Access Management). ebXML, Web Service등 XML을 기반으로 한 프레임워크들이 세계적으로 대두되면서 국내외 개발사에서는 이들을 개발하고 상용화하고 있으며, 국내의 전자상거래가 XML을 기반으로 발전을 하고 있다.[1][2] 이에 따라 현재 IT산업이 발전하면서 거대하고 복잡해짐에 따라 사용자의 정보보호가 중요과제가 되었듯이 XML을 사용하는 서비스들의 정보보호 역시 중요과제가 되었다.

기존 전자상거래 지원 보안서비스로는 Password, PKI, SSL기반에서 XML 정보보호기술로 발전하고 있다. XML 정보보호기술은 W3C와 OASIS에서 개발을 주도하고 있다. W3C에서는 XML보안을 위해 XML Encryption과 XML Signature와 같은 기술의 명세를 발표하였다. 이중 XML Encryption은 문서의 기밀성에 직접적인 영향을 주는 암호화를 다루고 있는 명세로서, 서비스의 이용자와 제공자 모두에게 기밀성을 제공해주는 필수적인 기술이다. 하지만 XML Encryption명세가 Recommendation으로 확정될 때 SEED 알고리즘은 ISO/IEC 국제표준회의에서 작업초안 단계를 통과 위원회 초안으로 넘어가던 시기로 XML Encryption은 SEED 알고리즘을 지원하지 않고 있다. SEED 알고리즘은 국산 128비트 대칭키 방식 암호 알고리즘으로 국내 표준 알고리즘이다. 이렇게 국내 표준 알고리즘의 미지원은 XML을 사용하는 서비스의 국내 표준화 진행시 아쉬운 점이다. 예를 들어보면 금융감독원에서 금융권 보안 시스템 강화 방안에서 국내 금융자동화 기기에 사용되는 암호 시스템으로 SEED 알고리즘을 채택하였는데 금융자동화기기와 XML 시스템을 연동할 경우 XML 시스템에서 SEED 알고리즘을 지원하지 못한다면 이 둘의 연동은 쉽지가 않을 것이다. 이에 본 논문에서는 XML Encryption에 SEED 암호화 알고리즘 연동을 위한 방안을 제시 한다.

## II. 관련연구

### 1. XML Encryption

기존 인터넷 암호화 방식은 IPSEC(IP Security

Protocol)이나 SSL(Secure Socket Layer) 같은 전송수준 암호화를 사용했는데 이 방식들은 확실한 보안을 제공하지만 XML의 장점을 상쇄하게 되는데 기존 방식은 XML 문서자체를 암호화하기 때문에 XML의 장점인 문맥상의 처리를 위한 최소한의 정보들마저 암호화를 하게 된다. XML Encryption 기술은 W3C에서 2002년 12월 10일 권고 되었다.[3]

XML Encryption을 사용하는 이유는 XML 도구로 생성되고 조정되며 분석되는 암호화 정보와 암호문을 갖기 바라기 때문이다[4]. XML 암호화는 데이터 암호화와 표현에 대한 처리의 결과를 XML로 명세한 것으로, 편리하게 XML 문서의 일부분 혹은 전체를 암호화할 수 있다. 때문에 XML 기반의 서비스에 자연스럽게 통합시킬 수 있다.

```
<MedInfo xmlns='http://mdedical.xx.yy'>
  <Identification>
    <Name>Micheal</Name>
    <Address>...</Address>
    <SSN>******-*******</SSN>
  </Identification>
  <Medical>a chest trouble</Medical>
  <Financial>
    <Insurance>...</Insurance>
    <Billing>...History...</Billing>
    <Payment>
      <Type>CreditCard</Type>
      <Number>1111 2222 3333 4444</Number>
      <Expires>04/02</Expires>
    </Payment>
  </Financial>
</MedInfo>
```

그림 1: 평문(Plain Text) XML문서

데이터를 암호화하기 위해 XML Encryption의 암호화 결과는 cipher data를 포함하거나 식별하는 'EncryptedData'요소이다. 'EncryptedData'요소는 암호화된 요소나 암호화된 내용을 대신하게 된다.

XML Encryption을 이용해서 XML문서를 암호화하는 단위는 XML요소 암호화와 XML요소 콘텐츠 암호화 그리고 다중 암호화 등이 있다.

[그림 1]은 기밀성을 요구하는 'Medical'요소와

```
<MedInfo xmlns='http://mdedical.xx.yy'
        xmlns:xenc='http://www.w3.org/2001/04/xmlenc#'>
  <Identification>
    <Name>Micheal</Name>
    <Address>...</Address>
    <SSN>******-*******</SSN>
  </Identification>
  <xenc:EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Element'>
    <xenc:CipherData>
      <xenc:CipherValue>A12B34C56</xenc:CipherValue>
    </xenc:CipherData>
  </xenc:EncryptedData>
  <Financial>
    <Insurance>...</Insurance>
    <Billing>...History...</Billing>
    <xenc:EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Element'>
      <xenc:CipherData>
        <xenc:CipherValue>D78E90F12</xenc:CipherValue>
      </xenc:CipherData>
    </xenc:EncryptedData>
  </Financial>
</MedInfo>
```

그림 2: Element 단위의 암호화

'Payment'요소 정보를 포함하고 있는 평문이다. [그림2]는 평문에서 'Medical'요소와 'Payment'요소의 문자 데이터를 선택해서 암호화한 결과 값이다. [그림 2]에서 점선으로 처리된 부분이 암호화를 거쳐서 'EncryptedData'요소로 변경된 부분이다. 위쪽에 있는 부분은 XML요소 암호화의 예이고 밑쪽에 있는 부분은 XML요소의 암호화와 XML요소 콘텐츠 요소 암호화의 예도 된다. XML 요소 콘텐츠 문자 데이터 암호화도 가능한데 [그림 3]이 그 예이다.

```
<Financial>
  <Insurance>...</Insurance>
  <Billing>...History...</Billing>
  <Payment>
    <Type>CreditCard</Type>
    <Number><xenc:EncryptedData
              Type='http://www.w3.org/2001/04/xmlenc#Content'>
      <xenc:CipherData>
        <xenc:CipherValue>D78E90F12</xenc:CipherValue>
      </xenc:CipherData>
    </xenc:EncryptedData></Number>
    <Expires>04/02</Expires>
  </Payment>
</Financial>
```

그림 3: XML요소 콘텐츠 문자데이터의 암호화

## 2. SEED 알고리즘

SEED는 대칭키 암호알고리즘으로, 블록 단위로 메시지를 처리하는 블록 암호알고리즘이다. 대칭키 블록 암호 알고리즘은 기밀성을 제공하는 암호 시스템의 중요 요소로 n비트 블록 암호 알고리즘이란 고정된 n비트 평문을 같은 길이의 n비트 암호문으로 바꾸는 함수를 말하며, 이 알고리즘을 통해 암·복호화를 수행하게 된다.[5]

SEED는 128비트 블록 암호 알고리즘으로 128비트의 평문 블록 단위당 128비트 키로부터 생성된 16개의 64비트 라운드 키를 입력으로 사용하여 총 16라운드를 거쳐 128비트 암호문 블록을 출력하게 된다.

### 1) SEED 알고리즘의 구조

SEED 알고리즘은 Feistel구조를 사용 다음과 같은 장점을 가진다.[6]

가. 라운드 함수에 관계없이 역변환이 가능하다.

나. 두 번의 수행으로 블록간의 완전한 diffusion이 이루어진다.

다. 알고리즘의 수행속도가 빠르다.

라. H/W 및 S/W로 구현이 용이하다.

마. 아직 구조상의 문제점이 발견되고 있지 않았다.

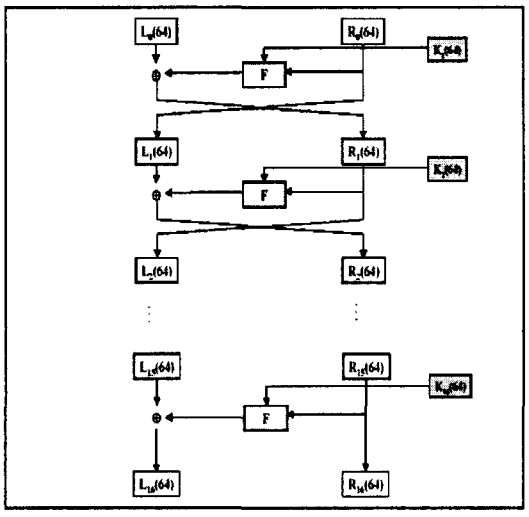


그림 4: SEED 알고리즘의 구조

### 2) SEED의 F함수

Feistel구조를 갖는 블록 암호알고리즘은 F함수의 특성에 따라 구분될 수 있는데 SEED의 F함수는 수정된 64비트 Feistel 형태로 각 32비트 블록 2개(C, D)를 입력으로 받아, 32비트 블록 2개(C′, D′)를 출력한다. 즉, 암호화 과정에서 64비트 블록 (C, D)와 64비트 라운드 키 Ki=(Ki,0Ki,1)을 F함수의 입력으로 처리하여 64비트 블록(C′, D′)을 출력한다.

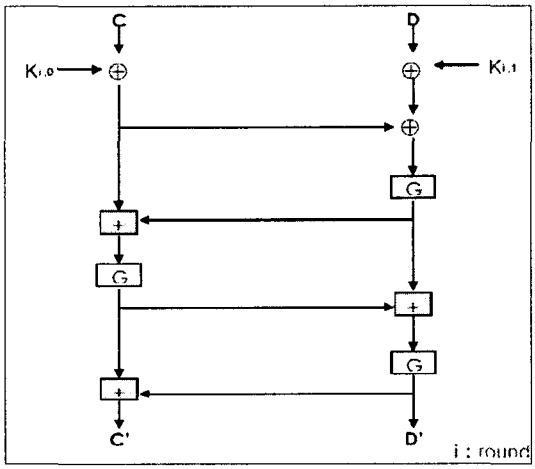


그림 5: F함수의 구조도

### 3) 라운드 키 생성과정 및 G함수

SEED의 라운드 키 생성과정은 128비트 암호키를 64비트씩 좌우로 나누어 이들을 교대로 8비트씩 좌/우로 회전이동한 후, 결과의 4워드들에 대한 간단한 산술연산과 G 함수를 적용하여 라운드 키를 생성한다. 라운드에 사용되는 라운드 키는 다음과 같은 방식으로 생성한다.

1) 128비트 입력키를 32비트씩 4객의 조각으로 나눈다. (A, B, C, D)

2) $K_{1,0} = G(A+C-KC_0)$ ; $K_{1,1} = G(B-D+KC_0)$로 1라운드 키를 생성(단, $KC_0$: 1라운드 상수)

3) $A||B = (A||B)^{>>8}$

4) $K_{2,0} = G(A+C-KC_1)$ ; $K_{2,1} = G(B-D+KC_1)$로 2라운드 키를 생성(단, $KC_1$: 2라운드 상수)

5) $C||D = (C||D)^{<<8}$

6) $K_{3,0} = G(A+C-KC_2)$ ; $K_{3,1} = G(B-D+KC_2)$로 3라운드 키를 생성(단, $KC_2$: 3라운드 상수)

7) 계속해서 16라운드 키를 생성할 때까지 반복

G 함수는 구현의 효율성을 위해 4개의 확장된 4바이트 SS-box들(4K bytes)의 exclusive-or로 구현할 수 있다.

8비트 입력 a, b, c, d,에 대한 G함수의 출력은 a′, b′, c′, d′이다.

a′ = (S1(a)&m0)⊕(S2(b)&m1)⊕(S1(c)&m2)⊕(S2(d)&m3)

b′ = (S1(a)&m1)⊕(S2(b)&m2)⊕(S1(c)&m3)⊕(S2(d)&m0)

c′ = (S1(a)&m2)⊕(S2(b)&m3)⊕(S1(c)&m0)⊕(S2(d)&m1)

d′ = (S1(a)&m3)⊕(S2(b)&m0)⊕(S1(c)&m1)⊕(S2(d)&m2)

m0 = 0xfc, m1 = 0xf3, m2 = 0xcf 그리고 m3 = 0x3f이다.

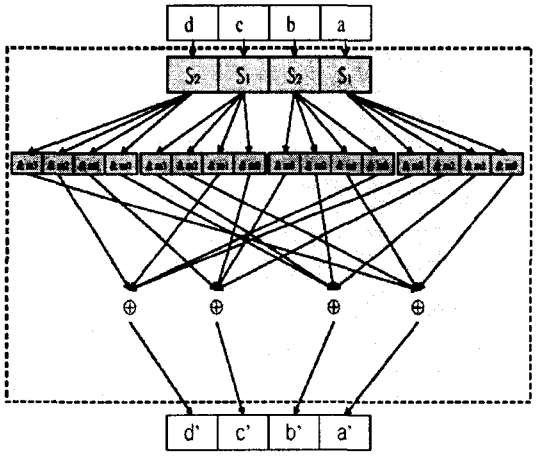


그림 6: G함수의 구조

# III. XML Encryption의 구현 방법

XML문서를 암호화하기 위해서는 문서 내에서 암호화할 부분과 암호화 알고리즘의 선택 그리고 키가 필요하다.

## 1. 암호화할 부분의 선택

관련연구에서 언급했듯이 XML의 암호화는 기존 암호화와 달리 문서의 일부분만 암호화할 수 있기 때문에 암호화 단위가 다양하다. 암호화할 부분을 선택할 때 단순히 요소 이름만 입력해서 구현 할 수도 있지만 XML문서의 특징상 다른 의미를 갖는 같은 이름의 요소가 존재할 수 있는 문제점과 요소의 콘텐츠 암호화에서 문자 데이터의 암호화 때문에 사용이 어렵다. 따라서 암호화 부분을 선택할 수 있는 가장 좋은 방법은 DOM 파서와 XPath를 사용하는 방법이다. DOM은 XML 문서의 내용에 접근할 수 있는 방법을 제공하는데 문서의 구조를 알아야 할 필요가 있을 때와 문서의 일부분을 변경하거나 이동시켜야 할 때 사용하고 XML문서를 객체로 표현한다. XPath는 XML 문서에서 논리적인 구조를 이용 문서의 일부분을 할당하는 것이다. XPath를 이용하면 요소 콘텐츠의 문자 데이터까지 정확하게 지정할 수 있다.

## 2. 암호화 알고리즘의 적용

XML Encryption 명세에서 요구하는 블록 암호화 알고리즘은 DES의 단점을 보안한 Triple-DES와 DES의 단점이 제시되면서 새로운 국제 표준으로 채택된 AES가 있다. 이러한 알고리즘을 라이브러리로 구현한 다음 앞서 선택한 암호화할 부분의 내용을 암호화하는 것이다. 여기서 암호화하는데 사용되는 키는 앞서 입력 받은 키를 이용하는데 키의 관리나 적용방법은 이 논문에서는 논하지 않는다.

암호화 알고리즘은 특정 알고리즘으로 고정되어 있거나 혹은 파라미터를 이용해서 입력받아 선택해서 암호화 할 수 있다. 암호화 알고리즘에 사용된 알고리즘은 암호문에서 'EncryptionMethod'요소의 'Algorithm'속성 값에 알고리즘의 식별자를 사용 암호문의 수신자가 확인을 할 수가 있다.

## 3. 암호화의 마무리

암호화 알고리즘의 결과로 나온 암호문을 다시 XML문서에 기술해주어야 한다. 앞서 언급했듯이 XML Encryption은 암호화한 부분을 'Encrypted-Data'요소로 대체를 하게 되는데, 암호문은 'EncryptedData'요소의 자식요소 'CipherData'의 자식요소 'CipherValue'의 문자 데이터 값이 된다. 그러므로 암호화의 마무리에서는 DOM을 사용 'EncryptedData'요소를 생성하고 XML Encryption 명세에 나와 있는 'EnryptedData'의 스키마를 준수해서 자식요소 및 속성을 생성한 다음, 암호화할 부분을 대체 XML 문서에 기술을 해 주게 된다.

# IV. SEED 알고리즘의 적용 방안

## 1. 적용 시나리오

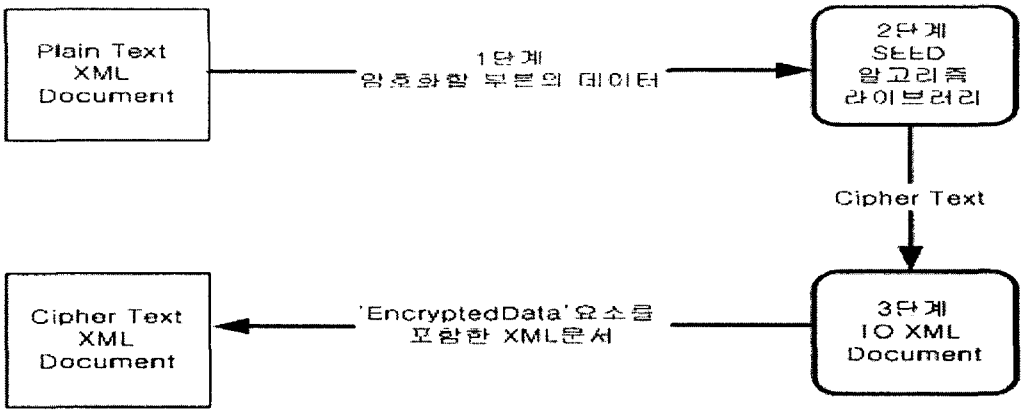


그림 7: SEED 알고리즘 XML Encryption 적용 시나리오

SEED 알고리즘의 적용 시나리오는 [그림 7]과 같다.

1) 암호화를 적용할 부분을 선택해서 데이터를 축출한다.

2) 축출한 데이터를 SEED 알고리즘 라이브러리를 이용 암호화 Cipher Text를 생성

3) Cipher Text를 이용 스키마를 적용 'EncryptedData'요소 생성 후 XML문서화 한다.

1단계는 기존 XML Encryption의 구현과 크게 다를 게 없으며, 2단계 역시 암호화 알고리즘을 SEED 암호화 알고리즘을 사용하면 되므로 큰 어려움은 없으며 암호화와 복호화의 처리는 XML 명세에 기술되어 있는 처리 규칙을 준수하면 된다. 하지만 3단계에서 XML문서화를 할 때에는 고려해야할 사항이 있다.

## 2. SEED적용시 고려해야할 사항

SEED 알고리즘이 W3C에서 권고하는 XML Encryption에 명세 되어 있지 않기 때문에, XML 문서화할 때는 'EncryptedData'요소의 자식요소 'EncryptionMethod'요소의 'Algorithm'속성의 값으로 들어가는 식별자를 고려해야 한다.

XML Encryption명세에서 요구하는 알고리즘의 식별자는 다음과 같다.

1) Required Triple-DES

http://www.w3.org/2001/04/xmlenc#tripledes-cbc

2) Required AES-128

http://www.w3.org/2001/04/xmlenc#aes128-cbc

3) Required AES-256

http://www.w3.org/2001/04/xmlenc#aes256-cbc

따라서 SEED 암호화 알고리즘의 식별자를 정의할 필요가 있다.

## V. 결론

국내외 전자상거래의 흐름이나 XML을 기반으로 한 서비스의 개발 및 발달은 XML 정보보안 기술의 국내 표준화가 필요로 할 것이며, XML Encryption의 국내 표준화시 국내 표준 암호화 알고리즘인 SEED 알고리즘의 지원이 필요할 것이다. 따라서 국내 표준 명세에서는 SEED 알고리즘을 위한 식별자 추가가 필요하며, 국가기관이나 산하연구소에서 식별자를 지원해 주어야 할 것이다.

## 참고문헌

[1] ebXML, http://www.ebxml.org
[2] Web Services, http://www.w3c.org/2002/ws/
[3] XML Encryption, http://www.w3c.org/Encryption/2001/
[4] E.Eastlke III, K.Niles, "XML 보안", 피어슨 에듀케이션 코리아
[5] SEED 암호화 알고리즘 명세 http://www.kisa.or.kr/seed/data/seed/Specification.pdf
[6] 128비트 블록암호화표준 알고리즘, TTA.KO-12.0004, 한국정보통신기술협회