스트림 인증에 적합한 개선된 HORS 기법
(An Improved HORS for Stream Authentication)

박 용 수 † 조 유 근 ††
(Yongsu Park) (Yookun Cho)

요 약 본 논문에서는 스트림 데이타를 인증하는데 적합한 개선된 HORS 일회용 서명 기법을 제시한다. 스트림 인증에 일회용 서명 기법을 사용할 경우, 가장 큰 문제점은 큰 서명 크기로 인해 네트워크 오버헤드가 많다는 점이다. 제시한 기법은 기존 일회용 서명 기법 중 서명 크기가 가장 작다. 또한, 제시된 기법의 검증 연산량은 매우 작다. 스트림 인증에 적합한 기존 기법과 온라인 서명 연산량을 비교하면, HORS보다는 다소 많지만, BiBa나 Powerball보다 매우 작다. 제시된 기법에서 서명 연산은 쉽게 병렬 처리가 가능하며, 특히 비밀키를 가지지 않는 서버의 도움을 받을 수 있기 때문에, 서명 서버의 부하를 손쉽게 그리고 안전하게 줄일 수 있다.

키워드 : 보안, 암호, 전자 서명, 스트림 배포

Abstract We propose an efficient one-time signature scheme for stream authentication by improving HORS. When one-time signatures are used for authenticating live streams, one of the most serious drawbacks is that its large signature size yields high communication overhead. Compared with the previous one time signature schemes, proposed scheme has the smallest signature size. Moreover, verification overhead is very low. Compared with the previous schemes for stream authentication, signing overhead of our scheme is larger than that of HORS but much lower than those of BiBa or Powerball. Moreover, signing operation can be trivially parallelized without any additional risk because it does not require sharing of the secret key between distributed servers.

Key words : security, cryptography, digital signature, stream distribution

1. 서 론

인터넷 사용자가 늘어나고 망 대역폭이 커짐에 따라 인터넷을 통하여 오디오나 비디오 같은 스트림 데이타를 제공하는 서비스가 늘어나고 있다. 또한, 인터넷이 상업적인 용도로 이용되면서 이런 서비스들도 점차 유료화되고 있는 추세이다.

상업적인 스트림 서비스가 널리 사용되기 위해서는 보안이 무엇보다도 중요하다. 일례로, 방송국에서는 임의의 해커가 방송 데이타에 상업적인 내용을 삽입하는 행위를 방지하고 싶어할 것이며, 방송 청취자는 증권 정보나 뉴스가 위/변조되지 않았음을 확인하고 싶어할 것이다. 이렇듯, 여러 가지 보안 요소 중 데이타의 위/변조를 방지하고 부인 방지를 제공하는 인증 기법이 무엇보다 중요하다[1].

실시간으로 생성/전달되는 스트림 데이타를 인증하는 문제는 일반 데이타를 인증하는 것과 다르다. 가장 큰 문제점은 데이타의 크기가 매우 크다는 점이다. 일반적으로 전자 서명이 많은 계산량을 요구하는 작업임을 생각해볼 때, 모든 데이타를 서명하는 것은 송신자에게 상당한 부하를 준다[1].

일회용 서명 기법은 일반적인 전자 서명 기법보다 서명 연산량이 매우 작아 스트림 인증에 적합하지만, 서명 크기가 커서 높은 통신 오버헤드를 야기하는 단점이 있다[2,3]. 일례로, SHA-1을 사용하는 경우, 160 비트 메시지에 대한 Lamport 일회용 서명 기법 [4]의 서명 크기는 3200 바이트이다. 서명될 스트림 청크의 크기가 통

† 비 회 원 : 서울대학교 전기컴퓨터공학부
yspark@ssrnet.snu.ac.kr
†† 종신회원 : 서울대학교 전기컴퓨터공학부 교수
cho@ssrnet.snu.ac.kr
논문접수 : 2003년 1월 10일
심사완료 : 2003년 5월 7일

1) 해쉬 연산을 이용하여 스트림 데이타의 크기를 줄인 후 서명하는 방법은 송/수신자 측에서 지연 시간을 야기한다. 또한, 패킷 손실이 발생할 경우 서명 검증이 실패하는 등 여러 가지 단점이 있다.

상 512 byte보다 작은 것을 생각해볼 때[1], 통신 오버헤드가 매우 높음을 알 수 있다. 기존 연구 중 [3]은 기존 일회용 서명 기법 대신 k 회용 서명 기법을 사용하여 네트워크 오버헤드를 줄였다. 또한, [1,2,5]은 서명 크기가 작은 일회용 서명 기법을 제시했다.

본 논문에서는 스트림 인증에 적합한 효율적인 일회용 서명 기법을 제시한다. 우리는 HORS(Hash to Obtain Random Subset)[5]를 개선하여 서명 크기를 감소시켰다. 기존 일회용 서명 기법과 비교해 볼 때, 제시된 기법은 가장 작은 서명 크기를 가진다. 또한, 제안 기법은 검증 연산량이 매우 작다. 온라인 서명 연산량은 HORS보다 다소 크지만, 기존 기법 중 스트림 인증에 적합한 기법인 BiBa(Bins and Balls)[2], Powerball[6] 보다는 매우 작다. 제시된 기법에서 서명 연산은 쉽게 병렬 처리가 가능하며, 특히 비밀키를 가지지 않는 서버의 도움을 받을 수 있기 때문에, 서명 서버의 부하를 손쉽게 그리고 안전하게 줄일 수 있다.

본 논문의 구성은 다음과 같다. 먼저 2 장에서 관련 연구를 서술한 후, 3 장에서 HORS를 설명한다. 4 장에서는 HORS를 개선한 제안된 기법을 제시한 후, 제안된 기법의 계산량, 안전성, 서명 크기 등을 분석하고 기존 기법과 비교한 결과를 설명한다. 마지막으로, 5 장에서 결론을 맺는다.

2. 관련 연구

스트림 데이타의 인증에 관한 연구는 크게 2 가지로 나뉜다. 첫째, 스트림 데이타에 적합한 매우 효율적인 서명 기법을 연구한 내용과, 둘째, 데이타를 그룹으로 묶은 후 서명 연산을 적용하여 많은 계산량을 요구하는 서명 연산의 횟수를 줄인 내용이다. 후자의 경우, 서버의 계산량이 매우 적으며 네트워크 오버헤드가 매우 작은 장점이 있으나, 데이타 생성 시 혹은 데이타 검증 시 지연 시간이 발생하며, 대부분 기법에서 수신한 데이타를 일부만 검증할 수 있다는 단점이 있다. 전자의 경우, 받은 데이타를 모두 검증할 수 있고 추가의 지연 시간이 발생하지 않지만, 통상 일회용 서명 기법을 사용하기 때문에 네트워크 오버헤드가 크다는 단점이 있다. 이들 두 가지 방법은 서로 독립적이며 병행해서 사용하면 보다 좋은 결과를 얻는다[7].

2.1 효율적인 스트림 서명에 관한 연구

보다 효율적인 서명에 대한 연구는 여러 가지 방법으로 이루어져왔다. 간단한 예로, RSA 서명 기법에서 중국인의 나머지 정리(Chinese Remainder Theorem)나 슬라이딩 윈도우 기법을 사용하면 보다 빠른 서명 연산을 수행할 수 있다[4]. 또한 Wong과 Lam은 많은 메모리를 이용하여 Feige-Fiat-Shamir 서명 기법을 보다 효율적으로 구현할 수 있는 방법을 제시하였다[8]. 제시한 방법에서 서명 연산 속도와 메모리 사용량은 상충관계(trade off)를 가지며, 최적화된 매개 변수를 선택하면 서명 연산을 DSA보다 2 배 빠르게 수행하면서도 서명 확인 속도를 작은 지수 값을 가지는 RSA와 비슷한 수준으로 만들 수 있다. 또한, 서명을 확인할 때 서명 확인자의 계산 능력치에 따라 서로 다른 수준의 신뢰를 주는 조정 가능하고 점진적인 서명 기법을 제안하였다.

효율적인 서명 기법으로 일회용 서명 기법이나 k 회용 서명 기법이 연구되었으며, Gennero와 Rohatgi는 이 기법을 이용하여 온라인 스트림 데이타에 대한 인증 기법을 제시했다[9]. 다만, 이들 서명 기법은 일반적인 전자 서명 기법과는 달리, 서명의 크기가 서명될 데이타의 크기에 비례하며, SHA 1 해쉬 함수를 사용하는 경우 통상 1000 바이트가 넘는다[7]. Rohatgi는 일회용 서명 대신 k 회용 서명을 사용하는 방법을 제안하였으며, 서명 크기를 줄이는 방법 및 네트워크 오버헤드를 줄이는 방법을 연구하였다[3]. Rohatgi는 첫째, 서약(commitment)과 TCR(Target Collision Resistant) 해쉬 함수를 사용하여 k 회용 서명 기법의 크기를 줄였으며, 둘째, k 회용 서명 기법에서 공개키 트리를 구성 할 때 TCR 해쉬 함수를 사용하여 공개키 크기를 줄였다. 셋째, 인증서의 크기를 최적화하여, 네트워크 오버헤드를 줄였다. 이 세 가지 방법을 동시에 사용하면, [9]의 온라인 방법을 사용할 때 스트림 청크 당 오버헤드가 약 300 바이트 정도이다[3].

최근, Perrig는 스트림 인증에 적합한 BiBa 일회용 서명 기법을 제시하였다[2]. BiBa 기법은 기존 일회용 서명 기법과는 달리, 생일 패러독스(birthday paradox)를 이용한다. 서명자는 비밀키로 다수의 난수값을 생성하며, 각 난수값을 공(ball)이라 부른다. 서명자는 각 공의 서약값(commitment)을 생성하여 이를 공개키로 한다. 서명될 메시지 값에 따라 각 공을 바구니(bin) 중 하나에 대응시킨 후, 바구니에 대응된 공 중에서 일정 패턴(일례로, 같은 바구니에 i 개 이상의 공이 대응된 경우)을 찾으면, 그 공들이 서명값이 된다. 이 기법에서 서명자는 다수의 공을 가지고 있기 때문에 패턴을 쉽게 찾을 수 있지만, 공격자는 가지고 있는 공의 수가 매우 적기 때문에 패턴을 쉽게 찾을 수 없는 성질을 가진다. BiBa는 기존 일회용 서명 기법에 비해, 서명 연산량이 많고 공개키의 크기가 크지만, 서명 크기가 매우 작으며 검증 연산량이 매우 작다. Mitzenmacher는 [6]에서

BiBa를 개선한 Powerball 서명 기법을 제안하였다. Powerball은 BiBa보다 온라인 서명 연산량이 2 배 많지만, 서명 크기와 검증 연산량이 더 작다.

Reyzin은 [5]에서 스트림 인증에 적합한 일회용 기법인 HORS를 제안하였다. HORS는 서명 연산량이 해쉬 연산 1번으로, BiBa나 Powerball에 비해 매우 작다. 하지만, HORS는 서명 크기가 BiBa나 Powerball에 비해 다소 크다. HORS에 대한 자세한 내용은 3 장에서 설명한다.

2.2 데이타를 그룹으로 묶는 방법에 대한 연구

데이타를 그룹으로 묶는 방법으로는 Gennaro와 Rohatgi가 제안한 방법으로 오프라인 방식이 있다[9]. 이 기법에서 패킷 그룹 내 k 번째 패킷은 $k+1$ 번째 패킷을 해쉬 함수에 입력한 결과값을 가지고 있으며 그룹 내 첫 번째 패킷만이 서명되어 전체적으로 서명의 횟수가 상당히 줄어든다. 하지만, 이 기법은 패킷 데이타가 한 개라도 전송 도중 유실될 경우 그룹 내 그 이후의 패킷에 대해 검증할 수 없다는 단점을 가진다. 또한, 이 기법은 그룹의 마지막 패킷부터 처음 패킷까지 역방향으로 계산하여야 한다. 따라서, 첫 패킷부터 차례대로 데이타를 생성하여 전송하는 온라인 스트림 데이타의 성질과 잘 맞지 않으며 오프라인 방식에 적합하다.

Wong과 Lam은 Merkle이 고안한 인증 트리 기법[10]을 이용하여 데이타를 그룹으로 묶어 서명하였다[8]. 이 기법에서 송신 서버는 먼저 해쉬 함수를 이용하여 트리 구조를 만든 후, 루트 노드를 서명한다. 각 패킷은 서명값과 트리 내 자신에게 해당하는 잎 노드부터 루트 노드까지의 경로 중 형제값을 모두 가진다. 이 기법의 장점은 그룹 내 패킷을 단 1 개만 수신하여도 수신자는 서명을 확인할 수 있다. 하지만, 각 패킷에 포함되는 인증 부가 정보량이 상당히 크다. 또한, 그룹의 크기가 송신 서버의 패킷 버퍼 크기로 제한되어 그룹의 크기를 늘리는 데 제약점을 가지며, 그룹 내 모든 패킷의 인증 정보가 한꺼번에 계산되어서 송신 서버가 보내는 데이타 패턴이 간헐적(bursty)으로 배출되는 단점이 있다.

Perrig, Canetti, Song, 그리고, Tygar는 TESLA (Timed Efficient Stream Loss-tolerant Authentication)와 EMSS라는 두 가지 기법을 제안하였다[1]. TESLA는 MAC(Message Authentication Code)을 이용하여 데이타를 인증하며, MAC을 생성하는 데 사용되는 키 값은 일정 시간 후에 공개된다. 이 기법은 송신 서버의 계산량이 매우 적고, 인증 부가 정보량이 적은 등 상당히 효율적이나 송신 서버와 수신자 사이에 클럭 값이 일정 오차 내로 동기화되어야 하며, 데이타의 전송 지연 시간의 한계값을 모든 수신자가 알고 있어야 하는 등 여러 가지 제약 조건을 가지고 있어 응용 범위가 제한된다. 또한, 이 기법은 부인 방지(non-repudiation)를 제공하지 못한다.

EMSS는 k 번째 패킷의 해쉬값이 $k+1$ 번째 이후의 여러 패킷에 포함되며, 서명 패킷은 그룹 내 마지막 패킷들의 해쉬값들과 이를 서명한 서명값을 가지고 있다 (해쉬 체인 기법을 사용). 이 기법은 매우 간단하며 데이타 유실이 발생하여도 검증 확률이 상당히 높으나, k 번째 패킷의 해쉬값을 가지는 패킷들이 난수로 결정된다. 따라서 이 기법은 결정적(deterministic)이지 못하다. 특히, 이 기법은 받은 패킷이 검증되는 시점을 보장하지 못하며, 최대 무한대까지 지연될 수 있다. 이러한 결점을 보완하기 위하여, 저자들은 수신자가 허락하는 검증 지연 시간의 최대 한계값이 t라고 할 때, 시간 t 동안 패킷들을 두 그룹 혹은 그 이상으로 나누어 패킷이 첫 번째 그룹에서 검증되지 못하면 다음 그룹에서 검증되게끔 제안하였다. 이 방법을 사용하면 시간 t 내 패킷의 검증 확률은 상당히 높아지지만 서명 횟수가 두 배 이상으로 늘어나서 계산량이 훨씬 늘어나게 된다. 같은 논문에서 저자는 IDA(Information Dispersal Algorithm) 기법을 사용한 확장된 EMSS를 제안하였다. 이 기법은 검증 확률이 경우에 따라서 EMSS보다 높지만 역시 위에서 언급한 문제를 그대로 가지고 있으며 계산량 또한 EMSS보다 IDA 기법의 오버헤드만큼 많다.

Golle와 Modadugu는 일반적으로 인터넷에서 패킷 손실이 연속적으로 일어난다는 사실에 주목하였으며, 이를 고려한 GM의 기법을 제시하였다[7]. 이 기법은 해쉬 체인 기법을 기반으로 하며, 송신 서버의 패킷 버퍼 메모리와 해쉬 버퍼 메모리가 제한되어있다고 가정할 때, 최소량의 인증 정보로 최대한의 연속된 패킷 손실을 견딜 수 있게끔 설계하였다. 또한, 이 기법은 결정적인 방법이며 검증 지연시간이 무한정 늦어지지 않으며, 송신 서버의 계산량이 매우 작다. 하지만, 이 기법은 인증 정보량을 조절할 수가 없어서 검증 확률을 원하는 만큼 얻을 수 없다.

3. HORS[5]

본 논문에서 사용하는 용어의 의미는 다음과 같다. $f(x)$는 l 비트 입력/출력 크기를 가지는 일방향 함수(one-way function[4])를 의미하며, $h(x)$는 크기 제한이 없는 입력과 $k\log_2 t$ 비트의 출력값을 가지는 일방향 해쉬 함수(one-way hash function[4])를 뜻한다(k, t는 보안 파라미터이다). 또한, C||D는 데이타 C와 D를 연

결시킨(concatenate) 것을 의미한다. $f(x)$와 $h(x)$는 표준 해쉬 함수로 (SHA-1 혹은 RIPEMD-160) 구현될 수 있다[5].

HORS는 키 생성, 서명 생성, 그리고 서명 검증의 3 가지 모듈로 구성된다. 첫째, 키 생성 모듈은 다음과 같다. 입력 매개변수 l, k, 그리고 t에 대하여, 서명자는 t 개의 l 비트 스트링 $s_1, s_2, \cdots, s_t$를 생성한다. 그 후, 서명자는 $v_i=f(s_i)$를 계산한다 $(1\le i\le t)$. 비밀키와 공개키는 각각 $SK=(s_1, s_2, \cdots, s_t)$, $PK=(v_1, v_2, \cdots, v_t)$이다. 둘째, 서명 생성 모듈은 다음과 같다. 서명될 메시지 m, 비밀키 SK가 있을 때, 서명자는 $h(m)$을 $h_1, h_2, \cdots, h_k$로 나눈다(각 h_j $(1\le j\le k)$는 $\log_2 t$ 비트이다). $h(m)$을 나누는 방식은 여러 가지가 있지만 본 논문에서는 $h(m)=h_1||h_2||\cdots||h_k$를 사용한다[2]. h_j를 정수값 i_j로 해석할 때 $(1\le j\le k)$, 메시지 m에 대한 서명값은 $SIG=(s_{i_1}, s_{i_2}, \ldots, s_{i_k})$이다. 셋째, 서명 검증은 다음과 같다. 메시지 m, 서명값 $SIG=(s_{i_1}, s_{i_2}, \ldots, s_{i_k})$, 그리고 공개키 $PK=(v_1, v_2, \cdots, v_t)$에 대하여, 검증자는 $h(m)$을 $h_1, h_2, \cdots, h_k$로 나눈다. 그는 h_j를 정수 i_j $(1\le j\le k)$로 해석한다. 만일 모든 j에 대해 $(1\le j\le k)$, $f(s_{i_j})=v_{i_j}$를 만족할 때, 그는 서명을 용인(accept)한다.

예 1. 일례로, l=80, k=3, t=8일 경우, 서명자는 비밀키 $SK=(s_1, s_2, \cdots, s_8)$을 생성한다. 이 때, 각 s_i $(1\le i\le 8)$는 80-비트 스트링이다. 그 후, 서명자는 공개키 $PK=(v_1, v_2, \cdots, v_8)$을 생성한다$(v_i=f(s_i)$ $(1\le i\le 8))$. 메시지 m에 대하여, 서명자는 $h(m)=h_1||h_2||h_3$을 계산한 후, 각 h_j를 정수 i_j로 해석한다$(1\le j\le 3)$. 일례로, i_1=2, i_2=3, i_3=5라고 가정하자. 그 후, 서명자는 메시지 m에 대한 서명값 $SIG=(s_2, s_3, s_5)$를 생성한다. 검증 절차는 다음과 같다. 메시지 m, 서명값 $SIG=(s_2, s_3, s_5)$, 그리고 공개키 $PK=(v_1, v_2, \cdots, v_8)$에 대하여, 검증자는 $h(m)=h_1||h_2||h_3$을 계산한 후, 각 h_j를 정수 i_j로 해석한다 $(1\le j\le 3)$. 만일 m이 변경되지 않았다면, i_1=2, i_2=3, i_3=5가 될 것이다. 만일 $f(s_2)=v_2$, $f(s_3)=v_3$, $f(s_5)=v_5$이면 검증자는 서명값을 용인(accept)한다.

4. 개선된 HORS

먼저, 4.1 절에서 HORS를 개선한 제안 기법에 대해 설명한 후, 4.2 절에서 제안된 기법의 계산량을 분석한다. 4.3 절에서 서명 크기, 공개키 크기, 그리고, 비밀키 크기를 분석하며, 4.4 절에서 제안 기법의 안전성에 대하여 분석한다. 마지막으로, 4.5 절에서 기존 기법과 비교 결과를 제시한다.

4.1 HORS를 개선한 제안 기법

HORS에서 공격자가 메시지 m에 대한 서명값 $SIG=(s_{i_1}, s_{i_2}, \ldots, s_{i_k})$를 가지고 있다고 가정하자. 만일 그가 $h(m')=h_2||h_3||h_1||\ldots||h_k$ 혹은 $h(m'')=h_3||h_2||h_1||\ldots||h_k$를 만족하는 m'이나 m''을 발견하면 m'에 대한 서명값 $(s_{i_2}, s_{i_3}, s_{i_1}, \ldots, s_{i_k})$ 혹은 m''에 대한 서명값 $(s_{i_3}, s_{i_2}, s_{i_1}, \ldots, s_{i_k})$을 위조할 수 있다(이 공격은 $h()$가 충돌 내성(collision reisistant) 함수일 때도 가능하다). 본 논문의 주 아이디어는 제한 조건을 추가함으로써 HORS의 이 약점을 보완하는 것이다. 그 결과, 4 장에서 언급하겠지만, 제안된 기법은 HORS는 물론 다른 기존 기법보다 작은 서명 크기를 가짐에도 불구하고 기존 기법들과 거의 같은 수준의 안전성을 가진다.

HORS와 마찬가지로 제안된 기법도 키 생성, 서명 생성, 그리고 서명 검증의 3 가지 모듈로 구성된다. 단, 제안된 기법은 $2|k$인 조건을 가진다. 첫째, 키 생성은 다음과 같다. 서명자는 t 개의 l 비트 스트링 $s_1, s_2, \cdots, s_t$를 생성한다. 그 후, 서명자는 $s'_i=f(s_i)$, $v_i=f(s'_i)$를 계산한다$(1\le i\le t)$. 비밀키와 공개키는 각각 $SK=((s_1, s_2, \cdots, s_t), (s'_1, s'_2, \cdots, s'_t))$, $PK=(v_1, v_2, \cdots, v_t)$이다(단, s_i, s'_j, v_u들은 모두 다른 값을 가져야 한다[3]). 둘째, 서명 생성은 다음과 같다. 메시지 m과 비밀키 SK가 있을 때, 서명자는 임의의 값 c를 선택한 후, $h(m||c)=h_1||h_2||\cdots||h_k$를 계산한다. 각 h_j는 정수값 i_j로 해석된다$(1\le j\le k)$. 만일 i_j $(1\le j\le k)$가 조건 $i_1<i_2<\cdots<i_{k/2}$, $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$를 만족시키지 못하면, 서명자는 c를 다른 값으로 다시 선택한 후, 위 작업을 반복한다. 그 결과, 서명자는 앞에서 언급한 조건을 만족하는 c, i_1, i_2, $\cdots$, i_k를 얻는다. 이 때, 서명값은 $SIG=(c, (s_{i_1}, s_{i_2}, \ldots, s_{i_k}), (s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}))$이다. 셋째, 서명 검증은 다음과 같다. 메시지 m, 서명값 $SIG=(c, (s_{i_1}, s_{i_2}, \ldots, s_{i_{k/2}}), (s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}))$, 그리고 공개키 $PK=(v_1, v_2, \cdots, v_t)$에 대하여, 검증자는 $h(m||c)=h_1||h_2||\cdots||h_k$를 계산한다. 각 h_j는 정수값 i_j로 해석된다 $(1\le j\le k)$. 만일 i_j $(1\le j\le k)$가 조건 $i_1<i_2<\cdots$

2) [5]에서는 $h(m)$을 어떤 방식으로 나누는지 설명하지 않았다. 일반적으로, $h(m)=b_1b_2\ldots b_{k\log t\ (2)}$일 때, P: $\{1, \cdots, k\log t\} \to \{1, \cdots, k\log t\}$인 치환(permutation) 함수 $p()$를 사용하여 $h_i = b_{p(1+(i-1)\log t)}b_{p(2+(i-1)\log t)}\cdots b_{p(i\log t)\ (2)}$ $(1\le i\le k)$로 나타낼 수 있으며, 이 경우에도 HORS 및 제안 기법을 적용시킬 수 있다.

3) 각 s_i, s'_i, v_u는 l 비트 스트링이며, 통상 $l\ge 80$이다. 4.5 절과 [6]에서 선택한 매개변수 값 $t=1024$ 일 때, 이들 s_i, s'_j, v_u $(1<i, j, u\le t)$ 값이 동일할 확률은 극히 낮다.

$<i_{k/2}$, $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j<k)\}=\varnothing$를 만족시키지 못하면, 서명 검증은 실패한다. ($1\le j\le k/2$)에 대하여 $f(f(s_{i_j}))=v_{i_j}$가 만족되고, ($k/2+1\le j\le k$)에 대하여 $f(s'_{i_j})=v_{i_j}$가 만족되면, 검증자는 서명을 용인(accept)한다.

예 2. 일례로, l=80, k=4, t=8일 경우, 서명자는 비밀키 $SK=((s_1, s_2, \cdots, s_8),(s'_1, s'_2, \cdots, s'_8))$을 생성한다. 이 때, 각 s_i, $s'_i(=f(s_i))$ ($1\le i\le 8$)는 80-비트 스트링이다. 그 후, 서명자는 공개키 $PK=(v_1, v_2, \cdots, v_8)$을 생성한다 ($v_i=f(s'_i)$) ($1\le i\le 8$)). 메시지 m에 대하여, 서명자는 임의의 c를 선택한 후, $h(m||c)=h_1||h_2||h_3||h_4$를 계산한다. 각 h_j를 정수 i_j로 해석한 다음 ($1\le j\le 4$), 다음 조건을 만족하는지 검사한다: $i_1<i_2$, $i_3<i_4$, $\{i_1, i_2\}\cap\{i_3, i_4\}=\varnothing$. 만일 이 조건을 만족시키지 못할 경우 다른 c에 대해서 위 작업을 반복한다. 일례로, $c=1001_{(2)}$일 때, i_1=2, i_2=7, i_3=4, i_4=8라고 가정하면, 조건을 만족하는 값이다. 그 후, 서명자는 메시지 m에 대한 서명값 $SIG=(1001_{(2)}, (s_2, s_7), (s'_4, s'_8))$을 생성한다. 메시지 m, 서명값 $SIG=(1001_{(2)}, (s_2, s_7), (s'_4, s'_8))$, 그리고 공개키 $PK=(v_1, v_2, \cdots, v_8)$에 대하여, 검증자는 $h(m||c)=h_1||h_2||h_3||h_4$를 계산한 후, 각 h_j를 정수 i_j로 해석한다. m, c 값이 변경되지 않았다면, i_1=2, i_2=7, i_3=4, i_4=8이 될 것이다. 만일 조건 $i_1<i_2$, $i_3<i_4$, $\{i_1, i_2\}\cap\{i_3, i_4\}=\varnothing$을 만족시키고, $f(f(s_2))=v_2$, $f(f(s_7))=v_7$, $f(s'_4)=v_4$, $f(s'_8)=v_8$이면, 검증자는 서명값을 용인(accept)한다.

4.2 제안 기법의 계산량 분석

본 절에서는 제안된 기법의 계산량을 분석한다. [5]와 마찬가지로, 키 생성, 서명 생성, 그리고, 서명 검증에 대하여 제안된 기법이 필요로 하는 일방 (해쉬) 함수의 호출 횟수를 계산하여 계산량을 분석한다.

키 생성에서는 $f()$를 $2t$ 번 호출한다. 그리고, 서명 검증 시 $h()$를 1 번 호출하고, $f()$를 $3k/2$ 번 호출한다. 서명 생성에서는 $h(m||c)$를 조건 $i_1<i_2<\cdots<i_{k/2}$, $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$를 만족시킬 때까지 호출한다. 평균 호출 횟수를 구하기 위해, 우리는 [5]의 가정처럼 $h()$가 랜덤 오라클 기반 일방 해쉬 함수 [11]라고 하고, i_1, i_2, $\cdots$, i_k가 고른 분포 (uniform distribution)을 가진다고 가정할 때, $i_1<i_2<\cdots<i_{k/2}$, $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$를 만족시키는 확률을 계산한다.

정리 1. i_1, i_2, $\cdots$, i_k가 $[1\sim t]$ 구간에서 고른 분포 (uniform distribution)을 가지는 독립 확률 변수일 때, $i_1<i_2<\cdots<i_{k/2}$, $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, 그리고, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$일 확률은

$$P(k,t)=\frac{t!}{t^k(\frac{k}{2}!)^2(t-k)!}$$ 이다.

증명. i) 우선, 확률 변수 i_1, i_2, $\cdots$, $i_{k/2}$가 $i_1<i_2<\cdots<i_{k/2}$ 조건을 만족시킬 확률을 구하면 다음과 같다. i_1, i_2, $\cdots$, $i_{k/2}$가 조건 없이 선택될 경우의 수는 t 개의 숫자 $0\sim t-1$에서 숫자 $k/2$ 개를 중복하여 선택한 후 선택한 순서대로 나열할 경우 수이며, 이는 $t^{\frac{k}{2}}$이다. 이들 숫자가 $i_1<i_2<\cdots<i_{k/2}$ 조건을 만족시킬 경우의 수는, t 개의 숫자 $0\sim t-1$에서 숫자 $k/2$ 개를 중복하지 않고 선택할 경우 수이며, 이는 $\binom{t}{\frac{k}{2}}$이다. 따라서, $i_1<i_2<\cdots<i_{k/2}$ 조건을 만족시킬 확률은 $\frac{\binom{t}{\frac{k}{2}}}{t^{\frac{k}{2}}}$이다.

ii) i_1, i_2, $\cdots$, $i_{k/2}$가 조건 $i_1<i_2<\cdots<i_{k/2}$를 만족시킬 때, $i_{k/2+1}$, $i_{k/2+2}$, $\cdots$, i_k 값에 대해 생각해보자. $i_{k/2+1}$, $i_{k/2+2}$, $\cdots$, i_k가 아무런 조건이 없이 선택될 경우의 수는 위에서 구한 것과 마찬가지로 $t^{\frac{k}{2}}$이다. 하지만, 조건 $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$을 만족시킬 경우의 수는 $\binom{t-\frac{k}{2}}{\frac{k}{2}}$이다. 따라서, i)이 만족될 때, 조건 $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$을 만족시킬 확률은 $\frac{\binom{t-\frac{k}{2}}{\frac{k}{2}}}{t^{\frac{k}{2}}}$이다.

i), ii)에 의해 조건 $i_1<i_2<\cdots<i_{k/2}$, $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$을 만족시킬 확률은

$$\frac{\binom{t}{\frac{k}{2}}}{t^{\frac{k}{2}}}\frac{\binom{t-\frac{k}{2}}{\frac{k}{2}}}{t^{\frac{k}{2}}}=\frac{\binom{t}{\frac{k}{2}}\binom{t-\frac{k}{2}}{\frac{k}{2}}}{t^k}=\frac{t!}{t^k(\frac{k}{2}!)^2(t-k)!}$$

이다. □

정리 1에 의하여, 서명 생성 시 $h()$의 호출 횟수는 평균 $\frac{1}{P(k,t)}=\frac{t^k(\frac{k}{2}!)^2(t-k)!}{t!}$이다.

제안된 기법에서 메시지 m에 대해 서명을 구할 때, 서명 서버는 조건 $i_1<i_2<\cdots<i_{k/2}$, $i_{k/2+1}<i_{k/2+2}<\cdots<i_k$, 그리고, $\{i_j|(1\le j\le k/2)\}\cap\{i_j|(k/2+1\le j\le k)\}=\varnothing$을 만족시키는 c를 찾는데 대부분의 시간을 사용한다. 이 작업은 비밀

키를 알 필요가 없으므로, 주위의 다른 서버의 힘을 빌려 쉽게 계산할 수 있다. BiBa나 Powerball 또한 병렬/분산 계산이 가능하지만, 이들은 모두 비밀키를 공유해야하는 단점이 있기 때문에, 비밀키 공유를 위한 추가의 오버헤드가 요구되며, 분산 서버 중 하나라도 공격당하면 비밀키가 노출되는 약점을 가진다. 하지만, 제안된 기법에서는 분산 서버 중에 단 1 개의 서버만이 비밀키를 가지므로, 나머지 서버가 공격받더라도 공격자는 키 값을 알 수가 없으므로, 병렬/분산 계산에의한 보안성 저하가 전혀 없다.

4.3 제안 기법의 서명 크기, 공개키 크기, 비밀키 크기

본 절에서는 제안된 기법의 서명 크기, 공개키 크기, 그리고 비밀키 크기를 분석한다. 우선, 제안된 기법의 공개키 크기는 tl 비트이다. 이 때, l의 크기는 $f()$의 출력 크기이며, 표준 해쉬 함수 SHA-1을 사용할 경우 20 바이트이다. 제안된 기법의 비밀키 크기는 $2tl$ 비트이나, tl 비트만으로도 나머지 부분을 계산할 수 있으며 이 작업은 오프라인으로 가능하다 (이것은 Powerball과 동일하다).

제안 기법에서 서명 크기는 $kl+|c|$ 비트이다. l의 크기는 $f()$의 출력 크기이고 c의 크기는 4.2 절에서 설명한 $P(k,t)$와 관련이 있다. 일례로, 4.5 절에서 사용한 매개변수 값 t=1024, k=8인 경우, 약 $P(k,t)$=0.00169이다. 이 경우 c 값을 바꾸어 $h(m||c)$를 계산해 볼 때, 평균 592 번에 한 번씩 올바른 서명값을 찾을 수 있다는 뜻이다. 따라서, 이 경우 $|c|$는 10 비트면 충분하다.

4.4 제안 기법의 안전성 분석

본 절에서는 제안된 기법의 안전성을 분석한다. 우리는 [5, 6]에서의 가정과 같이 $h()$가 랜덤 오라클 기반 일방 해쉬 함수 [11]라고 가정한다. 그러면, a, $b=h(a)$가 주어졌을 때, $h(a')=b$인 a와 다른 a'를 찾을 확률이 $1/2^{klogt}$이다. 또한, [5, 6]에서와 같이 $f(x)$가 주어졌을 때 x를 찾지(역상을 찾지) 못한다는(infeasible) 가정하에서[4] 제안 기법의 안전성을 분석한다.

정리 2. 공격자 A가 메시지 m과 이에 대한 서명값 $SIG=(c,(s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}),(s'_{i_{k/2+1}},s'_{i_{k/2+2}},\ldots,s'_{i_k}))$를 가지고 있을 때, 메시지 m' ($\neq m$)와 이에 대한 서명값 $SIG'=(c',(u_1,u_2,\ldots,u_{k/2}),(u_{k/2+1},u_{k/2+2},\ldots,u_k))$를 위조했다고 가정하자. 그러면,

$\{u_1,u_2,\ldots,u_{k/2}\}=\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$이고

$\{u_{k/2+1},u_{k/2+2},\ldots,u_k\}=\{s'_{i_{k/2+1}},s'_{i_{k/2+2}},\ldots,s'_{i_k}\}$이다.

증명. 공격자 A가 알고 있는 $f(f(y))=v_i$ ($1\le i\le t$)를 만족하는 y 값은 $\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$가 전부이다(그렇지 않으면, A는 $f()$의 역상을 적어도 하나 이상 계산해야하며 이는 4.4 절 처음 부분에서 언급한 가정에 위배된다). 또한, A가 알고 있는 $f(z)=v_i$ ($1\le i\le t$)를 만족하는 z 값은 $\{f(s_{i_1}),f(s_{i_2}),\ldots,f(s_{i_{k/2}}),s'_{i_{k/2+1}},s'_{i_{k/2+2}},\ldots,s'_{i_k}\}$가 전부이다(그렇지 않으면, A는 $f()$의 역상을 적어도 하나 이상 계산해야 한다). 이 때,

$\{u_1,u_2,\ldots,u_{k/2}\}\neq\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$,

$\{u_{k/2+1},u_{k/2+2},\ldots,u_k\}\neq\{s'_{i_{k/2+1}},s'_{i_{k/2+2}},\ldots,s'_{i_k}\}$

이 각각 모순임을 보임으로써 정리 2를 증명한다.

i) $\{u_1,u_2,\ldots,u_{k/2}\}\neq\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$라고 가정하자. 그러면, $u_x\notin\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$인 u_x ($1\le x\le k/2$)가 적어도 1 개 존재한다. SIG'가 m'에 대한 적합한 서명이라고 했으므로 다음과 같은 조건이 만족된다: $h(m'||c')=h'_1||h'_2||\cdots||h'_k$이고 i'_j는 h'_j를 정수값으로 해석한 값일 때 ($1\le j\le k$), $v_{i'_x}=f(f(u_x))$이다. $u_x\notin\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$이고 $f(f(u_x))=v_{i'_x}$이므로, 증명의 처음 부분에서 언급한 것 (A가 알고 있는 $f(f(y))=v_i$ ($1\le i\le t$)를 만족하는 y 값은 $\{s_{i_1},s_{i_2},\ldots,s_{i_t}\}$가 전부이다)과 모순이다. 그러므로, $\{u_1,u_2,\ldots,u_{k/2}\}=\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$이다.

ii) $\{u_{k/2+1},u_{k/2+2},\ldots,u_k\}\neq\{s'_{i_{k/2+1}},s'_{i_{k/2+2}},\ldots,s'_{i_k}\}$라고 가정하자. 그러면, $u_x\notin\{s'_{i_{k/2+1}},s'_{i_{k/2+2}},\ldots,s'_{i_k}\}$인 u_x가 적어도 1 개 존재한다($k/2+1\le x\le k$). SIG'가 m'에 대한 적합한 서명이기 위해서는 다음과 같은 조건을 만족해야 한다: $h(m'||c')=h'_1||h'_2||\cdots||h'_k$이고 i'_j는 h'_j를 정수값으로 해석한 값일 때 ($1\le j\le k$), $v_{i'_x}=f(u_x)$를 만족해야 한다.

$u_x\notin\{f(s_{i_1}),f(s_{i_2}),\ldots,f(s_{i_{k/2}})\}$인 경우와

$u_x\in\{f(s_{i_1}),f(s_{i_2}),\ldots,f(s_{i_{k/2}})\}$인 경우로 나누어 생각해 보자. 전자의 경우,

$v_{i'_x}=f(u_x)$이고

$u_x\notin\{f(s_{i_1}),f(s_{i_2}),\ldots,f(s_{i_{k/2}}),s'_{i_{k/2+1}},s'_{i_{k/2+2}},\ldots,s'_{i_k}\}$이므로 증명의 처음 부분에서 언급한 것에 모순이다. 후자의 경우, $u_x=f(s_{i_z})$인 z ($1\le z\le k/2$)가 존재한다. 모든 s_i ($1\le i\le t$)가 서로 다르고 i)에 의해 $\{u_1,u_2,\ldots,u_{k/2}\}=\{s_{i_1},s_{i_2},\ldots,s_{i_{k/2}}\}$이므로, $s_{i_z}=u_y$인 y가 존재한다($1\le y$

4) 실제, $f()$의 역상을 찾아낼 확률은 매우 낮지만 불가능하지는 않다. 따라서, 본 절에서 분석한 확률보다 약간 높은 확률로 공격자는 위조된 서명을 구할 수 있다. 하지만, 그 차이는 대단히 작으며, 기존 기법인 [5, 6]도 $f()$의 역상을 찾아내지 못한다는 가정 하에서 위조된 기법의 안정성(위조된 서명을 찾을 확률)을 분석했다.

$\le k/2$). SIG'가 적합한 서명이므로, $h(m'||c')=h'_1||h'_2||\cdots||h'_k$이고 i'_j는 h'_j를 정수값으로 해석한 값일 때 $(1\le j\le k)$, $v_{i'_j}=f(u_x)=f(f(u_y))$이다.

따라서, $v_{i'_j}$는 $v_{i_j}=f(f(u_y))$와 같은 값을 가진다. 이 때, 모든 si, s'j, vk (1≤i, j, k≤t)가 서로 다르므로, i'x=i'y이어야 한다. 1≤y≤k/2, k/2+1≤x≤k이므로, 이는 조건 {ij|(1≤j≤k/2)}∩{ij|(k/2+1≤j≤k)}=∅에 모순이다.

따라서, $\{u_{k/2+1}, u_{k/2+2}, \ldots, u_k\}=\{s_{i_{x-1}}, s_{i_{x+2}}, \ldots, s_{i_y}\}$이다. i), ii)에 의하여,

$\{u_1, u_2, \ldots, u_{k/2}\}=\{s_{i_1}, s_{i_2}, \ldots, s_{i_{k/2}}\}$이고

$\{u_{k/2+1}, u_{k/2+2}, \ldots, u_k\}=\{s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}\}$이다. □

정리 3. 공격자 A가 메시지 m과 이에 대한 서명값 $SIG=(c, (s_{i_1}, s_{i_2}, \ldots, s_{i_{k/2}}), (s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}))$를 가지고 있을 때, 메시지 m' $(\neq m)$와 이에 대한 서명값 $SIG'=(c', (s_{j_1}, s_{j_2}, \ldots, s_{j_{k/2}}), (s'_{j_{k/2+1}}, s'_{j_{k/2+2}}, \ldots, s'_{j_k}))$를 위조했다고 가정하자. 이 때, $i_1=j_1 \wedge i_2=j_2 \wedge \ldots \wedge i_k=j_k$이다.

증명. $i_1=j_1 \wedge i_2=j_2 \wedge \ldots \wedge i_k=j_k$는 다음과 같은 두 가지 경우로 나누어 생각할 수 있다.

i) $i_1=j_1 \wedge i_2=j_2 \wedge \ldots \wedge i_{k/2}=j_{k/2}$: 정리 2에 의해 $\{s_{i_1}, s_{i_2}, \ldots, s_{i_{k/2}}\}=\{s_{j_1}, s_{j_2}, \ldots, s_{j_{k/2}}\}$이다. $i_1<i_2<\cdots<i_{k/2}$, $j_1<j_2<\cdots<j_{k/2}$이고, 모든 s_1, s_2, …, s_t가 서로 다르므로 $i_1=j_1 \wedge i_2=j_2 \wedge \ldots \wedge i_{k/2}=j_{k/2}$이다.

ii) $i_{k/2-1}=j_{k/2-1} \wedge i_{k/2-2}=j_{k/2-2} \wedge \ldots \wedge i_k=j_k$: $i_{k/2-1}=j_{k/2-1} \wedge i_{k/2-2}=j_{k/2-2} \wedge \ldots \wedge i_k=j_k$가 아닌 경우를 생각해 보자. $i_{k/2-1}<i_{k/2-2}<\cdots<i_k$, $j_{k/2-1}<j_{k/2-2}<\cdots<j_k$이므로, $i_{k/2-1}=j_{k/2-1} \wedge i_{k/2-2}=j_{k/2-2} \wedge \ldots \wedge i_k=j_k$가 아닐 경우에는 $j_x \notin \{i_{k/2-1}, i_{k/2-2}, \cdots, i_k\}$인 x가 적어도 1 개 이상 존재한다. 이 때, 정리 2에 의해 $s'_{j_x} \in \{s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}\}$이어야 한다. 모든 s'_1, s'_2, …, s'_t가 서로 다르므로 $j_x \notin \{i_{k/2-1}, i_{k/2-2}, \cdots i_k\}$에 모순이며, 따라서, $i_{k/2-1}=j_{k/2-1} \wedge i_{k/2-2}=j_{k/2-2} \wedge \ldots \wedge i_k=j_k$이다.

i), ii)에 의하여, $i_1=j_1 \wedge i_2=j_2 \wedge \ldots \wedge i_k=j_k$이다. □

정리 4. 공격자 A가 메시지 m과 이에 대한 서명값 $SIG=(c, (s_{i_1}, s_{i_2}, \ldots, s_{i_{k/2}}), (s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}))$를 가지고 있을 때, 메시지 m' $(\neq m)$과 이에 대한 서명값 $SIG'=(c', (s_{i_1}, s_{i_2}, \ldots, s_{i_{k/2}}), (s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}))$를 위조한다고 가정하자. 이 때, A가 유효한 SIG'를 찾으려면, 평균 $h()$를 $2^{k\log t}$ 번 계산해야 한다.

증명. SIG'가 유효성을 가지려면 (검증자가 성공적으로 검증하려면) $h(m'||c')=h(m||c)=h_1||h_2||\cdots||h_k$이어야 한다 (만일, $h(m||c)=h_1||h_2||\cdots||h_k$이고 $h(m'||c')=h'_1||h'_2||\cdots||h'_k$일 때, $h_j \neq h'_j$인 h_j가 존재한다고 가정하자. i_j (i'_j)는 h_j (h'_j)를 정수값으로 해석한 값일 때 $i_j \neq i'_j$이다. 따라서, $1\le j\le k/2$이면 $v_{i'_j} \neq (f(f(s_{i_j}))=v_{i_j})$이며, k/2+1≤j≤k이면 $v_{i'_j} \neq (f(s'_{i_j})=v_{i_j})$이어서 SIG'가 유효하지 않기 때문이다).

$m'||c' \neq m||c$ 이므로 4.4 절 처음 부분의 가정에 의하여 공격자 A가 $h(m'||c')=h(m||c)=h_1||h_2||\cdots||h_k$를 만족시킬 m', c'를 찾을 수 있을 확률을 $1/2^{k\log t}$이다. 그러므로, A는 평균 $h()$를 $2^{k\log t}$ 번 계산해야 한다. □

정리 5. 공격자 A가 메시지 m과 이에 대한 서명값 $SIG=(c, (s_{i_1}, s_{i_2}, \ldots, s_{i_{k/2}}), (s'_{i_{k/2+1}}, s'_{i_{k/2+2}}, \ldots, s'_{i_k}))$를 가지고 있을 때, 메시지 m' $(\neq m)$와 이에 대한 서명값 $SIG'=(c', (u_1, u_2, \ldots, u_{k/2}), (u_{k/2+1}, u_{k/2+2}, \ldots, u_k))$를 위조한다고 가정하자. 이 때, A가 유효한 SIG'를 찾으려면, 평균 $h()$를 t_k 번 계산해야 한다.

증명. 정리 2, 3, 4에 의하여 A가 유효한 SIG'를 찾으려면, 평균 $h()$를 $2^{k\log t}=t^{k\log 2}=t^k$ 번 계산해야 한다. □

4.5 기존 기법과 비교 결과

본 절에서는 기존의 일회용 서명 기법과 제안된 기법에 대하여 서명 크기, 검증 연산량, 서명 연산량 그리고, 공개키 크기에 대해 비교한 결과를 제시한다. 표 1은 스트림 인증에 적합한 기존 기법인 BiBa[2], Powerball [6], HORS[5]와 제안 기법을 비교한 결과를 보여주고 있다.

표 1 스트림 인증에 적합한 기존 기법과 비교 결과

기법	서명 크기	검증 연산량	서명 연산량 (off line)	서명 연산량 (on line)	공개키 크기	위조 확률
BiBa	k	$2k+1$	t	$2t$	t	$k!/2t^k$
Powerball	k	$2k+1$	$2t$	$2t$	t	$(k-1)!/2t^k$
HORS	k	$k+1$	t	1	t	$(k/t)^k$
Our scheme	k	$3k/2+1$	$2t$	$\frac{t^k(\frac{k}{2}!)^2(t-k)!}{t!}$	t	$(1/t)^k$

표 2 동일한 안전성을 가지는 조건에서 기존 기법과 비교 결과

기법	서명 크기	검증 연산량	서명 연산량 (off-line)	서명 연산량 (on-line)	공개키 크기
Lamport [4]	80	80	160	1	160
Merkle-Winternitz [12]	23	169	355	1	1
Bleichenbacher-Maurer [13]	45	72	182	1	1
BiBa [2]	11	23	1024	2048	1024
Powerball [6]	10	21	2048	2048	1024
HORS [5]	13	14	1024	1	1024
The proposed scheme	8	13	2048	592	1024

표 1에서 연산량은 일방향 (해쉬) 함수의 수를 나타낸다. 서명 크기, 공개키 크기는 일방향 (해쉬) 함수 단위이며, 만일 출력 크기가 20 바이트인 SHA-1을 사용하는 경우 각 값에 20을 곱하면 바이트 크기가 된다 (단, 위 표에서 제안된 기법, BiBa, 그리고 Powerball의 서명 크기는 c의 크기를 제외한 값이며, 이를 추가하면 제안된 기법은 약 10 비트 정도 더 커지고, Powerball과 BiBa는 약 2 비트 정도 더 커진다).

각 기법에서 k, t 값이 동일한 경우, 제안된 기법은 기존 기법보다 매우 낮은 위조 확률을 가진다. [5]에서의 설정한 값인 t=1024이고 k가 8~13일 때, BiBa는 제안 기법보다 위조 확률이 약 10^5~10^{10} 배 높으며, Powerball은 약 10^4~10^9 배 높다. 통상 위조 확률이 $1/2^{80}$이하정도면 충분함을 생각해 보면 [6], 동일한 안전성을 보장할 때 제안된 기법은 기존 기법보다 서명 크기 혹은 공개키의 크기가 작음을 알 수 있다.

위에서 사용한 비교 방식(k, t 값이 동일한 경우)보다는, 각 기법이 동일한 안정성을 가질 때 서명 크기, 연산량 등을 비교하는 것이 좀 더 합리적이며, [5, 6]에서도 후자의 방법을 택했다. 표 2는 동일한 안전성 (위조 확률: $1/2^{80}$)에서 제안된 기법과 기존 일회용 기법과의 비교 결과를 보여준다. 제안된 기법, Powerball, HORS에서 t 값은 [5]의 설정에 따라 1024로 정하였다.

표 2에 나와있듯이, 제안된 기법은 기존 일회용 서명 기법보다 매우 작은 서명 크기를 가진다. 또한, 제안된 기법은 검증 연산량이 매우 작음을 알 수 있다.

스트림 인증을 위한 기존 기법(BiBa, Powerball, HORS)과 제안된 기법의 온라인 서명 연산량을 비교해 보면, HORS의 서명 연산량보다는 다소 많지만, BiBa나 Powerball보다는 작다. 해쉬 함수 연산이 매우 빠른 점을 생각해보면, 제안된 기법은 서명 서버에 크게 부하를 주지 않음을 알 수 있다. 또한, 4.3 절에서 언급하였듯이, 제안된 기법에서 서명 연산은 쉽게 병렬 처리가 가능하며, 특히 비밀키를 가지지 않는 서버의 도움을 받을 수 있기 때문에, 서명 서버의 부하를 손쉽게 그리고 안전하게 줄일 수 있다.

5. 결 론

본 논문에서는 스트림 데이타를 인증하는데 적합한 개선된 HORS 일회용 서명 기법을 제시했다. 스트림 인증에 일회용 서명 기법을 사용할 경우, 가장 큰 문제점은 큰 서명 크기로 인해 네트워크 오버헤드가 크다는 점이다. 제시된 기법은 기존 기법에 비해 가장 작은 서명 크기를 가진다. 또한, 제시된 기법의 검증 연산량은 매우 작다. 온라인 서명 연산량은 HORS보다 다소 많지만, 기존 기법 중 스트림 인증에 적합한 기법인 BiBa, Powerball보다 매우 작다. 또한, 제시된 기법에서 서명 연산은 쉽게 병렬 처리가 가능하며, 특히 비밀키를 가지지 않는 서버의 도움을 받을 수 있기 때문에, 서명 서버의 부하를 손쉽게 그리고 안전하게 줄일 수 있다.

참 고 문 헌

[1] Adrian Perrig, Ran Canetti, Dawn Song, and J. D. Tygar, Efficient Authentication and Signing of Multicast Streams over Lossy Channels, In Proceedings of IEEE Security and Privacy Symposium, May, 2000.

[2] A. Perrig, The BiBa One Time Signature and Broadcast Authentication Protocol, ACM CCS'01, 2001.

[3] Pankaj Rohatgi, A Compact and Fast Hybrid Signature Scheme for Multicast Packet Authentication, In 6th ACM Conference on Computer and Communication Security, pp. 93 100, November, 1999.

[4] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. Handbook of Applied Crypto

graphy. CRC Press, 1997.
[5] L. Reyzin, N. Reyzin, Better than BiBa: Short One-time Signatures with Fast Signing and Verifying, ACISP'02, 2002.
[6] M. Mitzenmacher, A. Perrig, Bounds and Improvements for BiBa Signature Schemes, Technical Report, 2002.
[7] Philippe Golle and Nagendra Modadugu, Authenticating Streamed Data in the Presence of Random Packet Loss. In NDSS'01, pages 13 22, 2001.
[8] Chung Kei Wong and Simon S. Lam, Digital Signatures for Flows and Multicasts, IEEE/ACM Transactions on Networking, 7(4):502 513, 1999.
[9] Rosario Gennaro and Pankaj Rohatgi, How to Sign Digital Streams. In CRYPTO'97, pages 180 197, 1997.
[10] Ralph C. Merkle, A Certified Digital Signature. In CRYPTO'89. pages 218-238, 1989.
[11] Mihir Bellare and Pillip Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. In 1st Conf. on CCS, ACM, pages 62 73, 1993.
[12] Ralph C. Merkle. A digital signature based on a conventional encryption function. In CRYPTO'87. pages 369 378, 1987.
[13] D. Bleichenbacher and U. Maurer. Directed acyclic graphs, one way functions and digital signatures. In CRYPTO'94, 1994.

박 용 수
1992년~1996년 한국과학기술원 전산학과(학사). 1996년~1998년 서울대학교 컴퓨터공학과(석사). 1998년~현재 서울대학교 컴퓨터공학과 박사과정. 관심분야는 정보보안, 네트워크보안, 암호학

조 유 근
1971년 서울대학교 건축공학과 학사 1978년 미네소타대학교 컴퓨터과학 박사 1979년~현재 서울대학교 컴퓨터공학부 교수. 1984년~1985년 미네소타대학교 교환 교수. 1993년~1995년 서울대학교 중앙교육연구전산원장. 1999년~2001년 서울대학교 공과대학 부학장. 2001년~2002년 한국정보과학회 회장. 관심분야는 운영체제, 알고리즘 설계 및 분석, 암호학