

제 2 장 공개키 기반구조

2.1 인증서

공개키 암호알고리즘을 암호프로토콜에서 활용할 경우 반드시 보장되어야 하는 것은 공개키의 인증이다. 즉, 주어진 공개키가 누구의 키인지 그것을 사용하는 프로토콜의 참여자가 확신할 수 있어야 한다. 만약 이것이 보장되지 않으면 다양한 공격이 가능하다. 공개키 방식에서 Alice가 Bob에게 메시지 M 을 비밀스럽게 전달하고 싶다. 이 경우에는 Bob의 공개키 $+K_B$ 로 M 을 암호화하여 전달하면 오직 Bob만이 자신의 개인키를 이용하여 복호화할 수 있다. 이 때 Carol이 Alice를 속여 Alice가 자신의 공개키 $+K_C$ 를 Bob의 공개키로 믿도록 하면 Alice는 $+K_B$ 가 아닌 $+K_C$ 로 메시지를 암호화하여 전달하게 된다. 그러면 Bob은 메시지를 얻을 수 없고, Carol만 메시지를 얻을 수 있다. 뿐만 아니라 만약 Carol이 자신의 개인키 $-K_C$ 를 이용하여 메시지에 서명하여 Alice에게 주면 Alice는 이것을 Bob이 서명한 것이라고 착각하게 된다.

공개키가 누구의 공개키인지 참여자들이 확신을 가질 수 있도록 하기 위해 사용하는 것이 인증서(certificate)이다. 인증서는 공개키와 그것의 소유자를 바인딩시켜주는 전자문서이며, Kohnfelder가 1978년에 처음으로 제안하였다. 인증서는 보통 신뢰할 수 있는 인증기관(CA, Certification Authority)이 전자서명하여 생성한다. 즉, 인증기관이 공개키를 공증해준다고 생각하면 된다. 오늘날 사용되는 대부분의 인증서는 현재 X.509 버전 3 표준을 따르고 있다. 이 표준 외에도 SPKI(Simple Public Key Infracstructure) 인증서, PGP(Pretty Good Privacy) 인증서가 있다.

2.2 인증서의 구성

인증서에는 다음과 같은 정보가 기본적으로 포함된다.

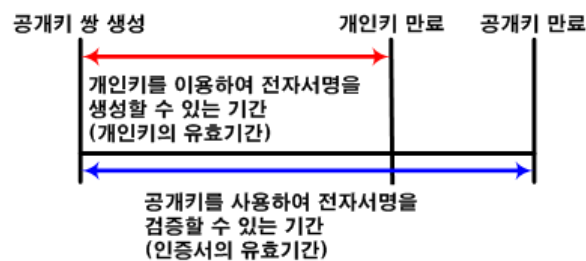
- 일련번호: 인증서를 발급한 인증기관 내의 유일번호이다.
- 버전: X.509 버전을 나타내며, 현재는 버전 3.0을 사용하고 있다.
- 서명 알고리즘: 이 인증서를 발급할 때 사용한 서명 알고리즘의 식별자
 - 예) SHA1 with RSA Encryption
- 발급자: 이 인증서를 발급한 인증기관의 DN(Distinguished Name)
- 유효기간: 이 인증서가 유효기간을 나타내며, 시작 날짜(not valid before)와 만료 날짜(not valid after)를 기록한다.
- 주체(subject): 이 인증서의 소유자 DN
- 주체의 공개키 정보: 공개키 값, 알고리즘 식별자 등으로 구성된다.

여기서 DN이란 X.500 표준에 따라 명명된 이름을 말한다. X.500은 C(Country), O(Organization), OU(Organizational Unit), CN(Common Name)과 같은 필드를 사용하여 명명한다. X.500을 이용한 명명의 예는 다음과 같다.

예) C=KR, O=한국기술교육대학교, OU=인터넷미디어공학부, CN=김상진

이 외에 다음과 같은 정보를 확장 필드에 포함한다.

- 키사용 용도: 인증서에 포함된 공개키의 사용 용도를 나타낸다.
 - 예) 전자서명, 부인방지, 키 암호화, 데이터 암호화, 키 동의, 인증서 서명 등
- 인증기관 키 식별자: 인증서를 확인할 때 사용할 공개키의 유일 식별자
- 주체 공개키 식별자: 이 인증서에 포함된 공개키의 유일 식별자
- CRL(Certificate Revocation List) 분배점(distribution point): 이 인증서의 폐지 여부를 확인하기 위한 인증서 폐지 목록(CRL)이 있는 위치
- 개인키 유효기간: 이 인증서에 바인딩되어 있는 공개키에 대응되는 개인키의 유효기간



<그림 2.1> 개인키의 유효기간

그림 2.1처럼 전자서명의 경우 서명할 수 있는 유효기간과 서명을 확인할 수 있는 유효기간이 보통 다르다. 이것은 개인키가 만료되어 더 이상 이 개인키로 서명할 수는 없지만 그 개인키로 기존에 서명된 전자서명은 검증할 수 있어야 하기 때문이다. 데이터나 키 암호화가 목적인 경우에는 개인키와 공개키의 유효기간이 보통 같다.

인증서에 포함된 모든 정보의 해쉬값을 계산하여 그 값에 전자서명하여 인증서가 발급된다.

2.3 인증서의 검증

인증서를 검증하기 위한 절차는 크게 다음과 같은 네 단계로 이루어진다.

- 단계 1. 인증서의 서명 확인
- 단계 2. 인증서의 유효기간 확인
- 단계 3. 인증서의 사용 용도 확인
- 단계 4. 인증서의 폐지 여부

인증서의 서명 확인은 인증서 확장 필드에 있는 인증기관 키 식별자 의해 식별되는 인증기관의 공개키를 이용하여 확인해야 한다. 이 때 인증기관의 공개키를 사용하기 위해서는 이 공개키가 포함된 인증서를 추가로 확인해야 한다. 인증기관의 인증서는 보통 상위 인증기관이 발급하며, 이 경로를 따라 올라가면 최상위 인증기관의 인증서까지 확인해야 한다.

이 때 최상위 인증기관은 자체 서명 인증서를 사용한다. 자체 서명 인증서란 인증서에 포함된 공개키의 대응되는 개인키로 인증서를 생성하는 것을 말한다. 이 처럼 인증서의 서명 확인하기 위해서는 추가적으로 여러 개의 인증서를 검증해야 한다. 하지만 한번 검증된 인증서를 매번 다시 검증하지 않는다. 인증서 정책에 따라 바뀔 수 있지만 보통 한번 확인된 인증서는 캐시에 보관되며 나중에는 유효기간과 폐지여부만 확인한다.

2.4 공개키 기반구조

공개키 기반구조(public key infrastructure)란 공개키 암호알고리즘을 안전하게 사용하기 위해 필요한 서비스를 제공하기 위한 기반구조를 말한다. 공개키 기반구조의 구성요소는 다음과 같다.

- 정책승인기관(PAA, Policy Approving Authority): 공인인증 서비스 전반의 정책과 절차 수립
- 정책인증기관(PCA, Policy Certification Authority): PAA에서 승인된 전반적인 정책을 확장하거나 세부화된 정책 생성
- 인증기관(CA, Certification Authority): 인증서를 발급하는 기관
- 등록기관(RA, Registration Authority): 가입자의 등록과 초기 인증(신원확인)을 담당
- 인증서 디렉토리(certificate directory, certificate repository): 인증서 검색 서비스를 제공
- 키 복구 서버(key recovery server)
- 타임 서버(time server)
- OCSP(Online Certification Status Protocol) 서버: 인증서의 폐지 여부를 온라인으로 확인하는 서비스를 제공하는 서버

최상위 인증기관이 보통 PAA 역할을 수행한다. PAA는 보통 PKI 전반에 사용되는 정책과 절차를 수립하며, 이 정책과 절차는 PKI의 모든 구성요소가 따라야 한다. PCA가 수립한 정책의 적법성을 검증하고 하부 PCA가 정해진 정책 준수 상태를 감시한다. PKI 내/외에서 상호인증(cross-certification)을 위한 정책을 수립하고 승인한다. 상호인증에 대해서는 2.x에서 자세히 설명한다.

PCA는 PAA에서 승인된 전반적인 정책을 확장하여 자신의 도메인 내의 가입자와 인증기관이 따라야 할 정책을 수립한다. 예를 들어 인증서의 유효기간을 얼마로 할 것인지, 인증서 폐지 목록 발급 주기 등을 결정한다.

등록기관은 가입자의 신원을 확인해주는 기관이다. 만약 등록기관이 없으면 인증기관이 등록기관의 역할까지 수행해야 한다. 오늘날 인터넷 बैं킹을 보면 은행이 등록기관 역할을 하고 있다고 보면 된다. 등록기관은 가입자의 신원을 확인한 다음에 신원 확인 증명서를 인증기관에게 전달한다. 등록기관은 또 가입자가 원하면 가입자를 대신하여 공개키 쌍을 생성해줄 수 있다. 이 경우 키 복구와 백업 서버 역할까지 등록기관이 할 수 있다. 가입자의 인증서 폐지 요청을 받아 그 요청의 정당성을 확인하는 역할도 수행한다.

인증기관은 PKI의 구성요소 중 가장 중요한 요소이다. 인증기관의 역할은 크게 다음과 같다.

- 인증서 발급
- 인증서 폐지
- 인증서 정지
- 인증서 갱신(certificate renewal), 재발급(certificate update)

인증서의 발급은 등록기관의 도움을 받아 신원을 명백하게 검증한 다음에 발급하여야 한다. 인증서의 발급은 인증기관의 서명키를 이용하여 발급하기 때문에 인증기관의 서명키가 공개키 기반구조의 안전성에 매우 중요하다. 이 때 적용되는 중요한 규칙이 하나 있다. 인증기관은 자신의 인증서 유효기간을 초과하는 인증서를 절대 발급하지 않는다.

인증서의 유효기간을 적절하게 설정하여도 유효기간 내에 인증서를 폐지하고 종종 새로 발급받을 필요가 있다. 이것은 신용카드와 그 원리가 비슷하다. 신용카드도 유효기간이 정해져 있지만 분실하거나 도난 되었을 경우에는 유효기간이 만료되지 않아도 새롭게 재발급 받아야 한다. 인증서도 이와 동일하게 개인키를 분실하였거나 도난당하였거나 인증서에 포함된 정보가 변경되면 인증서의 유효기간 만료되지 않아도 새로 발급받아야 한다.

인증서의 정지는 폐지와 달리 일시적으로 인증서의 사용을 정지하는 것이다. 예를 들어 해외로 출장을 때문에 한 달 동안 인증서를 사용할 필요가 없는 경우 인증서를 한 달 동안 정지시켜 놓을 수 있다.

인증서의 갱신은 인증서의 유효기간이 만료되어 다시 발급받는 것을 말하며, 이 때 유효기간만 변경(renewal)할 수도 있고, 공개키쌍을 새롭게 생성하여 발급(update)받을 수 있다. 전자를 갱신이라 하고, 후자는 **재발급**이라 한다.

인증서 디렉토리는 인증기관의 인증서, 가입자의 인증서, 인증서 폐지 목록의 공개 보관소이다. 즉, 사용자들은 이 디렉토리에 접근하여 자신이 필요한 상대방의 인증서를 얻을 수 있다. 국제 표준으로 X.500이 있지만 이 표준보다는 LDAP(Lightweight Directory Access Protocol)을 이용하여 보통 구현된다. 하지만 현재 인증서 디렉토리는 널리 사용되고 있지 못하다.

2.5 인증서 폐지

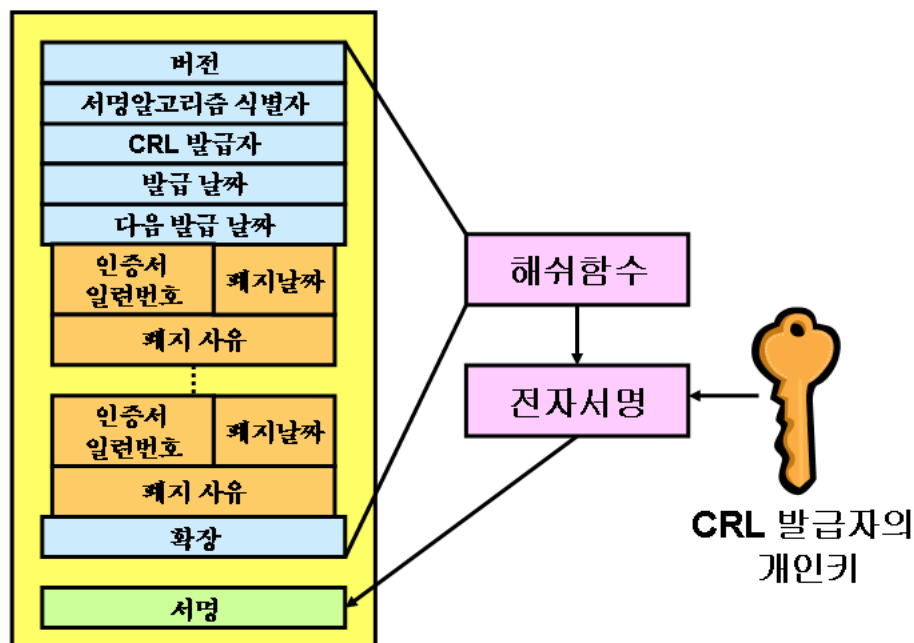
인증서가 어떤 이유에서든지 폐지되면 이 인증서를 더 이상 사용할 수 없으며, 그것의 사용은 매우 위험할 수 있다. 예를 들어 공격자가 인증서에 포함된 공개키에 대응되는 개인키를 확보하면 이 인증서는 더 이상 그 원래의 역할을 할 수 없다. 따라서 인증서 폐지의 목적은 시기적절하게 폐지를 처리하여 폐지된 인증서를 사용에 따른 피해를 줄이는 것이다.

앞서 언급한 바와 같이 신용카드와 인증서는 인증서 폐지 측면에서 유사한 점이 많다. 신용카드의 경우 초기에는 각 상점에게 폐지된 신용카드 번호 목록을 분해하여 신용카드 폐지를 처리하였다. 즉, 상점들은 신용카드를 승인하기 전에 이 목록에서 확인해야 했다. 현재는 온라인으로 신용카드의 폐지 여부를 확인한다. 인증서도 이와 비슷한 방법으로 인증서 폐지를 처리한다.

인증기관은 주기적으로 폐지된 인증서 목록을 발급하며, 이 목록을 인증서 폐지 목록(CRL)이라 하며, 이 목록도 X.509 표준에 정의되어 있다. 이 목록도 인증서와 마찬가지로 임의로 조작되거나 만들 수 없어야 한다. 따라서 이 목록도 인증서와 마찬가지로 인증기관이 전자서명을 하여 발급한다.

인증서 폐지 목록은 보통 폐지된 인증서들에 관한 정보만 유지한다. 이와 같은 접근 방법을 나쁜 목록(bad-list) 방법이라 한다. 이와 반대로 좋은 목록(good-list) 방법도 있다. 좋은 목록의 경우에는 이 목록에 포함된 인증서만 사용해야 하며, 나쁜 목록은 이 목록에 포함되지 않은 인증서만 사용해야 한다. 좋은 목록 방법은 나쁜 목록 방법에 비해 다음과 같은 이유에서 안전하다. 좋은 목록에 포함되어 있음에도 불구하고 유효하지 않은 인증서일 수 있고, 나쁜 목록에 포함되어 있지 않음에도 불구하고 유효하지 않을 수 있다. 이 측면에서 차이가 없다. 하지만 좋은 목록이 공개되어 있으므로 잘못된 인증서를 발견하여 조치를 취할 확률이 나쁜 목록에 포함되어 있지 않은 유효하지 않은 인증서를 발견하여 조치를 취할 확률보다 높다. 이 측면에서 좋은 목록 방법이 보다 안전하다. 하지만 좋은 목록은 나쁜 목록보다 상대적으로 매우 크며, 나쁜 목록보다 빠르게 갱신되어야 한다.

2.5.1 인증서 폐지 목록의 구성



<그림 2.2> 인증서 폐지 목록의 구성

인증서 폐지 목록은 그림 2.2처럼 각 폐지된 인증서마다 그것의 일련번호, 폐지날짜, 폐지 사유를 포함하고 있으며, 인증서와 비슷하게 기본적으로 다음과 같은 정보가 포함된다.

- 버전: 인증서 폐지 목록의 버전을 나타내며, 값으로 "version 2"를 사용하거나 이 필드를 사용하지 않는다. 이 필드를 사용하지 않으면 버전 1을 나타낸다.
- 서명 알고리즘: 이 인증서 폐지 목록을 발급할 때 사용한 서명 알고리즘의 식별자
- 발급자: 이 인증서 폐지 목록을 발급한 인증기관의 DN(Distinguished Name)
- 발급날짜: 이 인증서 폐지 목록을 발급한 날짜
- 다음 발급날짜: 차기 인증서 폐지 목록을 발급할 날짜

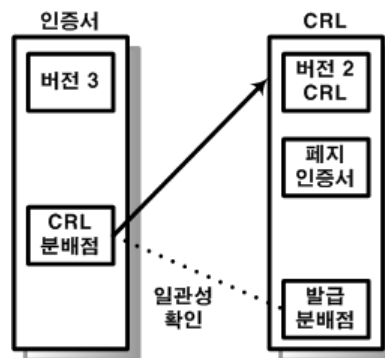
인증서 폐지 목록도 이 기본 정보 외에 확장 필드에 다음과 같은 정보를 포함할 수 있다.

- 인증기관 키 식별자: 이 인증서 폐지 목록을 확인할 때 사용하는 확인키에 대한 정보
- 델타 CRL 지시자(delta CRL indicator): 이 CRL이 델타 CRL인지 여부를 나타낸다.
- CRL 영역(CRL scope): CRL이 다양한 형태로 분할되어 제공될 경우 이 필드는 이 CRL이 어떤 기준으로 분할되어 있으며, 이 CRL에는 어떤 인증서들이 포함되어 있는지 나타낸다.
- 상태 참조(status referrals): 간접 CRL에서 사용
- 발급 분배점(issuing distribution point): CRL 분배점 이름과이 CRL에 포함되어 있는 인증서들의 종류를 나타낸다.

델타 CRL, 간접 CRL은 다음 절에서 보다 자세히 설명한다.

2.5.2 인증서 폐지 목록의 종류

인증기관의 도메인에 속한 폐지된 모든 인증서를 하나의 목록에 유지할 경우 이 목록을 **완전 인증서 폐지 목록(complete CRL)**이라 한다. 완전 인증서 폐지 목록은 목록이 너무 커질 수 있어 접근 비용이 증가하여 효율성이 떨어질 수 있다. 즉, 확장성 측면에서 바람직하지 못하다. X.509는 인증서 폐지 목록을 **EPRL(End-entity Public-key Revocation List)**과 **CARL(Certification Authority Revocation List)**로 나누어 발급하도록 하고 있다. EPRL은 최종 사용자들의 폐지된 인증서 정보를 유지하며, CARL은 인증기관의 폐지된 인증서 정보를 유지한다. CARL은 빈 목록이거나 크기가 매우 작을 것이므로 완전 인증서 폐지 목록 형태로 유지가 가능하지만 EPRL은 완전 인증서 폐지 목록 형태로 유지하는 것이 어렵다. 따라서 EPRL은 다양한 방법으로 분할하여 유지된다.

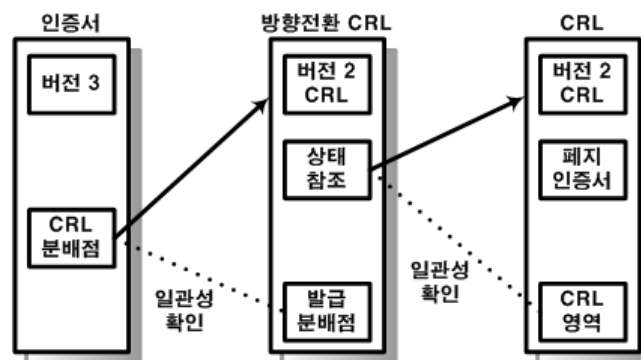


<그림 2.3> CRL 분배점

CRL 분배점은 단일 도메인의 폐지된 인증서들을 다양하게 분류하여 여러 개의 CRL에 나누어 관리할 수 있도록 해준다. CRL 분배점은 인증서의 확장 필드에 기록되는 정보로서 이 인증서와 관련된 폐지 정보가 수록된 CRL의 위치(DNS 이름, IP 주소 등)를 나타낸다. 이것을 사용하면 CRL을 관리할 수 있는 크기로 나누어 제공할 수 있으며, 사용자가 사전에 CRL이 어디에 있는지 알 필요가 없다. 즉, 그림 2.3과 같이 인증서를 확인할 때 CRL 분배

점 정보를 이용하여 그 위치에서 최신의 CRL를 다운받아 이 인증서의 폐지 여부를 확인할 수 있다. 그림 3에 일관성 확인은 CRL 교체 공격을 방지하기 위한 수단이다. CRL 교체 공격이란 원래 위치에 있는 CRL을 다른 위치에 CRL로 교체하는 공격을 말한다. 만약 사용자가 교체된 사실을 모르면 폐지가 된 인증서를 유효한 것으로 착각할 수 있다. 보통 일련번호, 인증서의 종류, 폐지 사유 등을 기준으로 CRL을 분할하여 유지한다. 이 때 폐지 사유 같은 경우에는 CRL 분배점만으로는 사용할 수 없는 기준이다.

CRL 분배점을 사용할 경우 CRL 분할이 고정된다. 즉, 인증서를 발급할 시점에서 그것의 CRL 위치를 미리 결정하여 인증서에 그 정보를 기록해야 한다. 따라서 인증서는 그 인증서의 유효기간이 끝날 때까지 CRL의 위치가 고정된다. 그러나 보다 유연하게 CRL을 관리하기 위해서는 CRL의 크기와 위치를 동적으로 변경할 필요가 있다.



<그림 2.4> 방향전환 CRL

방향전환 CRL(redirect CRL)은 CRL 분배점을 사용할 경우 각 인증서와 관련된 폐지 정보가 있는 위치가 고정된다는 문제점을 해결하고자 X.509 2000년 버전에 추가된 개념이다. 인증서를 발급할 때 그 인증서와 관련된 폐지 정보가 수록될 CRL의 위치를 CRL 분배점에 직접 기록하지 않고 이 인증서와 관련된 방향전환 CRL의 위치 정보를 기록한다. 이 인증서의 폐지 여부를 알고 싶으면 이 인증서와 관련된 CRL를 바로 다운받을 수 없고, 먼저 방향전환 CRL을 검색하여 실제 CRL의 위치를 알아내야 한다. 따라서 특정 인증서와 관련된 폐지 정보가 기록된 CRL의 위치를 변경하고자 하면 방향전환 CRL의 정보만 변경하면 된다.

CRL 분배점은 폐지된 인증서들을 다양한 기준에 따라 분할하여 제공할 수 있도록 하여 완전 인증서 폐지 목록의 크기 문제점을 극복하고 있다. 이 문제점을 극복하는 또 다른 방법은 **델타 CRL(delta CRL)**이다. 델타 CRL은 점진적으로 폐지 정보를 공개할 수 있도록 해준다. 델타 CRL은 그것에 기준이 되는 기준 CRL(base CRL)이 존재하며, 델타 CRL은 기준 CRL이 발표될 당시에 없었던 내용만 추가된다. 따라서 델타 CRL의 크기는 상대적으로 매우 작다. 이것의 효과는 다음과 같다. 사용자는 매번 완전 인증서 폐지 목록을 받을 필요가 없고, 새롭게 발급된 델타 CRL만 받아 이미 가지고 있는 기준 CRL과 델타 CRL을 이용하여 인증서의 폐지 여부를 판단하면 된다.

간접 CRL(indirect CRL)은 여러 인증기관이 발급한 CRL를 하나로 결합하여 사용할 수 있도록 해준다. 간접 CRL은 사용자들이 여러 개의 CRL를 다운받아 확인해야 하는 문제점을 해결하여 주지만 반대로 완전 CRL처럼 크기 문제가 발생할 수 있다.

2.5.3 온라인 질의 메커니즘

온라인 질의 메커니즘이란 온라인으로 특정 인증서의 폐지 여부를 문의하는 것이다. 이와 같은 메커니즘은 다음과 같은 측면에서 주기적 공표 메커니즘과 차이가 있다. 온라인 서버가 항상 서비스를 제공하고 있어야 한다. 주기적 공표 메커니즘은 사용자들이 정보를 캐쉬할 수 있으므로 오프라인 작업이 가능하다. 온라인 질의 메커니즘은 IETF PKIX 워킹그룹에서 주로 연구되었다. OCSP(Online Certificate Status Protocol)는 1999년에 RFC 2560으로 표준화되었다. OCSP는 매우 단순한 질의 프로토콜로서 OCSP 요청에 대해 OCSP 서버는 "good", "revoked", "unknown" 세 가지 방법으로 응답한다. 즉, 인증서가 유효하다, 유효하지 않다, 모른다, 세 가지 형태로 응답한다. OCSP 서버는 기존 CRL을 이용하여 사용자를 대신하여 상태를 검색할 수도 있고, 다른 방법을 통해 인증서의 폐지 여부를 확인하여 응답할 수 있다. OCSP의 응답은 반드시 서명되어 전달해야 한다. OCSP는 사용자의 요청과 응답에 대해서만 정의되어 있고, OCSP 서버가 어떻게 최신의 인증서 폐지 정보를 획득하는지에 대해서는 정의되어 있지 않다.

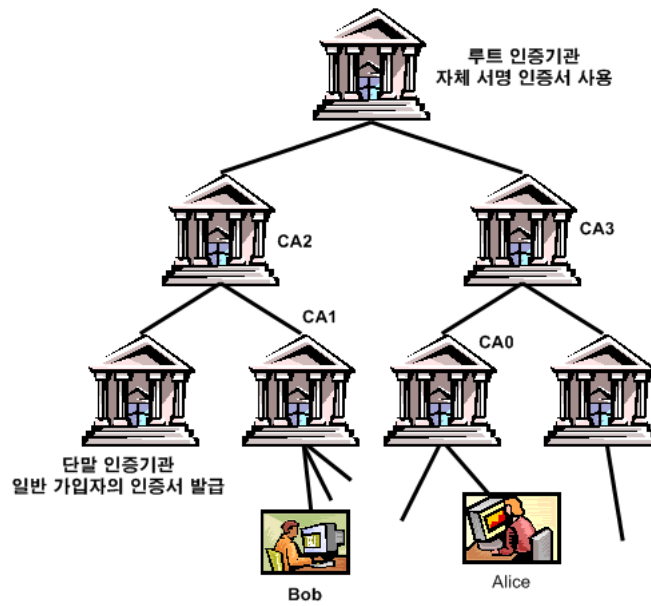
IETF PKIX 워킹그룹은 OCSP 외에 SCVP(Simple Certificate Validation Protocol)도 제안하고 있다. SCVP는 OCSP와 달리 인증서 폐지 정보만 알려주는 것이 아니라 인증서 검증 자체를 대신하여 주는 프로토콜이다.

2.6 신뢰 모델

하나의 인증기관만이 존재하는 것이 아니기 때문에 인증기관들 간에 신뢰 모델이 전체 PKI 동작에 중요한 역할을 하게 된다.

2.6.1 순수 계층구조 모델

모든 인증기관이 단일 계층구조를 형성하고 있으면 순수 계층구조 모델이라 한다. 이 계층구조에서 부모 인증기관이 자식 인증기관의 인증서를 발급하여 주며, **단말 인증기관**은 일반 가입자의 인증서를 발급하여 준다. **중간 인증기관(intermediate CA)**이란 루트 인증기관과 단말 인증기관을 제외한 나머지 인증기관들을 말하며, 중간 인증기관은 오직 다른 인증기관들만 인증을 한다. **루트 인증기관**은 자신을 인증해줄 인증기관이 없으므로 자체 서명 인증서를 사용한다. 따라서 순수 계층구조 모델에서 루트CA는 모든 신뢰 관계의 공통된 기준점(anchor)이 된다. 즉, 모든 가입자는 공통적으로 루트CA까지 신뢰를 하고 있으며, 루트CA를 포함하여 자신의 위치부터 루트CA까지 경로 상에 있는 모든 인증기관의 인증서를 자신의 시스템에 캐쉬하고 있다.

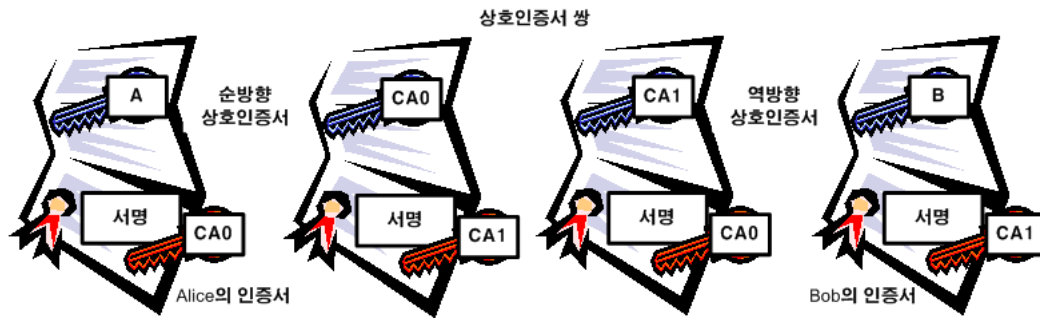


<그림 2.5> 순수 계층구조 모델

그림 2.5에서 Alice가 Bob의 인증서를 확인하기 위해서는 루트 인증서를 이용하여 인증기관 CA2의 인증서를 확인하고, 인증기관 CA2의 인증서를 이용하여 인증기관 CA1의 인증서를 확인한 다음, 인증기관 CA1의 인증서를 이용하여 Bob의 인증서를 확인해야 한다. 이 때 Alice가 Bob을 인증하기 위한 **인증 경로(certification path)**는 **ROOT→CA2→CA1**이 된다. Alice는 **순방향(forward)** 방식으로 이 경로를 형성한다. 즉, Bob의 인증서를 발급한 인증기관이 인증서를 얻고, 그 다음 이 인증기관의 인증서를 발급한 인증기관의 인증서를 얻는다. 이와 같은 방법으로 루트CA까지 찾아간다. 이 모델의 경우에는 하나의 가입자의 인증서를 확인하기 위한 인증 경로가 고정되어 있어 편리하지만 현실적으로 전세계의 다양한 인증기관을 이와 같은 순수 계층구조 모델로 구성하기가 어렵다는 문제점이 있다.

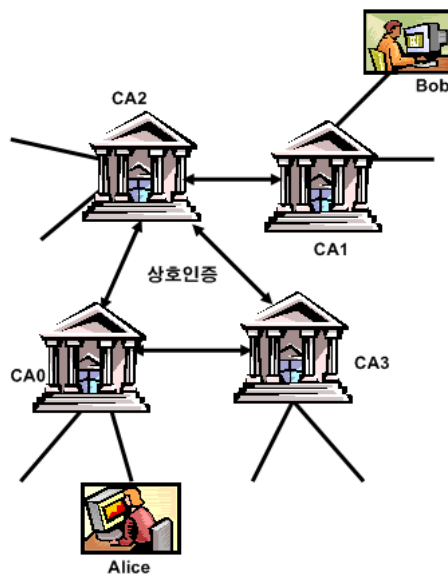
인증기관의 개인키가 유효기간 내에 노출되어 교체하여야 하는 경우 그 인증기관이 발급한 모든 인증서를 갱신해야 한다. 이것은 모델과 상관없이 어떤 모델에서도 인증기관의 인증서가 폐지되면 그 인증기관이 발급한 모든 인증서는 갱신되어야 한다. 여기서 갱신이란 인증서의 유효기간과 발급자 키 식별자 정보는 변경하여 인증기관의 새 개인키로 다시 발급하는 것을 말한다.

2.6.2 네트워크 구조 모델



<그림 2.6> 상호 인증

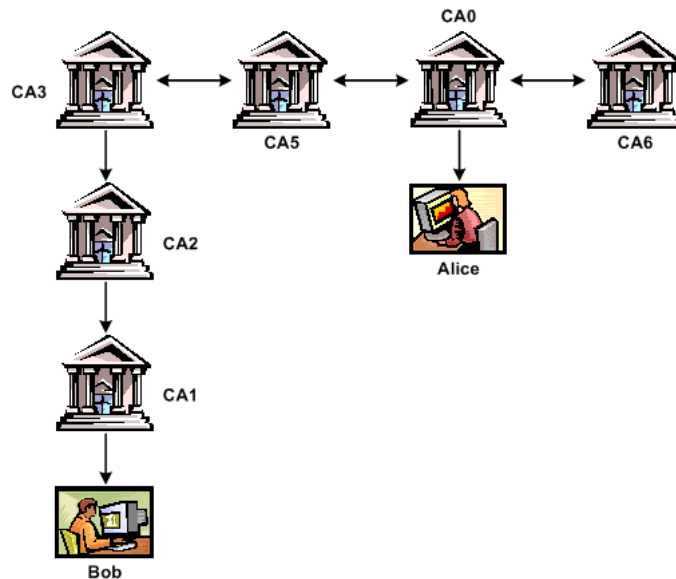
모든 인증기관들이 계층구조 모델에서 단말 인증기관 역할을 하고 있고, 인증기관들은 서로 **상호 인증(cross-certification)**을 통해 연결되어 있다면 **순수 네트워크 모델**이라 한다. 상호 인증이란 두 개의 인증기관이 서로 상대방의 공개키를 인증하여 주는 인증서를 발급하여 사용하는 경우를 말하며, 이와 같은 인증서를 상호인증서(cross-certificate)라 한다. 그림 2.7에서 Alice의 인증서는 인증기관 CA0가 발급하였으며, Bob의 인증서는 인증기관 CA1이 발급하였다. 또한 Alice는 CA0를 신뢰하고 있지만 CA1은 신뢰하고 있지 않으므로 Bob 인증서를 확인하기 위해서는 추가적인 인증서가 필요하다. 즉, Alice는 자신이 신뢰하는 인증기관이 발급한 CA1의 인증서가 필요하다. 상호인증서는 이 때 사용할 수 있다. Alice는 자신이 신뢰하고 있는 CA0가 발급한 CA1에 대한 역방향(reverse) 상호인증서가 있으면 이를 통해 Bob의 인증서를 확인할 수 있다. 순방향(forward), 역방향 상호인증서 용어는 X.509 1997년도 버전에 사용된 용어이다. 이 용어는 현재 사용되고 있지 않지만 CA0 입장에서 다른 인증기관이 발급한 자신의 인증서를 순방향 상호인증서라 하며, 다른 인증기관을 위해 자신이 발급한 인증서를 역방향 상호인증서라 한다. 그림 2.6에서 CA0 입장에서 CA1이 발급한 본인 인증서를 순방향 상호인증서이며, 자신이 발급한 CA1 인증서가 역방향 상호인증서가 된다. 2000버전에서는 이 용어 대신에 “issued to this CA”와 “issued by this CA”로 바꾸어 사용하고 있다.



<그림 2.7> 순수 네트워크 모델

순수 네트워크 모델은 순수 계층구조 모델과 달리 하나의 가입자의 인증서를 확인하기 위한 경로가 여러 개 존재한다. 따라서 경로를 찾는 비용이 계층구조 모델에 비해 비싸다. 그림 2.7에서 Alice가 Bob의 인증서를 확인하기 위한 인증경로는 CA0→CA2→CA1, CA0→CA3→CA2→CA1 등 다양하게 존재한다. 이 인증경로를 찾는 방법은 다양한 그래프 알고리즘을 활용할 수 있다. 이 모델에서도 순방향 방식으로 인증경로를 찾을 수 있다. Bob의 인증서는 CA1이 발급하였으며 CA1 디렉토리를 검색하면 CA2가 발급한 CA1 인증서가 있음을 알 수 있다. CA2 디렉토리를 검색하면 CA0와 CA3가 발급한 인증서가 있음을 알 수 있다. 그런데 CA0는 Alice의 신뢰 기준점이므로 인증경로를 찾는 것이 완료된 것이다. 이 예에는 간단하기 때문에 이와 같이 인증경로를 찾을 수 있지만 복잡한 네트워크 구조에서는 인증경로를 찾는 것이 쉽지 않을 수 있으며, 가장 효율적인 인증경로를 찾는 것은 어려울 수 있다. 다시 말하면 인증경로가 짧을 수록 인증서를 확인하는 비용이 저렴하지만 가장 짧은 인증경로를 찾는 것은 매우 어렵고, 그 자체가 비용이 많이 소요될 수 있다. 이것을 해결하기 위해 완전 그물형(full mesh) 구조를 사용할 수 있다. 즉, 모든 인증기관이 상호인증하도록 하는 것이다. 그러면 항상 인증경로의 길이는 2가 된다. 하지만 확장성이 떨어지며, 현실성이 없는 접근 방식이다. 따라서 현재에는 순수 계층구조와 네트워크 모델을 결합하여 루트CA 간에는 상호인증을 하는 방식을 사용한다. 이 방법을 혼합 모델이라 한다.

2.6.3 혼합 모델



<그림 2.8> 혼합 모델

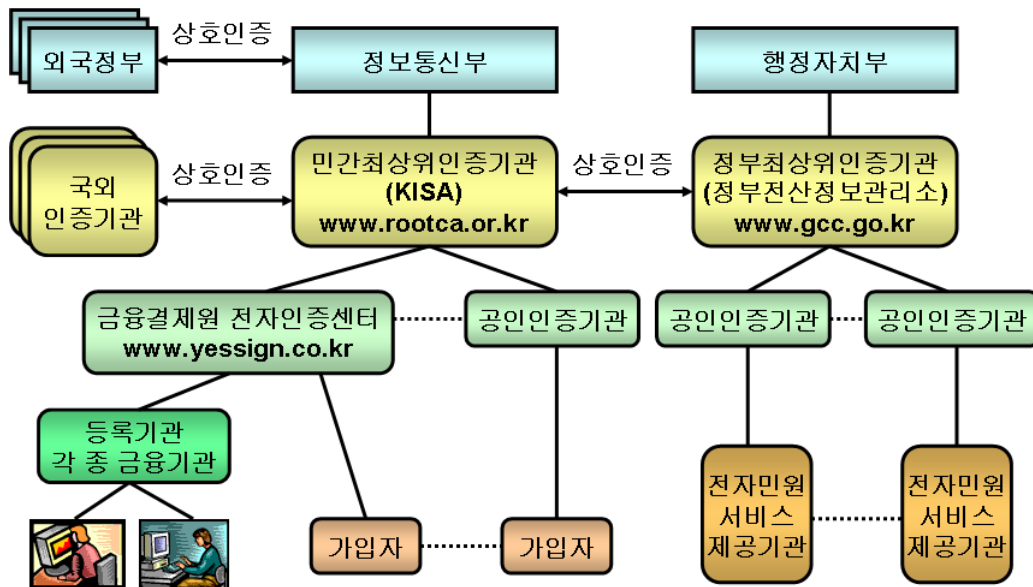
이 모델에서는 계층구조와 네트워크 모델을 혼합하여 사용한다. 즉, 전세계의 모든 인증기관을 단일 계층구조로 형성하기 어렵기 때문에 각 나라에서 하나 또는 두 개의 계층구조로 모든 인증기관을 구성하며, 계층구조 간에는 상호인증을 통해 관계를 형성한다. 예를 들어 그림 2.8에서 Bob이 전자서명한 전자우편을 Alice가 수신하였다고 하자. 이 때 Alice는 기존처럼 순방향 방식으로 CA3→CA2→CA1의 인증경로를 찾을 수 있다. 하지만 더 이상은 순방향 방식으로 찾아가기 어렵다. 따라서 이 때에는 역방향 방식으로 인증경로를 찾을 수

있다. 여기서 Alice의 신뢰 기준점은 CA0이다. 여기서부터 CA3까지의 경로를 찾아가는 것이다. 이 예에서는 CA0는 CA5와 CA6에 대한 역방향 상호인증서를 발급하였다. 따라서 CA5와 CA6의 디렉토리를 접근하여 검색한다. 이 경우 CA5 디렉토리에는 CA3에 대한 역방향 상호인증서가 있으므로 최종 인증경로는 CA0→CA5→CA3→CA2→CA1이 된다.

2.6.4 푸시 모델 vs. 풀 모델

푸시(push)와 풀(pull) 모델은 인증서를 획득하는 방법에 대한 모델이다. 푸시 모델은 하나의 인증서를 확인하기 위해 필요한 모든 인증서를 함께 전달하는 방식을 말하고, 풀 모델은 하나의 인증서만 전달하고, 이 인증서를 확인하기 위해 필요한 추가적인 인증서는 사용자가 문의하여 직접 가져오는 방식을 말한다. 순방향 방식의 탐색을 사용할 경우에는 인증서에 있는 발급자 정보를 이용하여 필요한 인증서를 추가적으로 받을 수 있지만 그렇지 않은 경우에는 가입자가 필요한 인증서를 찾는 것이 어려울 수 있다. CRL과 관련하여 이 용어가 사용되는 경우도 있다. 이 역시 사용자가 필요한 CRL를 문의하여 가져오는 방식이면 풀 모델이 되고, 주기적으로 보내주거나 인증서를 요청하였을 때 관련 CRL를 함께 보내면 푸시 모델이 된다.

2.6.5 국내 인증체계



<그림 2.9> 국내 인증체계

국내 인증체계는 그림 2.9와 같다.

2.6.6 기타 모델

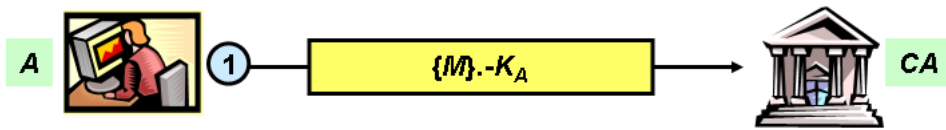
가장 널리 사용되는 신뢰 모델은 혼합 모델이지만 이 외에 독점 모델, 소수 독점 모델 (oligarchy) 등이 있다. 소수 독점 모델은 다른 말로 웹 모델이라 하며 몇 개의 CA만 존재하

는 모델로서, 현재 브라우저 등에서 널리 사용되는 모델이다. 이 모델에서 제품(브라우저)을 설치하면 여러 개의 CA의 인증서를 자체적으로 설치되며, 이들 CA가 발급한 인증서들은 사용자가 신뢰해야 하는 모델이다.

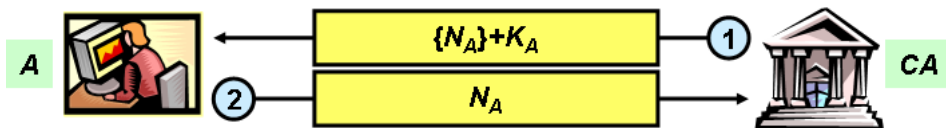
2.7 공개키 쌍 생성

가입자나 인증기관의 공개키 쌍의 생성은 크게 두 가지 방식으로 생성이 가능하다. 첫째는 기본 방식(BAS, Basic Authenticated Scheme)이라 하며, 이 경우에는 가입자가 직접 공개키 쌍을 생성하여 공개키만을 안전하게 인증기관/등록기관에 전달하여 인증서를 발급받는 방식이다. 이 방식은 인증기관이 가입자의 개인키를 모르기 때문에 부인방지에 보다 효과적이다. 두 번째 방법은 중앙집중 방식(centralized scheme)이며, 이 방식에서는 등록기관이 공개키 쌍을 생성하여 대신 인증서를 발급받아 준다. 이 경우 개인키는 안전하게 가입자에게 전달되어야 한다. 이 방식은 가입자의 개인키를 인증기관이 알고 있으므로 인증기관에 대한 신뢰가 절대적이어야 한다. 이 방식은 키 위탁과 키 복구가 용이하다는 장점을 가지고 있다.

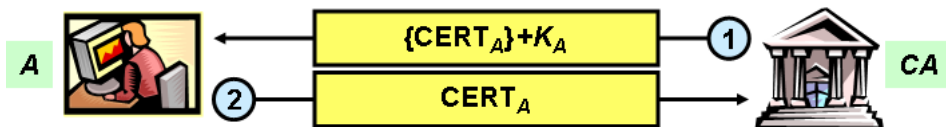
첫 번째 방식에서는 가입자가 요청한 인증서에 대응되는 개인키를 알고 있다는 것을 반드시 확인해야 한다. 이와 같은 확인은 그림 2.11부터 2.13에 기술된 프로토콜들을 이용하여 쉽게 확인할 수 있다.



<그림 2.10> 개인키 확인 프로토콜: 서명방식



<그림 2.11> 개인키 확인 프로토콜: 복호화 능력 검증 방식 1



<그림 2.12> 개인키 확인 프로토콜: 복호화 능력 검증 방식 2

그림 2.10 방식에서는 임의의 메시지를 서명하여 인증기관에 전달함으로써 공개키에 대응되는 개인키를 가지고 있음을 증명하는 방식이고, 그림 2.11과 2.12는 인증기관이 보낸 암호문을 복호화할 수 있음을 증명하여 개인키를 가지고 있다는 것을 입증하는 방식이다. 그림 2.12의 경우에는 인증서 자체를 공개키를 암호화하여 전달함으로써 이 방식에서는 대응되는 개인키가 없는 가입자는 인증서 자체를 얻을 수 없다.

참고문헌

[1] Carlisle Adams and Steve Lloyd, *Understanding PKI*, 2nd Ed., Addison Wesley, 2003.

[2] Steve Lloyd, *Understanding Certification Path Construction*, PKI Forum White Paper, 2002.