情報保護學會論文誌
第13券 第6號, 2003. 12

PKI환경의 OCSP 서버 부하 감소를 위한 OCSP 분산 기법

고 훈*†‡, 장의진**, 신용태*

A Study on Distributed OCSP for minimizing the load of OCSP Server based on PKI

Hoon Ko*†‡, Ui-jin Jang**, Yong-tae Shin***

요 약

공개키 기반구조에서 중요한 요소는 상대방에 대한 인증이다. 서로 간에 인증을 위해서 각각 인증서를 얻는다. 그리고 인증서 취소 목록을 통해서 인증서의 유효성 유무를 체크하게 된다. 그러나 CRL을 이용하는 것은 오프라인 상태에서 사용하는 방법이다. 따라서 최신의 정보를 참조하는 것은 불가능하고 시간이 지남에 따라서 CRL의 크기가 커지게 되어 다운받은 후에 사용하는 CRL 기법은 여러 가지로 불편하게 된다. 따라서 오프라인에서 사용하는 CRL 보다는 온라인에서 사용되는 OCSP를 선호하게 된다. 이에 본 논문에서는 인증서 저장소와 분산된 OCSP 서버에게 동일한 갱신 정보를 소유하게 해서 인증서 검증 요청이 있을 경우 인근의 OCSP 서버를 이용해서 빠른 검증 결과를 제공하는 기법을 제시하고자 한다.

ABSTRACT

The important factor in Public-Key Infrastructure is the authentication to correspondent. We receive the digital certificate for authentication between each other, and then we check the existence of validity on the certificate by Certification Revocation List(CRL). But, To use CRL is the scheme used in offline status. So, it is impossible to refer to the latest information and the CRL scheme which is used after downloading is variously unsuitable to getting bigger of the CRL size as time goes on. Therefore, we prefer OCSP(Online Certificate Status Protocol) used in online to CRL used in offline. Consequently, we propose the scheme which provides the request of fast verification in case of requesting the verification on the certificate by owning the same update information to Certificate Registry and distributed OCSP.

keyword : *OCSP, Public-Key, CRL, Authentication*

Ⅰ. 서 론

공개키 기반구조에서 중요한 요소는 상대방에 대한 인증이다. 서로 간에 인증을 위해서 각각 인증서를 얻고 인증서 취소 목록을 통해서 인증서의 유효성의 유무를 체크하게 된다. CRL(Certificate Revocation List)은 인증기관에 의하여 서명되어 발행되고 인증서 폐지목록은 주기적으로 갱신되기 때문에 해당 주

* 대진대학교 컴퓨터공학과 초빙교수(skoh21@daejin.ac.kr)
** 숭실대학교 컴퓨터학과 부교수(shin@comp.ssu.ac.kr)
*** (주)디지캡스 선임연구원(neon@digicaps.com)
† 주저자, ‡ 교신저자, 논문접수일 : 2003년 7월 30일, 심사완료일 : 2003년 12월 8일

기 내에 발생된 폐지 사실이 이용자에게 바로 반영되지 않고 시간이 지남에 따라서 크게 증가되는 단점이 있다. 인증서 취소 목록은 취소 사유를 포함하는 취소된 인증서들의 목록이다. 그러나 이런 방법은 오프라인에서 사용되는 방법이다. CRL을 이용하는 방법보다 빠른 방법은 OCSP(Online Certificate Status Protocol) 방법이다. OCSP서버는 인증서에 대한 상태 정보를 클라이언트에게 전달한다. OCSP는 인증서 폐지/효력 정지 상태 파악을 실시간으로 정확하게 할 수 있다는 특성이 있기 때문에 고가의 증권정보나 고액의 현금거래 등 중요성이 매우 높은 전자거래에 사용된다. 그러나 많은 클라이언트들이 OCSP서버에 인증에 대한 서비스를 요청할 경우 OCSP 서버의 부담은 증가하게 되고, 많은 갱신정보들이 집중화 될 때도 OCSP 서버는 부담을 갖게 된다. 이에 본 연구에서는 OCSP 서버의 부담을 최소화 하고 고속의 인증을 위한 방안을 연구하였다.

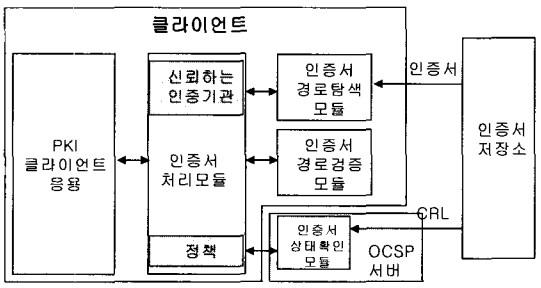
본 논문의 구성은 다음과 같다. Ⅱ장에서는 현재 OCSP 운영에 대한 설명을 하고, Ⅲ장에서는 제안하는 분산된 OCSP에 대한 설명, Ⅳ장에서는 제안된 방법에 대한 분석 및 실험을 하였고 실험결과를 설명하였다. 마지막으로 Ⅴ장에서는 결론을 맺고 마치고자 한다.

Ⅱ. OCSP 서버

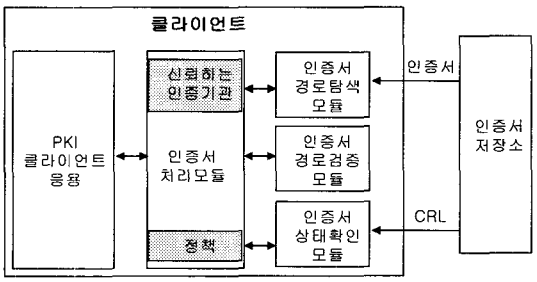
RFC2560에서 제안한 OCSP는 온라인 인증서 상태에 대한 인터넷 표준을 제정하고 있다. 인증서 폐지목록은 주기적으로 갱신되기 때문에 해당 주기 내에 발생된 폐지 사실이 이용자에게 바로 반영되거나 변경 혹은 폐지 사실이 이용자에게 전달되지 않는다. 이러한 문제를 해결하기 위해 IETF PKIXWG에서 1999년 6월 'X509 Public Key infrastructure Online Certificate Status Protocol(OCSP)' 버전 1.0을 발표한 후, 2001년 3월 버전 2.0을 드래프트 형태로 발표하였다. OCSPv2는 클라이언트가 온라인상에서 특정 인증서의 상태를 OCSP 서버에게 문의하거나 그에 대한 인증 경로를 획득 가능하게 하고 획득한 인증 경로의 유효성에 대해 검증할 수 있는 프로토콜로 제안되었다. OCSPv2가 포함하고 있는 서비스로는 온라인 취소 상태 확인 서비스(ORS), 대리 인증 경로 검증 서비스(DPV), 대리 인증 경로 발견 서비스(DPD)등이 있다[그림 1].

[그림 2]는 인증서 상태확인 모듈을 OCSP 서버가 대행하는 것을 나타낸다. OCSP는 위임받은 서버에게 인증서 상태확인을 의뢰한다.

〔그림 1〕 OCSP 서버 구성 요소



〔그림 2〕 OCSP를 이용한 인증서 검증



〔그림 3〕 CRL을 이용한 인증서 검증

[그림 2]에서 보는 것과 같이 클라이언트는 실시간에 가까운 인증서 폐지 상태 정보를 OCSP 서버를 통해서 실시간으로 얻을 수 있다.

[그림 3]은 클라이언트에서 CRL을 이용한 인증서 검증 모듈을 나타낸 것이다. 현행 CRL 인증서 검증의 문제점은 복잡한 인증서 검증 절차 수행, 온라인 실시간 처리의 문제, 클라이언트 모듈의 크기가 증가하는 문제점을 안고 있다. 이에 비해 OCSP는 인증서 폐지/효력정지 상태 파악을 실시간으로 정확하게 할 수 있다는 특성이 있기 때문에 실시간의 정보를 원하는 증권정보나 고액의 현금거래 등 금융권과 전자거래에 사용된다. 일반적으로 인증서 검증은 인증 경로상의 모든 인증서에 대하여 수행되며, 인증서 내에 존재하는 서명 검증, 폐지상태, 유효기간 검증, 정책

처리 / 검증 등으로 구성된다. 이를 3부분으로 나누면 인증서 경로 구축, 인증서 경로 검증, 인증서 상태 확인 등으로 구분된다.

2.1 ORS 프로토콜

ORS(Online Revocation Status) 서비스는 클라이언트가 서버에게 특정 인증서의 정보를 제공하고 서버는 클라이언트에게 특정 인증서의 취소 상태를 검사하여 알려준다. ORS 서비스를 사용하기 위해서는 클라이언트와 서버 사이의 메시지 형식이 id-pkix-ocsp-basic response type의 형태이어야 하며 양측 모두에게 이러한 형태의 필드 처리 능력이 요구된다.

[요구사항]

(1) 요구 메시지의 형식은 OCSP 요구 메시지를 따르고, 서명 후에 전송하여야 한다.

(2) 클라이언트 요구 메시지는 ORS 서비스를 사용하기 위해. OCSPRequest의 tbsRequest 필드에 옵션으로 처리되는 requestExtensions 필드 부분에 id-pkix-ocsp-ors-req의 OID 값이 포함되어 있어야 한다.

2.2 DPD 프로토콜

DPD(Delegated Path Discovery) 서비스는 서버가 인증서 결로 구축정책에 따라서 클라이언트를 대신하여 신뢰정점(TrustAnchor)까지의 인증서 경로를 구축하는 서비스이다[그림 4].

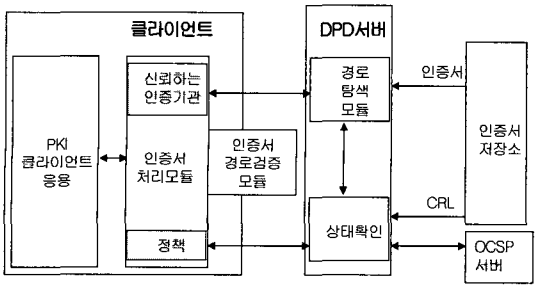
DPD 서비스에서 교환 요청 및 응답 메시지의 처리에 대한 요구사항들은 아래와 같다.

[요구사항]

(1) 인증서 경로구축 및 인증서 경로상의 인증서 각각에 대한 인증서 상태정보 요청

(2) 에러메시지를 포함하여 응답 메시지를 생성하여야 한다.

(3) 요청메시지에 포함된 사용자 인증서에 대해서 하나 이상의 인증서 경로를 획득할 수 있어야 한다.



〔그림 4〕 DPD 서버를 이용한 인증서검증

2.3 DPV 프로토콜

DPV(Delegated Path Validation) 서비스는 서버가 클라이언트로부터 인증서 검증을 위임받은 후, 인증서 검증 정책에 따라서 최신의 인증서 상태정보를 이용하여 인증서 검증을 수행한다[그림 5].

[요구사항]

(1) DPV 서버와 DPV 클라이언트는 서로에 대한 각종 에러에 대한 응답 메시지 그리고 각종 응답 메시지를 생성하여야 한다.

(2) DPV 클라이언트는 CA이름, 인증서 일련번호 ES의 ESSCertID, OtherSigningCertificate의 인증서 해쉬값을 사용할 수 있다.

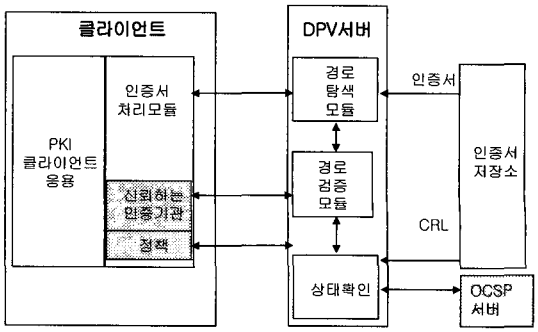
(3) DPV 서비스는 Replay Attack을 방지할 수 있어야 한다.

(4) 인증을 위해서 응답메시지에 서버의 전자서명이 포함되어야 한다.

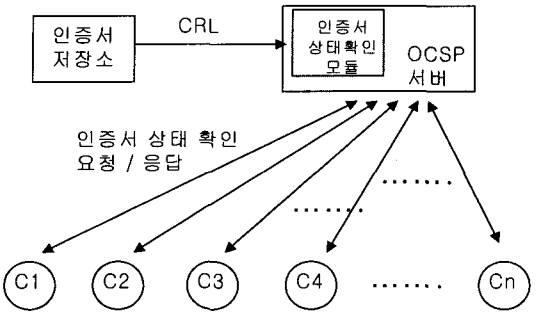
(5) DPV 서버는 하나 이상의 다른 DPV 서버의 서비스를 이용할 수 있도록 설정될 수 있어야 한다.

Ⅲ. 분산된 OCSP 서버

OCSP 방법은 기존의 CRL 방법에 비해서는 실시간 처리라는 관점에서 보았을 경우에 굉장히 효과적인 방법이다. 그러나 그림 6에서 보듯이 모든 클라이언트들이 OCSP 서버 한곳에 서비스를 요청한다면 OCSP 서버는 부담을 가지게 된다.



〔그림 5〕 DPV 서버를 이용한 인증서 검증



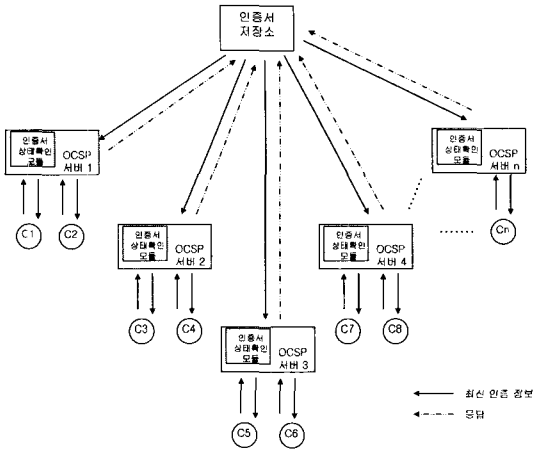
〔그림 6〕 OCSP 서버 집중화 현상

게다가 최근에는 전자상거래를 이용하는 사용자가 급격히 증가하고 있는 상태에서 이러한 서버의 부담은 가중될 것으로 추측된다.

이를 해결하기 위해서 분산된 OCSP 서버 모델을 제안하고자 한다. 제안하는 분산된 OCSP 서버는 인증기관에서 관리를 하게 되고, OCSP에 전달되는 CRL정보는 동일한 저장소를 이용하게 된다. 즉 많은 OCSP 서버가 각 지점에 분산되어 있을지라도 분산되어 있는 OCSP 서버에 CRL정보를 전달해 주는 인증서 저장소는 같음을 의미한다. OCSP 서버를 분산한 이유는 앞에서 설명했듯이 전자상거래를 사용하는 사용자가 계속적으로 증가하고 있다. 또한 전자상거래의 위험성을 제거하기 위해서 공개키 기반의 인증서라는 것을 발행해서 이를 이용해서 상대방을 인증하고 있으며 이 또한 이를 이용하는 사용자는 계속적으로 증가하고 있다. 그러나 이를 해결해 주기 위한 OCSP 서버는 현재까지는 고정되어 있다. 결국 계속적으로 증가하는 사용자가 고정된 위치의 고정된 개수의 OCSP를 이용해서 서비스를 받고자 한다고 가정할 때, OCSP의 서버에 부담을 줄 수 있다. [그림 7]은 분산된 OCSP 서버에 대해서 설명하고 있다.

인증서 저장소는 각 분산된 OCSP 서버들에게 최신의 갱신 정보들을 주기적으로 전송한다. 이때 인증저장소는 OCSP 서버에 정보를 전달할 때, 정보의 비밀성을 유지하기 위해서 인증서 저장소와 각 OCSP 서버간에 세션키를 이용해서 암호화 한 후에 각 OCSP 서버에 전달하게 된다. 또한 OCSP 서버의 특성에 맞게 인증서 저장소는 만약 갱신 신호가 있으면 갱신 정보를 바로 각 OCSP 서버에게 갱신된 최신의 정보를 전송하게 되어, 분산된 OCSP 서버 정보의 최신성을 보장하게 된다.

이렇게 수신된 갱신 정보에 대해서 각 OCSP 서버들은 인증서 저장소에게 수신확인 응답 메시지를 전달한다. 인증서 저장소는 모든 OCSP 서버로부터 일



〔그림 7〕 분산된 OCSP 서버 구성도

정 시간동안 응답메시지를 기다린 후, 모든 OCSP 서버로부터 응답메시지를 수신한 후에, 다시 각 OCSP 서버에게 확인 메시지를 전송하게 된다. 확인 메시지에는 모든 OCSP 서버가 동일한 정보를 가지고 있기 때문에 각각 분산된 OCSP 서버의 정보를 이용해도 된다라는 의미를 담고 있다. 그러나 인증서 저장소에서 일정기간이 지나도 특정 OCSP 서버로부터 응답이 없으면 만기시간 전까지 계속 재전송을 하게 된다. 그래도 응답이 없으면 다른 OCSP 서버에게 실패 메시지를 전송하게 되어 전에 보낸 갱신 정보의 폐기를 요청한다. 그러나 계속적으로 OCSP 서버로부터 응답이 없다면 전체 시스템에 갱신 정보를 수신하지 못하는 경우가 발생된다. 따라서 일정 시간이 지나도 OCSP 서버로부터 신호가 없으면 그 서버는 서비스 불가로 판정되어 OCSP 서버가 원상태로 돌아오기 전까지는 무시하게 된다.

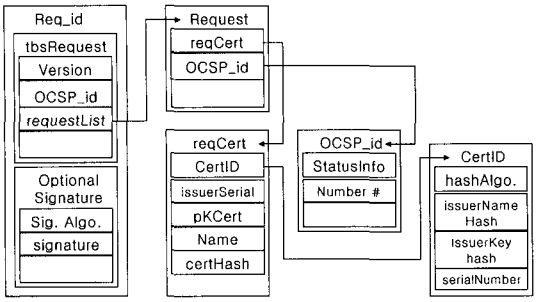
[고려사항]

분산된 OCSP 서버를 구축하기 위해서 몇 가지 고려사항이 있다.

- 모든 OCSP 서버의 정보는 같아야 한다.
- 모든 OCSP 서버에 최신정보를 전달할 때 속도적인 측면을 고려해야 한다. 즉 빨라야 한다.
- 인증서 저장소에서 각 OCSP 서버로 최신 정보를 전달할 때 정보의 보안을 유지해야 한다.
- 인증서 저장소와 각 OCSP 서버 사이에 전달되는 신호는 단순해야 한다.

[Notation]

- *OCSP_id* : 각 OCSP 서버의 id



〔그림 8〕 *Req_id* 메시지의 구성

OCSP_1, ..., OCSP_n

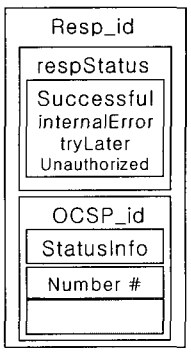
- *U_CRL* : 갱신 인증서 취소 목록
- *SKey* : 세션키
- *Resp_id* : 각 OCSP의 응답 메시지
- *Req_id* : 각 OCSP의 요청 메시지
- *Confirm_id* : 성공 메시지
- *Fail_id* : 실패 메시지
- *E(U_CRL)SKey* : 세션키로 *U_CRL*을 암호화
- *D(U_CRL)SKey* : 세션키로 *U_CRL*를 복호화
- $E(U\_CRL)_{id}$: 공개키로 *U_CRL*을 암호화
- $D(U\_CRL)_{id}$: 개인키로 *U_CRL*을 복호화
- *OCSP_idpub* : 각 OCSP_id의 공개키
- *OCSP_idPri* : 각 OCSP_id의 개인키:
- $S(U\_CRL)_{id}$: 분산된 *OCSP_id*의 키를 이용한 갱신된 CRL에 대한 서명
- $V(U\_CRL)_{id}$: 분산된 *OCSP_id*의 키를 이용한 갱신된 CRL에 대한 검증

3.1 요구 메시지 (*Req_id*)

요구 메시지 (*Req_id*)는 OCSP_id가 인증서 저장소에게 갱신된 인증서 취소 정보를 요청하는 메시지이다. 요구 메시지의 구성은 크게 버전과 OCSP_id, 서명 알고리즘 종류, 서명문, 공개키를 검증하기 위한 공개키의 인증 경로 등을 포함하는 optionalSignature 필드로 구성되어 있다[그림 8].

메시지의 구성 내용은 다음과 같다.

- **version** : OCSP의 버전을 나타낸다.
- **requestorName** : 옵션이며, *Req_id* 요구자의 *OCSP_id*를 나타낸다.
- **requestList** : 상태 검증을 요구하는 특정 인증서의 정보를 나타낸다.
- **ReqCert** : 발급자의 고유번호를 나타낸다.



〔그림 9〕 응답메시지 구성도

- **singleRequestExtensions** : 옵션이며, 요구 메시지의 확장 영역을 나타낸다.
- **requestExtensions** : *OCSP_id*와 인증서 저장소 간에 미리 합의된 확장 영역을 나타내 주는 필드이며 옵션으로 처리된다.
- **StatusInfo** : *OCSP_id*의 시간정보를 담고 있다.
- **number #** : request의 횟수를 나타낸다.

3.2 응답 메시지 (*Resp_id*)

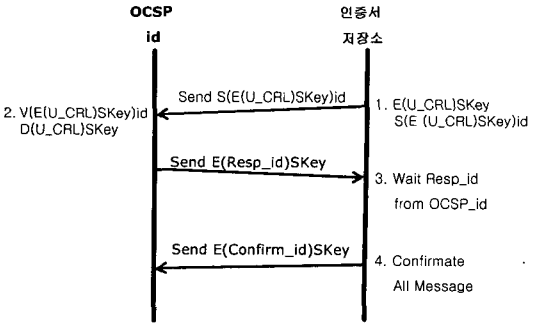
응답 (*Resp_id*)메시지는 *OCSP_id*가 인증서 저장소의 갱신 인증서 취소 목록 정보를 수신했을 경우에 *OCSP_id*가 인증서 저장소에 보내는 응답 메시지이다. 응답 메시지도 요구 메시지와 마찬가지로 서명되어 전송된다. 응답 메시지의 구성은 크게 응답의 상태를 나타내는 responseStatus필드로 구성되어 있다[그림 9].

메시지의 구성 내용은 다음과 같다.

- **sucessful** : 수신 메시지의 성공 여부를 나타낸다.
- **internalError** : 요구 메시지가 서버 내부의 상태에 일치하지 않는 내부 오류 상태임을 나타낸다.
- **tryLater** : 서버가 작동하고 있지만 인증서 상태를 알려줄 수 없는 경우에 사용
- **unauthorized** : 권한지 부여되지 않는 경우
- **StatusInfo** : *OCSP_id*의 시간정보를 담고 있다.
- **number #** : response의 횟수를 나타낸다.

3.3 시스템 구축도

아래의 그림들은 분산된 *OCSP_id*에 대해서 구축되는 몇 가지에 대해서 가정을 하고 거기에 대한 흐름을 분석하고자 한다.



〔그림 10〕 갱신정보 송수신 과정

[단계설명] : 갱신정보 송수신 과정

[1단계] $E(U\_CRL)_{SKey}$
$S(E(U\_CRL)_{SKey})_{id}$
send $S(E(U\_CRL)_{SKey})_{id}$ *to OCSP_id*
//먼저 세션키로 암호화 한후 U_CRL을 서명한다.
//그리고 해당되는 OCSP 서버에 전송한다.

[2단계] $V(E(U\_CRL)_{SKey})_{id}$
$D(U\_CRL)_{SKey}$
send $E(Resp\_id)_{SKey}$ *to 인증서저장소*
// $OCSP_id$에서 검증, 복호화 한다.
//그리고 인증 저장소에 수신확인 메시지를
//전송한다.

[3단계] *wait* $E(Resp\_id)_{SKey}$ *from* $OCSP\_id$
//인증서 저장소는 모든 OCSP 서버로부터 수신
//응답 메시지가 올 때까지 기다린다.

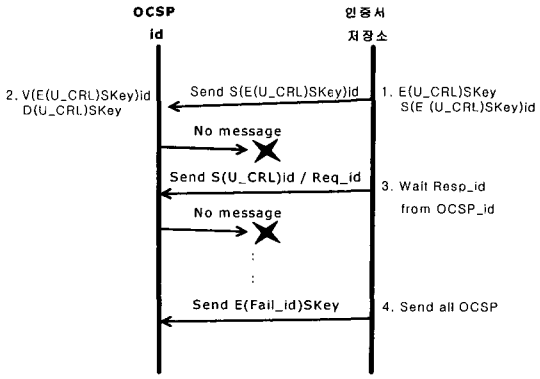
[4단계] *send* $E(Confirm\_id)_{SKey}$ *to* $OCSP\_id$
//인증서 저장소는 모든 OCSP 서버에게 성공
//메시지를 전송한다.

분산된 OCSP 모델의 가장 중요한 부분은 실시간으로 인증서 취소 목록을 확인 가능하다는 것이다. 그리고 각 OCSP 서버에서 참조하는 정보는 모든 OCSP 서버가 동일해야 한다는 것이다. 즉 인증서 저장소와 OCSP 서버간의 동기가 중요하다는 의미이다.

[그림11]은 인증서 저장소가 OCSP 서버들에게 갱신 메시지를 전송했으나, OCSP 서버로부터 응답이 없는 경우이다.

[단계설명] : OCSP 서버의 응답이 없는 경우

[1단계] $E(U\_CRL)_{SKey}$



〔그림 11〕 OCSP 서버의 응답이 없는 경우

$S(E(U\_CRL)_{SKey})_{id}$
send $S(E(U\_CRL)_{SKey})_{id}$*to OCSP_id*
//먼저 세션키로 암호화 한후 U_CRL을 서명한다.
//그리고 .해당되는 OCSP 서버에 전송한다.

[2단계] $V(E(U\_CRL)_{SKey})_{id}$
$D(U\_CRL)_{SKey}$
send $E(Resp\_id)_{SKey}$ *to 인증서저장소*
// $OCSP_id$에서 검증, 복호화 한다.
//그리고 인증 저장소에 수신확인 메시지를
//전송한다.

[3단계] *wait* $E(Resp\_id)_{SKey}$ *from* $OCSP\_id$
//인증서 저장소는 모든 OCSP 서버로부터 수신 //응답 메시지가 올 때까지 기다린다.

[4단계] *No message*
//인증서 저장소는 OCSP 서버로부터
//응답 메시지를 수신하지 못했다.

[5단계] *send* $S(E(U\_CRL)_{SKey})_{id}$ / Req_id *to* $OCSP\_id$
//인증서 저장소는 $OCSP\_id$ 에게 다시
// $S(E(U\_CRL)_{SKey})_{id}$ / Req_id 메시지를 전송
//한다.

[6단계] *No message*
//인증서 저장소는 OCSP 서버로부터
//응답 메시지를 수신하지 못했다.

[7단계] *Send* $E(Fail\_id)_{SKey}$ *to All* $OCSP\_id$
//모든 $OCSP_id$에게 $E(Fail\_id)_{SKey}$ 메시지를

//전송한다.

그러나 계속적으로 OCSP 서버로부터 응답이 없는 경우, 해당 서버의 서비스 불가로 판단되어 인증서 저장소는 이후에는 해당 OCSP서버를 무시하게 된다. 해당 OCSP 서버가 다시 서비스가 가능하게 되면 인증서 저장소에 신호를 보내어 다시 갱신 정보를 요청하게 된다.

[그림12]는 인증서 저장소로부터 응답이 없는 경우를 보여주고 있다. 인증서 저장소로부터 응답이 없는 경우는 특별한 경우이다. 인증서 저장소가 다운되는 경우는 극히 드문 경우이다. 왜냐하면 인증서 저장소는 CA기관에서 관리하기 때문이다. 따라서 이런 경우는 중간 네트워크에서 특정 문제가 발생했을 가능성이 크다.

[단계설명] : 인증서 저장소의 응답이 없는 경우

[1단계] $E(U\_CRL)_{SKey}$

$S(E(U\_CRL)_{SKey})_{id}$

send $S(E(U\_CRL)_{SKey})_{id}$ *to OCSP_id*

//먼저 세션키로 암호화 한후 *U_CRL*을 서명한다.
//그리고 .해당되는 OCSP 서버에 전송한다.

[2단계] $V(E(U\_CRL)_{SKey})_{id}$

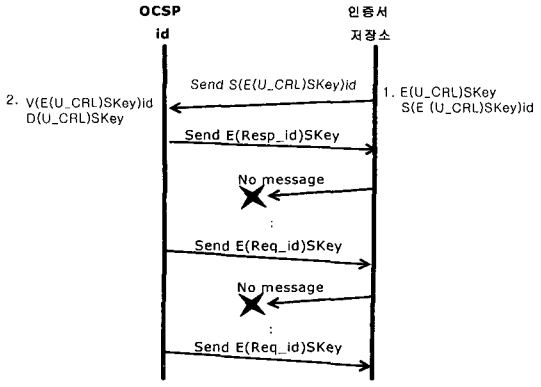
$D(U\_CRL)_{SKey}$

send $E(Resp\_id)_{SKey}$ *to 인증서저장소*

// *OCSP_id*에서 검증, 복호화 한다.
//그리고 인증 저장소에 수신확인 메시지를
//전송한다.

[3단계] *wait* $E(Resp\_id)_{SKey}$ *from OCSP_id*

//인증서 저장소는 모든 OCSP 서버로부터 수신



〔그림 12〕 인증서 저장소의 응답이 없는 경우

//응답 메시지가 올 때까지 기다린다.

[4단계] *No message*

send $E(Req\_id)_{SKey}$ *to 인증서 저장소*

// *OCSP_id*는 인증서 저장소로부터 서버로
//부터 *Confirm_id* 수신하지 못했다.
//그래서 인증서 저장소에 *Req_id* 메시지를
//전송한다.

[5단계] *wait* $E(Confirm\_id)_{SKey}$ *from 인증서 저장소*

// *Confirm_id* 메시지가 올 때까지 4단계를
//반복하면서 계속 기다린다.

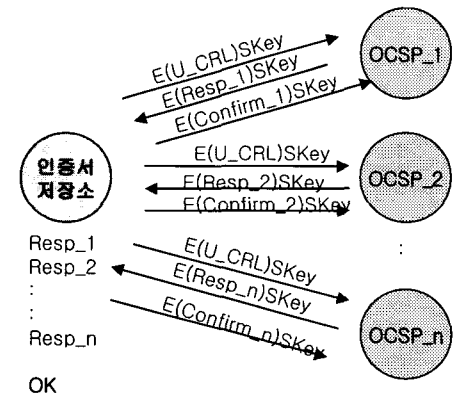
*OCSP_id*의 임무는 지역에 있는 사용자로부터 인증서에 대한 유효성을 체크해 준다. 인증서 저장소로부터 응답이 없다고 해서 중간에 기능을 포기해서는 안된다. 따라서 인증서 저장소로부터 응답이 없는 경우 *OCSP_id*는 계속적으로 인증서 저장소로부터 *Confirm_id* 메시지가 올 때까지 계속 대기한다. 그러나 계속적으로 *Confirm_id* 메시지가 오지 않을 경우 *OCSP_id*는 인근의 *OCSP_id*에게 *Confirm_id* 메시지를 요청하든지, 아니면 자신이 서비스 해 주고 있는 사용자들을 인근의 다른 *OCSP_id*에게 주는 방안도 있지만 본 논문에서는 포함하지 않았다.

Ⅳ. 모델 분석 및 실험

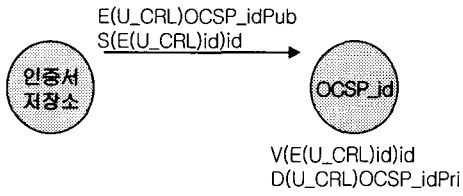
본 장에서는 분산된 *OCSP* 서버 모델에 대한 분석을 하고 실험을 통해 결과를 도출해 낸다.

4.1 갱신정보의 동시성 문제

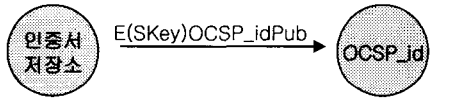
본 모델에서 가장 중요한 부분 중의 하나는 인증서 저장소에서 전송되는 *U_CRL*이 모든 *OCSP_id*가 동일한 정보를 가지고 있어야 한다는 점이다. 즉 모든 *OCSP_id*가 소유하고 있는 정보는 모두 동일해야 한다. 이를 해결하기 위해서 인증서 저장소는 *U_CRL*을 각 *OCSP_id*에 전송했을 때, *Resp_id*와 *Confirm_id*를 모든 *OCSP_id*와 동시에 처리했을 경우에 *U_CRL*을 인정하게 된다[그림 13]. 만약 특정한 *OCSP_id*로 부터 *Resp_id*가 수신되지 않는다면 *Confirm_id*를 보내지 않게 되고, 각 *OCSP_id*는 *Confirm_id*를 수신하지 않기 때문에 이전에 수신



〔그림 13〕 갱신정보의 동시성



〔그림 14〕 세션키 전송



〔그림 15〕 안전성 해결

했던 *U_CRL*을 무시하게 된다.

4.2 *U_CRL*의 안전성 문제

안전한 갱신 정보 전송을 위해서 서로 간에 미리 주고받은 공개키를 이용하게 된다. 서로간의 공개키를 이용해서 세션키를 암호화 후에 전송하게 된다. 암호화된 세션키를 수신한 후 각자 소유하고 있는 개인키를 이용해서 복호화 하게 된다[그림 14].

그리고 인증서 저장소는 전송할 *U_CRL*를 먼저 세션키를 이용해서 암호화 한 후 전자서명을 한다. 암호화/서명된 *U_CRL*를 *OCSP_id* 보내게 되고, *OCSP_id*는 서명된 문서를 검증 한 후에 세션키를 이용해서 복호화 하게 된다[그림 15]. 이러한 과정을 거치면 *U_CRL*의 안전성을 보장하게 된다.

4.3 *U_CRL*의 크기 문제

본 문제는 분산된 *OCSP_id* 모델에서 발생되는 문제가 아니고, 기존의 인증서 저장소를 이용하는 경우에 발생되는 문제이다. 본 모델은 인증서 저장소에서 각 *OCSP* 서버로의 갱신정보 전달보다는 각 서버의 성능을 위주로 분석하였다. 따라서 본 논문에서는 직접적으로 고려하지 않았고 실험할 때 약간의 전송 시간을 추가하여 실험하였다.

〔표 1〕 환경변수

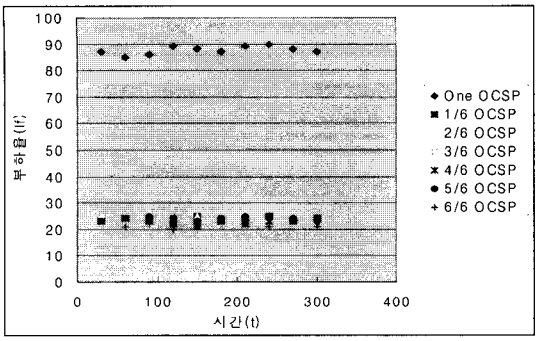
OCSP 서버 개수	6
Host(User) 개수	600
OCSP당 평균 Host 개수	100
암호화 알고리즘	DES
키 길이	56 bits
해쉬 알고리즘	SHA1
서명 알고리즘	RSA
서명 / 인증 키 길이	512 bits
링크 지연	10ms
대역폭	1.5mb, 10mb
Empty CRL 크기(구조)	55kb
인증서 크기	2kb
인증서 갱신 요청 시간	20-30sec
평균 서비스 요청 시간	10 sec
실험시간	300 sec

4.4 실험환경 및 실험결과

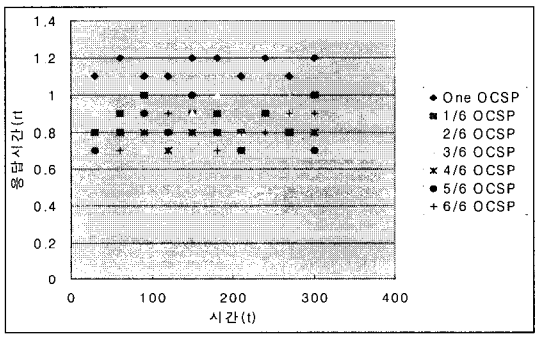
본 논문은 분산된 *OCSP* 서버들에게 인증서 저장소의 취소정보를 안전하게 전달하는 모델을 설명하였다. 이에 대한 실험을 하기 위한 실험환경은 아래 [표 1]과 같다.

실험 프로그램은 NSII을 이용하였다. [표 1]에 대한 설명을 하면 네트워크의 기본 환경은 기존의 실험환경을 그대로 적용하였다. 그리고 한국전자인증에서는 1000개당 하나의 CRL파일을 할당하고 있다. 그리고 빈 CRL은 55kb의 크기를 가지고 있으며, 인증서 한 개당 3kb의 크기를 할당한다고 한다. 그렇지만 본인의 인증서와 여러 사람의 인증서 크기를 분석한 결과 1kb를 약간 넘는 크기였다. 따라서 인증서의 평균 크기인 약 2kb의 크기로 정하였다. 이를 기반으로 *OCSP*가 한대 일 경우와 6대일 경우의 성능을 비교하였다.

[그림 16][그림 17]은 *OCSP* 서버가 한대일 경우와 여섯 대일 경우에 대해서 실험하였다. 먼저 [그림 16]은 각 환경에서 서비스를 요청했을 때 성능을 보



〔그림 16〕 성능비교



〔그림 17〕 서비스 요청에 대한 응답시간

여주고 있다. 한대일 경우는 600여대의 사용자가 요청하기 때문에 많은 부하가 있음을 보여 주고 있다. 평균 부하는 한대일 경우는 87.6, 여섯 대일 경우 평균은 23.25로 나타났다. [그림 17]은 사용자의 인증서 유효성 서비스 요청 시간에 대한 응답시간을 보여주고 있다. 이 또한 서버를 분산했을 경우에 한대일 경우보다 빠른 응답시간을 보여주고 있지만 [그림 16]에 비해서는 확실히 우월하다고 할 수는 없다. 왜냐하면 *OCSP*서버 6대를 구축했을 경우에는 인증서 저장소에서 *OCSP*서버로의 갱신정보 전달 시간을 추가했기 때문에 이에 대한 시간도 포함되었기 때문이다. 서비스 요청 평균 응답시간은 한대일 경우 1.15초, 여섯 대일 경우 0.825초로 여섯 대일 경우가 평균적으로 빠른 결과가 나왔다.

4.5 분산모델의 특징 요약

본 논문에서는 하나의 *OCSP* 서버가 많은 사용자의 서비스 요청이 들어왔을 경우 이를 해결해 주기 위해서 서버에 많은 무리가 따르기 때문에 서버의 무리를 최소화하기 위한 분산된 본 모델을 제시하였다.

〔표 2〕 기존 모델과 비교

항목	CRL	OCSP	제안모델
On-line	×	○	○
추가 장비	×	○	○
실시간 처리	×	○	○
서버부담	-	있다	없다
최신성	×	○	○
비밀성	×	×	○
통신량	많다	많다	적다
갱신정보 전달	×	×	○
암호화/인증처리	×	×	○

V. 결 론

본 논문은 기존의 하나의 인증서 저장소에서 인증서 취소 체크 서비스를 할 때, 순간 많은 사용자들이 서비스를 요청할 때의 문제점에 대해서 분석을 하고 이에 대한 해결 방안을 제시하였다. 본 논문에서는 기존 하나의 *OCSP* 서버가 담당하던 인증서 유효성 검사를 여러 *OCSP* 서버로 분산시키는 방법을 제시하였고 이를 실험하여 *OCSP* 서버가 여섯 대일 경우가 한대일 경우보다 부하감소와 서비스 요청 평균 응답시간이 월등함을 증명하였다. 또한 본 논문에서 중요한 요소 몇 가지가 있다. 하나는 인증서 저장소와 *OCSP_id* 간의 *U_CRL*전송 시 안전성의 확립문제인데 이는 인증서 저장소와 *OCSP_id* 사이의 *U_CRL* 전달 전에 세션키를 상대방의 공개키로 암호화해서 전송하고 이 세션키를 이용하여 전송되는 정보를 암호화하여 전송함으로써 안전성 및 기밀성을 해결하였다. 그리고 또 하나는 인증서 저장소와 *OCSP_id* 간의 최신정보의 동기를 부여하는 것이다. 이에 대한 설명은 본 논문의 구조에서 인증서 저장소는 인증서 취소에 대한 체크를 하는 역할이 아니고 단지 각 *CRL*의 갱신 유무를 체크하고 *OCSP_id*에게 *U_CRL* 정보만을 전달하고 신뢰성 있는 전송을 하였는지에 대한 책임만이 있을 뿐이다. 따라서 모든 *OCSP_id*의 *Confirm_id* 수신이 같은 시각이 아닐 것이기에 이에 대한 시간 차이는 아직 해결하지 못한 부분으로 남아 있다. 예를 들면 인증서 저장소에서 *Confirm_id*를 보냈을 지라도 *OCSP_id*을 동시에 수신하지 않을 것이다. 따라서 현재까지는 이런 동시성을 지니지 않기 때문에 최신 *U_CRL*을 사용하지 못하는 경우가 발생할 수도 있다. 이에 대한 문제를 해결하기 위해서 계속 연구 중에 있다. 마지막으로

빈번한 인증서 저장소의 *CRL* 갱신문제이다. 많은 사용자들이 동시에 사용하기 때문에 빈번한 갱신이 요구될 꺼라 생각된다. 너무 빈번한 갱신 요구 또한 서버의 부담이 가중된다. 이에 대한 해결방한도 향후 논의되어져야 할 것이다.

참 고 문 헌

[1] W.Diffie and M.Hellman, "New Directions In Cryptography", IEEE Trans on Information Theroy. Vol. IT-22, pp.644~654. Nov, 1976.

[2] R.Housley, W.Ford, W.Polk, D. Solo. RFC2459 "Intranet X.509 Public Key Infrastructure Certificate and CRL Profile", Jan.1999.

[3] M.Myers, R.Ankney, A.Malpani, S.Galperin, C.Adams, RFC2560 "Internet X.509 Public Key Infrastructure Online Certificate Status Protocol-OCSP", IETF Standard, June, 1999.

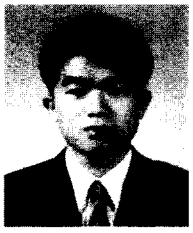
[4] M.Myers, R.Ankney, C.Adams. "On-line Certificate Status Protocol, cersion2", IETF Draft, draft-ietf-pkix-ocspv2-01.txt. Nov, 2000.

[5] M.Myers. S.Farrell, C.Adams. "Delegated Path Discovery with OCSP". IETF Draft, draft-ietf-pkix-ocsp-path00.txt. Sep, 1999.

[6] M.Myers, C.Adams, S.Farrell. "Delegated Path Validation", IETF Draft. draft-ietf-pkix-ocsp-valid-00.txt. Aug, 2000.

[7] RD.Pinkas. "Delegated Path Validation and Delegated Path Discovery Protocols", IETF Draft draft-ietf-pkix-dpvdld-00.txt, Jul. 2001.

〈著者紹介〉



고 훈 (Hoon Ko) 정회원

1998년 : 호원대학교 컴퓨터학과 졸업 학사
2000년 : 숭실대학교 컴퓨터학과 통신연구실 석사
2002년 : 숭실대학교 컴퓨터학과 통신연구실 박사수료
2000년 5월~2002년 7월 : (주)지오나스 선임연구원
2002년 9월~현재 : 대진대학교 컴퓨터공학과 초빙교수
<관심분야> 암호화프로토콜, 정보보안, 인터넷보안, 전자서명, 네트워크 보안

장 의 진 (Uijin Jang) 정회원

1999년 9월 : 숭실대학교 컴퓨터학과 졸업 학사
2002년 9월 : 숭실대학교 컴퓨터학과 통신연구실 석사
2002년 12월~현재 : 디지캡 기술연구소 선임연구원
<관심분야> DRM, 네트워크 보안, 암호화 프로토콜, 정보보안, 인터넷보안, 전자서명

신 용 태 (Yongtae Shin) 정회원

1985년 2월 : 한양대학교 산업공학 학사
1990년 12월 : Univ. of Iowa 전산학 석사
1994년 5월 : Univ. of Iowa 전산학 박사
1994년 5월~1994년 8월 : Univ. of Iowa computer Science Dept. 객원교수
1994년 8월~1995년 1월 : Michigan State Univ Computer Science Dept. 객원교수
1995년 3월~현재 : 숭실대학교 컴퓨터학과 부교수
2000년 4월~현재 : (주) 디지캡 대표이사
<관심분야> 암호화프로토콜, 정보보안, 인터넷보안, DRM