MHS에서 디지탈다중서명 설계 및 구현

°강은영 조성제 남길현
국 방 대 학 원

Design and Implementation of Digital Multisignature for Message Handling System

Eun-young Kang, Seong-Je Cho and Kil-Hyun Nam
National Defense University

요 약

현재 운용되고 있는 대부분의 MHS는 송신자가 수신자에게 메세지를 전달하는 과정에서 적절한 보안서비스를 제공하지 못하고 있기 때문에 민감하거나 비밀스런 자료에 적합하지 못하며, 특히 여러사람이 문서에 서명을 해야 하는데는 문제점을 안고 있다. 본 연구에서는 네트워크상에서 송신자의 신분인증과 메세지 인증뿐만 아니라 전자문서를 일반문서와 같이 처리할 수 있는 디지탈다중서명을 MHS에서 구현할 수 있는 방안을 제시하였다.

1. 서 론

종이없는 사무환경에서 전자문서(Electronic document)는 일반 문서와 같은 기능을 대신할 수 있다. 이러한 전자문서에 인감도장과 같은 역할을 할 수 있는 기능과 송신자의 문서에 대한 부인, 수신자의 문서 내용의 변경, 그리고 제3자의 문서의 위조 등 분쟁요인을 해결할 수 있는 암호응용 기술이 디지탈서명이다[1].

대부분의 사무 환경은 계층적인 구조로 이루어졌으며, 기안자는 문서를 작성하여 다른 부서의 협조 및 통제를 받은 후에 상급자의 결심을 위한 계층적인 다단계 서명이 필요하다. 또한 수입 승인을 위해서 은행, 무역업체, 세관 등의 기관 혹은 업체간의 연속적이고 계층적인 서명이 필요하다. 기존의 디지탈서명은 쌍방간에 한번의 서명만으로 이루어지는 단일 서명이 대부분이지만 실제적인 사무환경에서 전자문서에 디지탈서명이 이루어지려면 디지탈다중서명 기술이 필요하다.

2. MHS UA에서 보안서비스 구현 방안

2.1 UA의 기능 및 보안서비스

UA는 MHS 사용자에게 이용자 접속(interface)기능을 제공하기 위한 것이며, 사용자는 UA를 통해서 다른 사용자에게 보낼 메세지를 작성, 수정, 편집 할 수 있고, 상대방으로 부터 받은 메세지를 읽어 볼 수 있다. 또한 UA는 내부적으로 MTA와의 접속을 통하여 MTA로 메세지를 제출하거나 MTA로부터 메세지를 받아 Mail Box에 보관 되며, Mail Box는 작성한 메세지의 보관을 위한 OUTBOX 및 HOLDBOX, 수신한 메세지의 보관을 위한 MBOX, INBOX가 있다. UA간의 메세지 교환은 IMP(Interpersonal Messaging Protocol(P2))에 따르며, 모든 메세지는 X.409원칙에 따라 암호화된 형태로 주고 받는다.

MHS에서 필요로 하는 보안서비스는 ISO/IEC/JTC1/SC21에서 발표한 OSI 보안구조(ISO 7498-2)내에 정의된 신분확인(Authentication/Identification), 접근제어, 데이타 무결성, 데이타 비밀성, 부인봉쇄(Non-Repudiation) 서비스 등이 있다.

2.2 UA에서 디지탈다중서명 구현방안

MHS에서 디지탈 다중서명을 구현하는 방안으로는 다음 세경우로 나누어 볼 수 있다.

① MHS에서 비밀키 생성과 메세지 송·수신 기능만 담당하고, PC에서 메세지 작성, 서명 등의 기능을 수행하도록 한다.

② MHS에서 비밀키 생성, 메세지 작성, 디지탈 서명, 메세지 암호화 등의 기능을 담당하고, PC는 단지 MHS 시스템의 단말기능을 수행하도록 한다.

③ Center에서 비밀키를 생성하고 PC에 MHS Server와 같은 자체 처리기능을 갖도록 하여 메세지 작성, 디지탈 서명, 메세지 암호화 등의 기능을 할 수 있도록 한다.

첫번째 방법은 PC에서 모든 기능을 처리하므로 UA는 단지 메세지 전달 기능만을 한다. 따라서 MHS의 기능을 거의 이용하지 않는다. 두번째 방법은 MHS의 기능을 충분하게 이용할 수 있으나 실제 모든 처리가 Server에서 이루어지므로 통신로상에서 키, 랜덤정보, 서명정보 등이 노출되어 이에 대한 부가적인 보안이 필요하다. 세번째 방법은 PC 자체에 MHS Server의 UA와 같은 기능을 할 수 있도록 하는 RUA (Remote UA)에서 모든 작업을 수행하는 방법이다. 즉, PC 자체가 UA의 처리 기능을 하도록 하는 방법이다. 세가지 방안중에서 세번째 구현방안이 MHS의 고유기능을 최대한 활용하면서 비밀키를 가장 안전하게 이용하여 디지탈 서명을 실시할 수 있는 방안이나 Server와 PC간에 상호 연동할 수 있는 프로토콜의 개발이 선행되어야 한다.

위의 세가지 방안중 본 논문에서는 두번째 구현방안을 중심으로 하여 시스템을 설계하여 MHS에서 디지탈 다중서명이 수행되도록 시스템을 구성한다. 그리고 비밀키의 저장은 IC카드를 이용하는 방안이 안전하다고 생각하나 IC카드의 처리능력과 메모리의 부족 등을 고려하여 프로토타입 형태로 비밀키만을 디스켓에 암호화된 상태로 저장하여 사용한다. 따라서 암호화된 비밀키는 Server에서 복호화하여 사용해야 한다.

3 디지탈다중서명 기법 고찰

3.1 Brickell-Lee-Yacobi의 다중서명 방식

Brickell, Lee, Yacobi는 Fiat-Shamir의 서명방식을 변형하여 원격회의를 위한 n-Party 식별 및 무순차 다중서명 방식을 제안했다[4]. 이 방식의 가정사항은 각 서명자들이 bridge에 접속되어 있어야 하며, 서명할 메세지 M과 서명자들의 리스트 IDcm = $ID_1 \| ID_2 \| \cdots \| ID_m$을 가지고 있어야 한다. 또한 bridge는 각 서명자들로부터 서명정보를 수신하면 서명정보를 곱한 후에 모든 서명자들에게 동보 전송하는 기능을 가져야 한다.

● 키 생성 및 분배 절차

(1) 두개의 큰 소수 p,q를 선택하여 비밀리에 유지하고 N=p×q를 공개한다.

(2) $I_{ij} = f(ID_i\ j)$ $(i=1,2,\cdots,m,\ j=1,2,\cdots,t\times k)$와 $I_{ij}^{-1} = S_{ij}^2 \mod N$ 를 계산한다.
여기서 t,k는 보안변수로서 k는 해쉬함수의 출력비트 수를, t는 k의 반복회수를 나타낸다. I_{ij}^{-1}는 법 N에 대하여 이차잉여인 집합 전체를 의미하고
j는 편의상 $j=1,2,\cdots,k$로 표기한다.

(3) $(N,f,h,\ S_{i1},S_{i2},\cdots,S_{ik})$가 기록된 IC카드를 발급 배포한다.

● 다중서명 생성 절차

(1) 각 서명자 i는 랜덤수 R_i를 선택하고 X_i를 계산하여 bridge로 전송한다.
$X_i = R_i^2 \mod N \quad (i=1,2,\cdots,m)$

(2) bridge는 $X = X_1 \cdot X_2 \cdots X_m \mod N$를 계산하여 서명자들에게 동보 전송한다.

(3) 각 서명자는 bridge로부터 X를 수신하면 서명정보(Y_i)를 bridge로 전송한다.
$(e_{i1},e_{i2},\cdots,e_{ik}) = h(M,\ IDcm,\ X)$

$$Y_i = R_i \prod_{ej=1}^{k} S_{ij} \mod N \quad (i=1,2,\cdots,m,\ j=1,2,\cdots,k)$$

(4) bridge는 $Y = Y_1 \cdot Y_2 \cdot \cdots \cdot Y_m \mod N$을 계산하여 서명자들에게 동보 전송한다.

● 다중서명 검증 절차

(1) 각 서명자는 IDcm으로부터 I_{ij}를 계산한다. $I_{ij} = f(ID_i, j)$ (i=1,2,⋯,m, j=1,2,⋯,k)

(2) 각 서명자는 $(e_{i1},e_{i2},\cdots,e_{ik}) = h(M,IDcm,X)$ (i=1,2,⋯,m) 을 계산한다.

(3) 각 서명자는 Z를 계산하여 Z = X 이면 다중서명은 유효한 것으로 간주한다.

$$Z = Y^2 \prod_{i=1}^{m} \prod_{eij=1}^{k} I_{ij} \mod N \quad (j=1,2,\cdots,k)$$

3.2 Ohta-Okamoto 다중서명 방식

Ohta, Okamoto는 Brickell-Lee-Yacobi의 무순차 다중서명 방식을 변형하여 새로운 순차 다중서명 방식을 제안했다[5].

● 키 생성 및 분배 절차 : Brickell-Lee-Yacobi 방식의 키 생성 및 분배 절차와 같다.

● 다중서명 생성 절차

(1) 공통키 생성 단계

1) 서명자 1 : 서명할 사람의 순서를 결정하고 랜덤수 R_1를 선택하고

$X_1 = R_1^2 \mod N$을 계산하여 다음 서명자에게 X_1을 전송한다.

2) 서명자 n : 앞 서명자로 부터 X_{n-1}을 수신하면 랜덤수 R_n를 선택하고

$X_n=R_n^2 X_{n-1} \mod N$ 을 n+1서명자에게 전송한다.

(단 마지막 서명자 m은 기안자에게 전송한다.)

(2) 서명 생성 단계

기안자(n=1)는 서명자의 IDcm = $ID_1 \| ID_2 \| \cdots \| ID_m$을 구성한다.

1) 서명자 n-1로부터 서명메세지 $(M,IDcm,X_m,Y_{n-1})$을 수신하면

$(e_{n1},e_{n2},\cdots,e_{nk}) = h(M, IDcm, X_m)$와

$Y_n = Y_{n-1} \cdot R_n \prod_{enj=1}^{k} S_{nj} \mod N$ (j=1,2,⋯,k)을 계산한다. (단 $X_0,Y_0=1$ 이다)

2) 서명자 n은 서명정보 $(M,IDcm,X_m,Y_n)$를 ID_{n+1}에게 전송한다.

3) 마지막 서명자(n=m)이면 $(M,IDcm,(e_{m1},e_{m2},\cdots,e_{mk}),Y_m)$을 검증자에게 전송한다

● 다중서명 검증 절차

(1) $I_{ij} = f(ID_i, j)$ (i=1,2,⋯,m , j=1,2,⋯,k)

(2) $Z_m = Y_m^2 \prod_{i=1}^{m} \prod_{eij=1}^{k} I_{ij} \mod N$ 을 계산하여 $Z_m = X_m$ 이면 서명이 유효하다.

3.3 K-K 다중서명 방식

강창구,김대영은 Fiat-Shamir방식에 근거하고 있는 Ohta-Okamoto 방식에서 공통키 생성단계를 서명생성 단계에 포함되도록 변형한 순차 다중서명 방식을 제안하였다[6].

● 키 생성 및 분배 절차 : Brickell-Lee-Yacobi 방식의 키 생성 및 분배 절차와 같다.

● 다중서명 생성 절차

기안자(n=1)는 서명자의 IDcm = $ID_1 \| ID_2 \| \cdots \| ID_m$을 구성한다.

(1) 서명자 n은 랜덤수 R_n을 선택한 후

$X_n = R_n^2 \cdot X_{n-1} \mod N$, $(e_{n1},e_{n2},\cdots,e_{nk}) = h(M, IDcm, X_n)$

$Y_n = Y_{n-1} \cdot R_n \prod_{enj=1}^{k} S_{nj} \mod N$ (j=1,2,⋯,k) 을 계산한다. (단 $X_0,Y_0=1$ 이다)

(2) 서명자 n은 서명정보 $(M,IDcm,X_1,\cdots,X_n,Y_n)$을 ID_{n+1}에게 전송한다.

(3) 마지막 서명자(n=m)이면 서명정보 $(M,IDcm,X_1,\cdots,X_m,Y_m)$를 검증자에게 전송한다.

● 다중서명 검증

(1) 각 서명자의 $(e_{i1},e_{i2},\cdots,e_{ik}) = h(M,IDcm,X_i)$ (i=1,2,…,n-1)를 계산한다.
$(e_{11},e_{12},\cdots,e_{1k}),(e_{21},e_{22},\cdots,e_{2k}),\cdots,(e_{(n-1)1},e_{(n-1)2},\cdots,e_{(n-1)k})$

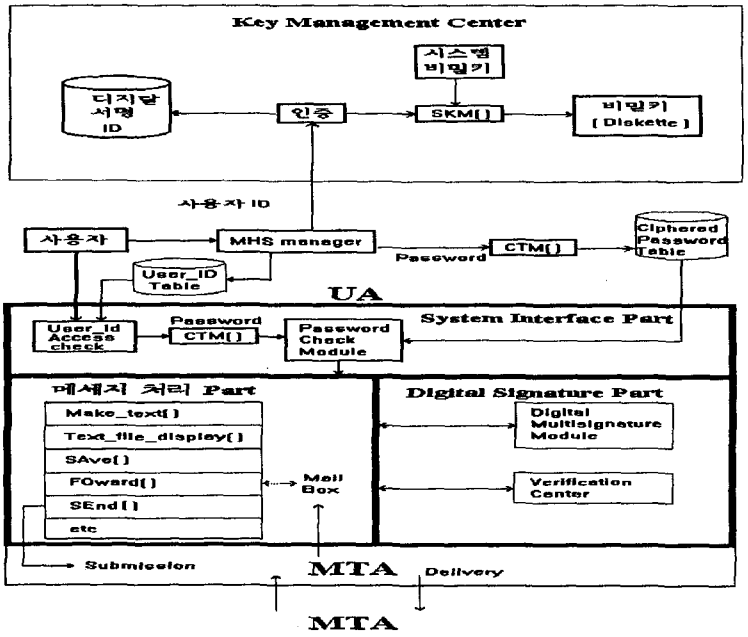
(2) 앞 서명자의 공개키를 계산한다. $I_{ij}=f(ID_i, j)$ (i=1,2,…,n-1, j=1,2,…,k)

(3) $Z_{n-1}=Y_{n-1}^2 \prod_{i=1}^{n-1} \prod_{enj=1}^{k} I_{ij} \bmod N$을 계산하여 $Z_{n-1}=X_{n-1}$이면 서명이 유효하다.

4 MHS의 UA에서 디지탈다중서명 설계 및 구현

4.1 MHS UA에서 디지탈 다중서명 설계

본 연구에서는 UA의 기존 메세지 편집 및 저장 기능을 그대로 유지하면서 디지탈다중서명이 수행될 수 있도록 시스템을 설계하는데 중점을 두었다.



<그림 1> MHS에서 디지탈 다중서명 구현을 위한 시스템 구성

디지탈다중서명을 위한 MHS UA의 구성은 <그림 1>과 같이 크게 시스템 접속, 메세지 처리 그리고 다중서명 세부분으로 구성되어 있다. 그리고 비밀키의 생성을 위한 MHS 중앙에 별도의 키관리 센타가 필요하다. 본 연구에서는 실행가능 명령어 중에서 메세지 처리에 관련된 5가지 주요 명령어(Compose, Send, Read, Forward, Save)를 중심으로 Mail Box내의 메세지 흐름에 따르는 디지탈 다중서명이 가능하도록 시스템을 구성하였다. 사용자 접근제어와 패스워드 확인모듈에서 적법한 절차에 의한 정당한 사용자임이 확인이 되면 MHS의 UA에서 제공하는 기능을 이용하여 메세지 생성, 메세지 송·수신 등을 할 수 있으며 필요시 디지탈 다중서명을 할 수 있다.

4.1.1 키관리 센타(Key Management Center)

본 연구에서는 공개키로 ID를 사용하고, 센타의 비밀정보를 이용하여 생성한 비밀키를 사용자에게 분배한다. 따라서 키관리 센타는 신뢰할 수 있어야 한다. 키관리 센타는 MHS 관리자에 의해 인증을 거친 사용자가 디지탈서명 서비스를 필요로 하는 사용자의 ID를 제출하면 비밀키 생성모듈(SKM :Security Key Module)에서 비밀키를 생성하여 디스켓에 저장하여 발급하고, 사용자의 ID를 공개키로 사용하기 위해 데이타베이스에 기록한다.

4.1.2 시스템 접속(System Interface Part)

MHS 관리자는 신규 가입자에 대한 User_ID, 암호 패스워드, 등록일자 등을 각각의 테이블에 기록하여 관리한다. 암호화된 패스워드는 MHS 관리자에게 사용자가 제출한 패스워드를 일방향 함수인 암호 패스워드 테이블 모듈(CTM:Ciphered password Table Module)에 의해 암호화된 상태로 패스워드 테이블에 기록된다.

사용자는 MHS에서 제공하는 접근제어를 통하여 시스템에 접근하게 된다. 접근제어는 단지 사용자의 User_ID만을 검사하고, 이 검사가 통과되면 시스템 인터페이스는 일방향 함수의 출력값과 패스워드 확인 모듈에서 암호화된 패스워드 테이블에 등록되어 있는 값을 비교, 확인하여 정당한 사용자에 대하여 MHS에 접속할 수 있도록 제어하는 기능을 한다.

4.1.3 메세지 처리

메세지 처리 부분은 시스템 인터페이스에서 정당한 권한자로 확인된 사용자가 MHS 메세지 처리기능을 이용하여 메세지의 작성, 수정, 삭제, 저장 ,전송을 수행할 수 있으며, 디지탈서명 서비스가 필요할 때에는 정당한 디지탈서명을 할 수 있는 자격이 있는지를 ID 데이타베이스를 검사하여 등록된 사용자는 다중서명 모듈을 이용하여 디지탈서명을 할 수 있으며, 서명정보는 메세지에 부가하여 송신한다.

4.1.4 디지탈서명(Digital Signature Part)

메세지 처리 부분에서 정당한 디지탈서명을 할 수 있는 사용자는 디지탈서명이 필요한 문서에 대해서 다중서명 모듈에 의해서 디지탈다중서명을 할 수 있으며, 수신자는 앞 서명자의 다중서명이 정당한가를 검증센타에 의뢰하여 검증할 수 있다.

4.2 디지탈다중서명 구현

4.2.1 키관리 센타

Fiat-Shamir방식을 응용한 디지탈다중서명의 비밀키 생성 알고리즘은 첫째, 소인수 분해 공격에 강한 256비트 두개의 소수(p,q) 생성, 둘째, 각 사용자의 공개키를 얻기위한 ID에 대한 해쉬함수 계산, 셋째, 사용자 ID의 해쉬값에 대한 역원(Inverse) 생성, 넷째, 르장드르 기호(Legendre Symbol)를 이용하여 그 역원의 이차잉여 존재여부를 판단, 다섯째, 합성수(N=pq)에 대한 이차잉여를 구하기 위해서 중국인의 나머지 정리(Chinese Remainder Theorem)[10]을 이용하여 비밀키를 생성하였다.

● 소수 생성

먼저 소수를 생성하기 위해서는 예측 불가능한 랜덤수를 생성해야 한다. 난수 생성은 시스템 시간을 이용하여 생성하였고, 이 수가 소수인지를 판별하기 위해서 확률론적 소수판별 알고리즘을 이용하여 구현하였다. 만일 n이 소수이고 $(n-1=2^k t)$로 표시할때(t는 홀수), 양의 정수 $a(0<a<n)$에 대해 $a^t \equiv 1 \bmod n$ 혹은 $0<i<k$인 i에 대해 $a^w \equiv \pm 1 \bmod n$ ($w = t \times 2^i$)이 성립된다는 것이 Miller에 의해 증명되었다[7]. 이 사실을 이용한 소수 판별법을 Miller-Rabin 소수 판별법이라 한다. 이것을 이용한 Knuth의 알고리즘이 <그림 2>에 있다[9]. 이 알고리즘 테스트에 합성수인 홀수 n이 패스할 확률은 1/4보다 작다고 알려져 있으므로 r번의 랜덤하게 선

택된 수 a에 대해 합성수 n이 패스할 확률은 $1/4^r$ 보다 작다. 그러므로 이 테스트를 통과한 정수 n은 소수일 확률은 크지만 이것이 소수임을 증명하는 것은 아니다.

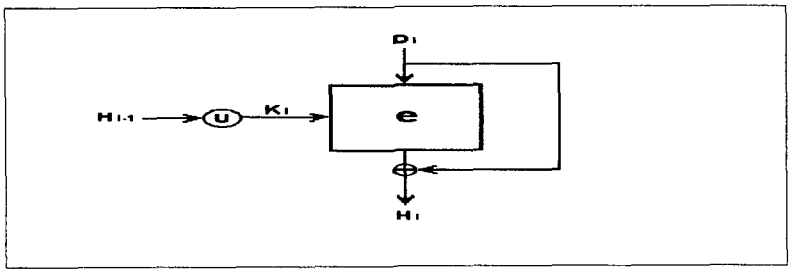
q가 홀수인 $n=2^kq + 1$ 을 계산한다.
단계 1 : 랜덤수 생성 $1<x<n$.
단계 2 : $j=0$, $y = x^q \bmod n$
단계 3 : $y = x^{q\times 2^j} \bmod n$; if (j=0 & (y=1 or y=n-1))
then n이 소수일 것으로 판단하고 알고리즘을 끝냄
if (j>0 & y=1) then 소수가 아니므로 goto 단계 5
단계 4 : j=j+1 if (j<k) then $y=y^2 \bmod n$ goto 단계 3
단계 5 : n이 소수가 아닐것이다. 알고리즘을 끝낸다.

<그림 2> Knuth의 소수 판정 알고리즘

실질적으로 n이 소수임을 증명하기 위해서는 결정적인 테스트(deterministic test)인 Jacobi sum 테스트와 complex multiplication 테스트가 빠른것으로 알려져 있으나 일반적으로 100 digit 근방의 소수를 분별해내는 것은 어려운 일이다.

본 논문에서 $k=100(4^{-100})$으로 하여 구현한 Knuth의 알고리즘은 대략 20분 정도에 256비트의 소수를 찾을 수 있었다. 실제로 k회 시행시 4^{-k}의 확률로 소수가 아님을 판정하므로 빠르게 높은 신뢰도를 갖는 소수를 얻을 수 있다.

● 해쉬함수



Di : n비트 블럭 데이타
u : n비트 블럭을 암호 알고리즘의 키로 변환
e : n비트 블럭 암호 알고리즘
Hi : eKi(Di) ⊕ Di

<그림 3> 단일 길이 해쉬함수 운영

본 논문에서의 해쉬함수는 94년도에 ISO/IEC/JTC1/SC27에서 발표한 n비트 블럭 암호 알고리즘을 이용한 IS 10118-2의 단일길이 운영 모드와 80-DES[8]로 실행되도록 설계 하였다. 블럭암호를 이용한 해쉬함수 설계방법의 장점은 단순하여 해쉬함수 설계에 필요한 노력을 최소화 할 수 있고, 블럭암호의 안전성을 바탕으로 한 해쉬함수를 설계함으로써 단순하고 안전하게 구현할 수 있다.

Differential Cryptanalysis와 선형 암호분석(Linear Cryptanalysis)에 강하도록 설계되어 있는 블럭 암호 시스템[8]을 이용함으로써 보다 안전한 해쉬함수를 구현할 수 있었다. 이러한 해쉬함수는 공개키를 얻거나 메세지 인증에 사용되기 위해서는 해쉬함수를 표준화하고 있는 ISO/IEC/JTC1/SC27에서의 요구조건인 계산의 효율성, 일방향성(one-wayness) 및 충돌 회피성(collision freeness)을 만족해야 한다.

● 역수(Inverse)

오일러의 정의에 따르면 n이 양의 정수이고 a가 정수일때, a와 n이 서로소이면 $a^{\phi(n)} = 1 \bmod n$ 의 식이 성

립한다. 여기서 $\phi(n)$은 오일러의 함수로 n보다 작은 양의 정수중에서 n과 서로소인 것의 갯수를 나타내는 함수이다. 그러므로 앞의 식은 $a \cdot a^{\phi(n)-1} = a^{\phi(n)} = 1 \bmod n$ 이 되고 a에 대한 $a^{\phi(n)-1}$은 모듈라 n의 역수가 된다. 또다른 방법으로는 유크리드(Euclid) 호제법을 이용해서 구할 수도 있다.

본 논문에서는 n=pq인 합성수 형태이고 $\phi(n)=(p-1)(q-1)$이 된다. 따라서 합성수에 대한 소인수 p,q는 키관리 센타에서 알 수 있으므로 모듈라 n의 역수는 $a^{(p-1)(q-1)-1} \bmod n$ 으로 쉽게 구할 수 있다.

● 이차 잉여(Quadratic Residue)

이차잉여의 존재여부를 알기 위해서는 르장드르 기호(Legendre Symbol)를 사용하면 쉽게 알 수 있다. 르장드르 기호는 p가 홀수인 소수이고 a는 p로 나누어지지 않는 수일때, $(\frac{a}{p}) \equiv a^{(p-1)/2} \bmod p$ 로 나타내고 이 값이 이차잉여이면 1의 값을 갖고, 이차비잉여이면 -1의 값을 갖는다. 만약 a가 법 p에 관한 이차잉여이면, 이차 합동식 $x^2 \equiv a \bmod p$ 는 꼭 두개의 해 $x \equiv x_0 \bmod p$, $x \equiv -x_0 \bmod p$ 를 가진다[10].

합성수 형태의 이차 합동식 $x^2 \equiv a \bmod pq$는 $x^2 \equiv a \bmod p$, $x^2 \equiv a \bmod q$와 동치이고, 이 두식이 해를 가질 때 위의 합동식은 4개의 해를 갖는다. 이차 합동식 $x^2 \equiv a \bmod p$, $x^2 \equiv a \bmod q$ 가 각각 두개의 해 $x \equiv \pm b \bmod p$, $x \equiv \pm c \bmod q$ 를 가진다고 하자. 이때, 이차 합동식 $x^2 \equiv a \bmod pq$ 의 해는 다음과 같은 4개의 연립일차합동식의 해이다.

$$\begin{cases} x \equiv b \bmod p \\ x \equiv c \bmod q \end{cases} \quad \begin{cases} x \equiv b \bmod p \\ x \equiv -c \bmod q \end{cases} \quad \begin{cases} x \equiv -b \bmod p \\ x \equiv c \bmod q \end{cases} \quad \begin{cases} x \equiv -b \bmod p \\ x \equiv -c \bmod q \end{cases}$$

위 4개의 연립일차 합동식을 풀기 위하여 중국인의 나머지 정리(Chinese Remainder Theorem)에 따라

$qN_1 \equiv 1 \bmod p$, $pN_2 \equiv 1 \bmod q$,

$x_0 = bqN_1 + cpN_2$, $-x_0 = -bqN_1 - cpN_2$,

$y_0 = -bqN_1 + cpN_2$, $-y_0 = bqN_1 - cpN_2$ 인 4개의 해를 구할 수 있다[10].

4개의 해 중에서 하나의 값만을 선택해서 비밀키로 사용하고 합성수 형태의 이차잉여를 계산하는 알고리즘은 <그림 4>와 같다.

```
Com_square(long input[], long output[]);
{  square(b, input, P);              /*  b^2 ≡ input mod P */
   if(value(b) == 0)  { printf("Not exist : square root"); return;);
   square(c, input, Q);              /*  c^2 ≡ input mod Q */
   if(value(c) == 0)  { printf("Not exist : square root"); return;);
   Inverse(Q, N1, P);                /*  Q×N1 ≡ 1  mod P */
   Inverse(P, N2, Q);                /*  P×N2 ≡ 1  mod Q */
   Multiply(Q, N1, work, N);         /*  x0 = bqN1 + cpN2 중에서 Q×N1 */
   Multiply(work, b, work, N);       /*  (Q×N1)×b          */
   Multiply(P, N2, work1, N);        /*  P×N2              */
   Multiply(work1, c, work1, N);     /*  (P×N2)×c          */
   Addition(work, work1, output);    /*  x0 = bqN1 + cpN2  */
   return;
}
```

<그림 4> 합성수에 대한 이차잉여 계산 알고리즘

4.2.2 디지탈다중서명 생성

키발급 센타에서 비밀키($S_{i1}, S_{i2}, \cdots, S_{ik}$)를 수록한 IC카드 대용으로 디스켓에 비밀키를 발급 받으면 디지탈다중서명을 생성하기 위해서 메세지 기안자는 먼저 메세지를 순차적으로 서명할 사람의 순서를 결정하고 $IDcm(ID_1 \| ID_2 \| \cdots \| ID_m)$을 구성한다. 본 연구에서는 K-K방식을 기본으로 하고, 각 서명자의 서명정보를 메세지에 포함하도록 하여 장차 각 서명자의 서명 정당성을 판정할 수 있도록 구현하였으며, 다중서명 생성은 <그림 5>의 알고리즘 절차에 따라 서명자의 수만큼 반복하여 수행하면 된다.

```
Multi_Signature()
  { int          i, j, n, r, chr;        char    ch;
    long   a[LLength], bit_leng, two[Length]={1,2}, c[Length], rd[Length],
           one[Length]={1,1}, work[Length], y[Length], key[80][79];
    unsigned char  r_val[8];
      printf("\n\n\t\t previous Signature Verification ? ");
      scanf("%d",&chr);
      if(chr==1)     Signature_Verification();
      do { printf("\n Sequence of Signature?"); scanf("%d",&n); } while(n<1 || n>10);
      Random_Num_Gen(rd, 512);
      Power(rd, 2, c, Nh);                    /*  c = rd^2 mod N          */
      Xn_file_read(work, n-1);                /*  Xn-1  read              */
      Multiply(c, work, a, Nh);               /*  Xn≡Xn-1 • R^2 mod N     */
      Xn_file_write(a, n);                    /*  Xn  write               */
      Mess_hash(Idcm, a, message, r_val);     /*  r_val:hash(idcm,x,m)    */
      Key_Read(key);                          /*  서명자 비밀키 read      */
      for(j=0; j<80; j++)                     /*  ∏(eij=1,80) Snj         */
      {  Key_Ith_Val(r_val, j, &r);           /*  i번째 bit 1 or 0        */
         if(r==1)
        {  Multiply(key[j], one, work, Nh);   /* 1일 경우에만 곱한다      */
              copy(work, one);
        }
      }
      Multiply(work, rd, y, Nh);      /* Rn • ∏(eij=1,80) Snj            */
      Sign_n_read(y, n-1);            /* Sign n-1  read                  */
      Multiply(y, work, c, Nh);       /* Yn ≡ Yn-1 • * Rn • ∏(eij=1,80) Snj */
      Sign_n_write(y, n);             /* Yn  write                       */
     return;
}
```

<그림 5> 디지탈다중서명 생성 알고리즘

4.2.3 디지탈다중서명 검증

서명자 n은 서명자 n-1로부터 다중서명 메세지 (M,IDcm,$X_1,\cdots,X_{n-1},Y_1,\cdots,Y_{n-1}$)를 수신하면 검증센타에 의뢰하여 검증 절차에 따라 다중서명을 검증할 수 있다.

본 논문에서의 검증센타는 서명자가 서명의 유효성을 판단할 수 있고 장차 다중서명의 정당성을 필요한 다중서명자를 선택하여 검증이 가능하다. 검증센타의 연산과정은 각 서명자의 메세지와 ID에 대한 해쉬값을 모두 구하는것 외에는 비밀키 및 다중서명 생성 절차의 연산과 동일한 방법으로 이루어진다.

4.2.4 디지탈다중서명 구현 결과

다중서명 알고리즘의 구현 결과는 소수, 랜덤정보, 사용자 ID 및 메세지 해쉬함수 출력값, 그리고 다중서명값 등이 있으며 각각의 값은 아래와 같다.

소수(256 비트) p : 10^{78}

208976487904102776269598866449518571379698076571170703903303747030319079088611

소수(256 비트) q : 10^{78}

127672627250559957449633642321695422110663111653986188465078190457502037368891

두 소수의 곱(512 비트) N = p×q : 10^{156}

026680577244311665441558818677440968802098322576465554169672788256648509523573626099937921454306506477827152723412650249631941959563355482855071404683800401

80비트 해쉬값 $I_{1j} = f(ID_1, j)$: 80비트 출력값으로 10^{24}

I_{11} : 270657635189742952212847

I_{12} : 675095143446994094843653

I_{13} : 926265577014120545146616

I_{14} : 128598494255063346748045

⋮ ⋮

ID 해쉬값에 따른 이차잉여 $I_{ij}^{-1} = S_{ij}^2 \mod N$을 만족하는 비밀키 S_{ij}

S_{11} : 009555712214295828697589643599138596234248999315590493967602270071431582725833
896477209529774897423741232511302511725946419327733646530859833857226433486714

S_{12} : 011230646603895281379665939336423763571985649990376706216560728179896413593575
056638415072856005643576380044847978696494149562167511773369274881848205157495

S_{13} : 020546496664184559177254206579004361563449748736804322628643943020177215898974
894655096855378605057536238171413929349450961436887008468944414357607670334060

S_{14} : 026044325695735658456241281494339859390946215118304679450496647069629140759608
975559676098125223675809068094093534159290209658175852136989092165495560060977

⋮ ⋮ ⋮ ⋮

기안자의 서명결과

R_1 : 010094866833195379346847765227092364708082844454370496894552883258858963816825
640642502529578141632282657278201113562662456182720448235466283610752208150482

$X_1 = R_1^2 \cdot X_0 \mod N$ (X_0=1) :
008142710475758452101395833069620978199775476393881076000729733143445742609071
227830679157286738796886627385203283446900475605272409660856434919431803976740

메세지 해쉬함수 출력비트(k=80) ($e_{11}, e_{12}, \cdots, e_{1k}$) : $h(IDcm, M, X_1)$
10100110100010100000110100000010010100011010101011001110110101111011101100001000

$Y_1 = Y_0 \cdot R_1 \prod_{e1j=1}^{k} S_{1j} \mod N$ (Y_0=1) :
015333147464158617826123530843131449404501480594837985831667906470362337532524
207434788661994341687599503514591676699351450698397924884675630641223841917213

이러한 단계를 서명자의 수만큼 반복하여 수행하면 다중서명이 된다. 예를들어 세명이 다중서명을 한 결과는 다음과 같다.

서명자 2의 $X_2 \equiv X_1 \cdot R_2^2 \mod N$:
023394487614588107711150189453881949338757803942161391397804260187332777942551
959465086726682850164432659194299525805010309613757206311970881413640862558753

서명자 2의 서명 $Y_2 = Y_1 \cdot R_2 \prod_{e2j=1}^{k} S_{2j} \mod N$:

010866863778034228737670276869204649846767090388733928678211687635907648217786
101801747881575607023381617429782919908422262069321770982790397818650365507454

서명자 3의 $X_3 \equiv X_2 \cdot R_3^2 \mod N$:
007938512879525989615096067487482681381095058216429834677991180979858610227213
777788689525525836395706817898693229733457512262433517469901134851514768769697

서명자 3의 서명 $Y_3 = Y_2 \cdot R_3 \prod_{e3j=1}^{k} S_{3j} \mod N$:

010613047416922071489130666585680799041713131692320505099233414095742596287082
811357771659719407395255887413093336020430725637559262312032498278315519095205

위와 같은 다중서명의 결과를 이용하여 정당한 다중서명의 결과는 검증단계에서 정당한 서명으로 검증이 성공되었으며 정당하지 않은 다중서명 즉,메세지를 위조하였을 경우 검증에 실패하였음을 보여줄 수 있었다.

5 결 론

정보 통신망이 급속히 보급되고 각종 통신 서비스가 제공됨에 따라 사용자 편의성과 업무의 효율성은 증대시켜 주었으나 반면에 중요정보 노출 및 침해가 주요 문제점으로 부각되게 되었다. 그러나 이러한 정보보호 문제와 상급자의 결심이나 업무의 협조 등과 같은 실제 사무환경에 맞는 전자문서를 일반문서의 처리와 동일하게 처리하기 위한 적절한 디지탈다중서명 기술이 제공되지 못하고 있는 실정이다. 정보화 사회에서 디지탈 다중서명 기술은 종이문서 체계를 대신하여 완전한 전자문서 처리를 가능하게 할 수 있는 핵심기술로서 산업계의 파급효과가 상당할 것으로 기대된다.

본 논문에서 구현한 Fiat-Shamir방식을 기반으로 하는 디지탈다중서명은 해쉬함수, 비밀키 등이 IC 카드에 저장되어야 하나 처리 능력과 메모리 부족으로 인하여 비밀키만을 디스켓에 저장하였고, 장차 IC카드의 계산 및 저장 능력이 확대되면 다중서명의 생성 및 검증까지도 IC카드에서 고속으로 처리할 수 있을 것이다. K-K 방식은 전체 서명자의 정당성만이 판정되므로 본 연구에서는 각 서명자의 서명정보를 메세지에 포함하도록 하여 장차 각 서명자의 정당성을 판정할 수 있도록 구현하였다. 소수를 찾는데 20분, 다중서명 생성시 4초, 검증시 40초 정도의 시간이 소요되었다. 검증의 대부분의 시간은 공개키를 얻기 위한 해쉬함수 수행시간에서 소모되었다.

따라서 실용화를 위해서는 해쉬함수 수행시간을 단축 시켜야 하며, 통신로 상에서 비밀성을 보장하기 위한 메세지 암호화 연구와 별도로 MHS 처리능력 향상과 보안성을 위해서 PC에 MHS Server와 같은 자체 처리 기능을 갖도록 하는 연구가 필요하다.

참 고 문 헌

[1] 남길현, 디지탈서명 보호규격 연구, 국방정보체계연구소, 1993.12

[2] 염흥렬,이만영, "이산대수 문제를 이용한 ID기본 암호시스템과 디지틀 서명 방식에 관한 연구," 91년도 데이타 보호기술 Workshop, pp.161-276, 1991

[3] A.Fiat and A.Shamir, "How to prove yourself : Practical Solutions to Identification and Signature Problems," Advances in Cryptology-Crypto'86, Lecture Notes in Computer Science 263, pp.186-199, 87

[4] E.F.Brickell,P.J.Lee and Y.Yacobi,"Secure Audio Teleconference," Advances in Cryptology-Crypto'87, Lecture Notes in Computer Science 293, pp.418-426, 1988.

[5] K.Ohta and T.Okamoto,"A Digital Multisignature Scheme Based on the Fiat-Shamir Scheme," Proceedings of Asiacrypt'91, pp.75-79, 1991

[6] 강창구,김대영, "새로운 순차 및 동시 다중서명 방식," 통신정보 보호학회 논문지, 제2권, 제1호, pp.36-44, 1992

[7] 최영주,"솟수 판별법," 통신정보보호학회지,제2권,제1호,pp.49-51,1992,3

[8] 문상재,남길현,노종선,이상재, CDMA방식 셀룰라 이동통신 시스템에 사용될 신호 암호화 기술 개발, 전자통신연구소, pp.74-113, 1994

[9] Knuth, SEMINUMERICAL ALGORITHMS,the art of computer programming,vol.2, second edition, ADDISON WESLEY, pp.374-380, 1980

[10] 박승안,김응태, 정수론 제3판, 경문사, pp.135-150, 1990