제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

이종의 시스템에서 스키마 결합을 통한 효율적인 인증서 상태 검증에 관한 연구

황민구*, 이용준*, 오해석*
*숭실대학교 컴퓨터학과
e-mail : hminkoo@hanmail.net, yjlee@koscom.co.kr, oh@computing.ssu.ac.kr

A Study on the Efficient Certificate Status Validation by Combining Schemes in Heterogeneity System

Min-Koo Hwang*, Young-Jun Lee*, Hae-Seok Oh*
*Dept of Computer Science, Soong-sil University

요 약

공개키 기반 구조에서 가장 비용이 많이 드는 부분이 인증서가 유효한지 여부를 검증하는 것이다. 특히 고액의 주식 거래나 전자상거래에서 실시간 인증서 상태 검증은 반드시 필요하다. 그동안 연구되어 왔던 인증서 폐지 여부를 확인하는 방법으로 CRL, delta-CRL, OCSP, CRTs, CRSL 등의 방법들이 제안되었다. 하지만, 이들 방법들은 즉시성과 네트워크 비용간의 trade-off가 발생하는 문제점이 있다. 본 논문에서는 이종의 시스템에서 인증서 사용 용도에 따라 실시간성이 요구되는 인증서는 OCSP를 통해서 그렇지 않은 인증서는 CRSL을 통해서 폐지 여부를 검증하여 각각에 맞는 서비스를 제공해 주는 방안을 제안한다.

1. 서론

인터넷이 대중화되면서 어떠한 거래를 할 때 서로 대면하지 않고도 인터넷상으로 신속하게 문서를 주고받을 수 있게 되었다. 하지만, 인터넷을 통해 흘러가는 정보가 전 세계의 네트워크 망에 노출되어있다는 사실 때문에 불법적인 도청이나 위조와 변조, 신분 위장 등 문제점이 발생한다.

이러한 보안성을 제공하기 위해서 개인키와 공개키를 사용하는 공개키 기반 구조(PKI : Public key Infrastructure)를 구축하고 있다[1]. 공개키 기반 구조는 기밀성, 인증성, 무결성, 부인방지를 모두 제공해준다.

그런데, 사용자의 공개키는 공개되어 있기 때문에 다른 사람으로부터 위조될 수 있다는 문제점을 가지고 있다. 이를 해결하기 위해서 신뢰할 수 있는 제 3자인 인증기관(CA : Certificate Authentication)이 공개키를 포함한 기타 정보를 가진 인증서를 발급하고 CA의 개인키로 서명하여 공개된 디렉토리 서버에 개시한다.

그런데, 만일 사용자의 실수로 개인키가 손상되었거나 노출되었거나 자격이 박탈되었을 경우 유효기간 내에 인증서를 폐지할 수 있다. 이러한 경우 사용자는 인증서를 사용하기 전에 반드시 인증서의 폐지 여부를 확인하는 인증서 검증 과정을 수행해야 한다.

인증서 폐지 여부를 확인하는 방법으로 인증서 폐지 목록(CRL), 온라인 인증서 상태 확인 프로토콜(OCSP), 인증서 폐지 트리(CRTs)등이 제안되었다. 하지만, 이들 방식은 실시간성과 네트워크 부하(비용)간의 trade-off가 발생하는 문제점이 있다.

본 논문에서는 이종의 시스템에서 효율적인 인증서 검증 서비스를 제공하기 위해 인증서 사용 용도에 따라 실시간이 요구되는 인증서와 그렇지 않은 인증서를 분류하여, 실시간성이 요구되는 인증서는 OCSP를 통해서 폐지 여부를 검증하고 그렇지 않은 인증서는 CRSL을 통해서 폐지 여부를 검증하여 각각에 맞는 서비스를 제공해 주는 방안을 제안한다.

2.관련연구동향

2.1인증서폐지목록(CRL)

CRL은 인증서의 폐지 여부를 확인할 수 있는 가장 일반적인 방법으로 CA가 인증서 폐지 목록을 주기적(보통 하루에 한번)으로 생성하여, 디렉토리에 개시한다. 인증서를 사용하기 위해 사용자는 디렉토리에서 인증서 폐지 목록을 다운로드 받아 사용한다.

CRL 방식은 주기적인 발행으로 인해 실시간 인증서 상태를 제공해 주지 못하며, 사용자는 CRL을 매번 다운로드 받아야 하기 때문에 네트워크 비용이 많이 든다는 비판을 받아왔다[2].

2.2delta-CRL

delta-CRL은 CRL의 단점을 개선하고자 가장 최근에 폐지된 인증서만을 포함하는 인증서 폐지 목록이다[3]. 따라서

사용자는 최초로 받은 CRL에 delta-CRL을 덧붙이기 때문에 네트워크 부하가 줄어든다.

delta-CRL은 CRL에 비해 실시간성을 주고 있지만, 이 역시 주기적으로 업데이트 되기 때문에 완벽한 실시간성을 제공해 줄 수없다는 단점이 있다.

2.3온라인인증서상태검증프로토콜(OCSP)

OCSP는 기존의 CRL 방식이 실시간 인증서 상태 정보를 제공해 주지 못한다는 문제점을 해결하기 위해 제안되었다[4]. 사용자는 특정 인증서의 상태를 OCSP 서버에게 요청하면, OCSP 서버는 요청된 인증서 상태 정보를 포함한 응답 메시지를 생성한 후, 이를 CA의 개인키로 서명하여 사용자에게 전송한다. 사용자는 이 응답 메시지의 서명을 CA의 공개키로 검증한 후, 인증서 폐지 여부를 확인 할 수 있다.

OCSP는 실시간성이 요구되는 고액의 증권 거래나 현금거래에서 이용된다. 하지만, 각각의 인증서에 대해 실시간적으로 인증서 폐지 여부를 확인하고 전자 서명해야 하므로 OCSP 서버는 과부하가 걸리게 된다는 문제점을 가지고 있다.

2.4인증서패기트리(CRTs)

CRTs는 인증서 폐지 목록을 이진 트리의 리프 노드로 지정하는 방식으로, 한 방향 해쉬 함수와 경량 전자 서명을 제공함으로써 CRL과 OCSP의 문제점을 해결하기 위해 제안되었다[5, 6]. 즉, CRL에서 디렉토리와 사용자간의 통신비용이 높다는 단점을 한 방향 해쉬 함수를 사용하여 해결하고, OCSP 서버에 부하가 많이 걸린다는 단점을 해쉬한 값을 서명함으로써 해결한다.

하지만 인증서가 폐지되어서 트리에 추가되거나 인증서가 만료되어서 트리에 삭제될 때 마다 트리를 변경해야 하는데, 이때 비용이 많이 든다는 단점이 있다. 또한 사용자가 인증서 폐지 여부를 검증하기 위해서 해쉬값을 여러 번 계산해야 된다는 번거로움이 있다.

2.5인증서폐지스킵리스트(CRSL)

CRSL은 CRTs의 이진 트리 대신에 skip lists를 사용하여 CRT의 단점인 트리를 변경하는데 부하가 많이 걸린다는 문제점을 해결하고자 제안한 것이다[7]. skip lists는 폐지된 인증서를 linked list의 순서로 표현하고, 맨 앞에 최소값을, 맨 뒤에 최대값을 삽입하여 맨 아래 레벨에 놓는다. 그리고 그 레벨에 있는 폐지된 인증서들을 임의로 1/2의 확률로 선택하고 해쉬하여 다음 레벨의 리스트로 만든다. 이것을 반복적용하고 루트 레벨의 최소값을 CA가 전자 서명한다.

하지만, 이 방식도 CRTs와 마찬가지로 인증서를 검증하기 위해서 사용자가 직접 해쉬값을 계산해야 된다는 문제점이 있다.

3.제안한방법

본 논문에서는 기존에 제안 했던 방법들이 사용했던 중앙집중 방식이 아닌 분산 방식을 이용하여, 고액의 증권 거래와 같은 실시간성을 요구하는데 사용하는 인증서인 경우 OCSP를 통해서 인증서 폐지 여부를 검증하고, 그렇지 않고 사적인 e-mail을 보내는데 사용하는 인증서인 경우 CRSL로 인증서 폐지 여부를 검증하는 것을 제안한다.

3.1제안한시스템구조

제안한 시스템 구조는 그림 1에서 보여준다.

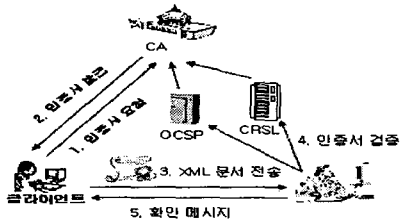

그림 1. 제안한 시스템 구조

만일 사용자가 증권 거래를 하기 위해 증권 사이트에 접속하게 되면, 사용자는 증권 사이트의 인증서를 OCSP를 통해서 검증하고, 증권 사이트는 사용자가 매매하고자 하는 정보들을 암호화된 XML 전자 서명 문서 형태로 받아 OCSP를 통해서 인증서를 검증한다. 검증이 성공하면, 증권 매매를 허용한다는 메시지를 사용자에게 보낸다.

위의 예제에서 W3C에서 표준으로 확정된 XML 전자 서명을 사용하는 이유는 어떠한 디지털 컨텐츠에도 적용이 가능하기 때문이다[8]. 그림 2는 XML 암호와 전자 서명을 한 예를 보여준다.

```
<?xml version="1.0">
<consumer>
  <order>
    <book serial_num="SKE7859" date="2002-06-23"> XML and Java </book>
    <quantity>3</quantity>
    <manufacture_name>Addison-Wesley </manufacture_name>
  </order>
  <payment>
    <consumer_name> kil-dong, hong </consumer_name>
    <card_number secure="true">UxoShznutAZxTK... </card_number>
    <expiration_date secure="true">Kdqw1n3+ veE=</expiration_date>
    <signature>mc0CFQCou78IIAMpmM1eknZAq2j3kfcCNJQndkx9Lu3...</signature>
  </payment>
</consumer>
```

그림 2. XML 암호와 전자서명의 예

3.2제안한알고리즘

[1] 사용자와 사용자 간의 거래

① 송신자는 문서에 신용카드 번호, 비밀번호, 유효기간 등의 중요한 부분이 있으면 그 부분을 선택하여 <secure>로 태깅을 한다. 만일 중요한 부분이 없으면 생략한다.

```
if is(secure_information in document ) then
  tagging(<secure>)
end
translate(XML document)
if is(<secure>) then
  validate(OCSP)
else
  validate(CRSL)
end
if valid certificate then
  if is(<secure>) then
    receiver_public_key(between <secure> to </secure>) ··· doc_1
  end
  sender_private_key(hash(doc_1)) ··· doc_2
  send(XML signature form(doc_1, doc_2), receiver)
 else
  print(invalid certificate)
  exit
end
```

그림 3. 송신자 알고리즘

```
receive(XML signature form)
extract (doc_1, doc_2)
receiver_private_key(doc_1) ··· doc_3
if is(<secure>in doc_3) then
  validate(OCSP)
else
  validate(CRSL)
end
if valid certificate then
  if sender_public_key(doc_2) == hash(doc_1) then
    send(sucess, sender)
  else
   print(failure)
  end
 else
  print(invalid certificate)
  exit
end
```

그림 4. 수신자 알고리즘,

② 그 문서를 XML 문서로 변환한다.
③ <secure>안에 있는 부분을 수신자의 공개키로 암호화를 한다. <secure>가 없으면 생략한다.
④ 암호화된 문서를 해쉬한 후, 자신의 개인키로 서명을 한다.
⑤ ③번 문서와 ④번 문서를 XML 전자 서명 형식으로 수신자에게 보낸다.
⑥ 수신자는 ④번 문서를 추출해서 송신자의 공개키로 서명을 풀고, ③번 문서를 추출해서 해쉬한 값과 비교하여 같은지 여부를 확인한다.

[2] 사용자와 Content Provider(CP) 간의 거래

```
connect (CP server)
if cash transactions then
  validate(OCSP)
  tagging(secure_information in form, <secure>)
else
  validate(CRSL)
end
if valid certificate then
  translate(XML document)
  if is(<secure>) then
    CP_public_key(between <secure> to </secure>) ··· doc_1
  end
sender_private_key(hash(doc_1)) ··· doc_2
  send(XML signature form(doc_1, doc_2), CP)
 else
  print(invalid certificate)
  exit
end
```

그림 5. 송신자 알고리즘

```
receive(XML signature form)
extract (doc_1, doc_2)
CP_private_key(doc_1) ··· doc_3
if is(<secure>in doc_3) then
  validate(OCSP)
else
  validate(CRSL)
end
if valid certificate then
  if sender_public_key(doc_2) == hash(doc_1) then
    send(sucess, sender)
  else
   print(failure)
  end
 else
  print(invalid certificate)
  exit
end
```

그림 6. CP 알고리즘

① 송신자는 CP의 서버에 접속을 한다.
② 만일 현금 거래와 관련된 서비스를 받고 있으면 OCSP로 CP의 인증서 폐지 여부를 검증하고 신용카드 번호 등의 비밀 정보를 <secure>로 태깅한다. 그렇지 않고 만일 현금 거래와 관련된 서비스를 받고 있지 않다면 CRSL로 CP의 인증서 폐지 여부를 검증한다.
③ CP의 인증서가 유효하다면, CP로 전송되는 폼을 XML 문서로 변형하고, <secure>안에 있는 부분을 CP의 공개키로 암호화를 한다. <secure>가 없으면 생략한다.
④ 암호화된 문서를 해쉬한 후, 자신의 개인키로 서명을 한다.
⑤ ③번 문서와 ④번 문서를 XML 전자 서명 형식으로 CP에게 보낸다.
⑥ CP는 ④번 문서를 추출해서 송신자의 공개키로 서명을 풀고, ③번 문서를 추출해서 해쉬한 값과 비교하여 같은지 여부를 확인한다.

3.3기대 효과

본 논문에서 제안하는 방식인 실시간성을 요구하는 인증서와 그렇지 않은 인증서를 분류함으로써 얻어지는 기대 효과는 다음과 같다.

1. OCSP보다 서버의 부하가 줄어든다.

실시간성을 보장하기 위해 모든 인증서가 OCSP를 통해서 검증하게 되면 OCSP 서버에 부하가 많이 발생한다. 본 논문에서는 고액의 현금 거래와 같은 안전성과 실시간성이 중요하게 요구되는 경우에만 OCSP 서버를 통해 인증서 폐지 여부를 검증하므로 서버의 부하를 줄일 수 있다.

2. CRSL보다 실시간성을 보장한다.

네트워크 부하(비용)를 줄이기 위해 모든 인증서가 CRSL을 통해서 검증하게 되면, 검증자가 직접 해쉬 값을 여러 번 계산해야 하므로 인증서 폐지 여부를 확인해야 하는데 시간이 걸린다. 따라서 현금 거래가 아니거나 기타 중요한 보안이 요구되지 않는 경우에만 CRSL을 통해서 인증서 폐지 여부를 확인하고, 중요한 서비스를 받아야 할 경우에는 OCSP를 통해 검증하면 보다 안전성과 실시간성을 보장할 수 있다.

3. 부분 암호화를 사용하여 암호화 속도가 개선된다.

송신자가 문서를 XML로 변환하고, 보안이 요구되는 정보만을 따로 <secure>로 태깅하여 그 부분만을 암호화한다. 따라서 빠르게 암호화를 할 수 있다는 장점이 있다.

4.결론 및 향후 연구 방안

인터넷 뱅킹이나 고액의 증권 거래, 현금 거래를 할 때 실시간적으로 인증서 폐지 여부를 확인해야 한다. 하지만, 기존의 인증서 폐지 여부를 확인하는 방법인 CRL, delta-CRL, OCSP, CRT, CRSL을 통해서는 실시간성과 네트워크 부하 간에 trade-off가 발생한다는 문제점이 있다. 본 논문에서는 보안성과 실시간성을 요구하는 경우와 그렇지 않은 경우를 분류하여, 보안성이 요구되는 경우에는 OCSP를 통해서 인증서 폐지 여부를 검증하고, 그렇지 않은 경우는 CRSL로 검증하는 것을 제안하였다. 이 방식은 CA가 OCSP와 CRSL을 모두 유지해야 한다는 단점이 있지만, 사용자의 요구에 맞게 효율적으로 서비스를 할 수 있다는 큰 장점이 있다. 향후 연구 방안으로는 요즘 이슈가 되고 있는 무선 PKI에서 효율적으로 인증서 폐지 여부를 검증할 수 있는 방안에 대해 연구하는 것이다.

참고 문헌

[1] Pay Hunt. "PKI and Digital Certification Infrastructure.", IEEE, 2001.

[2] Ronald L. Rivest. "Can We Eliminate Certificate Revocation Lists?" In Proceedings of Financial Cryptography 1998. Springer, February 1998.

[3] Warwick Ford and Michael S. Baum. "Secure Electronic Commerce." Prentice Hall PTR, 1997.

[4] Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Carlisle Adams. "X.509 Internet Public Key Infrastructure: Online Certificate Status Protocol." IETF RFC2560, June 1999.

[5] Paul Kocher. "A Quick Introduction to Certificate Revocation Trees(CRTs)." Technical report, ValiCert, 1999.

[6] Moni Naor and Kobbi Nissim. "Certificate Revocation and Certificate Update." In Proceedings of the 7th USENIX Security Symposium, 1998.

[7] M. T. Goodrich and R. Tamassia. "Efficient authenticated dictionaries with skip lists and commutative hashing." Technical Report, Johns Hopkins Information Security Institute, 2000.

[8] W3C, "http://www.w3.org/TR/xmldsig-core"