

공개키 확인서 취소 방식의 비교

오중효*, 박기철*, 이국희*, 조갑환**, 문상재*

* 경북대학교 전자전기공학부

** (주)대구종합정보센터

Comparison of Public Key Certificate Revocation Schemes

Joong-Hyo Oh*, Ki-Chul Park*, Kook-Heui Lee*, Kap-Hwan Cho**, and Sang-Jae Moon*

* School of Electronic and Electrical Engineering, Kyungpook National University

** Taegu Information Network Center, Inc

요약문

공개키 암호 알고리즘은 사용자의 공개키가 그 사용자와 대응되는지를 확인할 수 있는 방법이 필요하다. 이를 해결하기 위해 사용자들이 신뢰할 수 있는 인증기관에서 각 사용자의 공개키의 확인서를 발급한다. 그러나 비밀키의 노출, 사용자의 자격의 박탈 및 유효기간의 만료 등으로 인하여 확인서를 취소하여야 할 경우가 있다. 따라서 각 사용자는 상대방의 공개키가 유효한 것인지를 확인하여야 한다. 본 고에서는 지금까지 제시된 공개키 확인서의 취소 여부를 확인하는 방법들을 분석하고 효율적인 방법을 제시한다.

1. 서론

인터넷이 발달함에 따라 문서의 전송이나 전자 상거래 등 인간 활동의 많은 부분이 네트워크 상에서 이루어지고 있다. 그러나 인터넷이라는 매체가 불완전한 개방형 구조를 가지므로 인해 처리되는 정보의 보호가 큰 과제로 대두되었으며 이를 해결하는 열쇠로 암호학적인 방법이 강구되었다. 암호학적인 보호 방법에는 대칭키 암호시스템과 공개키 암호 시스템이 있으며 이중 공개키 암호 시스템은 효율적인 인증 및 디지털 서명의 제공으로 널리 사용되고 있다.

한편, 공개키 암호시스템은 공개된 사용자의 공개키가 그 소유자와 대응되는 지를 확인할 수 있어야 하며 그렇지 못할 경우 자신의 공개키를 다른 사람의 공개키로 위조할 수 있다. 이를 해결하기 위해 신뢰할 수 있는 제 3 자인 인증기관에서 각 사용자의 신분과 공개키를 확인한 후 공개키 확인서를 발급하고 사용자는 이를 검증하여 공개키 인증 문제를 해결한다. 그러나 사용자의 실수로 비밀키가 노출되었거나 자격의 박탈, 유효기간 만료 등의 이유로 확인서를 취소해

야 할 경우가 있다. 따라서 각 사용자는 수신한 공개키가 유효한 것인지를 확인해야 하며 이는 인증기관이 취소된 확인서에 대한 정보를 공개하거나 사용자가 원하는 확인서의 상태를 인증기관에 직접 의뢰를 함으로써 알 수 있다.

지금까지 공개키 확인서 취소 여부를 알 수 있는 일반적인 방법으로 인증기관이 취소된 확인서의 목록(certificate revocation list, CRL)을 공개하는 것이 있다[1,2]. 이는 사용자가 CRL을 수신하여 원하는 공개키가 목록에 포함되어 있는지를 확인하여 취소 여부를 판단한다. 이 경우 취소 확인서가 늘어나면 부하가 커지며 갱신을 주기적으로 하므로 확인서의 상태를 실시간으로 알 수 없는 단점이 있다. CRL의 단점을 개선하기 위해 Micali는 개개의 공개키 확인서에 대해 유효성 여부를 공개하는 방법[4,5]을 제시했으며 Kocher와 Naor는 트리를 사용하여 취소 확인서를 검색하는 방법을 각각 제안했다[6,7]. 또한 CRYPTO '98에서 Micali의 방법을 개선하여 통신량을 줄이는 방법을 제안했다[8]. 본 논문에서는 지금까지 제안된 공개키 확인서의 유효성 여부를 판별하는 방법을 살펴보고 비교 분석하여 가장 효율적인 방법을 제시한다.

2. 공개키 확인서 취소 여부 확인 방법

2.1. 기본 가정

본 논문에서 통신량을 분석할 때 필요한 인증기관의 구성은 하나 이상의 인증기관이 존재할 수 있다고 생각하며 디렉토리는 각 인증기관에 속해 있을 수 있으며 또는 전체에 하나의 디렉토리가 있을 수도 있다. 즉, 인증기관과 디렉토리 간 또는 사용자와 디렉토리 간의 통신량을 생각할 때 어느 특정 디렉토리를 대상으로 하는 것이 아니고 전체 디렉토리에 대해서 네트워크 상에 발생하는 통신량을 고려한다. 즉, 디렉토리가 전체에 하나가 있다고 가정하는 것과 동일하다.

전체 총 확인서의 수를 n 이라 하고 한 인증기관의 평균 확인서 수를 k 라 한다. 평균적으로 전체 확인서의 p 배가 취소된다고 가정하고 취소된 확인서의 수를 r 로 나타낸다. 즉, $r = n \times p$ 와 같이 구할 수 있으며 일반적으로 취소되는 비율은 10%정도로 본다[4]. 또한 얼마나 자주 갱신할 것인지를 결정해야 하는데 하루에 갱신하는 횟수를 T 로 표기하고 확인서의 일련번호를 l 비트로 표기 하며 사용자들이 하루동안 디렉토리에 정보를 요구하는 횟수를 q 라 가정한다.

2.2. 확인서 취소 목록(Certificate Revocation List, CRL)

공개키 확인서의 취소 여부를 알 수 있는 가장 일반적인 방법이며 현재까지 표준화 중인 여러 공개키 기반 구조에서 이 방법을 채택하고 있다[1,2]. 인증기관이 각 사용자의 공개키 확인서를 발급하며 발급된 확인서를 취소해야 할 요건이 발생할 경우 그 확인서의 일련번호를 확인서 취소 목록에 추가 한다. 이 목록은 주기적으로 갱신되며 모든 사용자들이 접근할 수 있는 공개 저장소에 공개한다. 사용자들은 주기적으로 갱신되는 확인서 취소 목록을 획득하여 수신된 공개키의 적법성 여부를 판별하여야 한다.

이 방법은 간단한 반면 확인서 취소 목록이 누적될 경우 다운로드 해야 파일의 크기가 커지

는 단점이 있으며 목록에서 대상이 되는 확인서가 존재하는지 확인해야 하므로 굉장히 많은 부하가 걸린다.

요구되는 통신량을 계산해 보면 인증기관과 디렉토리사이에는 하루에 취소된 확인서에 대해서 갱신히트 수 만큼 목록을 전송해야 하므로 $T \times r \times l = T \times n \times p \times l$ 비트의 양을 전송해야 하며 디렉토리가 하루동안 사용자들에게 전송해야 하는 양은 $q \times p \times k \times l$ 비트이다.

2.3. Certificate Revocation System(CRS)

확인서 취소 목록의 단점을 개선하기 위해 1995년 Micali가 제안한 방법으로 취소된 확인서의 목록을 만들어 공개하는 대신 모든 확인서에 대해서 인증기관에서 취소 여부를 알려주는 방법[4,5]이다. 즉, 인증기관에서는 모든 확인서에 대해 유효성 여부를 사용자가 확인할 수 있도록 관련 정보를 공개 디렉토리에 전송하고 사용자는 원하는 확인서에 대해서 이 정보를 이용하여 취소 여부를 검증한다. 여기서는 설명의 편의를 위해 매일 한번씩 정보를 전송하는 것으로 가정한다.

인증기관, 디렉토리 및 사용자로 나누어서 각각의 동작을 살펴보면 다음과 같다.

• 인증기관

사용자의 공개키 확인서를 발급할 때 사용자의 공개키, 사용자 이름, 날짜 및 유효기간 등의 X.509에 규정되어 있는 기존의 요소들 외에 100비트의 길이를 가지면서 확인서의 유효를 나타내는 Y 와 취소를 나타내는 N 을 포함시킨다. 여기서 확인서에 미리 발급되어지는 Y, N 값을 "0-Token"이라 한다. Y 는 먼저 비밀 값인 Y_0 를 생성하고 일방향 함수 F 를 365번 적용하여 $Y = Y_{365} = F^{365}(Y_0)$ 와 같이 구하고 N 은 $N = F(N_0)$ 로 구한다. 일방향 함수의 예로서는 해쉬함수 등을 사용할 수 있으며 인증기관은 공개 저장소인 디렉토리에 다음과 같은 정보를 전송한다.

- 1) 유효한 확인서와 발급된 확인서의 갱신된 리스트
- 2) 각각의 확인서에 대해서 유효한 경우에는 그 확인서가 발급된 날로부터 i 번째 날 일 경우 $Y_{365-i} = F^{365-i}(Y_0)$ 를 전송하고 확인서가 취소되었을 경우에는 N_0 를 전송한다.

Y_{365-i} 는 Y_0 에 F 를 365- i 번 적용하면 쉽게 구할 수 있다.

- 3) 그 날에 새롭게 발급된 확인서

• 디렉토리

인증기관으로부터 수신한 정보를 확인한다. 예를 들어 모든 확인서에 대해서 수신한 V 값에 대해 $F(V) = Y_{i-1}$ 를 만족하는 지를 확인한다. Y_{i-1} 은 전날의 확인서 취소 여부를 확인하는 값이다. 또는 $F(V) = N$ 인지를 확인한다.

만약 사용자가 디렉토리에 발급된 날로부터 i 번째 날에 어떤 확인서의 현재 상태를 질문하

면 디렉토리는 그 확인서와 관련된 100 비트의 값을 보내준다. 즉, 확인서의 상태에 따라 Y_{365-i} 나 N_0 을 전송한다.

• 사용자

사용자가 어떤 확인서에 대해서 유효한지를 알기 위해서 디렉토리에 그 확인서의 상태를 질의하게 되면 디렉토리는 이에 대한 정보를 전송한다. i 를 현재의 날짜, Y 를 확인서가 유효한 경우의 값, N 을 그렇지 못한 경우라 하자. Y 와 N 은 공개키 확인서내에 포함되어 있다. 사용자는 디렉토리로부터 관심있는 확인서의 상태에 대한 100 비트의 값, V 를 수신한 후 $F^i(V)=Y$ 를 만족하면 대상이 되는 확인서는 정당한 상태이며 $F(V)=N$ 이면 그 확인서는 취소된 것으로 간주한다. 두 경우가 모두 아닐 경우에는 디렉토리가 사용자에게 요구하는 확인서에 대한 증명값을 보내주지 못한 것이며 사용자는 디렉토리가 서비스를 거부한 것으로 간주한다.

이 방법은 일방향 함수의 역을 구할 수 없다는 것에 안전도를 기초하고 있으며 사용자가 디렉토리로부터 원하는 확인서에 대한 정보만 수신하면 된다. 인증기관에서 디렉토리로 매일 리스트와 "0-Token"을 전송해야 하므로 전체 $T \times n \times (l+100)$ 비트의 통신량을 가지며 디렉토리에서는 매일 사용자들에게 평균적으로 $q \times 100$ 비트를 전송한다.

2.4. Certificate Revocation Tree(CRT)

1900년 Kocher가 CRL의 비효율성을 보완한 방법[6]으로 인증기관에서는 확인서들의 상태를 표현하는 서술문을 해쉬하여 트리를 구성하고 디렉토리에 전송하며, 디렉토리는 트리의 일부만을 사용자에게 전송하여 사용자가 원하는 확인서의 유효성을 입증할 수 있게 함으로써 통신량을 줄일 수 있다.

인증기관에서의 트리 발행과 사용자의 확인서 유효성을 검증하는 두 단계로 나누어 살펴보면 다음과 같다.

CRT 발행

인증기관은 확인서의 상태를 표현하는 서술문의 생성한다. 모든 확인서는 인증기관의 공개키와 확인서의 일련번호로써 유일하게 식별할 수 있으며 각각의 공개키 해쉬값의 크기가 $CA_1 < CA_2 < CA_3$ 를 만족하는 세 인증기관에 확인서들이 속한다고 하자. 또한 CA_1 은 세 개의 취소된 확인서 {156, 343, 344}를 가지며, CA_2 는 확인서 모두가 유효하고, CA_3 는 한 개 취소된 확인서 {987}를 가지고 있다. 이 경우 인증기관은 어떤 확인서 X 에 대해 다음과 같은 서술문을 발행할 수 있다.

$If : -infinity < CA_x < CA_1$	Then: Unknown CA(revocation status unknown).
$If : CA_x = CA_1$ and $-infinity \leq X < 156$	Then: X is revoked if and only if..
$X = -infinity$	
$If : CA_x = CA_1$ and $156 \leq X < 343$	Then: X is revoked if and only if $X = 156$.
$If : CA_x = CA_1$ and $343 \leq X < 344$	Then: X is revoked if and only if $X = 343$.
$If : CA_x = CA_1$ and $344 \leq X < infinity$	Then: X is revoked if and only if $X = 344$.
$If : CA_1 < CA_x < CA_2$	Then: Unknown CA(revocation status unknown).
$If : CA_x = CA_2$ and	Then: X is revoked if and only if
$X = -infinity$	
$-infinity \leq X < infinity$	
$If : CA_2 < CA_x < CA_3$	Then: Unknown CA(revocation status unknown).
$If : CA_x = CA_3$ and $-infinity \leq X < 987$	Then: X is revoked if and only if
$X = -infinity$.	
$If : CA_x = CA_3$ and $987 \leq X < infinity$	Then: X is revoked if and only if $X = 987$.
$If : CA_3 < CA_x < infinity$	Then: Unknown CA(revocation status unknown).

또한 인증기관은 위의 각 문장들을 최종노드로 하여 그림 1 과 같이 이진 해쉬 트리를 구한다. 그림에서 $N_{0,0}, N_{0,1}, N_{0,2}, \wedge N_{0,10}$ 는 첫번째부터 열번째까지의 문장을 각각 해쉬한 값이며, 이들을 차례로 두개씩 연결시켜 상위 레벨의 노드들을 만들어내면서 루트 노드까지 구성한다. 즉, $N_{2,1} = H(N_{1,2} | N_{1,3})$ 로 계산이 되며, “|”표기는 연결(concatenation)을 나타낸다. 전체 트리를 구성한 후 무결성을 제공하기 위해 트리의 루트 노드의 값을 서명하여 트리와 함께 디렉토리에 전달한다.

확인서 증명 과정

확인서의 상태를 확인하기 위해 사용자는 적절한 서술문과 이를 증명할 데이터가 필요하다. 예를 들어 CA_1 에 속한 일련번호 600 번의 확인서 상태를 디렉토리에 질의 했을 경우, 적절한 서술문은 다음과 같다.

$If : CA_x = CA_1$ and $344 \leq X < infinity$ Then: X is revoked if and only if $X = 344$.

위의 문장은 CA_1 에 속한 일련번호 600 번의 확인서가 유효함을 나타낸다. 사용자는 디렉토

리로부터 받은 위 문장을 해쉬하여 $N_{0,5}$ 값을 만들 수 있으며, 그 무결성을 증명하기 위해 지원 노드들과 함께 루트 노드를 계산해 서명된 루트 노드값과 비교하여 일치여부를 판단한다. 여기서, 지원 노드란 $N_{0,5}$ 값과 함께 루트 노드를 계산해낼 수 있는 최소한의 노드들의 집합이다. 그림 1 에서 지원노드는 $N_{0,4}, N_{1,3}, N_{2,0}, N_{3,1}$ 이다. 결과적으로, 디렉토리는 사용자의 질의에 답하여 적절한 서술문과 그에 해당하는 지원노드, 서명된 루트 노드값을 전송하면 된다.

서명의 길이를 l_{sig} , 해쉬함수 출력의 길이를 l_{hash} 라 하자. 취소 리스트의 변화가 발생할 경우 변화 리스트와 변화된 CRT 의 루트 노드의 서명 값만 전송하면 디렉토리에서 CRT 를 재구성하고 서명을 검증할 수 있으므로 인증기관에서 디렉토리로 전송되는 양은 취소 리스트의 변화분과 갱신된 서명 값인 $\frac{n \times p \times l}{365} + T \times l_{sig}$ 비트이다. 디렉토리에서 사용자들에게 전송되는 비트는 $q \times l_{hash} \times \log_2(p \times k)$ 이다.

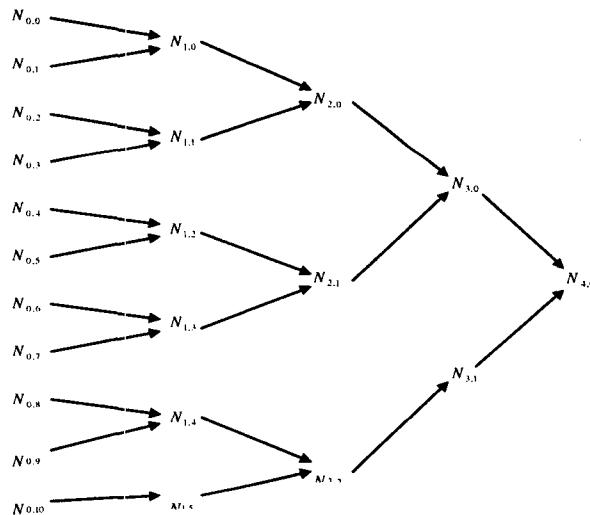


Fig 1. Certificate Revocation Tree

2.5. Certificate Revocation Method by Noar(CRM)

CRT 와 비슷한 방법으로 취소 리스트를 일련번호 순으로 나열해 확인서의 취소 여부를 검색하고 그 무결성을 증명하기 위해 각각의 취소 확인서 일련번호의 해쉬값을 최종 노드로 하는 트리구조를 사용한다[7]. 인증기관, 디렉토리 및 사용자로 구분하여 살펴보면 다음과 같다.

- 인증기관

시스템 초기화시 인증기관은 취소된 확인서들을 일련번호 순으로 나열하여 이들을 최종 노드로 한 후 각 최종 노드들을 두개 혹은 세개씩 연결하고 이를 해쉬하여 상위 노드를 생성한다. 이를 루트 노드까지 반복하여 2-3 트리를 생성한다. 인증기관은 2-3 트리의 모든 노드 값들을 저장하고 루트 노드 값, 트리의 깊이, 타임 스탬프 등을 서명하여 순서대로 나열된 취소 리스트와 함께 디렉토리에 전송한다. 갱신시 취소 리스트가 변화되었을 경우 인증기관은 새로운 트리를 생성하여 루트 노드를 생성하고 트리 깊이, 타임 스탬프 등과 함께 서명한 값을 구한다. 디렉토리로는 취소 리스트의 변화분과 갱신된 서명 값 만을 전송한다.

●디렉토리

취소 확인서 리스트를 받은 후 직접 2-3 트리를 생성하여 인증기관으로부터 수신한 서명값과 비교해 검증한다. 리스트가 갱신될 경우에는 인증기관이 보내온 취소 리스트의 변화분에 따라 트리를 재구성하고 그에 영향받는 노드 값들을 다시 계산하여 인증기관이 서명하여 보낸 값과 일치하는 지를 검증한다.

사용자가 특정 확인서의 유효성 여부를 질의 할 경우 디렉토리는 루트 노드 값, 트리 깊이, 타임 스탬프의 서명 값과 함께 다음을 전송한다. 먼저, 대상이 되는 확인서가 취소되었을 경우 그 확인서와 관련된 최종 노드와 루트 노드 까지 경로상의 노드 값과 그 노드들의 자식 노드 값을 사용자에게 전송하여 취소사실의 무결성을 증명하도록 한다. 확인서가 유효할 경우에는 질문 받은 확인서의 일련번호를 사이에 둔 두 연속한 취소된 일련번호들과 루트 노드까지의 각각의 경로상에 있는 노드 값들을 보낸다. 통신량을 줄이기 위해 사용자가 계산할 수 있는 노드 값들은 제외하고 전송할 수도 있다.

●사 용 자

사용자는 먼저 확인서의 만기일을 확인한 후 디렉토리에 질의한 일련 번호로써 질의한다. 디렉토리로부터 취소 확인서로 응답을 받을 경우 취소 확인서 일련번호와 디렉토리에서 수신한 노드 값들로부터 루트 노드 값을 계산해 인증기관이 서명한 값과 비교하여 검증한다. 유효한 확인서의 경우 디렉토리는 질의한 확인서의 일련번호를 사이에 둔 두 연속한 취소 확인서 일련번호들과 관련 노드들을 전송하여 사용자가 질의한 확인서가 취소리스트에 포함되어 있지 않음을 보여주며 사용자는 루트 노드 값을 계산하여 이를 검증할 수 있다.

인증기관에서 디렉토리로 전송하는 통신량은 취소 리스트의 변화분과 루트 노드, 트리 깊

이 및 타임 스탬프를 서명한 값을 보내면 되므로 $\frac{n \times p \times l}{365} + T \times l_{sig}$ 비트이다. 디렉토리에

서 사용자들에게 하루에 평균적으로 $2 \times q \times l_{hash} \times \log_2(p \times k)$ 비트이다. 이는 확인서가 유효한 경우에 2 개의 일련번호에 대한 노드 값을 보내야 하는 것에 기인한다.

2.6. Fast Digital Identity Revocation(FDIR)

CRYPTO '98 에서 Aiello 등이 CRS 에서의 인증기관과 디렉토리간의 부하를 줄이기 위해 제안한 것[8]으로 CRS 와 같이 모든 확인서에 대해 그날의 해쉬값을 제공하지 않고 확인서들간에 부분적으로 교차하는 여러 "0-Token" 값을 확인서에 포함시키고, 하나의 "0-Token" 값으로 여러 확인서의 유효성을 표현할 수 있게 함으로써 통신량을 줄인다. "0-Token"을 계층적으로 분배하는 방법으로 기본 개념을 살펴본다.

$N = 2^l$ 는 사용자 수를 나타내고 v 는 l 비트의 ID 를 나타낸다고 하자. 8 명의 사용자를 가정하면 $l=3$ 이 되며 그림 2 와 같이 계층적인 구조로 나타낼 수 있다.

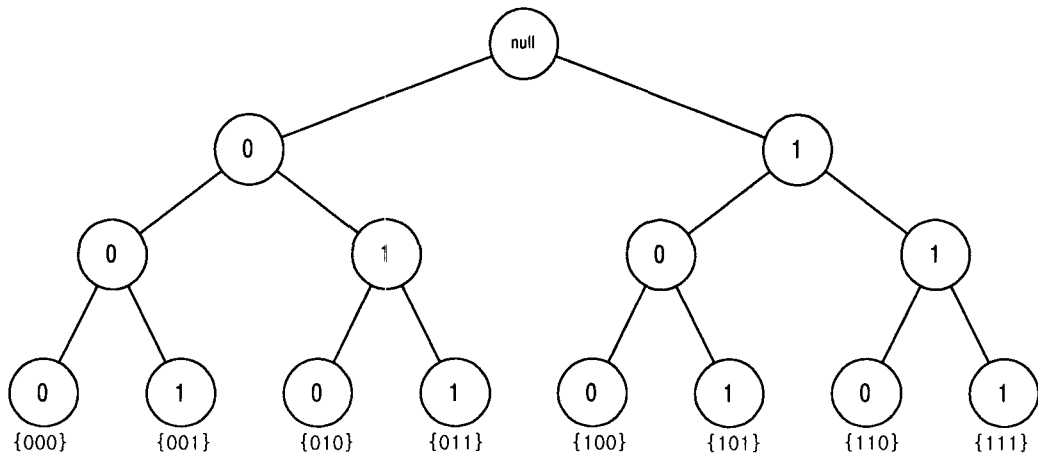


Fig. 2. Fast Digital Identity Revocation

그림 2 에서 루트 노드는 널(null)로 표시하고, 자손 노드들은 [0,1]로 표현하여 최하위 노드까지 l 비트 스트링으로 나타내며 최하위 노드인 [000, 001, ..., 111]는 확인서의 ID 를 의미한다. 즉, 상위 노드 s 에 대해 그 자손 노드는 [$s0, s1$]와 같이 나타내며 실제로 각 노드들은 " i -Token"의 값을 의미한다.

● 인증기관의 확인서 발급

인증기관은 사용자의 ID 경로에 따라 최하위 노드에서부터 큰 노드까지의 "0-Token"을 확인서 내에 포함시켜 발행한다. 그러므로, 확인서는 $\log_2 N + 1$ 개의 "0-Token"을 포함한다. 예를 들어 그림 2 에서 ID "010"인 사용자의 확인서는 유효를 나타내는 루트 노드 및 {0, 01, 010}노드의 "0-Token"과 취소를 나타내는 "0-Token"인 N 을 포함 한다. 취소를 나타낼 경우는 CRS 와 동일한 방법을 사용한다.

● 확인서 검증 노드 생성

R_i 를 i 번째 날에 취소된 확인서의 ID 들의 집합이라 하고 확인서를 유효 여부를 표현할 노드들의 부분 집합을 i 번째 날의 검증 노드라 하며 다음의 두 성질을 만족한다.

1) R_i 에 포함되지 않은 확인서 v 의 경로상에 존재하는 노드들 중 최소한 한 개 이상이 i 번째 날의 검증 노드에 포함되어야 한다.

2) R_i 에 포함된 확인서 r 의 경로상에 존재하는 어떤 노드도 i 번째 날의 검증 노드에 포함되어서는 안 된다.

인증기관은 i 번째 날의 검증 노드에 포함된 각 노드에 대해 " i -Token"을 계산할 수 있으며, 모든 " i -Token"의 집합을 i 번째 날의 검증 토큰으로 하여 디렉토리에 전송한다. 예를 들어 그림 2에서 만약 i 번째 날에 {000}과 {100}의 두 확인서가 취소 되었을 경우 i 번째 날의 검증노드는 {001, 01, 101, 11}이 되며 이 노드들에 대한 " i -Token" 값과 취소된 확인서 {000}과 {100}에 대한 N_0 값을 디렉토리에 전송한다.

• 사용자의 질문에 따른 디렉토리의 응답

i 번째 날의 확인서 u 에 대한 질의에 대해 u 가 취소되지 않았다면, 인증기관이 발행한 i 번째 날의 검증 토큰들 중 u 경로의 노드가 최소한 하나 이상이 포함되어 있어야 한다. 그러면, 디렉토리는 그 중 하나의 토큰을 질의에 대한 응답으로 사용자에게 전송한다.

만약, u 가 취소된 확인서이면 i 번째 날의 토큰들 중 u 의 경로에 포함된 토큰은 존재하지 않을 것이며, 디렉토리는 u 에 관련된 " i -Token"을 발행할 수 없다. 따라서 디렉토리는 CRS와 마찬가지로 취소된 확인서에 대한 N_0 을 전송한다.

• 사용자의 동작

디렉토리가 확인서 u 에 대한 사용자의 질의에 대해 유효하다는 응답을 할 경우, 사용자는 확인서 u 의 경로에 속한 하나의 " i -Token"을 디렉토리로부터 받는다. 사용자는 그 값의 무결성을 확인하기 위해 CRS와 같이 i 번 해쉬하여 확인서내의 " 0 -Token" 값과 동일한지 비교한다. 이에 반해 질의한 확인서가 취소된 경우, 디렉토리는 N_0 값을 보내오며, 사용자는 그 값을 한 번 해쉬하여 취소를 의미하는 " 0 -Token"인 N 과 동일한지 비교한다.

이 방법에서 인증기관에서 디렉토리로 발생하는 통신량을 살펴보면 그림 2와 같은 계층구조에서 취소 확인서의 수가 $r = 2^l$ 이고 이들이 트리의 최하위 노드에 균일하게 분포한 경우에 가장 많은 검증 토큰을 필요로 한다. 이 때 취소 확인서를 하나만 포함하는 서브 트리를 구성할 수 있고 그 깊이는 $l-t$ 가 된다. 각 서브 트리에서 유효한 확인서에 대해서 발급해야 하는 " i -Token"은 $l-t$ 이고 취소된 확인서에 대해서는 하나의 N_0 가 필요하므로 전체 $l-t+1$ 개

의 토큰이 필요하다. 서브 트리에는 하나의 취소 확인서만 존재하므로 취소 확인서의 수 만큼 서브 트리가 존재한다.그러므로 총 발급해야 하는 토큰 수는

$$r \times (l - t + 1) = n \times p \times \log_2 \left(\frac{1}{p} + 1 \right) \text{ 이고 "i-Token"의 길이가 CRS와 동일하게 100 비트}$$

라 하면 $n \times p \times \log_2 \left(\frac{1}{p} + 1 \right) \times (l + 100)$ 비트를 전송해야 한다. 디렉토리에서 사용자들에게 전송해야 하는 양은 하루에 q 번 질의 하므로 $q \times 100$ 비트이다.

3. 비교분석 및 고찰

앞 절에서 언급한 시스템들에서 가장 중요한 사항은 인증기관과 디렉토리 사이와 디렉토리 와 사용자 간의 통신량이다. 이 중에서도 특히, 디렉토리와 사용자간의 통신량이 시스템의 부하 와 성능에 영향을 미친다. 각 방식에 대해서 요구되는 통신량을 표 1 에 나타내었으며 이를 실제 의 값을 대입하여 구한 통신량을 표 2 에 나타내었다.

표 2 에서 결과에서 디렉토리에서 사용자들에게 전송되는 통신량이 가장 적은 것은 CRS 와 FDIR 이다. 이 들은 디렉토리에서 사용자가 원하는 확인서에 대한 100 비트의 값만 수신하면 되 므로 매우 효율적이다. 인증기관에서 디렉토리로 전송되는 비트들은 표 2 에서는 CRT 와 CRM 이 적다. 이 들은 확인서 취소 리스트에서 변경된 부분과 루트 노드의 서명 값만 전송하고 디렉 토리에서 트리를 다시 구성하므로 취소 리스트 갱신시에 효율적인 방법이다. 전체적으로 발생 하는 통신량을 보면 FDIR 이 가장 적은 통신량을 발생시킨다.

Table 1. Communication amount between CA and Directory

	인증기관→디렉토리 (bits/day)	디렉토리→사용자 (bits/day)
CRL	$T \times n \times p \times l$	$q \times p \times k \times l$
CRS	$T \times n \times (l + 100)$	$q \times 100$
CRT	$\frac{n \times p \times l}{365} + T \times l_{sig}$	$q \times l_{hash} \times \log_2(p \times k)$
CRM	$\frac{n \times p \times l}{365} + T \times l_{sig}$	$2 \times q \times l_{hash} \times \log_2(p \times k)$
FDIR	$n \times p \times \left(\log_2 \frac{1}{p} + 1 \right) \times (l + l_{char})$	$q \times 100$

인증기관, 디렉토리 및 사용자들이 필요로 하는 계산량을 비교할 수 있으나 언급한 모든 방 식들은 해쉬 함수와 한번의 서명 정도를 사용하므로 계산량에는 큰 차이가 없다. 또한 취소된 확인서에 대한 명확한 검증을 제공하는 지를 살펴보면 CRL 을 제외하고는 모두 제공된다. CRL

의 경우에는 사용자가 원하는 확인서가 확인서 취소 목록에 수록되어 있지 않다고 해서 그 확인서가 유효하는 것을 명확하게 제시하지 못하기 때문에 확인서 취소 여부에 대한 직접적인 증명을 제공하지 못한다.

Table 2. Practical communication amount ($n=3000000$, $k=30000$, $p=0.1$, $q=3000000$, $T=1$, $l=20$ bits, $l_{sig}=1000$ bits, $l_{hash}=128$ bits)

	인증기관에서 디렉토리로의 통신량	디렉토리에서 사용자들로 의 통신량
CRL	6×10^6	1.8×10^{11}
CRS	3.6×10^8	3×10^8
CRT	1.7×10^4	3.5×10^9
CRM	1.7×10^4	7×10^9
FDIR	1.6×10^8	3×10^8

표 2에 의하면 FDIR이 가장 적은 통신량을 가진다. 하지만 이는 하루에 한 번 확인서 취소 리스트를 갱신하는 경우이다. 그러나 좀 더 안전한 서비스를 위해 1시간에 한번씩 서비스를 제공할 경우에는 CRT 방법이 유리하며 갱신횟수가 많아짐에 따라 CRS와 FDIR은 해쉬해야 하는 횟수가 늘어나는 단점이 있다.

4. 결론

본 논문에서는 공개키 확인서의 취소 여부를 확인하는 방법들을 살펴보고 이들을 발생하는 통신량에 따라 비교 분석해 보았다. 공개키의 불법 사용을 막기 위해서는 공개키 확인서의 취소 여부를 반드시 확인해야 하며 이를 위해 CRL을 주로 사용하나 이의 많은 통신량으로 인해 몇 가지 대안이 제시되었다. 이들을 분석해 본 결과 취소 리스트의 갱신 횟수에 따라 달라지나 하루에 한 번 갱신할 경우에는 FDIR 방법이 가장 효율적이다. 그러나 보호의 정도를 높여 한 시간에 한 번 갱신할 경우에는 CRT의 방법이 효율적이며 임계치는 하루에 21회 정도이다.

공개키 암호 시스템의 사용을 위해서는 공개키 확인서의 도입은 필수적이며 이의 유효를 확인하는 방법 또한 제공되어야 한다. 통신 비용이 높은 현실에서 확인서의 유효성을 확인하는 효율적인 방법의 도입이 있어야 하겠다.

참고 문헌

- [1] NIST, *Federal Public Key Infrastructure Technical Specifications : Part A-Technical Concept of Operations*, TWG-98-59, 1998. 9.
- [2] Standards Australia, *Strategies for the Implementation of a Public Key Authentication Framework(PKAF) in Australia*, SAA MP75-1996.
- [3] IETF, *Online Certificate State Protocol*, RFC

- [4] S. Micali, "Efficient Certificate Revocation," Technical Memo MIT/LCS/TM-542b, 1996.
- [5] S. Micali, "Certificate Revocation System," U.S. Patent 5666416
- [6] P. Kocher, "A Quick Introduction to Certificate Revocation Trees(CRTs)," available at <http://www.valicert.com>
- [7] M. Naor and K. Nissim, "Certificate Revocation and Certificate Update," Proceedings of USENIX '98
- [8] W. Aiello, S. Lodha, and R. Ostrovsky, "Fast Digital Identity Revocation," Advances in Cryptology-CRYPTO '98, pp.137-152, 1998.