인증 경로의 유효성을 높이는 효율적인 인증 경로 설정 알고리즘

정회원 최 연 희*, 전 문 석**

An Efficient Certificate Path Discovery Algorithm Making High a Certificate Path Validity.

Yeon-hee Choi*, Moon-seog Jun** *Regular Members*

요 약

다수의 인증 경로가 존재하는 PKI 구조에서 인증 경로를 설정하는 것은 중요한 문제이다. 경로 설정은 다양한 검증을 통해 이루어지고 검증이 많을수록 경로의 유효성은 높아진다. 유효성 높은 경로의 선택은 경로 설정 및 검증 작업의 반복 횟수를 감소시킴으로서 고속의 인증서 검증이 가능하도록 하는 반면에 하나의 경로를 설정하는데 소요되는 시간 및 부담을 증가시킨다는 문제점을 가진다. 본 논문에서는 보다 적은 부담으로 경로의 유효성을 높일 수 있는 효율적인 인증 경로 설정 알고리즘을 제안한다.

key Words : PKI, Path Validity, Certificate Path Discovery Processing

ABSTRACT

To discover a certificate path is a very important topic in the PKI with a lot of candidate paths. The certificate path discovery processing is executed via many verifications and as the number of verification times increases, the validity of the discovered path becomes high. The selection of the path with high validity provides high-speed certificate validation by reducing the number of repetition times of path discovery and validation processing. Otherwise, there is a problem that the speed and computation overheads are increased. In this paper, we propose an efficient certificate path discovery algorithm can make high the certificate validity with low overhead.

Ⅰ. 서 론

공개키 기반구조(Public Key Infrastructure, PKI)는 정보화 사회로 발전하기 위한 핵심 기반기술로서 전자상거래, 정보 보안 등 다양한 응용분야에서 사용자들이 공개키 암호 시스템을 이용하여 안전하게 정보를 교환할 수 있도록 하는 기반이 된다. 일반적으로 PKI에서는 통신하고자 하는 상대방의 인증서를 수령한 후 수령한 인증서에 대한 검증 과정을 필요로 한다. 사용자는 원하는 인증기관의 검증된 공개키를 가지고 있지 않을 경우 그 공개키를 획득할 수 있는 추가적인 인증서 연결 구조를 필요로 한다. 이 연결 구조는 특정 인증기관에서 발행한 인증서와 다른 인증기관에서 발행한 인증서들로 구성되며 이를 인증경로라 한다[1]. 상대방 인증서의 유효성은 인증 경로의 설정 및 검증 작업을 통해 판단된다.

인증 경로를 설정하는 문제는 현재까지 표준화된 방안이나 공개된 연구가 거의 없지만 인증서 검증의 필요성이 증가하면서 점차로 대두되는 중요한 분야이다. 각 국가 및 기관에서 운영하는 인증기관들의 상

* 한국산업기술대학교 e-비지니스학과 겸임교수(lovejung22@naver.com), ** 숭실대학교 정보과학대학 정교수
논문번호 : KICS2004-05-015 접수일자 : 2004년 5월 21일

호 호환을 위해 주로 네트워크 PKI 구조나 혼합 PKI 구조가 사용되는데, 이러한 구조의 PKI에서는 다수의 인증 경로가 존재함으로서 가능한 빨리 유효한 인증 경로를 찾는 것이 효율적인 설정 작업을 위한 변수가 된다[2]. 이는 설정에 필요한 다운로드 양을 줄이고 설정 작업의 반복횟수를 줄임으로서 연산 및 속도의 부담을 감소시킬 뿐만 아니라 검증 작업의 횟수 또한 줄임으로서 인증서의 유효성이 빠르게 판단될 수 있도록 한다. 경로의 유효성을 높이기 위해서는 설정 작업에 올바른 경로 설정을 위한 다수의 검증을 포함시켜야 한다. 그러나 검증의 수가 많아질수록 하나의 경로를 설정하는데 필요한 연산의 양은 증가되고 그에 비례하여 속도가 저하되는 문제가 발생한다.

본 논문에서는 다수의 경로를 가지는 PKI 구조에서 적은 연산 및 속도의 부담으로 경로의 유효성을 높일 수 있는 효율적인 인증 경로 설정 알고리즘을 제안한다. 제안한 알고리즘은 각 인증기관들로 하여금 자신의 상호 인증기관이 발행한 인증서들과 사용자 인증서들의 서명을 검증하여 이 결과를 서명 검증 인증서 (Signature Validation Certificate : SVC)로 발행하도록 하였다. 제안한 알고리즘은 SVC를 전적으로 신뢰한다는 가정 하에서 경로 설정 시에 인증서 대신 SVC를 다운로드 받아 SVC의 서명 검증 결과를 확인함으로서 서명 검증과 관련한 어떠한 연산 없이도 경로의 유효성을 높일 수 있다. SVC의 전적인 신뢰로 인해 발생되는 문제를 해결하기 위해 SVC의 신뢰성을 확인하는 연산을 포함시키는 경로 설정 알고리즘 또한 제안한다. 이는 해쉬 연산을 통해 수행됨으로서 부담이 적을 뿐만 아니라 SVC 신뢰성 확인을 통해 보다 높은 경로 유효성을 제공할 수 있다는 장점을 가진다.

Ⅱ. 관련 연구들

1. 정방향 인증 경로 설정

상대방으로부터 신뢰 인증기관으로 인증서의 발행자 (issuer)를 찾아 경로를 설정하는 방식으로서 계층적 PKI 구조에서 사용하기 적합하며 신뢰 인증기관의 인증서가 맨 마지막에 검증되기 때문에 검증 과정이 모두 완료된 후에야 경로의 유효성을 판단할 수 있다. 계층적 PKI 구조에서는 각 주체의 발행 인증기관이 하나로서 오직 하나의 인증 경로만 설정되어 검증되기 때문에 정 방향 설정 방식이 효율적으로 적용될 수 있다[2]. 그림 1은 A가 F의 인증서를 정 방향으로 설정하는 과정을 보인다. A는 F로부터 자신의 신뢰

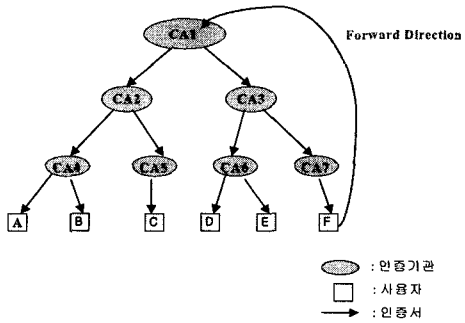

그림 1. 계층적 PKI 구조에서의 정방향 설정 방식

인증기관까지의 경로 상의 인증서들을 발행자를 찾아 <CA7,F>, <CA3,CA7>, <CA1,CA3>, <CA1,CA1>의 순서로 경로를 설정한다.

2. 역방향 인증 경로 설정

신뢰 인증기관으로부터 상대방으로 인증서의 주체(subject)를 찾아 경로를 설정하는 방식으로서 다수의 경로를 가지는 PKI 구조에서 사용하기 적합하다.

이 방식은 신뢰 인증기관이 맨 처음에 검증되기 때문에 작업의 수행 도중에 경로의 유효성이 판단될 수 있다. 따라서 하나 이상의 후보 경로가 존재함으로서 경로 설정이 복잡한 PKI 구조에서의 역 방향 설정은 보다 신속하게 경로가 검색되도록 한다[2][3]. 그림 2는 네트워크 PKI에서 A가 F의 인증서를 역방향으로 설정하는 과정을 보인다. A는 자신의 신뢰 인증기관인 CA2로부터 F까지의 경로 상의 모든 인증서들을 인증서 주체를 찾아감으로서 경로를 설정한다.

3. 인증 경로 설정 알고리즘

일반적으로 경로 설정 작업에서는 경로 상의 필요한 인

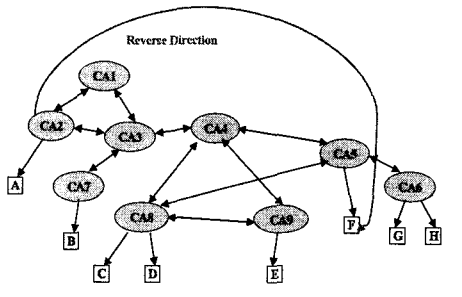

그림 2. 네트워크 PKI 구조에서의 역방향 설정 방식

증서와 CRL을 다운로드 받아 이들에 대해 검증 작업의 부분 집합 (이하 설정 작업)인 이름 연결, 유효기간 검사, 인증서 취소 상태 확인, 서명 검증 등을 수행함으로서 유효한 경로를 설정한다[4][5]. 그림 3은 역방향 설정 과정의 알고리즘을 나타낸 것이다. 알고리즘의 쉬운 이해를 위해 다음의 표기를 사용한다.

- <x,y> : x가 발행한 y의 인증서
- <x,y>-Discovery : 서명 검증을 제외한 <x,y>에 대한 인증 경로 설정 작업
- <x,y>-SV. : <x,y>의 서명 검증
- Cx : x가 발행한 인증서
- Certx : x의 인증서
- CAx : CA, x
- CC. : Crypto Calculation
- HC. : Hash Calculation

```
Start-CA = TrustedCA;
Start-cert = C_TrustedCA;
Final-cert = Cert_Target;
Next-CA = Subject of Start-cert;
WHILE (Start-cert ≠ Final-cert) {
  Start-CRL = Download CRL issued by Start-CA;
  CA-cert = Download C_Next-CA;
  Perform <Start-CA, Next-CA>-Discovery;
  Perform<Start-CA,Next-CA>-SV by CC.;
  IF (<Start-CA, Next-CA>-Discovery = No OR
      <Start-CA, Next-CA>-SV.by CC.= No) {
        Next-CA = Another-Subject of Start-cert;
        CA-cert = Download C_Next-CA;
        Perform <Start-CA, Next-CA>-Discovery;
        Perform <Start-CA,Next-CA>-SV. by CC.;}
      Include CA-cert on the Path;
      Start-CA := Next-CA;
      Start-cert := CA-cert;
      Next-CA := Subject of Start-cert;}
Include CA-cert on the Path;
Return the Final Certificate Path;
Finish Certificate Path Discovery;
```

그림 3. 인증 경로 설정 알고리즘

그림 3에서, 설정자는 각 인증기관이 발행한 인증서 및 CRL을 다운로드 받아 이들에 대해 설정 작업을 수행하여 유효한 인증서를 선택한 후 이를 경로에 추가한다. 경로에 추가한 인증서의 주체를 찾아 같은 방법으로 설정 작업을 수행함으로서 상대방 인증서까지의 최종 경로를 설정한다. 설정된 경로는 RFC 3280 인증서 검증 알고리즘[6]을 통해 검증되고 이 과정은 유효한 경로가 설정되어 인증서의 정확한 유효성이 판단될 때 까지 반복된다. 따라서 다양한 검증을 통해 유효성이 높은 경로를 찾는 것은 전반적인 설정 및 검증과 관련한 부담을 줄이고 고속의 검증을 가능하게 한다. 반면 하나의 경로를 설정하는데 따른 부담을 초래할 수 있다. 특히 서명 검증을 포함할 시에는 더욱 부담이 가중된다. 따라서 많은 경로 처리 구현은 서명 검증을 인증 경로 검증 작업 시에만 수행하도록 하는 추세이지만 설정 작업 시의 서명 검증은 유효한 경로를 설정할 확률을 높일 수 있다는 장점이 제공됨으로서 부담과 경로 유효성 사이의 트레이드 오프 문제가 발생하게 된다. 이러한 문제를 해결하기 위해 Dijkstra 알고리즘을 이용하는 등의 다양한 연구가 계속되고 있다[7].

Ⅲ. 제안한 알고리즘

1. 서명 검증 인증서 (Signature Validation Certificate : SVC)

제안한 알고리즘은 네트워크 PKI에서의 수행을 기본으로 함으로서 각 인증기관은 자신의 상호 인증기관의 공개키를 알기 때문에 상호 인증기관이 발행한 인증서들의 서명을 검증한다. 그림 4는 네트워크 PKI 구조에서 각 인증기관이 서명 검증한 인증서들의 예를 보인다. 그림 4에서 보이는 것처럼 네트워크 PKI 영역내의 모든 인증기관들은 자신이 발행한 인증서 주체인 사용자나 상호 인증기관들의 공개키를 알기 때문에 이들이 발행한 인증서들의 서명을 검증할 수 있다. 이 때, 각 인증기관은 상호 인증기관이 자신에게 발행한 인증서에 대한 서명 검증은 수행할 필요가 없다. 예로서, CA5는 자신의 상호 인증기관인 CA3와 CA4의 공개키를 알기 때문에 CA3가 발행한 c5,c11,c17와 CA4가 발행한 c2,c3,c6,c27,c28의 서명을 검증할 수 있다. 각 인증기관들은 서명 검증한 결과로서 SVC를 발행하여 이를 공개한다. SVC에는 검

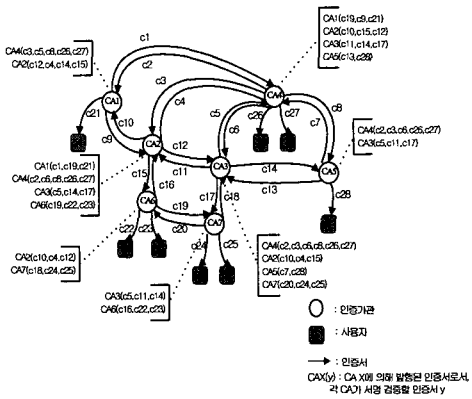

그림 4. 각 인증기관의 서명 검증할 인증서

```
Start-CA = TrustedCA;
Start-cert = C_TrustedCA;
Final-cert = Cert_Target;
Next-CA = Subject of Start-cert;
Start-CRL = Download CRL issued by Start-CA;
CA-cert = Download C_Next-CA;
Perform <Start-CA, Next-CA>-Discovery;
WHILE (<Start-CA, Next-CA>-Discovery = No) {
        Next-CA = Another-Subject of Start-cert;
        CA-cert = Download C_Next-CA;
        Perform <Start-CA, Next-CA>-Discovery;}
Include CA-cert on the Path;
Start-CA := Next-CA;
Start-cert := CA-cert;
Next-CA := Subject of Start-cert;
SVC-CA = TrustedCA;
WHILE ("c-subject"of Start-cert ≠ "subject name" of Final-cert) {
    Start-CRL = Download CRL issued by Start-CA;
    CA-cert = Download SVC for Next-CA issued by SVC-CA;
    IF (CA-cert = NULL){
        CA-cert = Download C_Next-CA;
        Perform <Start-CA, Next-CA>-Discovery;}
    Perform <Start-CA, Next-CA>-Discovery;
    Identify "valid status" of Start-SVC;
    WHILE (<Start-CA, Next-CA>-Discovery = No OR
            "valid status" of Start-SVC = invalid) {
        Next-CA = Another-Subject of Start-cert;
        CA-cert = Download SVC for Next-CA issued by SVC
        Perform <Start-CA, Next-CA>-Discovery;
        Identify "valid status" of Start-SVC; }
    Include CA-cert on the Path;
    SVC-CA := Start-CA;
    Start-CA := Next-CA;
    Start-cert := CA-cert;
    Next-CA := Subject of Start-cert;}
Include CA-cert on the Path;
Return the Final Certificate Path;
Finish Certificate Path Discovery;
```

그림 5. 제안한 인증 경로 설정 알고리즘 1

증자로 하여금 검증한 주체 인증서를 확인할 수 있도록 주체 인증서의 항목 중 issuer name, validity period, subject name, signature등을 포함하고 추가로 서명 검증한 결과와 검증자 측의 간단한 인증서 무결성 검증을 위해 필요한 주체 인증서의 내용에 해쉬 계산한 값을 포함한다.

각 인증기관은 상호 인증기관들의 행위를 감시하여 이들이 발행한 인증서들의 변화에 따라 새로운 SVC의 생성 및 기존 SVC의 내용을 갱신해야 한다. 제안한 방식은 SVC 갱신 작업을 제외한 모든 SVC 관련 작업은 인증기관의 여유로운 시간에 수행하도록 함으로서 인증기관 측의 부담이 최소한으로 감소될 수 있도록 하였다.

2. 인증 경로 설정 알고리즘 1

제안한 첫 번째 인증 경로 설정 알고리즘은 SVC를 전적으로 신뢰한다는 가정 하에서 수행된다. 이 알고리즘은 SVC의 서명 검증 결과 확인을 통해 서명 검증을 수행함으로서 추가되는 부담 없이 보다 유효한 경로를 설정할 수 있다. 제안한 알고리즘에서는 CRL과 더불어 인증서 대신 해당 인증서에 대한 SVC를 다운로드 받는다. 그림 5는 제안한 설정 알고리즘을 보인다. 그림 5에서, c-subject와 valid status는 SVC에 포함된 항목들로서, 이들은 각각 주체 인증서의 subject name으로부터 복사된 값과 주체 인증서의 서명 검증 결과를 나타낸다. 그림 5에서 보이는 것처럼 인증기관의 자체 서명 인증서를 제외한 경로 상의 첫 번째 인증서를 선택하기 위해 신뢰 인증기관이 발행한 CRL과 인증서들이 차례로 다운로드 되어 설정 작업이 수행된다. 경로 상의 다음 인증서를 선택하기 위해 CRL과 다음 인증서 후보들에 대한 SVC가 차례로 다운로드 되고 SVC의 주체 인증서 복사 항목을 이용하여 설정작업이 수행된다. 이 과정은 상대방 인증서에 대한 SVC가 발견될 때까지 계속된다.

제안한 알고리즘은 SVC가 전적으로 신뢰된다면 기존 알고리즘과 같은 조건 및 부담 하에서 유효한 경로를 설정할 확률이 높아지지만 SVC에 문제가 발생할 경우에는 불필요한 설정 및 검증 작업이 반복됨으로서 기존 알고리즘보다 더 복잡한 수행과 큰 부담을 야기할 수 있다. 이 문제는 SVC가 원래 인증서에 대한 검증 결과인지 혹은 SVC 발행 이후의 인증서 내용의 무결성 및 서명의 합법성을 확인하는 등의 SVC 신뢰성 확인 작업을 알고리즘에 포함시킴으로서 해결될 수 있다.

3. 인증 경로 설정 알고리즘 2

제안한 두 번째 알고리즘은 SVC의 신뢰성을 확인하는 과정을 포함함으로서 첫 번째 알고리즘의 SVC 의존에 따른 문제점을 해결하였다. 이 알고리즘은 기존 알고리즘의 암호화 계산을 통한 서명 검증을 통해 설정된 경로와 같은 정도의 유효성을 가지지만 훨씬 적은 부담으로 운영된다는 장점을 가진다. 그림 6은 제안한 두 번째 경로 설정 알고리즘을 보인다.

이 알고리즘은 첫 번째 알고리즘과 달리 설정 작업을 위해 CRL과 더불어 인증서와 SVC를 모두 다운로드 받는다. 신뢰 인증기관이 발행한 인증서들과 SVC 미발행 등으로 인해 SVC가 존재하지 않는 인증서들에 대해서는 서명 검증을 제외한 나머지 검증을 통한 설정 작업만을 수행한다. SVC가 존재하는 인증서들에 대해서는 제안한

```
Start-CA = TrustedCA;
Start-cert = C_TrustedCA;
Final-cert = Cert_Target;
Next-CA = Subject of Start-cert;
SVC-CA = Null;
WHILE ("c-subject"of Start-cert≠"subject name"of Final-cert) {
    Start-CRL = Download CRL issued by Start-CA;
    CA-cert = Download C_Next-CA;
    Start-SVC = Download SVC for Next-CA issued by Start-CA;
    IF (Start-SVC = NULL){
        Perform <Start-CA, Next-CA>-Discovery; }
    Identify "valid status" of Start-SVC;
    Perform <Start-CA, Next-CA>-SV.by HC.;
    Perform <Start-CA, Next-CA>-Discovery;
    WHILE(<Start-CA, Next-CA>-Discovery = No
            OR "valid status" of Start-SVC = invalid OR
            <Start-CA, Next-CA>-SV.by HC. = No){
      Next-CA = Another-Subject of Start-cert;
      CA-cert = Download C_Next-CA;
      Start-SVC=Download SVC for Next-CA issued by Start-CA;
      IF (Start-SVC = NULL){
          Perform <Start-CA, Next-CA>-Discovery; }
      Perform <Start-CA, Next-CA>-Discovery;
      Identify "valid status" of Start-SVC;
      Perform <Start-CA, Next-CA>-SV.by HC.; }
      Include CA-cert on the Path;
      Start-CA := Next-CA;
      Start-cert := CA-cert;
      Next-CA := Subject of Start-cert;
      SVC-CA := Start-CA;}
  Include CA-cert on the Path;
  Return the Final Certificate Path;
  Finish Certificate Path Discovery;
```

그림 6. 제안한 인증 경로 설정 알고리즘 2

첫 번째 알고리즘과 마찬가지로 기본 설정작업 외에 SVC의 "valid status" 항목을 통해 서명 검증 결과를 확인하는 작업이 수행된다. 여기에 추가로 SVC의 신뢰성 확인을 위한 인증서 서명 검증 작업이 추가된다. SVC를 통한 인증서 서명 검증 작업은 우선 인증서의 내용에 해쉬 계산을 한 후, 이를 SVC에 포함된 주체 인증서 내용의 해쉬 값과 비교함으로서 인증서의 무결성을 확인한다. 기존의 암호화 계산을 통한 무결성 검사도 계산된 해쉬 값과 암호화 계산을 통해 추출된 기존 해쉬 값의 비교를 통해 수행되기 때문에 제안한 무결성 검증은 기존의 서명 검증 방식과 같은 정도의 신뢰성을 갖는다. 다음의 서명 검증 과정으로서 인증서의 서명을 SVC에 포함된 주체 인증서의 서명과 비교하여 인증서의 합법성을 확인한다. 암호화 계산을 통해 검증된 서명이 SVC에 포함되었기 때문에 2개 서명 값의 비교만으로 서명의 합법성을 확인할 수 있다.

결과적으로 제안한 알고리즘은 오직 해쉬 계산을 통해 SVC의 신뢰성 및 인증서의 무결성과 합법성을 검증하기 때문에 이 알고리즘을 통해 설정된 경로는 기존 및 첫 번째 알고리즘보다 높은 유효성이 보장된다. 또한 암호화 계산을 통해 서명 검증을 수행하는 경우와 같은 정도의 유효성을 가지지만 해쉬 연산이 암호화 연산보다 훨씬 빠름으로서 연산 및 속도 측면에서 훨씬 부담이 감소된다는 장점을 가진다.

표 1. 기존 알고리즘과 제안한 알고리즘의 비교

	기존 알고리즘	제안 알고리즘 1	제안 알고리즘 2
다운로드시간	t_{down}	t_{down}	$t_{down}+\alpha$
전체검색시간	t_{search}	t_{search}	$t_{search}+\beta$
인증서 당 설정 시간	t_d	$t_d+\sigma$	$t_d+\sigma+t_{hash}$
전체설정 소요 시간	$n_{total} \cdot n_{oc} \cdot t_d \approx O(n^2t)$	$n_{total} \cdot n_{oc} \cdot t_d + (n_{total}-2) \cdot n_{oc} \cdot \sigma \approx O(n^2t)$	$n_{total} \cdot n_{oc} \cdot t_d + (n_{total}-2) \cdot n_{oc} \cdot (t_{hash}+\sigma) \approx O(2n^2t)-O(2nt)$
부담	*Low*	*Low*	*Some*
경로 유효성	*Low*	*Medium*	*High*

Ⅳ. 분 석

표 1은 제안한 알고리즘들을 기존 알고리즘과 다양한 측면에서 비교 분석한 것이다. 여기서, n_{total}, n_{oc}, t_{hash}는 각각 경로 상의 모든 인증서 수, 각 경로마다 검증해야 할 인증서 수, 서명 검증을 위한 인증서 당 해쉬 계산 시간을 나타낸다.

표 1에서 보이는 바와 같이, 제안한 알고리즘 1은 인증서 대신 SVC를 다운로드 받음으로서 다운로드와 관련한 추가 시간이 소요되지 않는 반면에 알고리즘 2는 기존 정보 외에 SVC를 요구함으로서 α의 시간이 추가로 소요된다. 이에 따라 SVC 검색과 관련한 β의 추가 시간이 발생한다. SVC의 "valid status" 검색 및 확인 시간인 σ와 SVC를 통한 서명 검증 시간인 t_{hash}가 제안한 알고리즘들에서는 추가로 소요된다. 사소한 α, β, σ의 시간을 무시하고 n_{total} 과 n_{oc}를 같은 크기로 가정했을 때, 하나의 경로를 설정하는 데 걸리는 시간은 기존 알고리즘과 제안 알고리즘 1이 $O(n^2t)$로 같은 시간이 소요되는 반면 제안 알고리즘 2는 $O(2n^2t)-O(2nt)$의 시간이 소요됨으로서 연산 및 속도의 부담이 야기될 수 있다. 그러나 이러한 부담은 제안 알고리즘 2의 높은 경로의 유효성을 가져온다. 제안 알고리즘 2의 부담을 분석하기 위해 각 알고리즘의 설정 시간을 측정하였다. 이를 위해 우선 다양한 해쉬 함수의 수행 속도를 측정하였다. 측정은 2개

의 UltraSPAC-II 400MHz CPU를 가진 Sun Enterprise 3500 서버에서 OPENSSL 라이브러리[8]를 통해 이루어졌다. 표 2는 측정된 해쉬 함수의 수행 속도를 보인다.

각 알고리즘의 설정 시간을 표 2의 측정 속도에 따라 계산하였다. 여기서는 t_d를 1ms로 가정하였고 n_{total}과 n_{oc}는 같은 값으로 계산하였다. 또한 보다 정확한 성능을 분석하기 위해 가장 큰 해쉬 크기인 8192바이트의 경우를 측정하여 비교하였다.

표 2. 해쉬 함수의 수행 속도

HSize (bytes)	MD4		MD5	
	λ_h *(bytes/ms)*	t_{hash} *(ms)*	λ_h *(bytes/ms)*	t_{hash} *(ms)*
8	4956	0.0016	3426	0.0023
64	21131	0.0030	14846	0.0043
256	34409	0.0074	23876	0.0107
1024	43373	0.0236	29404	0.0348
8192	47609	0.1720	31779	0.2577
HSize (bytes)	SHA1		RC4	
	λ_h *(bytes/ms)*	t_{hash} *(ms)*	λ_h *(bytes/ms)*	t_{hash} *(ms)*
8	2019	0.0039	26830	0.0002
64	8820	0.0072	38541	0.0016
256	15021	0.0170	40379	0.0063
1024	18575	0.0551	41012	0.0249
8192	19887	0.4119	40782	0.2008

표 3. 기존 및 제안 알고리즘의 설정 시간

n	기존 *(ms)*	제안 1*(ms)*	제안 2 *(ms)*			
			MD4	MD5	SHA1	RC4
3	9	9	9.516	9.7731	10.2357	9.6024
5	25	25	27.58	28.8655	31.1785	28.012
7	49	49	55.02	58.0195	63.4165	56.028
10	100	100	113.76	120.616	132.952	116.064
15	225	225	258.54	275.251	305.3205	264.156
30	900	900	1044.48	1116.47	1245.996	1068.672
50	2500	2500	2912.8	3118.48	3488.56	2981.92
100	10000	10000	11685.6	12525.5	14036.62	11967.84

표 3은 n 값과 표 2의 해쉬 함수 수행 속도에 따른 기존 알고리즘과 제안 알고리즘들의 설정 시간을 나타낸다. 표 3에서, 기존 및 제안 알고리즘 1은 해쉬 함수의 계산이 포함되지 않음으로서 해쉬 함수 속도와 상관없이 같은 설정 시간이 측정되었다. 제안 알고리즘 2는 각 해쉬 함수 속도와 n의 크기에 따라 다양한 설정 시간이 측정되었다. 측정 결과, 해쉬 함수 속도가 매우 빠르기 때문에 제안 알고리즘 2는 다른 알고리즘들보다 약간의 시간만이 더 소요됨을 알 수 있었다. 그림 7은 n이 10일 때의 각 알고리즘의 수행 시간을 비교한 것이다.

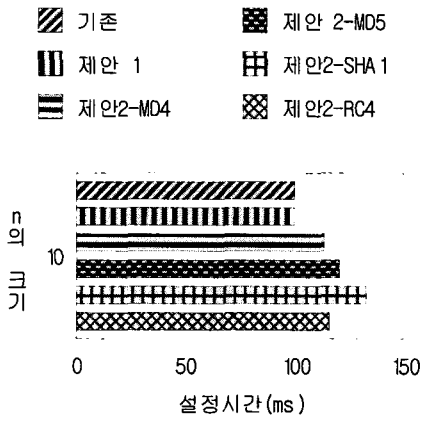

그림 7. n=10일 때의 각 알고리즘의 전체 설정 소요 시간

그림 7에서 n=10일 때의 속도 배율은 사용하는 해쉬 알고리즘에 따라 0.752부터 0.879사이의 값을 가진다. 표 3으로부터 n이 다른 값일 경우에도 0.945부터 0.712 사이의 속도 배율을 보임을 알 수 있다. 이는 n의 크기가 커져 해쉬 연산 횟수가 증가하더라도 기존 알고리즘과 제안 알고리즘 2의 속도 차이가 그다지 크지 않음을 의미한다. 결과적으로 설정 작업을 수행하는데 요구되는 기존 알고리즘과 제안 알고리즘 2의 전체적인 부담이 거의 비슷하다는 결론을 얻을 수 있었다.

V. 결 론

본 논문에서는 다수의 인증 경로를 가지는 PKI에서 높은 유효성을 가지는 경로 설정 알고리즘을 제안하였다. 제안한 알고리즘은 인증기관들로 하여금 서명 검증을 수행하여 검증 결과를 SVC로 발행하도록 하고 설정자는 SVC를 통해 서명 검증 결과를 확인함으로서 서명 검증과 관련한 기존의 연산적인 부담 없이 유효한 경로를 결정하도록 하였다. 또한 SVC의 신뢰성을 확인하는 과정을 추가하여 적은 부담으로 경로의 유효성을 더욱 높임으로서 검증 작업을 고속화시키는 결과를 가져올 수 있다.

SVC를 전적으로 신뢰한다는 가정 하에서 제안한

알고리즘은 SVC를 통해 서명 검증 결과를 확인함으로서 설정된 경로는 보다 높은 유효성을 가진다. 그러나 SVC의 전적인 신뢰는 SVC의 변조나 위조가 발생할 경우에는 올바르지 않은 경로를 설정할 확률이 높게 되고 이는 기존 알고리즘보다 비효율적인 설정 과정 및 결과를 야기할 수 있다. 이러한 문제를 해결하기 위해 제안한 두 번째 알고리즘에서는 SVC를 통한 인증서 서명을 검증하는 과정을 추가함으로서 다른 알고리즘들에 비해 검증의 유효성을 한층 높이는 결과를 가져왔다. 비록 해쉬 연산 시간의 추가로 인해 $O(2n^2t)$-$O(2nt)$의 설정 시간이 소요됨으로서 $O(n^2t)$의 시간이 소요되는 다른 알고리즘에 비해 연산에 대한 부담은 다소 증가하지만 이는 설정 시간의 분석을 통해 0.945-0.712의 속도배율이 측정됨으로서 해쉬 연산의 포함이 경로 설정 시간에 거의 영향을 미치지 않는다는 것을 확인할 수 있었다.

인증기관은 SVC를 인증서 발행 시 오직 한번 발행하고 그들의 자유로운 시간에 발행하는 것을 기본으로 하기 때문에 큰 부담은 되지 않지만 정확한 인증기관 측의 부담을 분석함으로서 제안한 알고리즘의 효율성을 보다 정확히 입증하는 방안이 향후 연구되어져야 할 것이다.

참 고 문 헌

[1] R.Housley, Russ, "Best Parctices Guide for Deploying Public Key Infrastructure", Wiley, 2001.

[2] Y. Elley, A. Anderson, S. Hanna, S. Mullan, R. Perlman and S. Proctor, "Building Certification Paths : Forward vs.Reverse", *In Network and Distributed System Security Symposium Conference Proceedings*, 2001.

[3] Peter M. Hesse and David P. Lemire, "Managing Interoperability in Non-Hierarchical Public Key Infrastructures", *In Network and Distributed System Security Symposium Conference Proceedings*, 2002.

[4] D. Pinkas and R. Housley, "Delegated Path Validation and Delegated Path Discovery Protocol Requirements", IETF RFC 3379, September 2002.

[5] 황보성, "서버 기반 인증서 검증", http://www.rootca.or.kr/down/down1/Server%20Based%20Certificate%20Validation.pdf"

[6] R. Housley, W. Polk, W.Ford, D. Solo, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile", IETF RFC 3280, January 2002.

[7] 유종덕, 이주남, 이구연, "Enterprise PKI에서의 고속 인증 경로 탐색 알고리즘의 설계 및 구현", *정보 보호 학회 논문지*, 제 2권, 제 2호, 2002년 4월.

[8] Eric A. Yong and Tim J. Hudson, OPENSSL toolkit, http://www.openssl.org.

최 연 희 (Yeon-hee Choi) 정회원

1991년 2월 : 목포대학교 전산통계학과 졸업

1993년 2월 : 숭실대학교 전자 계산 학과 석사

2004년 2월 : 숭실대학교 컴퓨터학과 박사

2004년 9월~현재 한국산업기술대학교 e-비지니스학과 겸임교수

<관심분야> 정보보안, 컴퓨터 통신, 암호학

전 문 석 (Moon-seog Jun) 정회원

1980년 2월 : 숭실대학교 전자계산학과 졸업

1996년 : University of Maryland 전산과 석사

1989년 : University of Maryland 전산과 박사

1989년 : Morgan State University 전산수학과 조교수

1989년~1991년 : New Mexico State University 부설 Physical Science Lab. 책임연구원

1991년~현재 숭실대학교 정보과학대학 정교수

<관심분야> 네트워크보안, 컴퓨터 알고리즘, 병렬처리, VLSI 설계, 암호학