KAIS '97 추계학술대회

전자상거래의 시큐리티 확보 방안

장근녕
부산여자대학교 경영정보학과

요약

전자상거래(Electronic Commerce)를 광범위하게 활용하기 위해서는 시큐리티 확보 방안이 먼저 마련되어야 한다. 전자상거래와 관련된 시큐리티 문제에는 통상적인 정보보안 이외에 메세지 발신자의 신원을 증명하는 신원인증과, 메세지 내용의 완전성 및 정확성을 보장하는 메세지의 완전성 보장, 그리고 메세지의 송·수신을 증명하는 부인방지 등이 있다. 본 논문에서는 전자상거래의 시큐리티 문제를 해결하는데 필요한 메세지 인증과 디지털 서명에 대해 소개하고, 또한 메세지 인증과 디지털 서명의 핵심적인 요소인 해쉬함수를 분석한다.

1. 서론

전자상거래를 활성화하기 위해서는 거래의 안전을 보호하기 위한 방안 마련이 필수불가결하다. 전자상거래에 필요한 시큐리티 서비스의 요건으로는 통상적인 정보보안(information security) 이외에 신원인증, 메세지의 완전성 보장, 부인방지 등을 들 수 있다.

신원인증이란 메세지 발신자의 신원을 증명하는 것으로 이름바 가장(무권한 사칭)이나 playback 또는 replay의 방지가 그 목적이다. 대책으로는 암호화, 패스워드, 디지털 서명(digital signature) 등이 있다. 메세지의 완전성 보장은 메세지 전체가 고의 또는 우연히 개변조되는 것을 방지하는 것이 목적으로, 대책으로는 암호화, 메세지 인증(message authentication) 등을 생각할 수 있다. 부인방지는 메세지가 송신되고 수신되었음을 증명하는 것으로, 메세지의 송·수신자가 송·수신에 대해 부정하는 것을 방지하는 것이 목적이다. 대책으로는 암호화, 디지털 서명, 공증기관 설치 등을 들 수 있다.

본 논문에서는 전자상거래의 안전 유지에 필요한 메세지 인증 및 디지털 서명기법을 소개하고, 이들 기법의 개발에 중요한 역할을 담당하는 해쉬함수(hash function)에 대해 살펴본다.

먼저 2절에서는 메세지 인증과 디지털 서명에 관하여 설명한다. 메세지 인증의 정의와 인증 방법을 설명하고 공개키 암호시스템과 해쉬함수를 이용하는 디지탈 서명기법을 설명한다. 3절에서는 메세지 인증과 디지털 서명에 사용되는 해쉬함수가 가져야 할 요건을 알아보고, 해쉬함수의 기본구조를 살펴본다. 4절에서는 기존의 해쉬함수와 이들에 대한 변조 방법을 살펴본다. 5절에서는 변동 법(molular)을 이용하는 해쉬함수를 제시하고, 마지막으로 6절에서는 결론을 맺는다.

2. 메세지 인증과 디지털 서명

2.1 메세지 인증

메세지 인증이란 둘이상의 정보교류자가 정보를 교환할 때, 수신한 정보가 제 3자에 의해 변조되지 않았는 가를 확인하는 절차이다. 정보교류자는 송신자(sender)와 수신자(receiver)로 구성된다. 송신자는 메세지를 보내는 사람이나 장치이며 수신자는 메세지를 받는 사람이나 장치를 의미한다.

메세지 인증을 통해 ① 메세지가 올바른 송신자에 의해 보내졌는가 (message's

origin), ② 메세지의 내용이 우연히 또는 고의적으로 변하였는가 (message's content), ③ 메세지를 송신된 순서로 받았는가 (message's sequence), ④ 메세지가 의도된 수신자에게 전달되었는가 (message's intended destination) 등을 결정할 수 있어야 한다[13].

그런데 메세지의 발신지, 도착지, 순서에 대한 인증은 메세지의 내용에 포함될 수 있다. 즉 발신지, 도착지를 인증하기 위해서 송신자는 자신과 상대의 ID(Identifier)를 메세지의 내용에 기록한다. 또한 메세지가 블럭별로 보내지는 경우 순서를 인증하기 위해 각각의 메세지 블럭에 보내진 시간을 기록한다. 수신자는 송신자로 부터 받은 메세지의 내용을 인증함으로서 발신지, 도착지, 순서를 인증할 수 있다[15].

메세지 인증에는 해쉬함수 h를 사용하는 방법이 가장 많이 사용된다. 해쉬함수 h는 임의의 크기를 갖는 메세지를 아주 작은 고정된 크기의 해쉬코드로 변환하는 함수이다[18]. 메세지가 변조된 경우 변조된 메세지의 인증값(해쉬코드)과 변조되기 전 메세지의 인증값은 다르게 되고 이를 통하여 변조를 알아낼 수 있다.

메세지 인증은 통산적인 정보보안과 독립적으로 사용된다. 즉 메세지가 암호문(cipher-text)으로 보내지는 경우나 원문(plaintext)으로 보내지는 경우에 상관없이 인증할 수 있는 방법이 필요하다. 송신자 A가 수신자 B에게 메세지를 원문으로 보내는 경우 송신자 A는 원문의 메세지 M을 통신채널(communication channel)을 이용하여 수신자에게 보낸다. 그리고 다른 통신채널을 이용하여 해쉬함수 h에 의해 얻어진 인증값 h(M)을 수신자 B에게 보낸다. 수신자는 A로부터 받은 메세지 M'을 해쉬함수 h에 의해 인증값 h(M')을 구하고 다른 채널을 통해 받은 인증값 h(M)과 비교한다. 이 두 값이 다를 경우 메세지가 변조된 것으로 인식한다.

송신자가 메세지 M을 암호화 알고리즘(encryption algorithm) E_k를 통해 얻은 암호문 $C(=E_k[M])$를 보내는 경우 수신자는 복호화 알고리즘(decryption algorithm)인 D_k를 이용하여 원문의 메세지 $M'(=D_K[C'])$을 얻는다. M'을 해쉬함수 h를 이용하여 인증값 h(M')을 얻고 송신자로 부터 받은 h(M)과 비교하여 메세지를 인증한다.

2.2 디지털 서명

메세지 인증은 수신자로 하여금 메세지가 변조되지 않았다는 것을 확인하게 할 수는 있지만, 제 3자가 앞에서 제시한 4가지 조건들을 확인할 수는 없다. 따라서 송신자는 수신자에 의해 메세지가 변조되었다고 주장할 수 있고, 또한 송신자는 그러한 메세지를 보내지 않았다고 주장할 수있다. 이러한 문제점들은 디지털 서명을 통하여 해결될 수 있다.

디지털 서명은 1976년에 Diffie와 Hellman이 처음으로 소개한 공개키 암호시스템(public key cryptosystem)이나 해쉬함수 등을 이용하여 이루어질 수 있다. 먼저 공개키 암호시스템을 이용하여 메세지 M을 서명하는 경우 사용자 A는 자신의 비밀키인 D_A를 이용하여 서명값 $S=D_A[M]$를 사용자 B에게 보낸다. 수신자 B는 서명값 S를 A의 공개키를 이용하여 서명을 확인한다 ($M=E_A[S]$). 이 경우 수신자나 제 3자는 A의 비밀키인 D_A를 알지 못하므로 메세지 M을 변조할 수 없다. 즉, 수신자가 메세지를 M'으로 변조한 경우 수신자는 변조한 메세지의 서명값인 $S'(=D_A[M'])$을 구할 수 없다. 또한 M에 대한 서명값 S는 A에 의해서만 얻어질 수 있기 때문에 송신자는 자신이 보낸 메세지 M을 부인할 수 없다.

다른 방법으로 해쉬함수를 부가하여 디지털 서명을 할 수 있다. 송신자 A는 메세지 M을 수신자 B의 공개키에 의해 암호화시킨 값 $C=E_B[M]$을 B에게 보낸다. 또한 디지털 서명을 위해 메세지 M을 해쉬함수 h에 의해 변환시키고 자신의 비밀키를 이용하여 서명값 $D_A[h(M)]$을 B에게 보낸다. 수신자 B는 받은 암호문 C를 자신의 비밀키를 이용하여 $M'=D_B[C]$을 구한 뒤 해쉬함수 h에 의해 h(M')을 얻고, 송신자로부터 받은 $D_A[h(M)]$을 A의 공개키에 의해 h(M)을 구한 후 h(M')과 비교한다. 이 두 값을 비교하여 메세지의 변조 여부도 확인할 수 있다.

3. 해쉬함수의 표준화

해쉬함수 h는 임의의 크기(m비트)를 갖는 메세지 M을 고정된 작은 크기의 n비트로 변환하는 함수이다.

$$h: \{0,1\}^m \rightarrow \{0,1\}^n$$

해쉬함수 h는 효율성과 안전성 측면에서 다음과 같은 조건을 만족해야 한다. 첫째, n은 m보다 아주 작은 크기를 가져야 하고, 메세지 M으로부터 해쉬코드 H(=h(M))를 빠른 시간 내에 쉽게 계산할 수 있어야 한다.

둘째, 해쉬함수 h와 해쉬코드 H가 주어졌을 때, H=h(M)을 만족하는 M을 구하는 것이 계산상 불가능(computationally infeasible)하여야 한다. 이 조건을 만족하는 해쉬함수를 약한 일방향성을 갖는 함수(week one-way function)라 한다.

셋째, 메세지 M과 h(M)이 주어졌을때 h(M')=h(M)이 되는 M'(≠M)을 발견하는 것이 계산적으로 불가능하여야 한다. 만약에 H(M')=H(M)인 M'(≠M)을 찾아낸다면 M을 M'으로 대체시킴으로서 메세지를 변조한다. 해쉬함수 h가 $\{0,1\}^m \rightarrow \{0,1\}^n$(m≫n)으로 변환하는 함수이므로 h(M')=h(M)인 M'은 반드시 1개 이상 존재한다. 무작위적으로 메세지 M'을 만들때 h(M')=h(M)일 확률은 2^{-n}이다. 따라서 해쉬함수는 확률적으로 2^{-n}이상의 확률로 해쉬코드가 같은 변조 메세지를 만들 수 없어야 한다. 이 조건을 만족하는 해쉬함수를 강한 일방향성을 갖는 함수(strong one-way function)라 한다.

넷째, 해쉬함수 h가 주어졌을 때, h(M)=h(M')을 만족하는 메세지 M과 M'을 구하는 것이 계산상 불가능하여야 한다. 이 조건을 만족하는 해쉬함수를 충돌저항성을 갖는 함수(collision-resistant function)라 한다.

여기서 첫번째 조건은 해쉬함수의 효율성 조건이라 할 수 있고, 나머지 조건은 해쉬함수의 안전성에 대한 제약이다. 두번째와 세번째 조건은 해쉬코드 H로부터 원래의 메세지 M을 역산 내지는 변조하는 것을 방지하기 위한 제약이고, 네번째 조건은 송신자가 처음에 메세지 M을 보내 놓고 나중에 M'을 보냈다고 주장하는 내부부정을 방지하기 위한 제약으로, 보다 중요시 여겨지는 조건이다.

해쉬함수는 그 사용하는 함수의 특성에 따라 블럭암호 알고리즘을 이용하는 해쉬함수, 전용 해쉬함수, 법(modular) 연산을 이용하는 해쉬함수 등으로 나눌 수 있고, 이에 대한 표준화가 각국에서 이루어지고 있다. 아래에서는 우리나라에서 해쉬함수의 기본구조로 고려하고 있는 내용을 간략하게 소개한다[3].

3.1 블럭암호 알고리즘을 이용하는 해쉬함수

메세지 M을 n비트 크기의 블럭 M_i로 분할한다. 마지막 블럭의 크기가 n보다 작을 경우에는 덧붙이기 방법을 이용하여 n비트의 블럭으로 만든다. 초기값(initial value) IV(H_0)는 변환 함수 u를 이용하여 블럭암호 알고리즘의 키 K_i로 변환한다. M_i와 K_i를 입력으로 하여 암호 알고리즘 e를 수행한 결과값과 M_i를 배타적 논리합한 것을 중간 결과값 H_i로 둔다. H_i를 다시 변환함수 u를 사용하여 변환한 후 메세지 블럭을 다음 입력 값으로 사용하여 반복적으로 수행한다. 해쉬코드 H는 마지막 출력 비트중 원하는 비트만큼 취함으로서 계산된다.

3.2 전용 해쉬함수

길이 L_M인 메세지를 512 비트 블럭의 배수가 되도록 덧붙이기를 하고 그 전체를 512 비트 블럭들로 분할한다. 분할된 첫 번째 메세지 블럭과 초기값을 사용하여 라운드 함수를 수행한 후, 그 결과값과 분할된 다음 메세지 블럭을 다시 입력값으로 사용하여 반복적으로 수행한다. H는 마지막 출력 블럭중 왼쪽의 원하는 비트만큼 취함으로서 계산되어진다.

3.3 법(modular) 연산을 이용하는 해쉬함수

길이 L_M인 메세지를 라운드 함수의 입력을 위해서 길이 L_H인 데이타 블럭들로 변환한다. 메세지의 길이 L_M이 $L_H/2$의 배수가 아니면 덧붙이기를 한다. 메세지의 끝에 길이가 $L_H/2$인 서브 스트링을 부가하는데, 이는 원래 메세지의 길이인 L_M의 이진 표현이다. 위의

모든 블럭의 길이를 $L_H/2$비트에서 L_H비트로 두배 확장한다. 라운드 함수로 초기값과 확장된 메세지의 첫 블럭을 이용하여 길이가 L_H인 블럭을 출력한다. 이 출력값과 확장된 다음 메세지 블럭을 사용하여 반복적으로 계산한다. H는 마지막 출력 블럭의 오른쪽 L_H비트를 취함으로써 계산되어진다.

3.4 메세지 인증 코드

메세지 인증 코드(MAC: Message Authentication Code)는 키를 사용하는 해쉬함수로서, 이는 앞에서 언급된 해쉬함수와 똑같은 성질을 만족한다. 그러나 차이점은 키의 존재 유무에 있다. 그러므로 동일한 키를 가진 자만이 해쉬코드를 검증할 수 있다.

블럭암호를 사용한 메세지 인증 코드 알고리즘은 세가지 부분으로 나누어진다. 메세지 M을 덧붙이기 방법을 사용하여 요구하는 해쉬코드 H의 길이(블럭암호의 블럭의 크기)의 배수가 되도록 한다. 이를 H 길이의 블럭 M_i로 분할한다. 비밀키 K를 사용하여 블럭암호 알고리즘 e(ECB 모드)를 반복 수행한다. 여기서의 최종 출력을 K로부터 유도되는 다른 키를 사용하여 다시 한번 암호화하여 H를 얻는다.

스트림 암호를 사용한 메세지 인증 코드는 비밀키 K와 초기값 IV를 사용하여, 스트림 암호(의사난수 발생기: pseudorandom number generator) 알고리즘을 수행하여 의사난수(키스트림)를 생성한다. 생성된 의사난수와 메세지 M을 비선형 함수(nonlinear function)의 입력값으로 사용한다. 해쉬코드 H는 비선형함수의 최종 출력값에서 원하는 비트 만큼 취함으로써 계산되어 진다.

4. 해쉬함수 분석

이 절에서는 메세지 인증과 디지털 서명에 사용된 몇가지 해쉬함수의 내용과 변조 방법을 소개한다.

4.1 블럭암호 알고리즘을 이용하는 해쉬함수

블럭암호 알고리즘을 이용하는 가장 기본적인 해쉬함수는, CBC(Cipher Block Chaining) 방법과 DES(Data Encryption Standard) 암호화 방법을 이용하여 메세지 M을 64비트 크기의 블럭 $M_1, \cdots, M_p$로 나누고 다음의 연산을 수행한다[5].

$H_0 = IV$

$H_i = E_k(H_{i-1} \oplus M_i)$

여기서 만약 마지막 블럭의 길이가 64비트가 되지 않으면 나머지 부분을 0으로 채워 64비트로 만든다. "⊕"는 배타적 논리합(XOR) 연산을나타내고 IV는 초기값(initial value)을 나타낸다. 암호화 과정 E_k로는 DES 암호화 방법을 사용한다. 동일한 메세지를 인증하는 경우에도 다른 초기값을 사용함으로써 매번 다른 해쉬코드를 가질 수 있다. 암호절차의 마지막 블럭 H_p가 메세지 M의 해쉬코드가 된다.

그런데 메세지 M과 해쉬함수 h를 알고 있는 경우 전방공격법(forward attack)이나 후방공격법(backward attack)을 통해 $h(M')=h(M)$인 변조 메세지 M'을 만들 수 있다.

<전방공격법>

단계 1: 변조 메세지 $M'_1, \cdots, M'_{q-1}$을 만들고 각각에 대해 H'_1, H'_{q-1}을 계산한다.

단계 2: 마지막 블럭 M'_q는 다음의 조건이 만족되도록 만든다.

$M'_q = H_{p-1} \oplus H'_{q-1} \oplus M_p$

이상의 방법으로 구한 변조 메세지 $M'(=M'_1, \cdots, M'_q)$의 해쉬코드 $H(M')$은 $H(M)$과 동일하게 된다.

$$\begin{aligned} H(M') &= E_k[\cdots E_k[E_k(H_0 \oplus M'_1) \oplus M'_2] \\ &\quad \oplus \cdots \oplus M'_q] \\ &= E_k[\cdots E_k(H'_1 \oplus M'_2) \oplus \cdots \oplus M'_q] \\ &\vdots \\ &= E_k(H'_{q-1} \oplus M'_q) \\ &= E_k(H'_{q-1} \oplus H_{p-1} \oplus H'_{q-1} \oplus M_p) \\ &= H(M) \end{aligned}$$

<후방공격법>

2번째 블럭에서 마지막 블럭까지의 의미

있는 변조 메세지 $M'_2, \cdots M'_q$를 만들고, 첫번째 블럭 M'_1는 다음의 절차에 따라 만든다.

단계 1: $M_p \oplus H_{p-1} = M'_q \oplus H'_{q-1}$인 H'_{q-1}를 찾는다.

단계 2: $H'_{q-2} = D_k(H'_{q-1}) \oplus M'_{q-1}$을 계산하여 H'_{q-2}를 찾고 유사한 방법으로 $H'_{q-3}, \cdots H'_1$을 구한다.

단계 3: 첫번째 블럭 M'_1는 $M'_1 = D_k(H'_1) \oplus H_0$으로 만든다.

이상의 방법으로 구한 변조 메세지 $M'(=M'_1, \cdots, M'_q)$의 해쉬코드 $H(M')$은 $H(M)$과 동일하게 된다.

$$\begin{aligned} H(M') &= E_k[\cdots E_k[E_k(H_0 \oplus M'_1) \oplus M'_2] \\ &\quad \oplus \cdots \oplus M'_q] \\ &= E_k[\cdots E_k[E_k(H_0 \oplus D_k(H'_1) \oplus H_0) \\ &\quad \oplus M'_2] \oplus \cdots \oplus M'_q] \\ &= E_k[\cdots E_k(H'_1 \oplus M'_2) \oplus \cdots \oplus M'_q] \\ &\vdots \\ &= E_k(H'_{q-1} \oplus M'_q) \\ &= E_k(H_{p-1} \oplus M_p) \\ &= H(M) \end{aligned}$$

한편, DES의 암호키를 고정된 값으로 하지 않고 각각의 메세지 블럭을 각 단계에서 암호키로 사용하여 인증값을 구하는 기법이 Rabin에 의해 제시되었다[18]. 이 방법은 메세지 M을 56비트 크기를 갖는 p개의 블럭으로 나눈다. 이 경우 각 블럭 M_i는 각 단계에서 DES 알고리즘의 암호키가 된다.

$H_0 = IV$

$H_i = E_{M_i}(H_{i-1})$

i단계에서, 블럭으로 나누어진 메세지 M_i를 DES 키로 사용하여 (i-1)단계에서 구해진 H_{i-1}을 DES 알고리즘에 의해 암호화하여 64비트의 H_i를 구한다. 이 과정을 반복하여 64비트의 해쉬코드 H_p를 구한다.

이 방법으로 해쉬코드를 구하면 전방공격법이나 후방공격법에 의해서 $H(M')=H(M)$이 되는 변조 메세지 M'을 만들 수 없다. 하지만 Birthday Paradox를 이용하여 변조 메세지를 구할 수 있다[5].

이외에 Quisquater와 Girault의 2n비트 해쉬함수[17], Miyaguchi 등의 N-해쉬함수[16], ISO/IEC 10118-2에 소개되어 있는 single 길이 해쉬코드 생성함수(hash-functions providing a single length hash-code)와 double 길이 해쉬코드 생성함수(hash-functions providing a double length hash-code)[2,10] 등이 이 범주에 속한다.

4.2 전용 해쉬함수

ISO/IEC 10118-3에 소개되어 있는 미국에서 표준화한 SHA와 유럽에서 표준화한 RIPEMD 등이 이 범주에 속한다[11]. 참고문헌 [20]과 [1]에도 SHA와 RIPEMD에 대한 내용이 자세히 제시되어 있다.

4.3 법 연산을 이용하는 해쉬함수

법 연산을 이용하는 가장 기본적인 해쉬함수는 임의의 양의 정수인 N을 선택하고 메세지 M을 k비트 크기의 p개의 블럭으로 나눈 후에 다음의 연산을 수행한다[14].

$H_0 = IV$

$H_i = (H_{i-1} \oplus M_i)^2 \bmod N$

i단계에서, 메세지 블럭 M_i와 (i-1)단계의 해쉬코드 H_{i-1}의 배타적 논리합 연산값을 제곱한 후에 N으로 나누어 얻은 나머지 값을 i단계의 해쉬코드로 한다. 이러한 절차를 메세지의 마지막 블럭까지 반복하여 얻은 값 H_p를 M에 대한 해쉬코드로 한다.

그러나 이 해쉬함수의 경우 앞에서 언급한 전방공격법에 의해 변조 메세지를 만들 수 있다. 즉, 이 방법으로 구한 변조 메세지 $M'(=M'_1, \cdots, M'_q)$의 해쉬코드 $H(M')$은 $H(M)$과 동일하게 된다.

$$\begin{aligned} H(M') &= (H'_{q-1} \oplus M'_q)^2 \bmod N \\ &= (H'_{q-1} \oplus H_{p-1} \oplus H'_{q-1} \oplus \\ &\quad M_p)^2 \bmod N \\ &= (H_{p-1} \oplus M_p)^2 \bmod N \\ &= H(M) \end{aligned}$$

Davies와 Price는 전방공격법으로 변조

메세지를 만들 수 없도록 위의 방법을 확장하여 다음의 방법을 제시하였다[7].

단계 1: 512비트 크기의 양의 정수 N을 선택한다.
단계 2: 메세지 M을 448비트 크기를 갖는 p개의 블럭 $M_1, \cdots, M_p$로 나눈다. 그리고 각각의 블럭 M_i의 왼쪽에 64비트를 0으로 채워 넣어 512비트 크기로 만든다.
단계 3: 다음 식을 이용하여 M에 대한 해쉬코드 $h_p = h(M)$을 구한다.

$$H_0 = IV$$
$$H_i = (H_{i-1} \oplus M_i)^2 \bmod N$$

이상의 방법으로 해쉬함수를 설계하면 전방공격법에 의해 M을 변조할 수는 없다. 그러나 Euclidian 알고리즘을 이용하면 이 방법을 공략할 수 있다.

이외에 Girault[8] 등이 제시한 해쉬함수와 ISO/IEC 10118-4에 소개되어 있는 MASH-1 및 MASH-2 [12] 등이 이 범주에 속한다.

5. 변동 법을 이용하는 해쉬함수

여기에서는 변동 법을 이용하는 해쉬함수를 제시한다. 메세지를 각 블럭이 k비트의 크기를 갖도록 나눈 후에, 각 블럭 M_i를 확장한다. 그리고 확장된 수를 다시 k비트 크기를 갖는 블럭으로 나누어 블럭중에 하나를 법으로 사용한다. 따라서 법으로 사용되는 값은 메세지의 내용에 따라 다른 값을 갖게 된다. 이와 같은 방법으로 해쉬함수를 설계하는 경우 안전성을 보다 높일 수 있을 것으로 생각된다.

각 블럭 M_i를 확장하는 방법으로 여러 가지를 고려할 수 있다. 본 논문에서는 제곱을 하는 방법과 Fibonacci 수를 이용하는 방법을 사용한다.

Fibonacci 수는 다음과 같이 정의된다[6].

$$F_0 = 0$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}, \quad k \geq 2$$

따라서 Fibonacci 수는 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, … 등으로 표현된다. 즉, 각 수는 앞의 두 수의 합으로 표현된다. 2개의 연속적인 수 (F_k, F_{k+1})은 $g=(1+\sqrt{5})/2$의 비율로 접근하여 간다. 따라서 64비트의 이진수를 표현하기 위해서는 92개의 Fibonacci 수가 필요하다.

모든 양의 정수 x는 Fibonacci 수에 의해 유일하게 표현할 수 있다.

$$x = F_{k_1} + \cdots + F_{k_t}, \quad k_1 \gg \cdots \gg k_t \gg 0$$

여기서 $t \geq 0$이고, $k_i \gg k_j$는 $k_i - k_j \geq 2$을 의미한다. 0보다 큰 정수 x를 Fibonacci 수에 의해 표현하기 위해서는 먼저 $F_{k_1} \leq x < F_{k_1+1}$인 k_1을 선택한다. 그러면 $x - F_{k_1} < F_{k_1+1} - F_{k_1} = F_{k_1-1}$ 식이 만족된다. 다시 $F_{k_2} \leq x - F_{k_1} < F_{k_2+1}$인 k_2를 찾는다. 이를 계속적으로 반복하면 위의 식을 얻을 수 있다.

제곱하여 메세지를 확장하는 경우의 해쉬함수 VMHA-1을 정리하면 다음과 같다.

단계 0: $H_0 = IV$, $i=1$.
단계 1: M_i를 제곱하여 앞의 k비트를 M_i^1, 뒤 k비트를 M_i^2라 한다.
단계 2: $H_i = (((H_{i-1} \oplus M_i^1)^2 \bmod M_i^2) \oplus H_{i-1})$.

Fibonacci 수를 이용하여 확장하는 경우의 해쉬함수 VMHA-2를 정리하면 다음과 같다.

단계 0: $H_0 = IV$, $i=1$.
단계 1: M_i를 Fibonacci 수로 표현한다.

$$M_i = F_{i_1}, \cdots, F_{i_t}, \quad F_{i_1} \gg \cdots \gg F_{i_t}$$

단계 2: 확장된 메세지 블럭 $(F_{i_1}, \cdots, F_{i_t})$를 k비트 크기의 블럭 $X_i^1, \cdots, X_i^L$로 나누고, 마지막 블럭이 k비트가 되지 않으면 제거한다.
단계 3:
$L > 1$이면,

$$H_i = (((H_{i-1} \oplus X_i^1 \oplus \cdots \oplus X_i^{L-1})^2 \bmod X_i^L) \oplus H_{i-1}),$$

L = 1이면,

$H_i = (H_{i-1} \oplus X_i^1)$.

제시된 VMHA-1과 VMHA-2의 안전성을 평가하기 위해 메세지내의 한개 문자의 변화와 한개 문자의 삽입이 해쉬코드에 미치는 영향을 측정하였다. 실험에서는 임의로 작성한 3,392비트 크기의 메세지를 이용하였고, 메세지 블럭과 해쉬코드는 각 64비트로, 초기값 IV는 0으로 정하였다. 실험 결과 VMHA-1의 경우에는 해쉬코드의 약 35%가 바뀌고, VMHA-2의 경우에는 약 50%가 바뀌는 것으로 나타났다. 물론, 이러한 결과가 해쉬함수의 안전성을 보장하지는 않는다. 한편, 해쉬코드 생성에 드는 시간은 예상대로 VMHA-2의 경우가 상대적으로 많았다.

6. 결론

본 논문에서는 전자상거래의 시큐리티 확보에 필요한 메세지 인증 및 디지털 서명 구현에 중요한 역할을 담당하는 해쉬함수를 분석하고, 새로운 해쉬함수를 제시하였다. 다만 제시된 해쉬함수의 경우 그 안정성과 효율성에 대한 보다 면밀한 검토가 추가되어야 할 것으로 생각된다.

참고문헌

[1] 김철, 안금혁, 염창선, "ISO/IEC JTC1/SC27의 해쉬함수에 대한 조사 연구," 통신정보보호학회지, 5권, 2호, 1995, PP.74-89.

[2] 이필중, "ISO/IEC JTC1/SC27의 국제표준소개(11): ISO/IEC IS 10118-2," 통신정보보호학회지, 6권, 1호, 1996, PP.79-88.

[3] 한국전산원, 디지털 서명을 위한 해쉬함수의 표준화 연구, 1995.

[4] 황석근, 조한혁, "디지털 서명과 해쉬함수," 통신정보보호학회지, 2권, 1호, 1992, PP.23-29.

[5] S.K. Akl, "On the Security of Compressed Encoding," Crypto'83, 1983, PP.209-230.

[6] D.M. Burton, Elementary Number Theory, Wm. C. Brown Publisher, 1989.

[7] D.W. Davies and W.L. Price, "The Application of Digital Signatures based on Public Ket Cryptosystem," International Conference on Computer Communication, 1980.

[8] M. Girault, "Hash-Functions using Modulo-n Operation," Eurocrypto'87, 1987, PP.218-226.

[9] ISO/IEC 10118-1: Information Technology - Security Techniques - Hash Functions - Part 1: General, 1994.

[10] ISO/IEC 10118-2: Information Technology - Security Techniques - Hash Functions - Part 2: Hash Functions using an n-bit Block Cipher Algorithm, 1994.

[11] ISO/IEC 10118-3: Information Technology - Security Techniques - Hash Functions - Part 1: Dedicated Hash Function, 1996.

[12] ISO/IEC 10118-4: Information Technology - Security Techniques - Hash Functions - Part 4: Hash Functions using Modular Arithmetic, 1996.

[13] R.R. Jueneman, S.M. Matryas and C.H. Meyer, "Message Authentication with Manipulation Detection Codes," Symposium on Security and Privacy, 1983, PP.33-54.

[14] R.R. Jueneman, S.M. Matryas and C.H. Meyer, "Message Authentication," IEEE Communications Magazine, Vol.23, 1985, PP.29-40.

[15] C.H. Meyer and S.M. Matyas, Cryptography: A New Dimension in Computer Data Security, John Wiley and Sons, 1982.

[16] S. Miyaguchi, M. Iwaia, and K. Ohta, "New 128-bit Hash Function," International Joint Workshop on Computer and Communication, 1989,

PP.279-288.

[17] J.J. Quisquater and M. Girault, "2n-bit Hash Function using n-bit Symmetric Block Cipher Algorithm," In Abstracts of Eurocrypto'89, 1989.

[18] R.L. Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Crptosystems," Communication of the ACM, Vol.21, 1978, PP.120-126.

[19] http://203.254.87.2/~media/ec/data/ec21.htm.

[20] http://csrc.ncsl.nist.gov/fips/fip180-1.txt.

[21] http://csrc.ncsl.nist.gov/fips/fip186.txt.