通信情報保護學會誌
第9券 第2號, 1999. 06

공개키 확인서 취소 방식에 관한 연구 동향

오 중효*, 박 기철**, 이 국희**, 문 상재**

요 약

공개키 암호 알고리듬은 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 살펴보고 이들을 발생되는 통신량 측면에서 비교 분석한다.

Ⅰ. 서 론

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 이 중 공개키 암호 시스템은 효율적인 인증 및 디지털 서명의 제공으로 널리 사용되고 있다.

한편, 공개키 암호시스템은 공개된 사용자의 공개키가 그 소유자와 대응되는 지를 확신할 수 있어야 하며 그렇지 못할 경우 자신의 공개키를 다른 사람의 공개키로 위조할 수 있다. 이를 해결하기 위해 신뢰할 수 있는 제3자인 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 취소해야 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 의뢰[1]를 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[2,3]. 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 여러 공개키 확인서 취소 방식이 제안되었는데 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 고에서는 지금까지 제안된 공개키 확인서의 취소 여부를 판별하는 방법을 살펴보고 전체적인 통신량의 발생 정도를 비교 분석하여 각 방식의 효율성을 알아본다.

Ⅱ. 공개키 확인서 취소 방식

1. 기본 가정

본 고에서는 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 디렉토리간의 통신량을 생각할 때 어느 특정 디렉토리를 대상으로 하는 것이 아니고 전체 디렉토리에 대해서 네트워크 상에 발생되는 통신량을 고려한

* 금융결제원 전자금융연구소 기술연구실(jhoh@kftc.or.kr)
** 경북대학교 전자전기공학부({maginga, lkh}@palgong.kyungpook.ac.kr, sjmoon@ee.kyungpook.ac.kr)

다. 즉, 디렉토리가 전체에 하나가 있다고 가정하는 것과 동일하다. 발생되는 통신량을 분석하기 위해 표 1과 같은 표기를 사용하며 1일을 기준으로 통신량을 나타낸다. 일반적으로 확인서가 취소되는 비율은 10%정도로 한다[4].

표 2. 표기

n	전체 총 확인서의 수
k	한 인증기관에 속한 평균 확인서 수
p	취소되는 확인서의 비율
r	취소된 확인서의 수($r=n\times p$)
T	취소 리스트의 1일 갱신 횟수
l	확인서의 일련번호를 나타내는 비트 수
q	사용자들이 인증기관에 확인서의 유효 여부를 질의하는 횟수(1일)

2. 확인서 취소 목록 방식

공개키 확인서의 취소 여부를 알 수 있는 가장 일반적인 방법이며 현재까지 표준화 중인 여러 공개키 기반 구조에서 이 방식을 채택하고 있다[2,3]. 인증기관이 각 사용자의 공개키 확인서를 발급하며 발급된 확인서를 취소해야 할 요건이 발생할 경우 그 확인서의 일련번호를 확인서 취소 목록에 추가한다. 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 커지는 단점이 있으며 목록에서 대상이 되는 확인서가 존재하는지 확인해야 하므로 굉장히 많은 부하가 걸린다. 이를 개선하기 위해 전체 CRL을 다루지 않고 최근 CRL이 발행된 시점에서 새로운 CRL이 발행된 시점까지의 목록을 모아둔 delta-CRL을 사용하나 이 또한 이전의 목록을 저장하고 있어야 한다.

요구되는 통신량을 계산해 보면 인증기관과 디렉토리사이에는 하루에 취소된 확인서에 대해서 갱신 횟수 만큼 목록을 전송해야 하므로 $T\times r\times l = T\times n\times p\times l$ 만큼의 비트를 전송해야 하며 디렉토리가 1일 동안 사용자들에게 전송해야 하는 양은 $q\times p\times k\times l$ 비트이다.

3. Certificate Revocation System(CRS)

CRL 방식에서 취소 목록의 전송량이 늘어나는 단점을 개선하기 위해 1995년 Micali가 제안한 방법으로 취소된 확인서의 목록을 만들어 공개하는 대신 모든 확인서에 대해 취소 여부에 관한 정보를 인증기관에서 디렉토리에 공개하는 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.

1)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 Y와 취소를 나타내는 N을 포함시켜 발급한다. 미리 발급되어 지는 Y, N 값을 *0-Token*이라 한다. 이들을 생성하기 위해 인증기관은 먼저 비밀 랜덤수 Y_0와 N_0를 생성하고 일방향 함수 F를 365번 적용하여 $Y = Y_{365} = F^{365}(Y_0)$를 구하고 N은 $N=F(N_0)$와 같이 구한다. F는 일방향 함수이므로 Y로부터 Y_0를 구하는 것은 불가능하다. 일방향 함수로는 해쉬함수 등을 사용할 수 있으며 인증기관은 공개 저장소인 디렉토리에 다음과 같은 정보를 전송한다.

- 유효한 확인서와 발급된 확인서 리스트
- 각각의 확인서에 대해 확인서가 유효하고 그 확인서가 발급된 날로부터 i번째 날이면 $Y_{365-i} = F^{365-i}(Y_0)$를 전송하고 확인서가 취소된 경우에는 N_0를 전송한다. Y_{365-i}는 Y_0에 F를 365-i 번 적용하면 쉽게 구할 수 있다. i 번째 날의 토큰을 *i-Token*이라 한다.
- 그 날에 새롭게 발급된 확인서

인증기관 외에 악의를 가진 개체가 *i-Token*인 Y_{365-i}로부터 다음날의 *Token*인 $Y_{365-i-1}$을 구하는 것은 일방향 함수의 역함수를 구하는 것으로 불가능하다.

2)디렉토리

인증기관으로부터 수신한 정보를 확인한다. 예를 들어 각각의 확인서에 대해 수신한 값을 V라 하면

이에 대해 $F(V)=Y_{i-1}$이 만족하는 지를 확인한다. Y_{i-1}은 전날의 확인서 유효성을 나타내는 값이다. 취소된 확인서일 경우 $F(V)=N$ 를 확인한다.

만약 사용자가 발급된 지 i 번째 날이 되는 확인서의 취소 여부를 디렉토리에 질의하면 디렉토리는 그 확인서와 관련된 100비트의 값을 사용자에게 보내준다. 즉, 확인서의 상태에 따라 Y_{365-i}나 N_0을 전송한다.

3)사용자

사용자가 어떤 확인서에 대해서 취소 여부를 알기 위해 디렉토리에 그 확인서의 상태를 질의하게 되면 디렉토리는 이에 대한 정보를 전송한다. i를 확인서가 발급된 후의 경과 일자, Y를 확인서가 유효한 경우의 값, N을 그렇지 못한 경우라 하자. Y와 N은 공개키 확인서 내에 포함되어 있다. 사용자는 디렉토리로부터 관심 있는 확인서의 상태에 대한 100비트의 값 V를 수신한 후 $F^i(V)=Y$를 만족하면 대상이 되는 확인서는 취소되지 않은 상태이며, $F(V)=N$이면 그 확인서는 취소된 것으로 간주한다. 두 경우가 모두 아닐 경우에는 디렉토리가 요구하는 확인서에 대한 검증값을 사용자에게 보내 주지 못한 것이며 사용자는 디렉토리가 서비스를 거부한 것으로 간주한다.

4) 통신량

이 방식은 일방향 함수의 역을 구할 수 없다는 것에 안전도를 기초하고 있으며 CRL과 달리 사용자가 디렉토리로부터 원하는 확인서에 대한 정보만 수신하면 된다. 인증기관에서 디렉토리로 매일 리스트와 *Token* 값을 전송해야 하므로 전체 $T\times n\times(l+100)$ 비트를 전송해야 하며 디렉토리는 매일 사용자에게 질의할 때마다 100비트를 전송해야 하므로 평균적으로 $q\times 100$ 비트를 디렉토리에서 사용자에게 전송한다.

4. Certificate Revocation Tree(CRT)

Kocher가 CRL의 비효율성을 보완한 방법으로 미국의 Valicert사에서 제품으로 내놓은 방식이다[6]. 등록된 확인서들의 상태 정보를 이진 트리를 사용하여 효율적으로 관리한다. 인증기관에서는 각각의 확인서의 상태를 표현하는 서술문의 해쉬값을 구하고 이를 이용하여 트리를 구성한다. 구성된 트리는 인증기관이 디렉토리에 전송하며 사용자가 디렉토리에 확인서의 상태에 대한 질의를 하면 디렉토리는 트리의 일부만을 사용자에게 전송하여 확인서의 취소 여부를 검증할 수 있게 하는 방식이다.

인증기관에서의 트리 발행과 사용자의 확인서 취소 여부를 검증하는 두 단계로 나누어 동작을 살펴보면 다음과 같다.

1) CRT 발행

인증기관은 먼저 모든 확인서의 상태를 나타내는 그림 1과 같은 서술문을 생성한다. 예를 들어 3개의 인증기관이 존재하며 모든 확인서는 인증기관의 공개키와 확인서의 일련번호로써 유일하게 식별할 수 있으며 세 인증기관은 각각의 공개키를 해쉬한 값의

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

그림 1. 확인서 취소 트리 방식의 서술문

크기가 $CA_1 < CA_2 < CA_3$ 를 만족한다고 가정하자. 또한 CA_1은 3개의 취소된 확인서를 가지며 그 일련번호가 156, 343, 344라 하고 CA_2는 소속된 모든 확인서가 유효하며 CA_3는 한 개의 취소된 확인서를 가지며 그 일련번호가 987이라 하자. 이 경우 인증기관은 어떤 확인서 X에 대해 그림 1과 같은 서술문을 발행하게 된다.

또한 인증기관은 그림 1의 각 문장들을 해쉬하고 이 값들을 이용하여 그림 2와 같이 이진 해쉬 트리를 구성한다.

그림 2에서 $N_{0,0}, N_{0,1}, N_{0,2}, \cdots, N_{0,10}$는 첫 번째부터 열한번째까지의 문장을 각각 해쉬한 값을 의미하며 이들을 차례로 두개씩 연접한 후 다시 해쉬한 값을 상위 레벨의 노드 값으로 한다. 이 과정을 반복하여 루트 노드까지 이진 해쉬 트리를 구성한다. 즉, $N_{2,1} = H(N_{1,2} \| N_{1,3})$와 같이 상위 노드가 계산되며 ‖는 연접(concatenation)을, H는 해쉬함수를 나타낸다.

전체 이진 해쉬 트리를 구성한 후에 무결성을 제공하기 위해 루트 노드 값을 인증기관이 서명하여 트리와 함께 디렉토리에 전달한다. 인증기관이 트리를 처음 생성할 경우에는 디렉토리에 전체 트리를 전송해야 하지만 시스템의 운용 중에 취소 확인서 목록이 변경되어 트리가 변경되어야 할 경우에는 변경된 목록과 새로운 루트 노드의 서명 값만 전송하여 디렉토리에서 재생성 하도록 한다. 디렉토리는 수신한 루트 노드의 서명 값을 자신이 생성한 트리의 루트 노드를 이용하여 올바르게 생성되었는지를 검증하면 된다.

2) 확인서 검증 과정

확인서의 상태를 확인하기 위해 사용자는 적절한 서술문과 이를 증명할 데이타가 필요하며 이를 디렉토리로부터 얻는다. 예를 들어 설명하면 사용자가 CA_1에 속한 일련번호 600번의 확인서 상태를 디렉토리에 질의했을 경우 해당되는 적절한 서술문은 다음과 같다.

$$If: CA_X = CA_1 \text{ and } 344 \le X < infinity$$

Then: X is revoked if and only if $X = 344$

위의 문장은 CA_1에 속한 일련번호 600번의 확인서가 유효함을 나타낸다. 사용자는 디렉토리로부터 받은 위 문장을 해쉬하여 $N_{0,5}$ 값을 구할 수 있으며 그 무결성을 검증하기 위해 인접한 지원 노드들과 함께 루트노드를 계산해 서명된 루트 노드 값과 비교하여 일치 여부를 판단하여 일치할 경우 이 확인서는 취소되지 않은 것이다. 여기서 지원노드란 $N_{0,5}$ 값과 함께 루트 노드를 계산해 낼 수 있는 최소한의 노드들의 집합이다.

그림 2에서 $N_{0,5}$의 지원노드는 $N_{0,4}$, $N_{1,3}$, $N_{2,0}$, $N_{3,1}$이다. 결과적으로 디렉토리는 사용자의 질의에 답하여 적절한 서술문과 그에 해당되는 지원노드, 서명된 루트 노드 값을 전송하면 된다.

3) 통신량

서명 값의 비트 길이를 l_{sig}, 해쉬함수 출력의 비

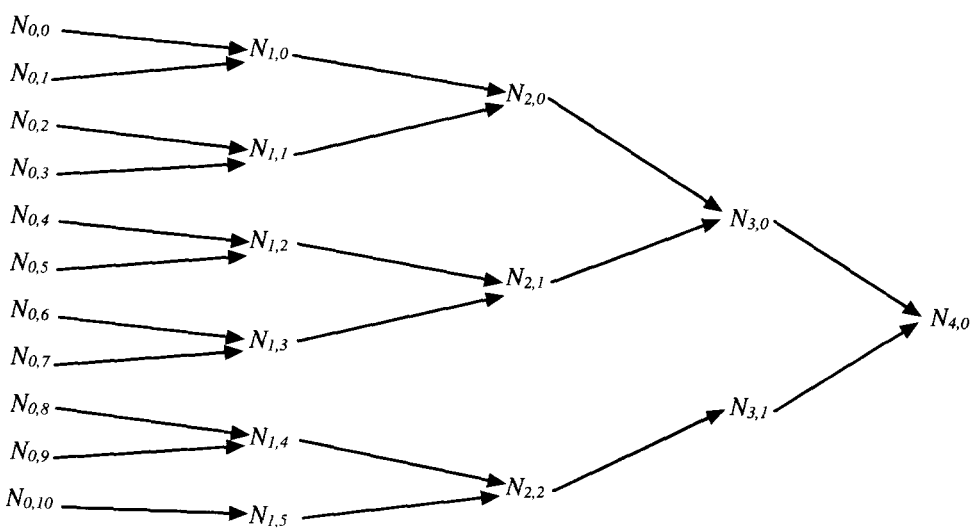


그림 2. 확인서 취소 트리

트 길이를 l_{hash}라 하고 발생되는 통신량을 살펴보면 다음과 같다. 먼저 인증기관에서 디렉토리로 전송해야 하는 정보는 처음 트리를 구성한 이후에는 취소 리스트의 변화가 발생할 경우 변화 리스트와 변화된 트리의 루트 노드의 서명 값만 전송하면 디렉토리에서 트리를 재구성하고 서명을 검증할 수 있다. 하루에 취소되는 확인서의 수는 $\frac{n\times p}{365}$이며 $\frac{n\times p\times l}{365}$비트이다. 1일 갱신횟수는 T이므로 하루에 전송해야 하는 서명 값은 $T\times l_{sig}$이다. 따라서 인증기관에서 디렉토리로 전송하는 정보는 취소리스트의 변화분과 갱신된 서명 값인 $\frac{n\times p\times l}{365}+T\times l_{sig}$비트이다.

사용자가 디렉토리에 확인서의 상태를 질의했을 경우에 디렉토리로부터 사용자에게 전송되는 정보는 질의한 확인서의 상태를 나타내는 서술문의 해쉬값, 루트 노드 값을 구하는데 필요한 노드 값과 루트 노드의 서명 값을 전송하면 된다. 따라서 디렉토리가 전송하는 비트는 $q\times(l_{hash}\times\log_2(p\times k)+l_{sig})$이다.

5. *Naor* 방식

1998년 *Naor*에 의해 제안된 방식이며 취소 확인서 리스트를 확인서의 일련번호 순으로 나열해 사용자가 원하는 확인서의 취소 여부를 검색하고 그 리스트의 무결성을 증명하기 위해 각각의 취소 확인서 일련번호를 최종 노드로 하여 CRT와 같이 트리를 구성한다. 루트 노드 값은 인증기관이 부가 정보와 함께 서명한다. CRT와 비슷한 방법이나 서술문을 사용하지 않고 취소 확인서를 일련번호 순으로 나열해 검색하는 것이 다른 점이다. 인증기관, 디렉토리 및 사용자로 구분하여 동작을 살펴본다.

1) 인증기관

시스템 초기화시 인증기관은 취소된 확인서들을 일련번호 순으로 나열하여 이들을 최종 노드로 한 후 각 최종 노드들을 두개 혹은 세개씩 연접하고 이를 해쉬하여 상위노드를 생성한다. CRT에서는 최종 노드값이 서술문의 해쉬 값이었으나 이 방식은 취소 확인서의 일련번호가 최종노드가 된다. 이를 루트 노드까지 반복하여 2-3트리[9]를 생성한다. 인증기관은 2-3트리의 모든 노드 값들을 저장하고 루트 노드 값, 트리의 깊이, 타임 스탬프 등을 서명하여 일련번호 순서대로 나열된 취소 리스트와 함께 디렉토리에 전송한다.

초기화 후에 확인서의 취소 리스트가 변화되었을 경우 인증기관은 새로운 트리를 생성하여 루트 노드를 생성하고 트리의 깊이, 타임 스탬프 등과 함께 서명한다. 디렉토리로는 취소 확인서 리스트의 변화분과 갱신된 서명 값만 전송하면 된다.

2) 디렉토리

인증기관으로부터 취소 확인서 리스트와 서명 값을 받은 후 직접 2-3트리를 생성하여 수신한 서명 값을 검증하여 수신한 정보가 맞는지를 확인한다. 취소 리스트가 갱신되었을 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 취소 여부를 질의할 경우 디렉토리는 루트 노드 값, 트리 깊이, 타임 스탬프의 서명 값과 함께 다음을 전송한다. 먼저, 대상이 되는 확인서가 취소되었을 경우에는 그 확인서와 관련된 최종 노드와 루트 노드까지의 경로에 있는 노드 값과 그 노드들의 자식 노드 값을 사용자에게 전송한다. 확인서가 유효할 경우에는 질문 받은 확인서의 일련번호를 사이에 둔 두 연속한 취소된 일련번호들과 루트 노드까지의 각각의 경로 상에 있는 노드 값들을 보낸다. 통신량을 줄이기 위해 사용자가 계산할 수 있는 노드 값들은 제외하고 전송할 수도 있다.

3) 사용자

사용자는 먼저 확인서의 만기일을 확인한 후 디렉토리에게 일련번호로써 질의한다. 질의 결과 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 취소 리스트에 포함되어 있지 않음을 보여주며 사용자는 루트 노드 값을 계산하여 이를 검증할 수 있다.

4) 통신량

인증기관에서 디렉토리로 전송하는 통신량을 살펴보면 초기화 후에는 취소 리스트의 변화분과 루트 노드, 트리 깊이 및 타임 스탬프를 서명한 값만 보내면 되므로 $\frac{n\times p\times l}{365}+T\times l_{sig}$비트를 보내면 된다.

사용자들은 하루에 디렉토리에 q번 질의하므로 최대 $q\times(2\times l_{hash}\times \log_2(p\times k)+l_{sig})$비트를 디렉토리가 사용자에게 전송한다. 이는 확인서가 유효한 경우에 2개의 일련번호에 대한 노드 값을 보내야 하는 것에 기인한다.

6. Fast Digital Identity Revocation

CRYPTO'98에 Aiello등이 CRS 방식의 인증기관과 디렉토리간의 통신량을 줄이기 위해 CRS를 개선하여 제안한 방식[8]으로 CRS와 같이 확인서의 취소 여부를 검증할 수 있는 정보를 모든 확인서에 대해 각각 디렉토리에 전송하지 않고 여러 확인서를 동시에 검증할 수 있는 하나의 *Token*을 전송하여 통신량을 줄이는 방식이다. 이는 확인서를 발급할 때 확인서들 간에 부분적으로 교차하는 여러 *Token*을 확인서에 포함시켜서 발급하므로 가능하다.

기본 동작을 *Token*을 계층적으로 분배하는 방법으로 살펴본다. 먼저 $N=2^m$은 사용자 수를 나타내고 v는 m비트의 ID를 나타낸다고 하자. 사용자가 8명인 경우를 생각하면 m=3이 되며 그림 3과 같이 계층적인 구조로 사용자의 ID를 나타낼 수 있다. 루트 노드는 null로 표기하고 자손 노드를 0과 1로 구분하여 나타낸다. 즉, 현재 노드 s에 대해 그 자손 노드는 [s0, s1]와 같이 나타내며 최하위 노드는 확인서의 ID를 의미한다. 실제로 노드들은 확인서의 유효함을 나타내는 *Token* 값들을 나타낸다.

1) 인증기관의 확인서 발급

CRS 방식은 발급시 확인서의 유효와 취소를 나타내는 100비트의 Y와 N을 각각의 확인서에 포함시켜 발급하였으나 이 방식은 확인서를 발급할 때 사용자의 ID 경로에 따라 최하위 노드에서부터 루트 노드까지의 0-*Token*들을 확인서 내에 모두 포함시켜 발행한다. 그러므로 확인서는 $\log_2 N+1$개의 0-*Token*들을 포함하고 있다. 예를 들어 그림 2에서 ID가 "010"인 사용자의 확인서는 유효를 나타내는 0-*Token*으로 {null, 0, 01, 010}노드의 값들을 가지며 취소를 나타내는 *0-Token*으로 CRS와 같이 N을 가진다.

2) 확인서 검증 노드 생성

R_i를 i 번째 날에 취소된 확인서 ID들의 집합이라고 하면 i 번째 날에 확인서의 유효를 표현할 노드들의 부분 집합을 i 번째 날의 검증노드라 하고 다음의 두 성질을 만족해야 한다.

(1) R_i에 포함되지 않은 확인서의 경로 상에 존재하는 노드들 중 최소한 한개 이상이 i 번째 날의 검증노드에 포함되어야 한다.

(2) R_i에 포함된 확인서 r의 경로 상에 존재하는 어떤 노드도 i 번째 날의 검증노드에 포함되어서는 안된다.

인증기관은 i 번째 날의 검증노드에 포함된 각 노드에 대해 CRS와 동일하게 $F^{365-i}(\,)$를 적용하여 *i-Token*을 계산할 수 있으며 모든 *i-Token*의 집합을 i 번째 날의 검증토큰으로 디렉토리에 전송한다.

그림 3을 예로 설명하면 만약 i 번째 날에 확인서 {000}와 {100}의 두 확인서가 취소되었을 경우 i 번째 날의 검증노드는 {001, 01, 101, 11}이 되며 이 노드들에 대한 *i-Token* 값과 취소된 확인서 {000}와 {100}에 대한 N_0 값을 디렉토리에 전송한다.

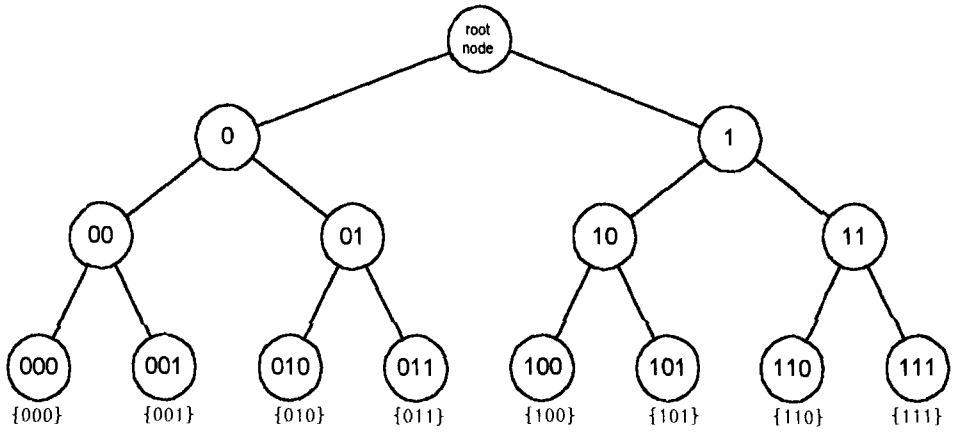


그림 3. FDIR 방식의 계층적 구조

3) 사용자의 질문에 따른 디렉토리의 응답

i 번째 날에 사용자가 확인서 u의 취소 여부를 알기 위해 디렉토리에 정보를 요구했을 경우 u가 취소되지 않았다면 인증기관이 발행한 i 번째 날의 검증노드들 중에 u의 경로가 최소한 하나 이상 포함되어 있어야 하므로 디렉토리는 그 중 하나의 *i-Token*을 사용자에게 전송한다.

만약 u가 취소된 확인서이면 i 번째 날의 검증노드에는 u의 경로가 존재하지 않을 것이며 디렉토리는 취소된 확인서를 알리는 N_0을 사용자에게 전송한다.

4) 사용자의 동작

디렉토리가 확인서 u에 대한 사용자의 질의에 대해 유효하다는 응답을 할 경우, 사용자는 확인서 u의 경로에 속한 하나의 *i-Token*을 디렉토리로부터 받는다. 사용자는 그 값의 무결성을 확인하기 위해 CRS와 같이 i 번 해쉬하여 확인서 내의 *0-Token* 값과 동일한지 비교한다. 이에 반해 질의한 확인서가 취소된 경우, 디렉토리는 N_0값을 보내오며 사용자는 그 값을 한 번 해쉬하여 취소를 의미하는 *0-Token*인 N과 동일한지 비교한다.

5) 통신량

이 방식의 통신량을 살펴보면 먼저 인증기관에서 디렉토리로 전송하는 경우에는 *i-Token*의 개수를 구해야 한다. 그림 3과 같은 계층구조에서 취소 확인서의 수가 $r=2^t$이고 이들이 트리의 최하위 노드에 균일하게 분포한 경우에 가장 많은 검증토큰을 필요로 한다. 이 때 취소 확인서를 하나만 포함하는 서브트리를 생각할 수 있고 그 갚이는 $l-t$가 된다. 각 서브트리에서 유효한 확인서에 대해서 발급해야 하는 *i-Token*은 $l-t$이고 취소된 확인서에 대해서는 하나의 N_0가 필요하므로 전체 $l-t+1$개의 토큰이 필요하다. 서브트리에는 하나의 취소 확인서만 존재하므로 취소 확인서의 수 만큼 서브트리가 존재한다. 그러므로 총 발급해야 하는 토큰 수는 $r\times(l-t+1)=n\times p\times\log_2(\frac{1}{p}+1)$이고 *i-Token*의 길이가 CRS와 동일하게 100비트라고 하면 1일 동안 $T\times n\times p\times\log_2(\frac{1}{p}+1)\times(l+100)$ 비트를 전송해야 한다.

디렉토리에서 사용자에게 전송해야 하는 양은 CRS와 동일하게 질의할 때마다 100비트의 토큰을 전송하면 되므로 $q\times 100$ 비트이다.

Ⅲ. 비교분석 및 고찰

앞 절에서 언급한 시스템들에서 가장 중요한 사항은 인증기관과 디렉토리 사이와 디렉토리와 사용자 간에 발생되는 통신량이다. 이 중에서도 특히, 디렉토리와 사용자간의 통신량이 시스템의 부하와 성능에 영향을 미친다. 각 방식에 대해서 1일에 평균적으로 발생 가능한 통신량을 표 2에 나타내었으며 이를 실제의 값을 대입하여 구한 통신량을 표 3에 나타내었다.

표 3의 결과에서 디렉토리에서 사용자들에게 전송되는 통신량이 가장 적은 것은 CRS와 FDIR이다. 이 들은 디렉토리에서 사용자가 원하는 확인서에 대한 100비트의 값만 전송하면 되므로 매우 효율적이다. 인증기관에서 디렉토리로 전송되는 비트들은 표 3에서 CRT와 *Naor* 방식이 적다. 이 들은 확인서 취소 리스트에서 변경된 부분과 루트 노드의 서명 값

표 3. 인증기관 및 디렉토리에서 발생하는 통신량

	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	$\frac{n\times p\times l}{365}+T\times l_{sig}$	$q\times(l_{hash}\times\log_2(p\times k)+l_{sig})$
Naor's	$\frac{n\times p\times l}{365}+T\times l_{sig}$	$q\times(2\times l_{hash}\times\log_2(p\times k)+l_{sig})$
FDIR	$T\times n\times p\times\log_2(\frac{1}{p}+1)\times(l+100)$	$q\times 100$

표 4. 통신량의 실례(n=3000000, k=30000, p=0.1, q=3000000, T=1, l=20bits, l_{sig}=1000bits, l_{hash}=128bits)

	인증기관→디렉토리(bits/day)	디렉토리→사용자(bits/day)
CRL	6×10^{6}	1.8×10^{11}
CRS	3.6×10^{8}	3×10^{8}
CRT	1.7×10^{4}	7.4×10^{9}
Naor's	1.7×10^{4}	1.2×10^{10}
FDIR	1.6×10^{8}	3×10^{8}

만 전송하고 디렉토리에서 트리를 다시 구성하므로 취소 리스트 갱신시에 효율적인 방법이다. 전체적으로 발생하는 통신량을 보면 FDIR이 가장 적은 통신량을 발생시킴을 알 수 있다.

인증기관, 디렉토리 및 사용자들이 필요로 하는 계산량을 비교할 수 있으나 언급한 모든 방식들은 해쉬 함수와 한번의 서명 정도를 사용하므로 계산량에는 큰 차이가 없다. 또한 취소된 확인서에 대한 명확한 검증을 제공하는 지를 살펴보면 CRL을 제외하고는 모두 제공된다. CRL의 경우에는 사용자가 원하는 확인서가 확인서 취소 목록에 수록되어 있지 않다고 해서 그 확인서가 유효한 것을 명확하게 제시하지 못하기 때문에 확인서 취소 여부에 대한 직접적인 증명을 제공하지 못한다.

표 3에 의하면 FDIR이 가장 적은 통신량을 가진다. 하지만 이는 하루에 한 번 확인서 취소 리스트를 갱신하는 경우이다. 그러나 좀 더 안전한 서비스를 위해 30분에 한번씩 서비스를 제공할 경우에는 CRT 방법이 유리하며 갱신횟수가 많아짐에 따라 CRS와 FDIR은 해쉬해야 하는 횟수가 늘어나는 단점이 있다.

Ⅳ. 결 론

본 고에서는 공개키 확인서의 취소 여부를 확인하는 방법들을 살펴보고 이들을 발생되는 통신량에 따라 비교 분석해 보았다. 공개키의 불법 사용을 막기 위해서는 공개키 확인서의 취소 여부를 반드시 확인해야 하며 이를 위해 CRL을 주로 사용하나 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 갱신의 정도를 많이 할 경우 CRT의 방법이 효율적이며 임계치는 하루에 42회 정도이다.

공개키 암호 시스템의 사용을 위해서는 공개키 확인서의 도입은 필수적이며 이의 유효를 확인하는 방법 또한 제공되어야 한다. 통신 비용이 높은 현실에서 확인서의 취소 여부를 검증하는 효율적인 방식에 관한 연구가 있어야 하겠다.

참 고 문 헌

[1] IETF, Online Certificate State Protocol

[2] NIST, Federal Public Key Infrastructure Technical Specifications : Part A-Technical Concept of Operations, TWG-98-59, 1998. 9.

[3] Standards Australia, Strategies for the Implementation of a Public Key Authentication Framework(PKAF) in Australia, SAA MP75-1996.

[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

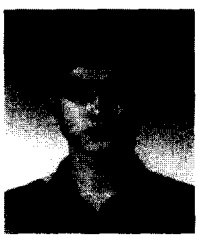
[8] W. Aiello, S. Lodha, and R. Ostrovsky, Fast Digital Identity Revocation, Advances in Cryptology-CRYPTO 98, pp. 137-152, 1998.

[9] A.V.Aho, J.E.Hopcroft, and J.D.Ullman, Data Structures and Algorithms, Addison-Wesley, 1983.

오 중 효(Joong-Hyo Oh)

1995년 2월 : 경북대학교 전자공학과 졸업
1997년 2월 : 경북대학교 전자공학과 석사
1997년 3월~현재 : 경북대학교 전자공학과 박사과정
1999년 3월~현재 : 금융결제원 전자금융연구소 기술연구실

박 기 철(Ki-Chul Park)



1998년 2월 : 경북대학교 전자공학과 졸업
1998년 3월~현재 : 경북대학교 전자공학과 석사과정

이 국 희(Kook-Heui Lee)

1993년 2월 경북대학교 전자공학과 (학사)
1995년 2월 경북대학교 전자공학과 (공학석사)
1995년 3월 ~ 현재 경북대학교 대학원 전자공학과 박사과정
주관심 분야 : 암호이론, 이동통신

문 상 재(Sang-Jae Moon)

1972년 2월 서울대학교 공과대학 공업교육과(전자공학, 공학사)
1974년 2월 서울대학교 대학원 전자공학과(통신공학, 공학석사)
1984년 6월 미국 UCLA(통신공학, 공학박사)
1984년 6월 ~ 1985년 6월 UCLA Postdoctor 근무
1984년 6월 ~ 1985년 6월 미국 OMNET컨설턴트
1994년 ~ 현재 경북대학교 공과대학 전지전자 공학부 교수

주관심 분야 : 정보보호, 디지털 통신, 정보통신망