54 1992. 12. 정보과학회지 제 10 권 제 6 호

□特 輯□

컴퓨터 계산 이론을 응용한 암호알고리즘

국방대학원 전자계산학과 남 길 현*

목 차

I. 서 론

2천년대 정보화사회 진입을 눈 앞에 두고 컴퓨터와 통신망 기술은 국가전략산업의 중추적 역할을 담당하고 있으며, 지금까지 문서화된 상태로 여러 곳에 분산 관리되어 오던 각종 자료들이 컴퓨터에 입력되면서 개인의 프라이버시 침해나 중요정보의 노출 또는 파괴 등의 역기능적인 문제가 심각하게 우려되고 있다.

컴퓨터와 통신에서의 정보보호방안은 기술적인 분야뿐만 아니라 법제도적인 분야, 교육 및 요원관리 분야 등을 포함하는 통합적인 의미를 갖고 있으며 그 중에서도 암호학은 높은 수준의 보안성 유지를 위해서 필수적인 활용분야라고 할 수 있다. 이제는 군이나 정부기관에서 뿐만 아니라 일반산업체에서도 컴퓨터 정보보호 문제가 중요하게 인식되고 있으며 선진 외국에서는 1970년대 후반부터 대학교나 연구소를 중심으로 암호학에 대한 연구가 활발히 이루어지고 있다.

전통적인 암호시스템은 대부분 메세지를 재배열하거나 다른 글자로 대치시켜서 복잡하게 만듦으로써 제3자가 해독할 수 없도록 하는 방법을 사용하였으나 근래에는 컴퓨터가 발전되면서 수치계산을 빠르고 정확하게 할 수 있게 됨에 따라 컴퓨터 계산이론의 복잡도에 근거를 두고 이를 응용한 암호 알고리즘들이 많이 제안되고 있다.

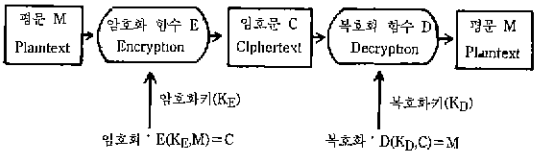
* 종신회원

본고에서는 컴퓨터의 계산이론을 근간으로 하여 제안된 암호알고리즘을 몇 가지 소개함으로써 계산이론의 활용에 대한 연구의욕을 북돋을 수 있는 계기를 마련하고자 한다.

II. 공개키 암호시스템

전통적인 개인키 암호시스템(Private-key Cryptosystem)은 평문(Plaintext)을 암호문(Ciphertext)으로 변환시켜 주는 암호화(Encryption) 알고리즘과 암호문을 다시 본래의 평문으로 회복시켜 주는 복호화(Decryption) 알고리즘으로 구성되어 있으며 암복호화 하는데는 특정한 키(key)를 사용함으로써 키를 모르고서는 누구도 암복호화를 할 수 없도록 하는 시스템이다. 그러나 개인키 암호시스템은 통신하고자 하는 상대방 마다 각각 다른 비밀키를 간직해야 하므로 가입자가 많은 전산망에서 일대일 통신을 하기 위해서는 비밀키의 수가 $O(n^2)$으로 증가하기 때문에 비밀키의 생성 및 분배에 큰 어려움을 갖게 되며, 이를 극복하기 위해서 새로운 개념의 공개키 암호시스템(Public-key Cryptosystem)이 제안되었다.

공개키 암호시스템은 (그림 1)에서와 같이 암호화할 때 사용하는 키와 복호화할 때 사용하는 키가 서로 다르며 암호화키를 알고 있더라도 복호화키를 찾아낼 수 없어야 한다.



(그림 1) 공개키 암호시스템의 구성

따라서 암호화키를 공개하면 누구든지 암호화할 수 있지만 복호화키를 비밀로 간직함으로써 복호화키를 모르는 제3자는 메세지 내용을 해독할 수 없도록 하여 비밀성을 유지할 수 있다. 또한 전산망 가입자의 수가 많아지더라도 $O(n^2)$의 비밀키만 소요되는 효과적인 암호시스템이다. 이러한 암호시스템의 개념은 비밀키의 관리를 편리하게 할 수 있고 여러가지 암호프로토콜을 개발하는데 획기적인 역할을 담당할 수 있으나 실제 응용가능한 시스템은 종류가 많지 않으며 대부분의 시스템들이 계산량의 오버헤드를 감수하여야 한다.

III. RSA 시스템

RSA 시스템은 1978년 Rivest, Shamir, Adleman이 제안한 대표적인 공개키 암호시스템으로서 Euler의 정리와 큰 합성수의 소인수분해가 어렵다는 계산 이론에 근거를 두고 있다.

먼저 RSA 시스템에서 응용되는 Euler의 Totient 함수와 Euler의 정리 및 고속지수 계산법과 합동식에서의 역원 계산 알고리즘에 대해서 알아보고자 한다.

[Euler의 Totient 함수]

양의 정수의 집합 {1, 2, …, n−1}의 원소 중에서 n과 서로소의 관계에 있는 원소의 갯수를 $\varphi(n)$으로 나타내고 이를 Euler의 Totient 함수 (φ-function)라고 한다.

특히 P가 소수일때 $\varphi(p)=p-1$이다. (단, $\varphi(1)=1$)

일반적으로 GCD(p,q)=1이면

$$\varphi(pq)=\varphi(p)\varphi(q)$$

이고, 임의의 n에 대하여

$$n=p_1^{e1}\ p_2^{e2}\cdots p_t^{et}$$

으로 표준 소인수분해 된다고 하면,

$$\varphi(n)=\prod_{i=1}^{t} p_i^{e_i-1}(p_i-1)$$

이 된다. 따라서 $\varphi(n)$을 구하기 위해서는 n을 소인수분해하여야 하며 소인수분해를 하지 않고서는 $\varphi(n)$을 구하기가 매우 어렵다고 본다.

[Euler의 정리]

임의의 양의 정수 a와 m에 대하여 GCD(a, m)=1일 때

$$a^{\varphi(m)}\equiv 1 \pmod m$$

이 성립하며 이를 Euler의 정리라고 한다.

Euler의 정리에 의하면

$$e\cdot d\equiv 1 \bmod \varphi(m)$$

이 성립되는 두 정수 e와 d에 대해서 $M\in[0, \cdots, m-1]$이고 GCD(M, m)=1인 임의의 M에 대하여

$$(M^e \bmod m)^d \bmod m\equiv M$$

이 성립된다. 왜냐하면 어떤 정수 k에 대하여

$e\cdot d=k\ \varphi(m)+1$이므로

$$\begin{aligned} M^{ed} \bmod m &\equiv M^{k\varphi(m)+1} \bmod m \\ &\equiv M\ M^{k\varphi(m)} \bmod m \\ &\equiv M(M^{k\varphi(m)} \bmod m) \bmod m \\ &\equiv M(M^{\varphi(m)} \bmod m)^k \bmod m \\ &\equiv M \bmod m \\ &\equiv M \end{aligned}$$

이기 때문이다.

[합동식에서의 곱셈역원(multiplicative inverse)]

정수의 집합 {0, 1, 2, …, m-1}의 원소들 중에서

$$ax\equiv 1 \pmod m$$

인 관계가 성립되는 x를 (mod m)에 관하여 a의 곱셈역원이라고 하며 a^{-1}로 나타냈다.

GCD(a, m)=1인 경우에 (mod m)에 관하여 a의 곱셈역원을 구하기 위해서는 Euler 정리를 이용하여

$$a^{-1}\equiv a^{\varphi(m)-1} \pmod m$$

식을 사용할 수 있으나 더욱 간편한 방법으로 Euclid 호제법을 이용하여 다음과 같이 구할 수 있다.

```
Algorithm Inverse(a. m)/* GCD(a, m)=1 */
  begin/* return x, ax≡1 (mod m), 0<(a, x)<m*/
    g0: =m,     g1: =a;
    u0; =1;     u1: =0;
    v0; =0;     v1; =1;
    i : =1;
```

```
while (g_i≠0) do/* g_i=u_m+v_m */
  begin
    y:=g_{i-1} div g_i;
    g_{i+1}:=g_{i-1}-y*g_i;
    u_{i-1}:=u_{i-1}-y*u_i;
    v_{i+1}:=v_{i-1}-y*v_i;
    i   :=i+1;
  end
  x:=v_{i-1};
  if x<0 then Inverse:=x+m
         else Inverse:=x
  end;
```

〈역원(Inverse)계산 알고리즘〉

[예제] 7x≡1 (mod 11)인 경우에 역원계산알고리즘을 이용하여 x를 구하면 다음과 같다.

i	g_i	u_i	v_i	y
0	11	−1	0	
1	7	0	1	1
2	4	1	−1	1
3	3	−1	2	1
4	1	0	−3	3
5	0			

$v_4=-3<0$이므로 x=−3+11=8 즉, 법 11에 관하여 7의 곱셈역원은 8이다. 실제로 7*8=56≡1 (mod 11)이 된다.

[고속지수계산 알고리즘]

Euler의 정리나 합동식에서 역원을 계산하는 과정 등에서는 지수계산을 필요로 하며 지수값이 클 때는 지수를 계산하는데 많은 시간이 소요된다. 그러나 합동식에서는 반복제곱에 의한 지수계산법이 효율적으로 사용된다.

이것은 지수부를 2진수로 표현하고 지수부가 2의 멱승이 되는 인수의 곱으로 분해한 후 작은 인수부터 순차적으로 법에 관한 곱셈을 계산해 나간다.

[예제]

$x\equiv 8^{26} \pmod{55}$를 구해보기로 한다.

$26=(11010)_2$이므로 $8^{26}=8^{2}8^{8}8^{16}$

$8^2\equiv 9 \pmod{55}$

$8^4\equiv 9\cdot 9\equiv 26 \pmod{55}$

$8^8\equiv 26\cdot 26\equiv 16 \pmod{55}$

$8^{16}\equiv 16\cdot 16\equiv 36 \pmod{55}$

따라서

$8^28^8\equiv 9\cdot 16\equiv 34 \pmod{55}$

$8^28^88^{16}\equiv 34\cdot 36\equiv 16 \pmod{55}$

이와 같은 고속지수계산법을 알고리즘으로 표현하면 아래와 같다. 이 방법을 사용하면 곱셈과 나눗셈의 횟수는 대략 2(log y)+1 정도 소용된다.

```
Algorithm Fastexp (a, y, m)
  begin/* return x≡a^y (mod m) */
    x:=1;
    while y>0 do
      begin
        while(y≡0 (mod 2)=0) do
          begin
            y:=y div 2;
            a:=(a*a) (mod m)
          end;
        y:=y-1;
        x:=(x*a) (mod m)
      end;
    Fastexp:=x
  end
```

〈고속지수계산 알고리즘〉

또 다른 방법으로는 Euclid의 호제법을 응용하여 a의 역원을 구할 수 있으며 이때는 φ(m)을 꼭 알 필요는 없다.

지금까지 소개한 정리와 알고리즘을 바탕으로한 RSA 시스템의 구성은 다음과 같다.

1) 두개의 큰 소수 p, q를 선정하고 n=pq를 계산하여 n을 공개함
2) GCD(e, φ(n))=1 되는 e를 선정하여 공개함
   (여기서 φ(n): Euler의 Totient 함수)
3) e·d≡1 (mod φ(n))되는 d를 계산(역원계산 알고리즘 사용)하여 비밀키로 함
   (φ(n)=(p−1)(q−1))
4) 암호화 $C=M^e \pmod{n}$, $0\le M<n$
5) 복호화 $M=C^d \pmod{n}\equiv M^{ed} \pmod{n}\equiv M$
   공개키 : n, e
   비밀키 : p, q, d (p와 q는 e와 d가 결정된 후 폐기할 수 있음)

[예제]

p=7, q=11이라고 하면 n=77,

$\varphi(n)=(7-1)(11-1)=60$

e=13이라고 하면 d=Inverse (13, 60)=37

따라서, 평문 메세지 M=18인 경우에

암호문 $C = M^e \pmod n$
$= 18^{13} \pmod{77}$
$= 18 * 18^4 * 18^8 \pmod{77}$
$\equiv 18 * 25 * 9 \pmod{77}$
$\equiv 46$

이를 다시 복호화 하면

평 문 $M \equiv C^d \pmod n$
$= 46^{37} \pmod{77}$
$= 46 * 46^4 * 46^{32} \pmod{77}$
$\equiv 46 * 60 * 37 \pmod{77}$
$\equiv 18$

RSA시스템은 공개키 n과 e를 가지고 비밀키 d를 구할 수 있으면 해독이 가능해진다. d를 찾아내기 위해서는 $\varphi(n)$을 계산할 수 있으면 가능하지만 n의 소인수 분해를 모르고서는 $\varphi(n)$을 결정하기 어렵다. 만약 p와 q가 약 100자리수의 소수이고 n이 약 200자리수를 갖는 합성수라고 하면 현재의 기술로 n을 인수분해하는 것은 거의 불가능하다고 판단된다.

지금까지의 연구결과로서 n을 소인수분해 하는데 소요되는 복잡도 T는

$$T = \exp\left((\ln n)^{1/2} (\ln \ln n)^{1/2}\right)$$

로 표현될 수 있다. 여기에서 $\ln$은 자연대수를 나타내며 exp는 $\ln$의 역(inverse)을 표시한다.

이 공식을 이용하여 한번 계산단위를 1 μsec라고 가정하면 200자리수를 갖는 합성수(약 664 bit수)를 소인수분해하는데 소용되는 시간은 약 10^{12}일 (day)이 소요되어 수억년이 필요하다는 계산이 나온다. 최근에는 여러가지 더욱 발전된 소인수분해법들이 개발되고 있으나 아직까지는 200자리수를 이용하는 RSA 시스템은 안전하다고 판단한다.

더욱이 RSA 시스템의 강점은 만약 200자리수의 합성수에 대한 소인수분해가 가능해진다면 다시 300자리수의 합성수로 확장시킬 수 있다는 점이다.

RSA 시스템은 공개키 e를 전화번호부처럼 공개하므로 누구든지 메세지를 암호화하여 상대방에게 보낼 수 있고 상대방의 비밀키를 알고 있어야만 복호화가 가능하기 때문에 비밀성이 보호된다. 한편 RSA시스템은 200자리수의 정수에 대한 멱승을 필요로 하기 때문에 계산에 소요되는 시간이 오래 걸린다는 단점이 있으며 큰 소수를 판별하는 방법, 역원을 계산하는 방법 등도 일반사용자에게는 친숙하지 못한 방법들이라고 볼 수 있다. 그러나 RSA시스템은 중요한 정보의 컴퓨터 통신에 필요한 디지탈 서명을 구현할 수 있는 아주 좋은 도구가 될 수 있다.

이 외에도 소인수분해의 어려움에 근거를 두면서 RSA 시스템과 비슷한 Rabin 암호시스템, Williams 암호시스템 등이 있다.

IV. RSA 시스템을 이용한 디지탈 서명 방식

지금까지의 종이서류에 의한 문서수발업무가 컴퓨터 통신망을 통해 빠르고 경제적으로 이루어지고 있으나 중요 메세지 전송시 상대방의 신분을 확인하고 메세지 내용에 대한 책임소재를 명확하게 해줄 수 있는 통상적인 인감도장과 같은 역할을 해줄 수 있는 제도와 기술적 절차가 필수적인 사항으로 부각되고 있다.

보통의 인감도장이 찍힌 종이서류와는 달리 컴퓨터 통신으로 전송된 메세지는 수신자가 고의적으로 메세지 내용을 변조하거나 부인함으로써 송수신자 사이에 분쟁이 발생될 수 있다. 이러한 문제점을 해결하기 위한 것이 디지탈 서명(Digital Signature)이라고 하며 암호시스템을 이용하여 구현하는 방안이 많이 연구되고 있다.

일반적으로 암호시스템이 추구하는 기본목표는 비인가자에게 자료의 노출을 방지하는 비밀성(Secrecy) 유지와 비인가자에 의한 자료의 변경을 막아주는 인증성(Authenticity) 확보에 있다고 볼 수 있다. 이러한 인증성은 수신자가 자료를 보낸 송신자와 보내온 메세지를 신뢰할 수 있다는 의미가 되며 암호시스템을 이용한 디지탈 서명 시스템도 비슷한 목표를 지니게 된다.

디지탈 서명이 필요로 하는 요구 특성은 다음과 같다.

첫째, 수신자는 메세지의 서명을 통하여 송신자를 인증할 수 있어야 한다.

둘째, 수신자를 포함한 어느 누구도 송신자의 서명을 위조할 수 없어야 한다.

세째, 송신자는 추후에 메세지 내용을 변경하거나 보낸 사실을 부인할 수 없어야 한다.

넷째, 만약 송수신자 사이에 분쟁이 발생하였을 때는 판정관이 누가 옳고 그름을 판별해 줄 수 있어야 한다.

이러한 디지탈 서명을 구현하는 방안은 여러가지가 있으나 RSA 공개키 암호시스템을 이용하는 방식은 제3의 증재자를 필요로 하지 않기 때문에 가장 바람직한 디지탈 서명 방식 중의 하나라고 볼 수 있다.

송신자(서명자) A가 수신자 B에게 디지탈 서명을 보내는 방식은 A와 B가 각각 RSA 시스템과 같은 비밀키와 공개키를 갖고 있을 경우에 다음과 같이 설명할 수 있다.

E_A, E_B : A와 B의 공개키를 이용한 암호화

D_A, D_B : A와 B의 비밀키를 이용한 복호화

M : 송신하고자 하는 메세지
S : 디지탈 서명
서명생성(A) : $S=E_B(D_A(M))$
복 호 화(B) : $M=E_A(D_B(S))$
$=E_A(D_B(E_B(D_A(M))))$
$=E_A(D_A(M))$
$=M$

즉 송신자(서명자)는 자신의 비밀키와 상대방의 공개키를 이용하여 서명 S를 생성하고 수신자는 송신자의 공개키와 자신의 비밀키를 복호화하여 메세지 내용을 확인하고 본래 받은 서명 S를 증거물로서 메세지 M과 함께 보관한다. 이렇게 생성된 서명은 오직 송신자 A만이 만들 수 있으므로 수신자는 서명자를 인증할 수 있고 또한 정당한 수신자 B만이 복호화 할 수 있으므로 메세지의 비밀성을 유지할 수 있다. 서명의 암복호화시에 사용되는 합성수 n_A와 n_B의 대소에 따라 서명 생성 과정이 약간 바뀔 수 있으나 별 다른 문제점은 없다고 본다.

다만 RSA 시스템이 근본적으로 안고 있는 취약점이 계산량의 오버헤드인데 서명을 생성하고 복호화하는데 전체 메세지에 대해서 이중으로 암복호화가 수행되어야 하기 때문에 이러한 방법은 더욱 과중한 계산량 부담을 지니게 된다. 이와같은 계산량의 오버헤드를 감소시킬 수 있는 대안으로서는 압축 알고리즘(Hash 함수)을 사용하여 전체 메세지를 적당량의 길이로 압축시켜서 짧은 함축문을 만들고 함축문에 대해서만 앞에서 기술한 방법으로 서명을 생성하는 방법이 제안될 수 있다. 이때는 압축 알고리즘 개발에 세심한 주의를 필요로 하며 본래 메세지의 비밀성을 유지할 수 있는 다른 암호기법의 선택이 요구된다.

이와같은 디지탈 서명은 각종 전산망의 확장과 더불어 정보화 사회를 앞당길 수 있는 중요한 도구가 될 것이며 이러한 시스템이 하루 빨리 표준화되고 법적인 효력을 갖을 수 있도록 법적, 제도적, 뒷받침이 되어야만 할 것으로 생각된다.

V. ElGamal 암호시스템

p가 소수이고, g를 법 p에 관한 원시근(primitive root)이라고 하면 $y\equiv 0 \pmod p$인 임의의 y에 대해,

$$y\equiv g^x \pmod p$$

를 만족하는 $x\in[0, \cdots, (p-1)]$이 반드시 존재한다.

p가 홀수인 소수일때는 항상 $\varphi(p-1)$개의 원시근이 존재하며 여기에서 g가 법 p에 관한 원시근이라 함은 다음의 특성을 만족시킨다.

1) $g^i\equiv g^j \pmod p \Leftrightarrow i\equiv j \pmod{\varphi(p)}$
2) 집합 $[g, g^2, g^3, \cdots, g^{\varphi(p)}]$는 법 p에 관한 기약 잉여계이다.

일반적으로 앞의 식에서 x, g, p가 주어졌을 때는 p가 비교적 큰 정수라해도 y는 고속지수계산 알고리즘을 사용하여 쉽게 구할 수 있지만, y, g, p가 주어졌을 때 x를 구하는 문제는 매우 어려운 것으로 인정되고 있으며 이를 이산대수 문제(discrete logarithm problem)라고 한다. 이산대수문제의 복잡도는 소인수분해의 복잡도와 비슷한 것으로 판단된다.

ElGamal 암호시스템은 이산대수문제가 매우 어렵다는 가정하에서 제안된 공개키 암호시스템이다. 이제 A가 B에게 메세지 $M(0\le M\le p-1)$을 보내는 경우를 생각해 보자.

1) 준비 단계
－먼저 큰 소수 p와 법 p에 관한 원시근 g를 결정하여 공개함
－사용자 개인은 각각 자신의 비밀키(x_i)를 선정하고 공개키(y_i)를 계산하여 공개함.

$$y_B\equiv g^{X_B} \pmod p,\ (0<X_B\le p-1)$$

2) 암호화
－A는 임의의 키 $k(0<k\le p-1)$를 선정한 후 T를 계산함

$$T\equiv y_B^k \pmod p$$

－암호문 $C=(C_1, C_2)$를 계산하여 B에게 보냄

$$C_1\equiv g^k \pmod p$$
$$C_2\equiv TM \pmod p$$

3) 복호화
－B는 먼저 T를 구함

$$T\equiv y_B^k\equiv(g^{X_B})^k\equiv(g^k)^{X_B}\equiv C_1^{X_B} \pmod p$$

(X_B는 B 자신만이 알고 있으므로 쉽게 계산됨)

－평문 메세지 M을 구함

$$M\equiv\frac{C_2}{T} \pmod p$$

이산대수문제의 어려움에 근거를 두고 있는 ElGamal시스템과 비슷한 시스템들이 키분배를 위해 많이 연구되고 있으며 최근 미국에서는 디지탈 서명을 표준

화하는데 ElGamal기법을 응용한 알고리즘을 제안하여 의견을 수렴하고 있다. ElGamal암호시스템도 약 200자리수 크기를 갖는 소수 p를 사용한다.

VI. Knapsack 암호시스템

일반적인 Knapsack문제는 박스의 길이가 C이고 주어진 n개의 다양한 길이를 갖고 있는 막대 길이를 $H=[a_1, a_2, \cdots, a_n]$이라고 할 때 C박스를 정확하게 채울 수 있는 막대들을 선정하는 문제로써 NP-complete 문제라는 것이 널리 알려져 있다. Knapsack 문제를 이용한 암호화 방법을 생각해 보면 n개의 다양한 정수값을 선정해 놓고 평문 메세지를 n-bit 이진벡터(binary vector)로 생각하여 C 값을 구하면 C 값은 바로 메세지에 대응되는 암호문이 된다.

즉, $C=\sum_{i=1}^{n} a_i \cdot m_i$, ($m_i$: 0또는 1인 메세지)

만약 n이 20 정도이면 컴퓨터로 모든 조합을 조사할 수 있으나 n이 100이상 될때는 가능한 조합의 수가 $2^{100} \approx 10^{30}$으로써 정확한 해를 구하기는 매우 어렵다고 본다.

그러나 일반적인 Knapsack문제는 C와 a_i만을 알고서는 암호문을 받은 수신자도 마찬가지로 복호화가 어렵게 되어 암호시스템으로서의 역할을 할 수 없게 된다. 그러므로 복호화를 쉽게할 수 있도록 하기 위해서는 새로운 $S=[s_1, s_2, \cdots, s_n]$를 다음과 같이 선정하면 복호화가 용이해 진다.

$$s_i > \sum_{j=1}^{i-1} s_j, \quad i=2, 3, \cdots, n$$

이와 같은 특성을 갖는 것을 Superincreasing Knapsack이라고 하며 복호화는 맨끝의 s_n부터 역순으로 C 값이 s_i보다 크거나 같으면 해당되는 m_i는 1이 되고 s_i 값을 C 값에서 빼고, 그렇지 않으면 m_i는 0이 되고 S_{i-1}을 다시 비교하는 방법으로 계속해 나가면 된다.

한편 S를 공개키로 공개한다면 S를 아는 모든 사람이 복호화가 가능해지므로 Superincreasing Knapsack문제의 S를 변환시켜 일반적인 Knapsack 문제의 H로 변환시켜 주고 어떤 비밀키를 알고 있으면 H로부터 S를 유도하여 쉽게 복호화할 수 있도록 한 것이 Knapsack 암호시스템이다. Knapsack 암호시스템을 소개하면 다음과 같다.

1) Superincreasing Knapsack 문제를 변환

－다음과 같은 특성을 갖는 p, $S=[s_1, s_2, \cdots, s_n]$, w를 선정

$$s_i > \sum_{j=1}^{i-1} s_j \ (i=2, 3, \cdots, n), \quad \sum_{i=1}^{n} s_i < p$$

－비밀키 p와 w는 서로 소. 따라서 $W \cdot W^{-1} \equiv 1 \pmod p$인 w^{-1}이 존재함.

－공개키 $H=[a_1, a_2, \cdots, a_n]$ 계산

$$a_i \equiv w \cdot s_i \pmod p, \quad i=1, \cdots, n$$

2) 암호화

메세지 $M=(m_1, m_2, \cdots, m_n)$, $m_i \in \{0, 1\}$

$$C=H \cdot M=\sum_{i=1}^{n} a_i \cdot m_i$$

3) 복호화 $C' \equiv w^{-1} C \pmod p$

$$\equiv w^{-1} \sum_{i=1}^{n} a_i \cdot m_i \pmod p$$

$$\equiv w^{-1} \sum_{i=1}^{n} (w \cdot s_i \pmod p) \cdot m_i \pmod p$$

$$\equiv \sum_{i=1}^{n} (w^{-1} w \cdot s_i)\ m_i \pmod p$$

$$\equiv \sum_{i=1}^{n} s_i \cdot m_i \pmod p$$

$$\equiv \sum_{i=1}^{n} s_i \cdot m_i$$

즉, 수신자는 $C' \equiv w^{-1} C \pmod p$로서 C'를 구하고

$C' \equiv \sum_{i=1}^{n} s_i \cdot m_i$으로부터 m_i를 구할 수 있게 된다. (s_i는 비밀키 이므로 수신자는 알고 있음)

[예제] 먼저 Superincreasing Knapsack의 $S=[1, 2, 4, 9]$라 하고 $p=17$, $w=15$라고 하자. 비밀키는 S, p, w이다.

공개키 $H=[a_1, a_2, a_3, a_4,]$을 계산해 보면

$a_1 \equiv 15*1 \pmod{17} \equiv 15$

$a_2 \equiv 15*2 \pmod{17} \equiv 13$

$a_3 \equiv 15*4 \pmod{17} \equiv 9$

$a_4 \equiv 15*9 \pmod{17} \equiv 16$

즉 $H=[m_1, m_2, m_3, m_4]=[0100, 1011, 1010, 0101\rangle$라고 할 때 암호문 $C=[C_1, C_2, C_3, C_4]$를 구해보면

$C_i=H \cdot m_i$

$C_1=[15, 13, 9, 16]*[0, 1, 0, 0]=13$

$C_2=[15, 13, 9, 16]*[1, 0, 1, 1]=40$

$C_3=[15, 13, 9, 16]*[1, 0, 1, 0]=24$

$C_4=[15, 13, 9, 16]*[0, 1, 0, 1]=29$

즉 $C=[13, 40, 24, 29]$가 된다.

복호화를 위해서 $15*8 \equiv 1 \pmod{17}$이므로 $w^{-1}=8$이다.

$m_i = w^{-1}C_i \pmod p$

$m_1 = 8*13 = 104 \pmod{17} = 2 = [0100]$

$m_2 = 8*40 = 320 \pmod{17} = 14 = [1011]$

$m_3 = 8*24 = 192 \pmod{17} = 5 = [1010]$

$m_4 = 8*29 = 232 \pmod{17} = 11 = [0101]$

(S=[1, 2, 4, 9]를 알고 있으므로 마지막 단계를 쉽게 계산할 수 있음)

여기서 소개한 Knapsack 암호시스템은 $n \approx 200$ 정도이고 s_i가 200비트 이상의 정수들로 구성된 공개키 암호시스템으로 제안되었으나 그 후에 여러가지 공격분석에 의해서 비교적 취약한 시스템이라는 결론에 도달했으며 이제는 여러가지 변형된 시스템들이 제안되고 있다.

VII. Subkey 암호시스템

1981년 Davida, Well, Kam은 중국인의 나머지 정리를 이용하여 레시듀 숫자와 소인수분해의 어려움에 근거를 둔 암호시스템을 제안하면서 데이타베이스를 암호화하는데 적합하다고 하였다. 이 시스템은 각 레코드(record)를 암호화 단위로 하여 레코드암호문으로부터 각 필드(field)는 각각에 해당하는 Subkey만 가지고 복호화할 수 있도록 하는 Subkey 암호시스템으로서 일반적인 공개키 암호시스템과는 다른 특성을 갖고 있다. 즉, 암호화키를 알고 있는 사람은 복호화키를 알 수 있으며 반면에 특정한 필드에 관한 복호화 키만을 알고서는 암호화키를 알아낼 수 없도록 설계되어 있으므로 공개키 암호시스템이라고는 할 수 없지만 복호화키와 암호화키를 분리시킨 암호시스템이라고 볼 수 있다.

Subkey 암호시스템에서는 암호화를 위한 write subkey $e_1, e_2, \cdots, e_t$와 복호화를 위한 read subkey $d_1, d_2, \cdots, d_t$를 분리하여 사용한다.

먼저 각 필드에 대응되는 최대 필드값 보다 큰 임의의 소수 d_i를 선정하여 read subkey로 한다. 각 레코드는 t개의 필드로 구성되어 있다고 가정하면, $d_1, d_2, \cdots, d_t$가 선정되며 $n = \prod_{i=1}^{t} d_i$라고 할 때 write subkey는 다음과 같이 계산된다.

$$e_i = \left(\frac{n}{d_i}\right) y_i, \quad (i = 1, \cdots, t)$$

$(y_i = \text{Inverse}(n/d_i, d_i))$

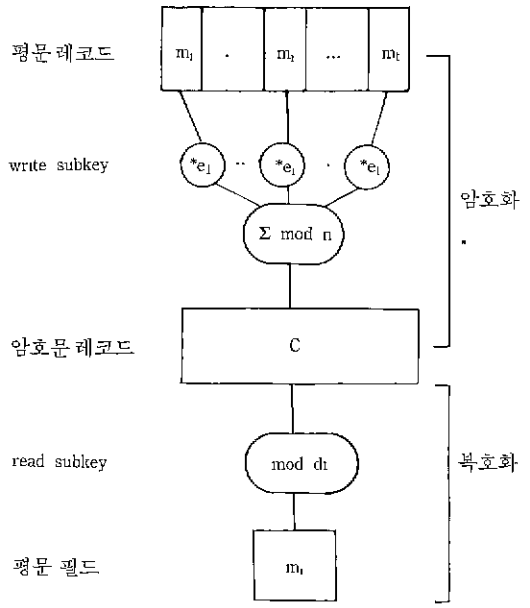
평문레코드 M은 t개의 필드 $m_1, m_2, \cdots, m_t$로 구성되어 있다고 하면 암호화는 다음과 같이 수행되며 (그림 2)와 같이 표현할 수 있다.

1) 암호화 : write subkey를 모두 알고 있는 사람이 레코드 단위로 암호화 함.

$$C \equiv \sum_{i=1}^{t} e_i m_i \pmod n$$

2) 복호화 : 각 필드별로 read subkey를 알고 있는 사람은 해당 필드만 복호화 함.

$m_i \equiv C \bmod d_i$



(그림 2) Subkey 암호시스템의 암복호화 과정

[예제] $t=3$, $d_1=7$, $d_2=11$, $d_3=5$라고 하면

$n = 7*11*5 = 385$

$y_1 = \text{Inverse}(385/7, 7) = \text{Inverse}(55, 7) = 6$

$y_2 = \text{Inverse}(385/11, 11) = \text{Inverse}(35, 11) = 6$

$y_3 = \text{Inverse}(385/5, 5) = \text{Inverse}(77, 5) = 3$

따라서, $e_1 = 55*6 = 330$, $e_2 = 35*6 = 210$, $e_3 = 77*3 = 231$이 된다. 평문 레코드 M=(4, 10, 2)를 암복호화 해보자

암호화 : M에 해당하는 암호문레코드 생성

$C \equiv (e_1m_1 + e_2m_2 + e_3m_3) \pmod n$

$\equiv (330*4 + 210*20 + 231*2) \pmod{385}$

$\equiv (1320 + 2100 + 462) \pmod{385}$

≡32

복호화 : 각 필드별로 복호화

$m_1 \equiv C \pmod{d_1} \equiv 32 \pmod{7} \equiv 4$

$m_2 \equiv C \pmod{d_2} \equiv 32 \pmod{11} \equiv 10$

$m_3 \equiv C \pmod{d_3} \equiv 32 \pmod{5} \equiv 2$

지금까지 설명한 Subkey 암호시스템은 여러 개의 필드를 통합하여 레코드 단위로 암호문을 만들고 각 필드별로 비밀키를 갖고 복호화함으로써 read만을 필요로 하는 데이타베이스 사용자들에게 액세스 권한을 분리할 수 있다는 특성을 갖고 있으며 write보다 read의 속도가 훨씬 빠르다는 장점을 갖고 있다.

그러나 여러가지 암호분석가들에 의한 공격을 막기 위해서는 각 필드마다 32비트 이상의 랜덤수를 첨가시켜야 하고 추가적인 필드가 필요하며 레코드의 크기도 매우 커야 한다는 등 여러가지 실제 활용하기에는 부적합한 특성들이 많이 포함되어 있다. 또한 각 필드단위로 암복호화를 수행하고 암복호화는 Subkey 암호시스템과 반대가 되도록 하는 방법도 제안되고 있으나 실제 응용하는데는 더 많은 연구가 필요할 것이다.

VIII. 결 론

지금까지 계산이론에 근거를 둔 공개키 암호시스템들에 대한 소개를 하였다. 큰 합성수의 소인수분해의 어려움에 근거를 둔 RSA 시스템, 유한체에서 이산대수문제의 어려움에 근거를 둔 ElGamal 암호시스템, NP-Complete 문제의 복잡도에 근거를 둔 Knapsack 암호시스템, 그리고 이들과는 약간 다른 특성을 갖고 있지만 복호화키와 암호화키를 분리시킨 Subkey 암호시스템 등이다.

이밖에도 소인수분해와 합동식의 제곱근을 구하는 문제의 어려움에 근거를 둔 Ong-Schnorr-Shamir의 인증시스템, 타원 곡선상의 이산대수문제에 바탕을 둔 Diffie-Hellman의 키분배 방식, 부정연립방정식의 해를 구하기 어려운데 근거를 두고 오류제어부호(error control code)를 이용한 McEliece의 암호시스템 등 수많은 암호시스템들이 제안되고 있으며, 이제는 컴퓨터 기술의 발달과 더불어 이론적 근거를 배경으로 실용화 단계로 개발되고 있는 추세이다.

암호시스템의 활용분야도 정보의 비밀성을 보호할 뿐만 아니라 디지탈 서명, 키의 분배, 암호프로토콜, 신원확인, 액세스 제어 및 바이러스 방지 대책 등 광범위하게 확산되고 있다.

컴퓨터 이론 연구가 자칫 이론 자체에만 국한하는 경향이 있으나 암호시스템은 계산이론을 바탕으로 활용에 중점을 두고 있으며, 정보보호문제는 정보화 시대를 앞당기는데 선결과제라고 볼 수 있다. 외국으로부터 컴퓨터 보안 관련 제품의 수입이 어려운 현실을 감안하여 볼 때 국내 대학이나 연구소를 비롯하여 산업체에서도 암호학에 대한 연구에 더욱 관심을 갖고 활성화 될 수 있는 계기가 되길 바랄 뿐이다.

참 고 문 헌

1. 남길현, 선진국의 정보보호 기술동향 분석, 한국전자통신연구소, 4, 1991.
2. 남길현, 암호시스템을 이용한 디지탈 서명 시스템, 한국통신정보보호학회지, 4, 1991.
3. Dorothy E. Denning, Cryptography and Data Security, Addison Wesley, 1982.
4. Charles P. Pfleeger, Security in Computing, Prentice Hall, 1989.
5. Kil-Hyun Nam. TRN Rao. "Private-key Algebraic-Code Encryption". IEEE Trans on Information Theory, Vol. 35, No. 4, July. 1989.
6. R. Rivest, A. Shamir, L. Adleman, "A Methed for obtaining Digital Signatures and Public-key Cryptosystem", Comm. ACM, Vol. 21(2), 1978.
7. J. Seberry, J. Pieprzyk, Cryptography-An Introduction to Computer Security, Prentice Hall, 1989.
8. M. R. Schroeder, Number Theory in Science and Communications, Springer-Verlag, 1984
9. NIST, "Responses to NIST's Proposal". Comm. ACM, Vol. 35, No. 7, July 1992.

남 길 현

1969 육군사관학교 졸업
1973 서울대학교 토목학과 졸업
1979 미 해군대학원(전산학 석사)
1983 미 위스콘신(메디슨) 주립대학교(전산학 석사)
1985 미 루이지아나 주립대학교(전산학 박사)
1985 ~ 현재 국방대학원 전산학과 부교수로 재직
관심 분야 : 컴퓨터보안, 암호학, 데이타베이스 알고리즘 분석