論 文

선형 이동 Knapsack 공개키 암호화 시스템의 구현에 관한 연구

正會員 車 均 鉉* 正會員 白 京 甲* 正會員 白 寅 天* 正會員 朴 商 奉*

A Study on the Implementation of Linearly Shift Knapsack Public Key Cryptosystem

Kyun Hyon TCHAH*, Kyeong Kap PAEK*, In Cheon PAIK*, Sang Bong PARK*

Regular Members

要 約 본 논문에서는 공개키 시스템을 위한 새로운 knapsack 알고리즘의 설명과 난이도 시험 및 이를 구현하기 위한 병렬 구조를 제안하였다. 기존의 Merkle Hellman의 knapsack은 선형 대수법에 의한 다른 쉬운 수열로의 사상 효과 등으로 Shamir나 Brickell 등의 attack에 약했으나, 선형이동 knapsack 시스템은 이러한 약점을 보완한다. 그리고 Brickell 및 Lagarias, Odlyzko의 저밀도 attack 알고리즘의 구현으로 새로운 knapsack 시스템과 기존의 knapsack 시스템을 비교 평가하였다. 또한 이 선형 이동 knapsack 시스템의 병렬 구현을 위한 VLSI 구조를 제안하였다.

ABSTRACT In this thesis, explanation of new knapsack algorithm for public key system, difficulty test and parallel architecture for implementation are suggested. Past Merkle-Hellman's knapsack is weak in Shamir or Brickell's attack by the effects of mapping into other easy sequences. But linearly shift knapsack system compensates them. And this system is compared with past knapsack system by implementation of low density attack in Brickell and Lagarias, Odlyzko's method. Also the VLSI architecture for parallel implementation of this linearly shift knapsack system is presented.

I. 서 론

1970년대 이후로 보인 반도체 및 회로기술의 급격한 발달로 암호화를 위한 장비들이 상품시장에 널리 퍼지게되고, 컴퓨터 네트워크, 기타 정보전달을 위한 통신이 급격히 증가함에 따라 키의 유실확률이 크고 교환이 불편한 기존의 대칭형 암호시스템으로부터 교환성이 용이하고 안정성이 높은 비대칭형 암호시스템을 구상하였고[2] 1976년 Diffie와 Hellman에 의해 최초로 이러한 요구를 만족시키는 공개키 시스템이 제안되었다. 최초로 공개키 암호화 시스템의 구현은 Rivest와 Merkle에 의해 공개되었다. 이것은 큰 수를 인수분해하는 것이나 subset problem의 복잡도에 근거하여 제시된 것이고 이들은 NP-complete 문제로 알려져 있었다.[1]

하지만 이는 기본적으로 선형성을 갖고, Brassard 에 의해 제시된 복잡도 문제에서 NP-completeness에 의심을 갖기 시작했다.[11] 이후에 Merkle-Hellman 이 Lenstra의 integer programming 알고리즘과 diophantine approximation 알고리즘으로 단일 반복 기본 knapsack을 다른 쉬운 knapsack으로 바꿀 trapdoor pair를 찾을 수 있음을 보여줬다. Odlyzko는 반복된 knapsack에 대해 UGSDA(Unuusally Good Simultaneous Diophantine Approximation) 방법을 사용하여 attack 할 수 있는 방법을 제시했고

*高麗大學校 電子工學科
Dept. of Electronic Eng., Korea University
論文番號 : 91-82(接受1991. 5. 30)

[8], Brickell은 저밀도 knapsack에 대한 attack이 L^3 basis reduction을 사용한 lattice내의 short vector를 찾음으로 이루어짐을 보였다.[7]

Desmedt는 왜 이러한 knapsack 암호시스템이 break 되는 가를 보였다. Merkle-Hellman 이나 Graham-Shamir scheme 으로부터 얻을 수 없는 디코딩이 가능한 암호화 키가 있음을 보였다. 이러한 얻을 수 없는 키가 knapsack 문제의 최악의 경우이므로 이런 키를 배제하면 knapsack 문제를 풀 수 있다. Desmedt와 Shamir 등이 여러 일반 knapsack 공개키 암호화 시스템에서 불필요한 암호화 키를 줄일 수 있음을 제안했다. 이것은 쉬운 복호화 키를 선형대수를 사용한 임의의 복호화키로 대체하는 것이다. 그러나 이것도 modular multiplication의 제약에 의해 충분한 불필요한 키를 얻을 수 없다. 게다가 data expansion이 매우 크고 복호화 속도가 매우 떨어진다. 또 저밀도 attack을 당하기 쉽다.

Goodman과 McAuley에 의한 knapsack은 modular multiplication과 CRT에 의해 이루어지나 linear integer programming이나 lattice reduction, short vector를 찾음으로 역시 쉽게 break 될 수 있음이 보여졌다.[1]

Pieprzyk는 GF(2) 상에서 다항식에 근거한 knapsack 형태의 공개키 시스템을 제안했다. 이것은 다항식을 사용한다는 것을 제외하고는 Goodman McAuley knapsack과 유사하다. 이 시스템 역시 gcd attack이나 선형대수 방법에 의해 breakable하다.[11]

Chor-Rivest는 finite field에서의 arithmetic에 근거한 knapsack 형태의 암호화 시스템을 제안하였으나 키 생성이 어렵고 복호화 속도가 느리다. Jau-Yien Lee[6]는 키 생성이 용이하고 임의의 다른 수열에 modular multiplication이나 다른 방법을 적용하여 얻을 수 없는 키 생성법을 제안했다. 또한 여기서 생성된 암호화 키는 NP completeness 를 갖는 worst knapsack으로 사상될 확률이 매우 크다. 따라서 Shamir의 attack 이나 저밀도 attack이 적용되지 않는다. 따라서 본 연구에서는 Merkle Hellman knapsack과 선형이동 knapsack 암호화 시스템에 대한 알고리즘의 설명, 위의 암호화 시스템에 대해 Brickell의 저밀도 knapsack의 break 알고리즘과 Lagarias와 Odlyzko의 Short Vector 알고리즘을 이용한 break test, 그리고 선형이동 knapsack 암호화 시스템의 VLSI 구현을 위한 병렬 구조를 제시하였다.

Ⅱ. Merkle-Hellman 암호화 시스템과 Polynomial Attack

Merkle Hellman이 제안한 암호 시스템은 최초로 송신자가 초증가 수열(superincreasing sequence) $A'=(a_1', a_2', \cdots, a_n')$ (즉, $a_i' > \sum_{j=1}^{i-1} a_j'$)을 선택하고 이것에 다음의 변형을 가하여 pseudorandom 수열 $A=(a_1, a_2 \cdots, a_n)$을 만든다.

$$a_i = a_i' \times W \bmod U$$

여기서 $U > 2a_n'$, $\gcd(W, U)=1$로 U과 W를 선택한다.
따라서, A를 공개하고 송신자는 이진 메세지 $M=(m_1, m_2, \cdots, m_n)$을 $S=\sum_{i=1}^{n} m_i a_i$로 암호화한다. 이제 송신자는 일반 통신 채널을 통하여 수신자에게 S를 보낸다. A가 공개되어 있고 도청자가 S를 가로채도 A의 부분합 S가 되는 것을 찾는 것은 NP complete 문제이다.

하지만 올바른 수신자, 즉 W와 U을 알고 있는 수신자이면 아래의 계산에 의해 메세지 $M(m_1, m_2, \cdots, m_n)$을 얻을 수 있다.

$$\begin{aligned} S' &= W^{-1} S \bmod U \\ &= W^{-1} AM \bmod U \\ &= W^{-1}(WA')M \bmod U \\ &= A'M \bmod U \\ &= AM \qquad W^{-1} = \mathrm{inv}(W, U) \end{aligned}$$

초증가 수열에서 m_i는 많아야 n번의 감산에 의해 계산될 수 있다. Merkle Hellman knapsack

은 다음 2가지의 단점이 있다. 첫째로 밀도가 0.5보다 작아 공개 화일 크기에 비효과적이고 둘째로, 복호화 키가 쉬운 수열 이라서 깨지기 쉽다. Desmedt는 공개키 A를 다른 초증가 수열로 바꾸고 infinite pair(v, U′)이 존재함을 보였다. 또한 Lenstra 등에 의해 제안된 L^3 reduction 알고리즘을 이용한 저밀도 knapsack의 해법이 제시되었다.

이러한 두 단점에도 불구하고 암호화와 복호화의 속도가 빠르다는 잇점이 있어 연구대상이 되고 있고 다음 절에서 이를 보완한 두 알고리즘을 설명한다.

Ⅲ. 고밀도 Knapsack 알고리즘

본 절에서는 기존 Merkle Hellman의 저밀도 knapsack 수열을 고밀도 knapsack으로 바꾸는 방법을 소개한다. 이를 위해서 정리 1을 사용한다.

정리 1 : a_i'가 초증가 수열이고 정수 v가

$a_i' > \sum_{j=1}^{i-1} a_j' + v$, $i=1, 2, \cdots, n$과 $U > \sum_{i=1}^{n} a_i'$

를 만족하고 $0 \leq b_i \leq v$인 정수 b_i에 대해 $c_i = a_i' - b_i$는 $U > \sum_{i=1}^{n} c_i$에 대해 초증가 수열을 형성한다.

증명 : $c_i = a_i' - b_i$이고 $0 \leq b_i \leq v$이므로

$$\sum_{j=1}^{i-1} c_j = \sum_{j=1}^{i-1} a_j' - \sum_{j=1}^{i-1} b_j \leq \sum_{j=1}^{i-1} a_j'$$

$a_i' \geq \sum_{j=1}^{i-1} a_j' + v$이므로

$$c_i = a_i' - b_i \geq \sum_{j=1}^{i-1} a_j' + v - b_i \geq \sum_{j=1}^{i-1} a_j' \geq \sum_{j=1}^{i-1} c_j \; U > \sum_{i=1}^{n} a_i' \geq \sum_{i=1}^{n} c_i$$

b_i가 $\{0 \leq b_i \leq v\}$내에 제한되어 있을지라도($\times W \bmod U$)의 변환을 통하여 $[0, vW]$에 임의로 분포되어 있어 고밀도로 된다. 고밀도 수열을 생성시키는 절차는 아래와 같다.

단계 1 : 임의의 초증가 수열 $A' = (a_1', a_2', \cdots, a_n')$, $\gcd(W, U) = 1$인 W, U을 선택하고, $v = \lfloor W / U \rfloor$(여기서 $\lfloor x \rfloor$는 floor function)

단계 2 : 암호화 키 $a_i = a_i' \times W \bmod U$ 계산.

단계 3 : $h_i = a_i \bmod W$를 계산하여 고밀도 knapsack 을 생성한다.

단계 4 : $b_i = [a_i / W]$, $0 \leq b_i \leq v$, 를 계산하고, 복호화 키 $c_i = a_i' - b_i$를 계산한다.

$c_i = a_i' - b_i$를 계산한다.

c_i가 초증가 수열임을 쉽게 보일 수 있고 $U > \sum_{i=1}^{n} c_i$를 만족한다. 원래 Merkle-Hellman의 키가 $[1, U]$에 분포되는 반면, 같은 security로 a_i'는 $[1, W]$내에 분포된다. 밀도는 U내에서 W를 적절히 선택함으로 조절할 수 있다.

Merkle Hellman은 $n=200$일 때 a_i'이 $[(2^{n+1}-1)2^{200}+1, (2^{n+1})2^{200}]$에 존재하고 U는 $[2^{401}+1, 2^{402}-1]$에 존재하고, W는 $[2, U-2]$에 존재할 것을 제안했다. 위의 경우 고밀도 알고리즘에 의한 밀도는 0.94 정도이어서 기존의 Merkle Helman 의 0.5에 비해 월등히 크다.

Ⅳ. 선형 이동 Knapsack 암호화 시스템

위의 고밀도 knapsack은 Merkle-Hellman scheme의 특별한 경우이므로 Shamir나 Brickell, Adleman과 Lagarias의 알고리즘에 의해 깨질 수 있다. 따라서 선형 이동 방법에 의해 수정될 수 있다. 이것은 한번 또는 여러번의 반복 operation 에 의해서는 얻을 수 없는 방법이다. 따라서 기존의 분석방법으로는 이것을 break 할 수 없다. 선형 이동 knapsack 암호화시스템은 다음과 같다.

단계 1 : 임의의 초증가 knapsack 수열 $A' = (a_1', a_2', \cdots, a_n')$선택

단계 2 : 고밀도 knapsack 알고리즘을 이용한

hard knapsack 수열 A로 변환

단계 3 : random 이진 수열 $Q=(q_1, q_2, \cdots, q_n)$ 선택

정수 k선택, $o<k<\min(a_i)$ for $q_i=1$

공개키 $e_i=a_i-kq_i$계산

따라서 복호화 키는 (A ′, k, W, U)가 된다. 만일 수신자가 $S=\sum_{i=1}^{n} a_i m_i$, 여기서 $M=(m_1, m_2, \cdots, m_n)$을 받으면 기존의 방법에 의해 S를 복호화할 수 있다. 그러나 $S'=\sum_{i=1}^{n} e_i m_i$를 받으면 다음의 식에 의거하여 복호화 해야 한다.

$$S \times W^{-1}=(\sum_{i=1}^{n} a_i m_i) \times W^{-1} \bmod U$$
$$=\sum_{i=1}^{n} (e_i+kq_i)\, m_i \times W^{-1} \bmod U$$
$$=S' \times W^{-1}+r \times \sum_{i=1}^{n} q_i m_i \bmod U$$

여기서 $r=kW^{-1} \bmod U$

Q와 M이 이진 수열이므로 윗식의 $\sum_{i=1}^{n} q_i m_i$는 $o \leqq \sum_{i=1}^{n} q_i m_i \leqq y \leqq n$ (여기서, $y=\sum_{i=1}^{n} q_i$)의 범위에 존재한다. 따라서 수신자는 일대일이면 위에서 추측한 해가 정상적인 암호화 과정에 의해 입증될 수 있다. Shamir의 이론에 따르면 n 차원을 갖고 modular U를 갖는 random modular knapsack 시스템은 $n<(\log_2 U)/2$일때 일대일이기 쉽고 그렇지 않으면 일대일이 아니다. 따라서 U가 2^{2n}보다 큰 부분에서 선택되면 일대일이기 쉽다.

암호문 S ′는 비밀키 pair(W^{-1}, U)에 의해 S로 변환된다. 암호화 키가 이동됐으므로 수신자는 S에 $j \times r$, $j=0, 1, \cdots, y$를 더하고 초증가 수열 A ′를 사용하여 메세지 m_j, $j=0, 1, \cdots, y$를 복호화 한다. m_j는 올바른 메세지 M을 포함하므로 이것을 선택해내야 한다. 따라서 메세지 m_j를 다시 암호화하여 원래 메세지를 찾아낸다. 복호화의 연산수는 1회의 승산, n회의감산, n+1회의 합산이 된다.

더 좋은 불확정성을 위하여 Q는 $\sum_{i=1}^{n} q_i = y = n/2$이거나 q_i가 $\{-1, 0, 1\}$이고 $\sum_{i=1}^{n} q_i = 0$인 $Q=(q_1, q_2, \cdots, q_n)$의 형태가 좋다고 제안한다.

$E=(e_1, e_2, \cdots, e_n)$을 ($W^{-1}$, U)에 의해 역변환을 수행하였을 때, 쉬운 knapsack 수열을 형성하지 않음을 보면

$$e_i \times W^{-1}=(a_i-kq_i) \times W^{-1} \bmod U$$
$$=a_i \times W^{-1}-kq_i \times W^{-1} \bmod U$$
$$=\begin{cases} U+a_i'-r & \text{for } q_i=1,\ a_i'<r \\ a_i'-r & \text{for } q_i=1,\ a_i' \geqq r \\ a_i' & \text{for } q_i=0 \end{cases}$$

이 된다.

여기서 $r=k \times W^{-1} \bmod U$

분명히 암호 사용자가 r을 잘 선택하여 역변환이 쉬운 수열이 되지 않게 할 수 있다.

사실 E가 다른 쉬운 수열이나 $U>\sum_{i=1}^{n} a_i'$를 만족하는 random 수열로부터 사상된다면 기존의 knapsack과 다를 것이 없다. 어느 random 수열 k가 modular 변환하에 초증가 수열의 사상이 될 확률은 $2\exp(-{}_nC_2) \times (\sum_{i=1}^{n} e_i)^2$보다 작다. 또 임의의 수열이 배른 임의 수열의 사상이 될 확률은 매우 작음은 정리 2로부터 설명이 된다.

정리 2 : $E=(e_1, e_2, \cdots, e_n)$이 균일하게 분포되어 있고 [1, U] 내의 독립난변수라 할때, modular transformation의해 E가 수열 H의 사상이 된 확률은

$P<(\sum_{i=1}^{n} e_i)/n!<nU/n!$을 만족한다.

증명 : [1, U]로부터 n개의 정수 h_i를 임의로 선택한다면 $\sum_{i=1}^{n} h_i<U$을 만족하는 확률 P는 $P<1/n!$

함수 $g_i(t)=e_i t-s_i U$에서 $\sum_{i=1}^{n} e_i$ interval에 대해 많아야 $\sum_{i=1}^{n} e_i$ minima가 있다. 여기서 $s_i=[e_i \cdot t/U]$ 이다. 그러므로 [1, U] 사이의 모든점들을 테스트 해보면 modular transformation(w_j, U)가 있으면, $1<j<e_i'$에 존재한다. 따라서 성공확률의 범위는

$P<(\sum_{i=1}^{n} e_i)/n!<nU/n!$이 된다.

따라서 n=200 일때 선형이동 키 e_1가 초증가 수열의 확률은 2^{4536} 보다 작고 random 수열의 image 일 확률은 10^{95} 보다 작다. 달리 말하면 knapsack problem의 worst case로 떨어질 확률이 매우 크다. 그리고 다른 키 생성 알고리즘으로는 얻을 수 없고 기존의 Merkle Hellman의 knapsack은 이 알고리즘에서 $q_1=0$ for all i 인 특별한 경우이다.

V. Knapsack의 Break Test

본 절에서는 기존의 Merkle-Hellman의 knapsack과 선형이동 knapsack의 break test 결과를 제시한다. 사용한 알고리즘은 Brickell 이 제시한 L^3 basis reduction을 사용한 attack 방법과 Lagarias와 Odlyzko의 Short Vetcor 알고리즘이다. Brickell의 방법과 Lagarias, Odlyzko의 방법은 근본적으로 같은 원리로 동작한다. Brickell의 방법은 해석하기는 곤란하지만 하나의 공개키 A에 대하여 break를 성공하면 같은 키로 암호화한 S에 대해 부분합 문제를 $O(n^2(\log(S+|a|))^2)$의 시간 이하에 풀 수 있다.[10]

방법 1 : Brickell의 저밀도 knapsack의 break 알고리즘

Brickell의 방법은 기본적으로 density가 $1/\log_2 n$ 보다 작은 knapsack에 적용되고 더 큰것은 density를 낮추어 이전의 break 알고리즘을 적용한다. 이 방법은 주어진 weight $a_1, a_2, \cdots, a_n$에 대해 $O(n^4(\log n)^3)$의 연산을 요한다.[7] 구현한 알고리즘의 흐름도는 그림 1과 같다.

밀도가 $1/\log_2 n$ 보다 클 때에는 $W<U$, $n^n<U<2n^n$, $\gcd(W, U)=1$인 W, U를 선택하여

$b_i=a_i W \bmod U \quad 1\leq i\leq n$

을 만든다. 이때 b_i는 least nonnegative residue라 한다. Lattice는 공개키 vector$(a_1, a_2, \cdots, a_n)$에 대하여 다음의 vector로 생성된 R^n 내의 lattice를 만든다.

$$\begin{bmatrix} 1 \\ na_2 \\ na_3 \\ \cdot \\ \cdot \\ \cdot \\ na_n \end{bmatrix} \quad \begin{bmatrix} 0 \\ na_1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} \quad \cdots\cdots \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ na_1 \end{bmatrix}$$

Lenstra-Lenstra-Lovasz(L^3)에 의한 lattice reduction은 $O(n^4\log B)$, 여기서 $B\in R$, $B>2$이고 $|b_i|^2<B$ for $1\leq i\leq n$,의 산술 연산을 필요로 한다.[9] Reduction 알고리즘은 lattice에 vector $b_1, b_2, \cdots, b_n\in R^n$에 대해 Gram-Schmidt orthogonalization process를 거쳐 vectors b_i^*와 u_{ij}를 다음의 계산으로 구한다.

$$b_i^*=b_i-\sum_{j=1}^{i-1}u_{ij}b_j^*$$
$$u_{ij}=(b_i, b_j^*)/(b_j^*, b_j^*) \quad (,) : \text{inner product}$$

lattice L에 대해 기저(basis) $b_1, b_2, \cdots, b_n$이 reduce 됐다 함은 아래의 식을 만족한다.

$$|u_{ij}|\leq 1/2, \text{ for } 1\leq j<i\leq n$$
$$|b_i^*+u_{i,i-1}b_{i-1}^*|^2\geq y|b_{i-1}|^2 \text{ for } 1<i\leq n$$
$|\ |$: Euclidean length, $1/4<y<1$

reduction 과정은 최초 초기화 과정에서 $|b_i^*|^2$과 u_{ij}를 구한 후 이 두 값이 위의 두 조건을 만족하도록 b_i를 조절한다. 과정중 b_i는 변하지만 L에 대해 기저를 여전히 형성한다.

Short enough는 L내의 vector $X=(x_1, x_2, \cdots, x_n)$에서 $\sum_{i=1}^{n}x_i<na_1$임을 검증함으로 이루어진다. Vector X가 short enough이면 $a_1, a_2, \cdots, a_n$의 $x_i \bmod a_i$에 의한 modular mapping은 small sum property를 갖는다. 따라서 L내의 모든 vector가 short enough이면 $a_1, \cdots, a_n$에 의한 $n-1$개의 independent SSMM을 찾을 수 있다. $(W_1, U_1), \cdots, (W_{n-1}, U_{n-1})$을 지금까지 절차에

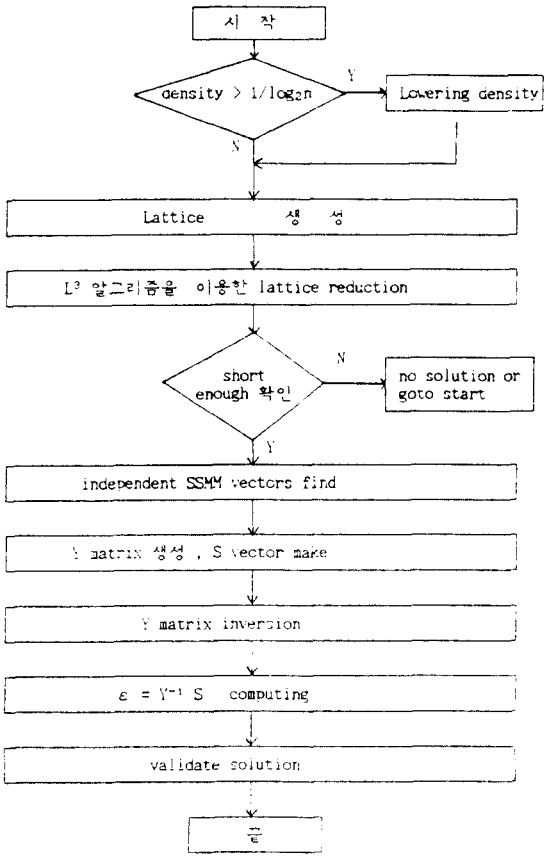


그림 1.Brickell의 저밀도 Knapsack의 Break 알고리즘

의해 구한 n−1개의 independent SSMM이라 하면 Y matrix는 다음과 같이 구할 수 있다.

$$Y=\begin{cases} a_1 & a_2 & \cdots & a_n \\ a_1W_1 \bmod U_1 & a_2W_1 \bmod U_1 & \cdots & a_nW_1 \bmod U_1 \\ \vdots & \vdots & & \vdots \\ a_1W_{n-1} \bmod U_{n-1} & a_2W_{n-1} \bmod U_{n-1} & \cdots & a_nW_{n-1} \bmod U_{n-1} \end{cases}$$

$\sum_{i=1}^{n}\varepsilon_i a_i=S$(여기서 ε_i는 0−1)에서 $S''=SW \bmod U$이고

$$S''=\begin{cases} S' & \text{if } S'\leq B \\ S'-U & \text{if } S'>B \text{ 이면} \end{cases}$$

$S''=\sum_{i=1}^{n}\varepsilon_i b_i$ 이므로

$\sum_{i=1}^{n}\varepsilon_i(a_iW_j \bmod U_j)=s_j''$ $1\leq j\leq n-1$
$S=(s, s_1'', \cdots, s_{n-1}'')$라면

$Y\varepsilon=S$, 따라서 $\varepsilon=Y^{-1}S$

식의 계산에 의해 ε가 0−1 vector인지 $\varepsilon \cdot a=S$가 되는지를 보아 해의 진위여부를 검증할 수 있다. 또한 밀도를 줄였을때, $S'=SW \bmod U$ 이라하면 (S'는 nonnegative residue)이라 한다.

$\sum_{i=1}^{n}\varepsilon_i b_i \in \{S', S'+U, \cdots, S'+(n-1)U\}$

여기서 b_i는 a_i의 nonnegative residue.

ε가 위식에 대해 만족하는 모든 해를 찾고 원래 weight에 적용하여 S가 나오는지 시험하여 해의 진위를 검증한다.

지금까지의 절차에 따라 알고리즘을 구현하여 기존의 Merkle Hellman knapsack 시스템과 선형이농 knapsack 알고리즘에 적용한 결과는 표 1과 같다.

표 1. Brickell의 방법을 이용한 Merkle-Hellman과 선형이농 Knapsack의 Break Test 결과

차원	density	시행수	Merkle Hellman knapsack		선형 이농 knapsack	
n	M H shift		success	fail	success	fail
3	0.51 0.59	30	30		25	5
4	0.42 0.52	30	30		24	6
5	0.46 0.55	30	30		25	5
6	0.5 0.57	30	30		24	6
7	0.47 0.58	30	30		22	8

방법 2 : Lagarias와 Odlyzko의 Short Vector 알고리즘

Short Vector 알고리즘은 부분합 문제를 공개키 A와 메세지 합 S로 구성한 lattice L 내에의 short vector e를 찾는 문제로 변환한다. short vector를 찾기 위해 구성된 lattice에 L^3 basis reduction 알고리즘을 적용하여 e로부터 해를 구하는 방법이다. 알고리즘은 아래와 같다.

단계 1 : n+1 차원의 정수 lattice L=L(A, S)을 위한 basis $[b_1, b_2, \cdots, b_{n+1}]$를 아래와 같이 구한다.

$b_1=(1, 0, \cdots 0, -a_1)$
$b_2=(0, 1, \cdots 0, a_2)$

$b_n=(0, 0, \cdots 1, -a_n)$
$b_{n+1}=(0, 0, \cdots 0, S)$

단계 2 : L^3 알고리즘을 사용하여 L의 reduced basis $[b_1^*, b_2^*, \cdots, b_{n+1}^*]$를 찾는다.

단계 3 : $b_i=(b_{i,1}^*, \cdots, b_{i,n+1}^*)$가 $1 \leq j \leq n$에 대해 모든 b_{ij}^*가 0이나 고정수 λ로만 구성되어 있는 지를 체크하고 그러한 b_i^*에 대해 $x_j=\lambda^{-1}b_{i,j}^*$에서 vector x가 해인지를 보아 해이면 종료하고 아니면 계속한다.

단계 4 : $S'=\sum_{i=1}^{n} a_i - S$로 S를 대체하고 단계 1-단계 3을 계속한다. 그리고 종료한다.

실제 SV 알고리즘은 $S<\sum_{i=1}^{n} a_i$이고 $\max(a_i) \leq B$라 할때 많아야 $O(n^6(\log nB)^3)$의 bit operation 내에 종료하는 것으로 알려져 있다.[12] SV 알고리즘을 사용하여 Merkle Hellman knapsack과 선형이동 knapsack의 break test 결과는 표 2와 같다.

본 알고리즘은 APOLLO Workstation 상에서 C언어로 구현하였다. Brickell의 경우 SSMM의 pair 계산에 있어 차원이 높아짐에 따라 시험에 precision의 문제가 있었다. 이것은 multiprecision을 제공하는 장비에서 시험하는 것이 요망된다. 하지만 낮은 차원에서도 선형이동 knapsack에는 저밀도 attack이 쉽게 적용되지 않음을 볼 수 있다. 차원이 높아지면 정리 2에 따라 break될 확률이 급격히 적어진다.

표 2. Lagarias와 Oldyzko의 방법을 이용한 Knapsack의 break Test 결과

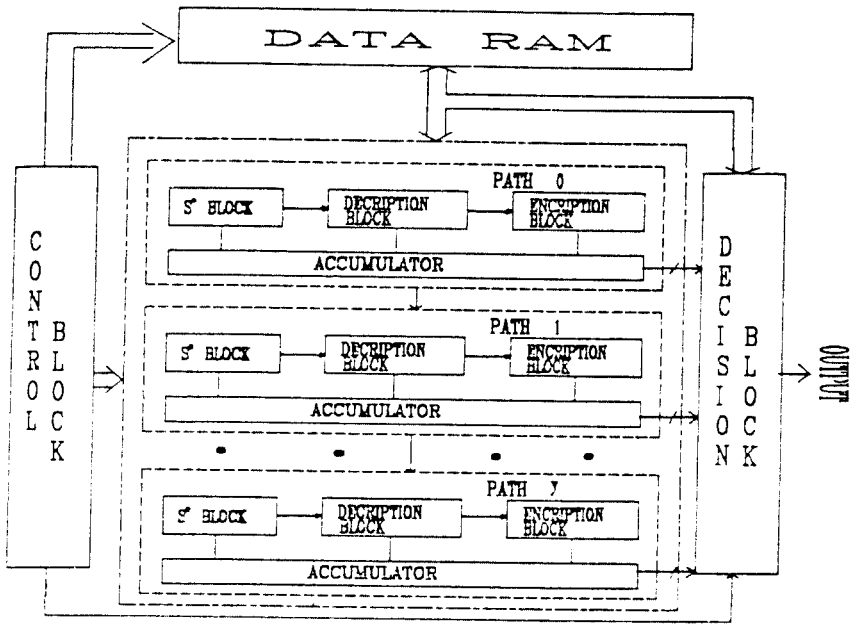
Dimension	log B	Merkle Hellman knapsack			선형 이동 knapsack		
		density	test 수	success	density	test 수	success
5	5	0.51	30	30	0.56	30	25
	8	0.39	30	30	0.42	30	29
10	10	0.5	30	30	0.73	30	24
	12	0.45	30	30	0.63	30	28
15	10	0.6	30	30	0.89	30	6
	15	0.5	30	30	0.79	30	13
16	14	0.53	30	30	0.89	30	0
	16	0.5	30	30	0.8	30	0
20	20				0.84	20	0

Ⅵ. 선형이동 knapsack 시스템의 구현을 위한 병렬구조

본 절에서는 Ⅴ에서 제시한 선형이동 knapsack 암호화시스템의 병렬구현을 위한 구조를 제안한다. knapsack 시스템은 RSA와 같이 exponentiation cipher에 비해 메모리 공간이 문제로 대두된다. n=100일 경우 200×100=20k의 공간이 weight를 위하여 필요하다. 고밀도 knapsack이나 선형이동의 경우 공간이 적게 소모되지만 병렬처리 구조시는 각각의 경로에 따라 고유 메모리가 필요하므로 n=100일 경우 y=100/2=50개의 병렬 경로가 소요됨으로 많은 메모리 공간이 요구된다. 따라서 본 제안에서는 각 경로에서 회귀시간이 적어 전체 성능에 영향이 적은 것은 순차적으로 하고 복호화나 암호화와 같이 큰 영향을 미치는 부분은 동기식으로 병렬 수행할 수 있는 구조를 설계하였다.

전체 구조는 그림 2와 같고, Data RAM에는 필요한 계수 S', $\bar{S}$, r, k, W, U(B, E, A : 필요에 따라 외부 RAM에 저장)가 저장되어 있다.

전체 구성은 추정합계 $S''=\bar{S}+i \cdot r(i=0, \cdots,$



n/2 τd 3n τd nτd n/2 τd 3n τd

그림 2. 병렬 처리를 위한 전체 블럭 구성도

y)를 연산하는 블럭, 복호화 블럭, 암호화 블럭, 결정블럭, 제어 블럭의 5개 블럭으로 구성된다.

-추정합계 연산 블럭은 $S''=\overline{S}+yr$을 계산하면 되므로 누산기와 n bit 가산기가 필요하다. 추정합계 연산결과는 각 path에서 순차적으로 (path 0 : $So''=\overline{S}+Or$, path 1 : $S_1''=So''+r$, …, path y : $S_y''=S_{y-1}''+r$) 누산되어져 내려간다.

-복호화 블럭은 위의 S''을 받고 비밀키 A'을 이용하여 snap의 연산[1]을 행하므로 n번의 합산이 필요하다. 구조를 그림 3에 보였다. modulo n의 하향 계수기로 각 반복을 제어하고 계수기의 출력에 따라 동기식으로 RAM으로부터 A'의 값을 읽어 A'와 S register에서 비교기의 출력에 따라 감산을 수행하고 메세지 register에 결과를 출력한다.

-암호화 블럭은 공개키 E가 필요하다. 제어신호

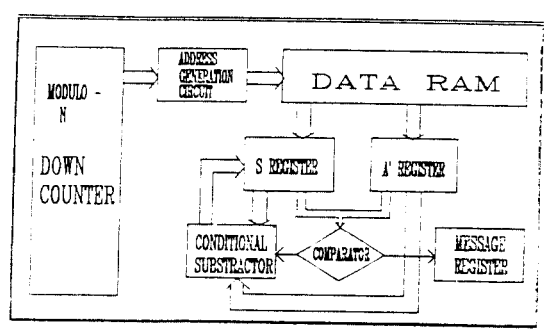


그림 3. 복호화 블럭 구성도

에 따라 E와 전단에서 계산된 메세지 bit에 따라 누산기에서 암호화 연산을 수행한다.

-결정블럭

결정블럭은 각 path의 메세지 bit 데이타의 출력을 결정하는 MUX 회로, MUX 제어신호, 비교기로 구성된다. 비교기는 단순히 XOR 연산을

수행하면 된다. 각 bit을 몇개의 group으로 나누어 회로를 구성하면 효율적이다.

－제어블럭

데이터를 RAM으로 부터 읽고 쓰기위한 동작, 각 단계별 동작신호, 동기신호등의 발생이 필요하다. 이는 기본 클럭 발생기, 분주회로, 기타 신호발생을 위한 random logic으로 구성할 수 있다.

각 연산블럭의 최대지연시간을 τ_d unit time이라 할때 n bit 메세지 추정합계연산 블럭과 결정블럭은 직렬 및 병렬구조에서 각각 $n/2 \cdot \tau_d$unit time이 소요되나(y=n/2라 할때) 복호화 블럭에서는 비교, bit loading, 감산의 소요시간을 고려하면 직렬의 경우 $3n^2\tau_d$, 병렬의 경우 $3n\tau_d$, 암호화 블럭에서 직렬의 경우 $n^2\tau_d$, 병렬의 경우 $5n\tau_d$가 된다. 만일 전 병렬의 경우 $(4n+2)\tau_d$의 동작이 요구된다. 따라서 제안한 구조가 메모리 공간의 제한 때문에 전 병렬의 구조는 아니지만 회귀시간이 적은 부분만을 직렬화 하였으므로 전체적인 성능면에서 크게 떨어지지 않음을 알 수 있다.

Ⅶ. 결 론

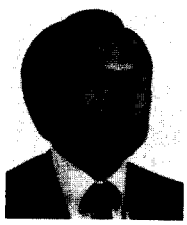
본 논문에서는 Merkle-Hellman이 제시한 공개키 시스템을 개선한 선형이동 knapsack 알고리즘을 설명했고 이의 성능 평가 및 구현을 위한 병렬구조를 제안하였다. Merkle Hellman의 knapsack이나 기타 여러 knapsack의 약점을 제시하고 생성된 암호키가 일대일임을 보장 못하는 것이 약점이지만 위의 기존 knapsack에 대한 단점을 보완할 수 있는 고밀도 knapsack과 선형이동 knapsack 시스템을 설명하였다. 그리고 Brickell이 제안한 L^3 basis reduction을 사용한 저밀도 knapsack break 및 Lagarias와 Odlyzko의 Short Vector 알고리즘을 구현하여 Merkle-Hellman knapsack과 선형이동 knapsack에 적용하여 평가하므로 선형이동 knapsack 시스템의 우수성을 입증하였다. 또한 이 선형 이동 knapsack 시스템의 구현을 위해 병렬 구조를 제안하였다. Knapsack 시스템은 RSA와 같이 exponentiation cipher 보다 더 많은 메모리 공간을 필요로 하지만 속도는 월등히 빨라진다. 선형이동 시스템은 데이터의 압축으로 메모리 공간이 덜 소모되나 실제 병렬 구현에 있어서는 여전히 많은 공간을 필요로 한다. 따라서 전 병렬(fully parallel) 구조를 변형하여 많은 시간이 요구되는 부분에 대해 동기식 병렬처리하도록 구성하였다. 이를 통해 분산체제 시스템, 기타 고속의 보안이 요구되는 컴퓨터 및 통신망의 물리적인 계층에 적용할 수 있으리라 사료된다. 따라서 앞으로 제안한 구조의 VLSI화의 연구가 기대된다.

본 연구는 1990년 산학재단학술 연구비 지원에 의하여 연구되었음.

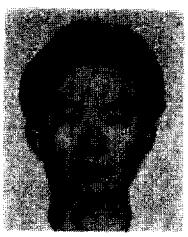
참 고 문 헌

1. Dorothy Elizabeth Robling Denning, "Cryptography and Data Security", Addison Wesley Publishing Co., 1982.
2. Jennifer Seberry, Josef Pieprzyk, "CRYPTOGRAPHY : An Introduction to Computer Security", Prentice Hall, 1989.
3. Dominic Welsh, "Codes and Cryptography", Clarendon Press, 1988.
4. R. Merkle and M. E. Hellman, "Hiding information and signatures in trapdoor knapsack", IEEE Trans. Inform. Theory, vol. IT-24, pp. 525-530, Sept 1978.
5. Chi-Sung Laih, Jau-Yien Lee, etc, "Linearly Shift Knapsack Public-Key Crytosystem", Trans of Comm. IEEE, 1989.
6. E. F. Brickell, "SOLVING LOW DENSITY KNAPSACKS", Proceedings of Crypto, pp. 25-37, 1983.
7. J.C. Lagarias, "KNAPSACK PUBLIC KEY CRYPTOSYSTEMS AND DIOPHANTINE APPROXIMATION", Proceedings of Crypto, pp 3-23, 1983.
8. A. K. Lenstra, H. W. Lenstra, Jr. , and L. Lovasz, "Factoring Polynomials with Rational Coefficients",

Math. Ann. 261, pp 515-534, 1982.
9. J.C. Lagarias and A.M. Odlyzko, "Solving Low Density Subset Sum Problems", Journal of the Assoc. comp. Mach Vol. 32, No.1 Jan 1985, pp 229-246.
10. E.F. Brickell and A.M. Odlyzko, "Cryptanalysis : A Survey of Recent Results", Proc. of IEEE, Vol. 7 6, No. 5, May 1988.
11. N. Weste, Kamran Eshranghian, "Principles of CMOS VLSI Design", Addison Wesley, 1985.
12. M. Annaratone, "Digital CMOS Circuit Design", Kluwer Academic Publishers, 1986.



車 均 鉉(Kyun Hyon TCHAH) 正會員
1939年 3月 26日生
1965年 : 서울大學校 工學士
1967年 : 美國일리노이大學校 工學碩士學位 取得
1976年 : 서울大學校 工學博士學位 取得
1977年 ~ 現在 : 高麗大學校 電子電算工學科 敎授



白 寅 天(In Cheon PAIK) 正會員
1963年 1月 14日生
1985年 : 高麗大 電子科 卒業
1987年 : 高麗大 大學院 電子科(碩士)
1987年 ~ 現在 : 高麗大 大學院 電子科 博士課程

白 京 甲(Kyeong Kap PAEK) 正會員
1965年 11月 8日生
1987年 2月 : 高麗大 電子工學科 卒業
1990年 2月 : 高麗大 大學院 電子工學科 卒業(工學碩士)
1990年 ~ 現在 : 高麗大 大學院 電子工學科 博士課程
※ 主關心分野 : VLSI / CAD, 通信시스템 設計 등

朴 商 奉(Sang Bong PARK) 正會員
1962年 3月 8日生
1985年 : 光云大 電子材料科 卒業
1987年 : 高麗大 大學院 電子工學科 卒業(工學碩士)
1987年 ~ 現在 : 高麗大 大學院 電子工學科 博士課程