通信情報保護學會誌
第8卷 第4號. 1998. 12

RSA 암호방식의 안전성에 관한 연구

A Study on the Notion of Security for Public-key Encryption Schemes

조 동 욱*, 김 영 수*, 정 권 성*, 원 동 호*

요 약

본 고에서는 큰 수의 인수분해가 어렵다는 사실에 기반한 암호방식인 RSA 암호시스템의 안전성에 영향을 미치는 여러 요소에 대하여 고찰하였다. 먼저 RSA 시스템에 대하여 간단하게 살펴보고, 다음으로 모듈러 n의 인수분해를 통한 공격 방식을 살펴본다. 그리고 마지막으로 RSA 시스템에 대한 공격 방식을 homomorphic 공격과 다항식 공격으로 나누어 살펴본다.

1. 서 론

최근 컴퓨터와 정보통신기술이 발달하면서 정보의 축적, 처리 전달 능력이 획기적으로 증대되고 있다. 이와 비례하여 정보가 지닌 가치의 중요성이 날로 배가되고 있는 실정이다. 따라서, 정보의 불법 유출, 삭제, 수정 등에 대한 적절한 보호조치가 없다면 이러한 불법적인 사고로 인하여 개인 사생활 침해뿐만 아니라 막대한 경제적 손실을 가져올 수 있으므로, 정보 시스템의 정상적인 기능유지는 효율적이고 건전한 정보화 사회의 밑거름인 동시에 반드시 보호되어져야 한다.암호시스템은 크게 관용키 암호시스템과 공개키 암호시스템으로 나눌 수 있으며, 이중 공개키 암호 시스템은 소인수 분해 문제의 어려움에 기반을 둔 RSA와 이산대수에 기반을 둔 많은 시스템들이 사용되고 있다. 본고에서는 RSA의 안전성에 관한 문제에 대하여 살펴보고자 한다. 먼저 2절에서는 RSA 암호 시스템에 대하여 간략하게 기술하였으며, 3절에서는 큰 수의 소인수 분해의 어려움에 근거를 둔 RSA의 인수분해 방법을 살펴보며, 4절에서는 RSA 시스템에 대한 공격 방식을 homomorphic 특성을 이용한 공격과 다항식 공격 방식으로 나누어 살펴보고자 한다.

* 성균관대학교

※ 본고는 1998년 한국과학재단의 공모과제(97-01-13-01-05)에 의해 연구되었음.

2. RSA 시스템

1976년 Diffie와 Hellman이 공개키 암호 방

식의 개념을 발표한 후[7], 1978년 MIT의 Rivest, Shamir와 Adleman이 처음 RSA라고 하는 공개키 암호 방식을 제안하였다. 이들은 큰 합성수의 소인수 분해의 어려움을 이용하여 RSA 암호 방식을 실현하였다.

RSA 시스템에 대하여 간략하게 소개하면 다음과 같다.

- 시스템 설정 (각 사용자)
  - 큰 소수 p, q를 선택하여 두 수를 곱하여 n을 구한다.
  - $(p-1)(q-1)$과 서로 소인 e를 선택한다.
  - $e \cdot d \equiv 1 \pmod{\phi(n)}$인 d를 선택한다.
  - n과 d는 공개하고, 소인수 p, q, d는 비밀 정보로서 비밀리에 간직한다.
- 사용자 A (비밀 메시지 m을 B에게 안전하게 보내기를 원하는 사용자)
  - B의 공개정보를 얻어서, $c \equiv m^{e_B} \pmod{n_B}$로 암호화한다.
  - B에게 c를 전송한다.
- 사용자 B
  - A로부터 암호문 c를 받아서, $m \equiv c^{d_B} \bmod n_B$을 이용하여 복호화 한다.

가입자 A	공개정보 e_A, n_A, e_B, n_B	가입자 B
p_A, q_A $n_A = p_A \cdot q_A$ $\phi(n_A) = (p_A-1)(q_A-1)$ $\gcd(e_A, \phi(n_A)) = 1$ $e_A \cdot d_A \equiv 1 \bmod \phi(n_A)$ $c \equiv m^{e_B} \bmod n_B$	C →	p_B, q_B $n_B = p_B \cdot q_B$ $\phi(n_B) = (p_B-1)(q_B-1)$ $\gcd(e_B, \phi(n_B)) = 1$ $e_B \cdot d_B \equiv 1 \bmod \phi(n_B)$ $m \equiv c^{d_B} \bmod n_B$

[그림 1] RSA 암호시스템

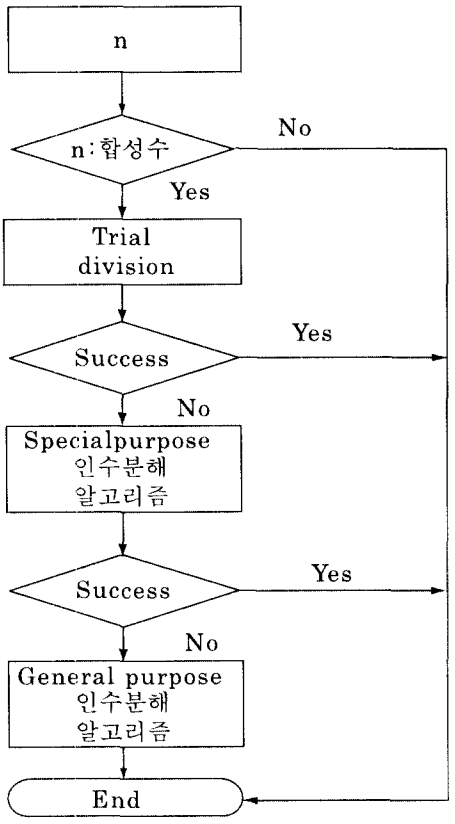
3. 인수분해

RSA 암호시스템은 큰 합성수의 인수분해가 어렵다는 가정에 기반을 두고 있으며, 인수분해가 가능하다면 RSA 암호시스템은 더 이상 안전하지 못할 것이다. 본 절에서는 인수분해의 절차와 인수분해 알고리즘들에 대해서 살펴본다.

3.1 일반적인 인수분해 절차

큰 수의 인수분해는 일반적으로 [그림 2]와 같이 다음 순서로 이루어진다.

1) n이 합성수인지를 검사한다.
2) n이 합성수로 판명되면 trial division을 수행하여 혹시 있을 지도 모를 작은 소인수를 찾는다.
3) Trial division에 실패하면, n의 소인수들이 가질지도 모를 특성을 이용하는 special purpose 인수분해 알고리즘을 이용한다.
4) Special purpose 인수분해 알고리즘으로 인수분해되지 않은 수는 general purpose 인수분해 알고리즘을 이용하여 인수분해한다.



[그림 2] 인수분해의 일반적인 절차

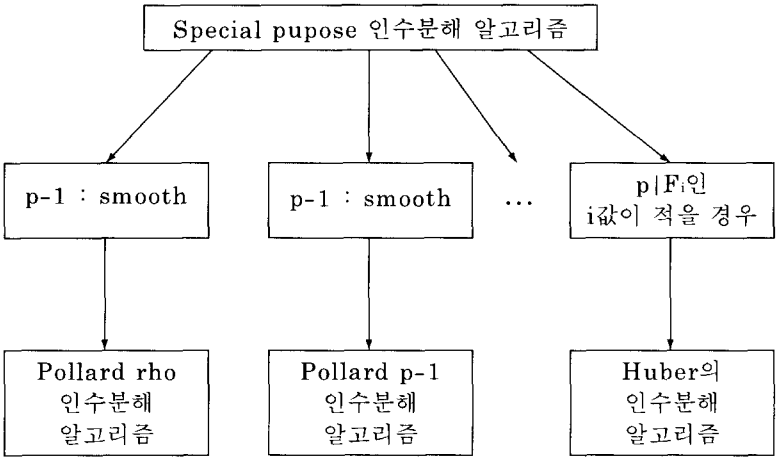
3.2 Trial division

(가) 알고리즘

- 입 력 : 합성수 n
- 출 력 : n의 trivial 소인수
- 과 정 : $\sqrt{n}$ 이하의 소수들로 n을 나누어서, n의 소인수를 찾는다.

3.3 Special purpose 인수분해 알고리즘

몇몇 인수분해 알고리즘들은, 정수 n이 특별한 특성을 가질 때 보다 잘 수행되는데, 이런 인수분해 알고리즘들을 special purpose 인수분해 알고리즘이라고 한다. Special purpose 인수분해 알고리즘의 수행시간은 보통 n의 길이보다는 n의 소인수들이 가지는 어떤 특징에 의존한다. 이 절에서는 special purpose 인수분해 알고리즘들 중 Pollard rho 인수분해 알고리즘, Pollard p-1 인수분해 알고리즘, 그리고 Huber의 인수분해 알고리즘에 대해서 알아보고자 한다.



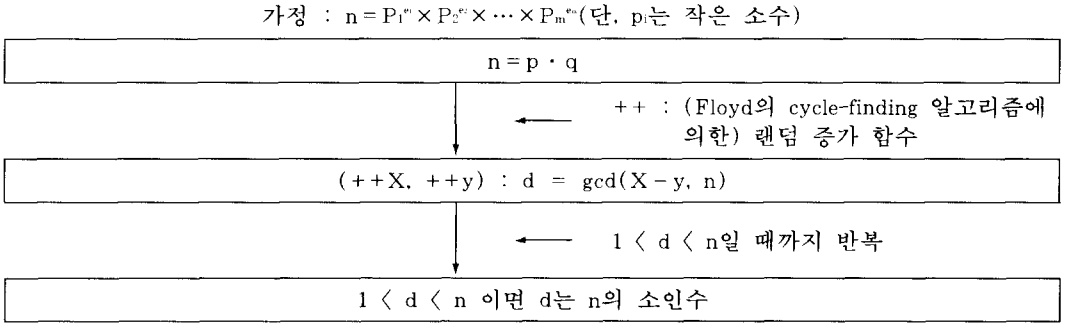
단, F_i는 Fibonacci 수열

[그림 3] Special purpose 인수분해 알고리즘

1) Pollard rho 인수분해 알고리즘[24]

(가) 개 요

Pollard rho 인수분해 알고리즘은 70년대 후반에 John M. Pollard에 의해서 개발되었으며, 합성수의 작은 소인수를 찾기 위한 special purpose 인수분해 알고리즘으로 Floyd의 cycle-finding 알고리즘을 이용하여 인수분해를 한다.[18]



[개념도 3.1] Pollard rho 인수분해 알고리즘

(나) 알고리즘

Floyd의 cycle-finding 알고리즘은 함수 $x_{i+1}=f(x_i)=x_i^2+1$을 이용하여 $x_m \equiv x_{2m} \bmod p$인 (x_m, x_{2m})의 쌍을 찾는 알고리즘이다.

Pollard의 인수분해 알고리즘은 실제 p를 모르므로 $\gcd(x_m - x_{2m}, n)$을 이용하여 n의 소인수 p를 찾는다.

- 입 력 : 합성수

- n출력 : n의 소인수
- 과 정 :
  ① a←2, b←2로 설정한다.
  ② $a \leftarrow a^2+1 \bmod n$, $b \leftarrow (b^2+1)^2+1 \bmod n$을 계산한다.
  ③ d=gcd(a − b, n)를 계산한다.
  ④ 1<d<n : d를 출력
  d=1 : 과정 ②로 돌아감
  d=n : 실패로 끝남

(라) 예 제

Pollard rho 인수분해 알고리즘을 사용하여 n=455459를 인수분해한 것은 다음과 같다.

① a←2, b←2로 설정한다.
② 알고리즘의 과정 ②, ③을 이용하여 다음 값들을 구한다.

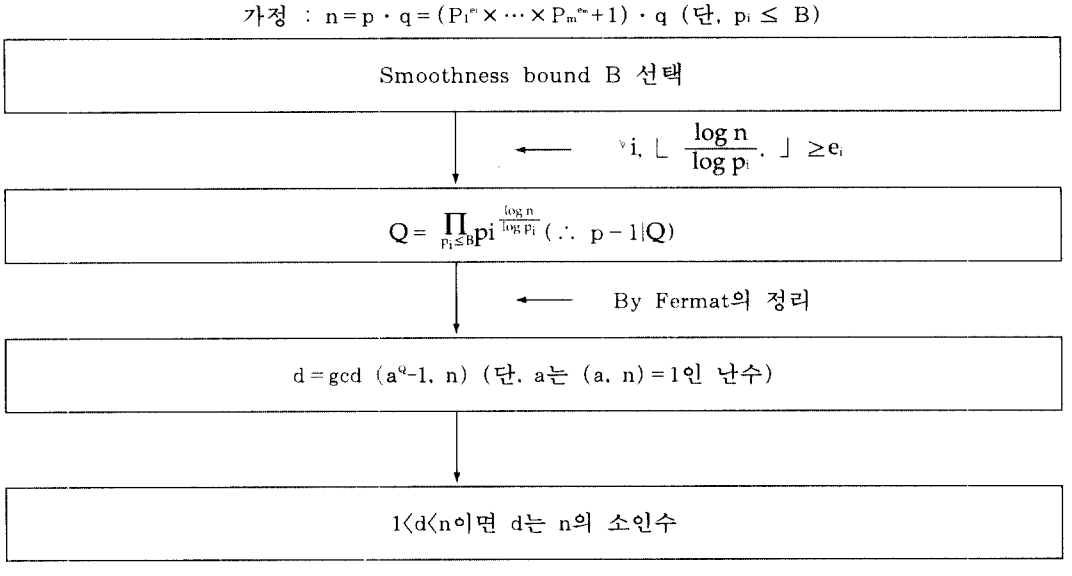
a	b	c
5	26	1
26	2871	1
677	179685	1
2871	155260	1
44380	416250	1
179685	43670	1
121634	164403	1
155260	247944	1
44576	68343	743

③ 1<d=743<7이므로 n=743 · 613이다.

2) Pollard p-1 인수분해 알고리즘[25]

(가) 개 요

Pollard p-1 인수분해 알고리즘은 1974년 John M. Pollard에 의해서 제안되었으며, p − 1이 B-smooth1) 인 합성수 n의 소인수를 효율적으로 찾는 special purpose 인수분해 알고리즘으로 Fermat의 정리를 이용한다. Fermat의 정리에 의하여 $a^{p-1} \equiv 1 \bmod p$이며, 이는 $a^{p-1}-1 \equiv 0 \bmod p$, 즉, $a^{p-1}-1=kp$이다. 따라서, $(a^{p-1}-1, n)=p$, 즉 n의 소인수이다.



[개념도 3.2] Pollard p-1 인수분해 알고리즘

1) 어떤 수 x를 B이하의 소수들의 곱으로 표현할 수 있을 때, x가 B-smooth하다고 한다.

(나) 알고리즘

- 입 력 : 합성수 n
- 출 력 : n의 non-trivial 소인수 d
- 과 정 :
  ① Smoothness bound B를 선택한다.
  ② $a \in_R Z_n - \{0, 1\}$를 선택, d=gcd(a, n)를 계산한다. 만일 $d \geq 2$ 이면 d를 출력한다.
  ③ 각 소수 $p_i < B$ 에 대하여 과정 ④, ⑤를 수행한다.
  ④ $l_i = \lfloor \frac{\log n}{\log p_i} \rfloor$, ⌋를 계산한다.
  ⑤ $a \leftarrow a^{p_i^{l_i}} \bmod n$를 계산한다.
  ⑥ d=gcd(a − 1, n)를 계산한다.
  ⑦ 만일 d=1이나 d=n이면, 실패로 알고리즘을 끝낸다. 그렇지 않으면 d를 출력한다.

(다) 예 제

Pollard p − 1 인수분해 알고리즘을 이용하여 n=19048567의 인수분해하는 과정은 다음과 같다.

① Smoothness bound B=19를 선택한다.
② a=3을 선택하고, gcd(3, n)=1을 계산한다.
③ $a \leftarrow a^{p_i^{l_i}} \bmod n$을 계산한다.

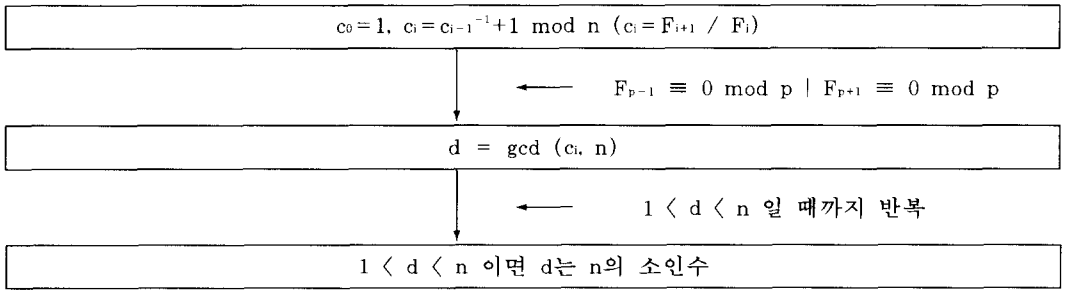
q	l	a
2	24	2293244
3	15	13555889
5	10	16937223
11	6	9685355
13	6	13271154
17	5	11406961
19	5	554506

④ d=gcd(554506 − 1, n)=5281 계산한다.
⑤ n의 두 소수 p=5281과 q=n/p=3607이다.

3) Huber의 인수분해 알고리즘[11]

(가) 개 요

이 알고리즘은 Klaus Huber가 제시한 인수분해 방법으로 모든 소수 p는 Fibonacci 수열 F_i에 대하여 $F_{p-1} \equiv 0 \bmod p$, 또는 $F_{p+1} \equiv 0 \bmod p$이다. 즉, 모든 소수 p에 대하여 $p \mid F_i$가 존재한다. 따라서, 이 알고리즘은 n의 소인수 p를 포함하는 Fibonacci수 F_{u_p}의 인덱스 u_p가 적을 경우에 인수분해를 쉽게 할 수 있다.



단, F_i : Fibonacci 수열

[개념도 3.3] Huber의 인수분해 알고리즘

(나) 알고리즘

- 입 력 : 합성수 n
- 출 력 : n의 non-trivial 소인수
- 과 정 :
  ① c ← 1를 입력한다.
  ② n의 소인수를 구할 때까지 절차 ③, ④를

되풀이한다.

③ $c \leftarrow c^{-1}+1 \pmod n$를 구한다.

④ $d=\gcd(c, n)$를 계산한다.

(다) 예 제

Huber의 인수분해 알고리즘을 이용하여 n=323의 소인수를 구한 결과는 아래와 같다.

① $c \leftarrow 1$를 입력한다.

② n의 소인수를 구할 때까지 알고리즘의 ③, ④를 되풀이하여 다음 값을 얻는다.

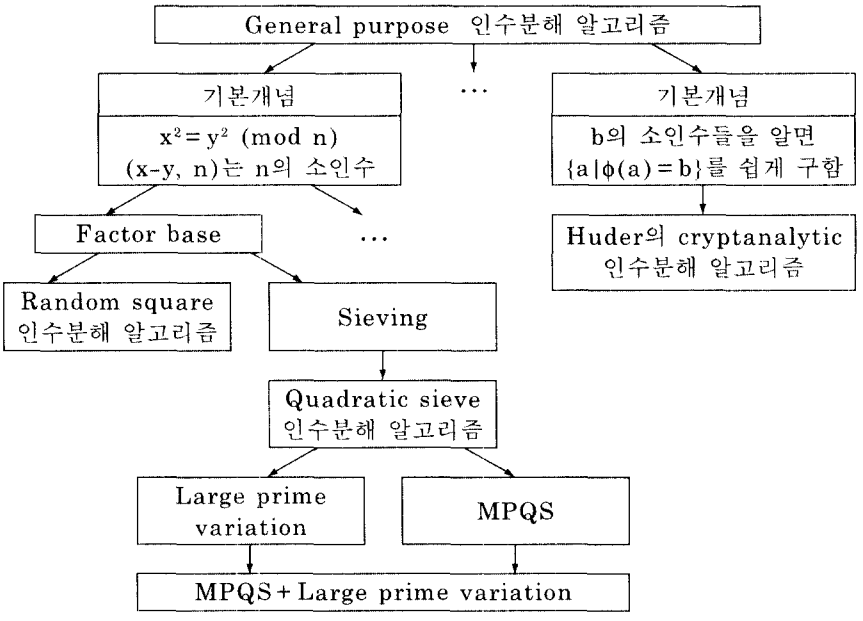
q	$c^{-1}+1 \bmod n$	a
1	2	1
2	163	1
163	217	1
217	260	1
260	42	1
42	101	1
101	17	17

③ 따라서 n의 소인수는 17과 19가 된다.

3.4 General purpose 인수분해 알고리즘

본 절에서는 인수분해 시간이 합성수 n의 길이에만 의존하는 general purpose 인수분해 알고리즘들 중 x, y와 n이 정수이고, $x^2 \equiv y^2 \bmod n$ $(x \not\equiv \pm y \bmod n)$이라면, $\gcd(x-y, n)$는 n의 non-trivial 소인수가 되는 성질과 factor base[2]을 이용하는 알고리즘들 중에서, random square 인수분해 알고리즘과, sieving[2] 기법을 이용한 quadratic sieve 인수분해 알고리즘, 그리고 이의 변형 방식들을 논한다.

[그림 4]는 본 절에서 다루는 인수분해 방식들과의 상호관계를 나타낸 것이다.



[그림 4] General Purpose 인수분해 알고리즘

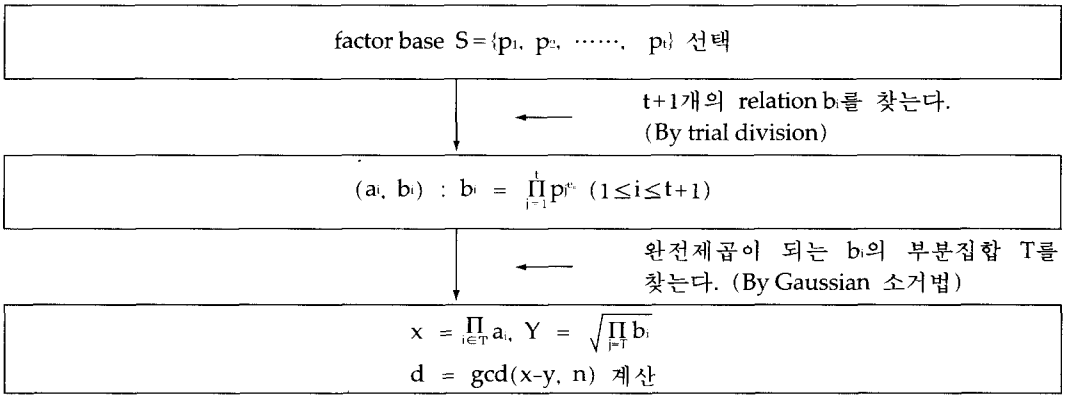
1) 전체 집합 N의 부분집합 S의 원소들의 곱으로 N의 모든 원소들을 표현할 수 있을 때, factor base라고 한다. 본 절의 general purpose 인수분해 알고리즘은 이차잉여를 이용하여 인수분해를 수행하므로, factor base $S = \{pi \mid (\frac{n}{p_i})=1, p_i \le B, p_i$는 소수$\}$ (단, B:Smoothness bound)를 이용하며, relation은 $\Pi p_i^{e_i}$의 형태로 나타난다.

2) Sieving이란 quadratic 다항식 $q(x)$가 $p_i|q(x)$이면, 임의의 상수 k에 대하여 $p_i|q(x+kp_i)$가 되는 특성을 이용하여 p_i-smooth를 검사할 $q(x)$값을 선택하는 것을 말한다.

1) Random square 인수분해 알고리즘

(가) 개 요

Random square 인수분해 알고리즘은 큰 합성수의 non-trivial 소인수를 찾는 general purpose 인수분해 알고리즘이다.



단, $a_i^2 = b_i \bmod n$

[개념도 3.4] Random square 인수분해 알고리즘

(나) 알고리즘

- 입 력 : 합성수 n
- 출 력 : n의 non-trivial 소인수
- 과 정 :
  ① 크기가 t인 factor base S를 선택한다.
  ② 과정 ③ - ⑤를 통하여 t+1 개의 relation을 찾는다.
  ③ a_i를 선택하고, $a_i^2 \equiv b_i \bmod n$을 계산한다.
  ④ $b_i = \prod_{j=1}^{t} p_j^{e_{ij}}$, $e_{ij} \ge 0$인지를 검사. 그렇지 않으면 과정③으로 되돌아간다.
  ⑤ $1 \le j \le t$에 대해 $e_{ij} = v_{ij} \bmod 2$인 $v_i = (v_{i1}, v_{i2}, \cdots, v_{it})$를 생성한다.
  ⑥ Gaussian 소거법을 사용하여 $T \subseteq \{1, 2, \cdots, t+1\}$에서 $\prod_{i \in T} v_i = 0$인 것들을 찾는다.
  ⑦ $x = \prod_{i \in T} a_i$, $y = \sqrt{\prod_{i \in T} b_i}$ 로 한다.
  ⑧ 만일 $x \equiv \pm y \bmod n$ 이면 과정 ⑥으로 되돌아간다.

그렇지 않으면 $d = \gcd(x-y, n)$를 계산하고 d를 출력한다.

(다) 예 제

Random square method를 사용한 n=15770708441의 인수분해는 다음과 같다.

① S=2, 3, 5, 7, 11, 13이라고 하고, 다음 세 수를 고려하면,

② 알고리즘 ② - ⑦을 적용하여 다음 세 수를 찾는다

$8340934156^2 \equiv 3 \times 7 \bmod n$

$12044942944^2 \equiv 2 \times 7 \times 13 \bmod n$

$2773700011^2 \equiv 2 \times 3 \times 13 \bmod n$

③ x, y값을 계산한다.

$(8340934156 \times 12044942944 \times 2773700011)^2 \equiv (2 \times 3 \times 7 \times 13)^2 \bmod n$

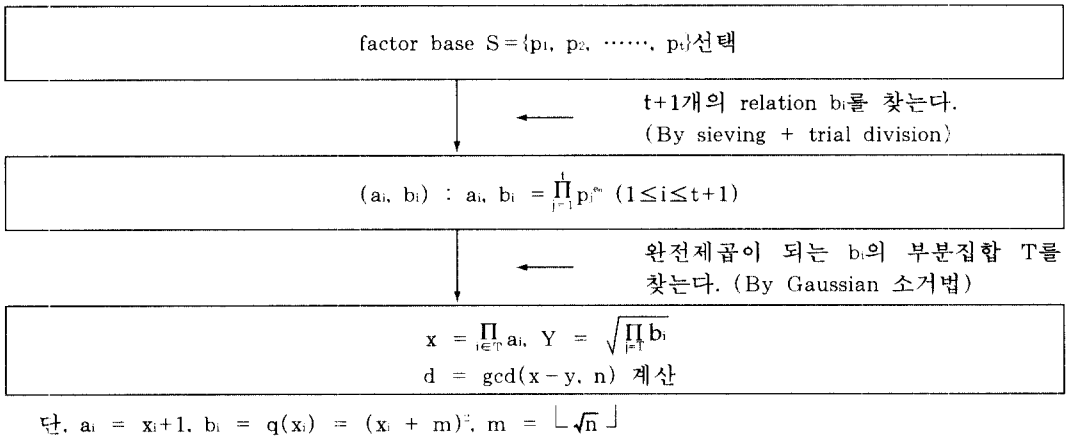
$9503435785^2 \equiv 546^2 \bmod n$ 이 므 로 x=9503435785, y=546

④ $\gcd(9503435785 - 546, 15770708441) = 115759$가되고, 115759는 n의 소인수이다.

2) Quadratic sieve 인수분해 알고리즘[27]

(가) 개 요

Quadratic sieve 알고리즘은 작은 소인수들을 가지지 않는 매우 큰 합성수들을 인수분해하기 위한 general purpose 인수분해 방법이다.



단, $a_i = x_i+1$, $b_i = q(x_i) = (x_i + m)^2$, $m = \lfloor \sqrt{n} \rfloor$

[개념도 3.5] Quadratic sieve 인수분해 알고리즘

(나) 알고리즘

Quadratic sieve 인수분해 알고리즘은 random square 방식과는 달리 $a_i^2 \equiv b_i \pmod n$을 만족하는 (a_i, b_i)을 찾기 위하여 quadratic 다항식, $q(x)=(x+m)^2 - n(m=\lfloor \sqrt{n} \rfloor)$을 이용하며, relation을 찾는 속도를 개선하기 위하여 sieving 기법을 사용하고 있다.

[Quadratic sieve 인수분해 알고리즘]

- 입 력 : 합성수 n
- 출 력 : n의 non-trivial 소인수 d
- 과 정 :
  ① Factor base $S=\{p_1, p_2, p_3, \cdots, p_t\}$를 선택한다.
  ② $q(x)=(x+m)^2 - n(m=\sqrt{n})$을 만족하는 모든 값들에서 다음의 [Sieving 알고리즘]을 이용하여, smoothness를 검사할 값들을 선택한다.
  ③ 과정 ④ - ⑦을 수행하여 t+1개의 (a_i, b_i)의 쌍을 수집한다.
  ④ 선택된 q(x)들 trial division을 이용하여 p_t - smooth한지를 검사한다.
  ⑤ pt - smooth하면, $b_i=\prod_{j=1}^{t} p_j^{e_{ij}}$로, $a_i=(x+m)$로 설정한다.
  ⑥ $1 \le j \le t$에 대해 $e_{ij}=v_{ij} \bmod 2$인 $v_i=(v_{i1}, v_{i2}, \cdots, v_{it})$를 생성한다.
  ⑦ Gaussian 소거법을 사용하여, $T \subseteq \{1, 2, \cdots, t+1\}$에서 $\prod_{i \in T} v_i=0$인 것들을 찾는다.
  ⑧ $x=\prod_{i \in T} a_i$, $y=\sqrt{b_i}$를 계산한다.
  ⑨ 만일 $x \equiv \pm y \bmod n$ 이면 과정 ⑦로 되돌아간다. 그렇지 않으면 d=gcd (x - y, n)계산하고 d를 출력한다.

[Sieving 알고리즘]

- 입 력 : Sieving 구간내의 모든 q(x)값들
- 출 력 : Smooth할 가능성이 높은 q(x)들
- 과 정 :
  ① $Q[x]=\lfloor \log_2 |q(x)| \rfloor$로 초기화한다. ($-M \le x \le M$)
  ② 모든 p_i에 대해 과정 ③ - ⑤를 수행한다.
  ③ $q(x) \equiv 0 \bmod p_i$인 x_1, x_2를 계산한다.
  ④ 모든 l값에 대해 $p_i \mid q(x_i+lp_i)$인 q(x)를 검색한다.
  ⑤ $Q[y]=Q[y] - \log_2 p_i$를 수행한다.

⑥ Q[x]의 값 중에서 0에 가까운 q(x)들을 선택한다.

(다) 예 제

Quadratic sieve 인수분해 알고리즘을 이용한 n=1042387의 인수분해는 다음과 같다.

① 크기 t=8인 factor base S={2, 3, 5, 7, 11, 13, 17, 19, 29, 31}을 선택한다.

② $m=\sqrt{1042387}=1020$을 계산한다.

③ $q(x)=(x+\sqrt{m})-n$을 이용하여 sieving 구간의 모든 값을 생성한다.

③ sieving과 trial division에 의해 다음의 relation을 찾는다.

i	x	q(x)	a_i	v_i
1	1	54	1021	(1,1,0,0,0,0,0,0)
2	7	12342	1027	(1,1,0,1,0,0,0,0)
3	10	18513	1030	(0,0,0,1,0,0,0,0)
4	41	8334	1061	(1,1,0,1,1,0,1,0)
5	92	194157	1112	(0,1,0,1,0,0,0,1)
6	109	232254	1129	(1,1,1,1,0,1,0,0)
7	128	275517	1148	(0,0,1,0,0,1,0,0)
8	155	338238	1175	(1,0,0,0,1,1,1,0)
9	197	438702	1217	(1,1,1,0,0,1,0,0)
10	370	889713	1390	(0,0,0,0,1,0,1,0)
11	500	1268013	1520	(0,1,0,1,0,0,0,1)

④ Gaussian 소거법에 의해 $v_1+v_2+v_3=0$를 찾는다.

⑤ $x=a_1a_2a_3 \bmod n=111078$을 계산한다.

⑥ $y=23311217 \bmod n=111078$을 계산한다.

⑦ $111078\equiv111078 \pmod n$이므로 다른 것이 필요하다.

⑧ Gaussian 소거법에 의해 $v_5+v_{11}=0$을 찾는다.

⑨ $a_5a_{11} \bmod n=647853$을 계산한다.

⑩ $b_5b_{11} \bmod n-496179$를 계산한다.

⑪ $647853 \neq \pm 496179 \bmod n$이므로, $\gcd(x-y, n)=\gcd(647853-496179, 1042387)=1487$이다. 따라서, 소인수는 1487이다.

3) Large prime variation 인수분해 알고리즘[26]

(가) 개 요

Large prime variation은 quadratic sieving의 변형으로 큰 소수의 인수분해시 더 많은 relation이 필요하며, 이러한 relation들을 찾기 위해 더 많은 시간이 소요되는 문제점을 해결하기 위하여, 단순히 relation을 생성할 때 factor base 밖의 몇몇 큰 소수들의 사용을 허용하는 것으로 sieving의 속도를 향상시켰다.[6] Relation을 이용하여 우리가 구하고자 하는 값은 완전제곱 형태인 y^2이므로, 같은 소수를 포함하는 relation들은 반드시 두 개 이상이 있어야 되며, 그렇지 않으면 인수분해에 이용할 수가 없다. Relation들의 형태들과 특성들은 아래의 [표 1]과 같다.[19]

형 태	특 성
$b\equiv\Pi p^e \bmod n$ (Small relation)	- 가장 일반적인 형태 - 인수분해할 숫자가 커지면 relation을 찾는데 시간이 많이 걸림
$b\equiv q_1\Pi p^e \bmod n$ (Partial relation)	- large prime variation의 일종 - Small relation들만을 사용하는 방법의 속도 향상 - sieving 단계에서 small 보다 많은 relation을 생성 - partial들을 결합하여 big relation을 만들어서 이용함 - large prime에 의해 정렬을 한 후 같은 large prime을 가진 것들을 이용

형 태	특 성
$b_1b_2 \equiv q_1^{e_1}\Pi p^{e_i+e_j} \bmod n$ (Big relation)	- partial relation들의 소수 q1이 같은 경우 - q1에 의해서 나누어질 수 있으므로, big relation은 small relation처럼 이용가능
$b \equiv q_1q_2\Pi p^{e_i} \bmod n$ (Partial-partial relation)	- relation에서 두 개의 큰 소수를 갖도록 확장 - partial 보다 더 많은 relation이 있음 - Partial-partial relation은 다른 relation들과 결합하여, small relation이나 big relation과 같이 유용하게 됨 - partial relation과의 결합으로 많은 relation을 생성

단, p:p∈factor base, q_i:소수, $q_i \notin$factor base

[표 1] Relation들의 형태

(나) 예 제

[Partial relation을 이용한 quadratic sieve]

- 입 력 : 합성수 n=391(23×17)
- 출 력 : n의 소인수 23, 17
- 과 정 :
  ① factor base={-1, 2, 3,}를 선택한다 (t=|factor base|=3)
  ② $m=\lfloor \sqrt{n} \rfloor$ 계산한다
  $\lfloor \sqrt{391} \rfloor = 19$
  ③ 과정 ④와 같이 t+1개의 (a_i, b_i)쌍을 찾는다.
  ④ x에 0, ±1, ±2, ±3을 대입하여 trial division을 이용하여 relation을 찾는다.
  i) factor base만을 사용한 경우

i	x	q(x)	q(x)의 인수분해	$a_i(x+m)$	v_i
1	1	9	32	20	(0,0,0)

⇒ t+1개의 relation을 발견하지 못함

ii) factor base와 factor base 외의 소수 5를 사용한 경우

i	x	q(x)	q(x)의 인수분해	$a_i(x+m)$	v_i
1	0	-30	$-1\times2\times35$	19	(1,1,1,1)
2	1	9	3^2	20	(0,0,0,0)
3	2	50	2×5^2	21	(0,1,0,0)
4	-3	-135	-1	16	(1,0,1,1)

⇒ t+1개의 relation을 발견

⑤ 과정 ④의 ii)의 relation에 Gaussian 소거법을 적용한다.

$v_1+v_3+v_4=0$

⑥ $x=a_1 \cdot a_3 \cdot a_4 \pmod n = 128$, $y=2\times3^2\times5^2 \pmod n = 59$를 계산한다.

⑦ $128 \neq \pm 59 \bmod n$이므로, gcd(128-59, 391)=23 이다.

$\therefore n=391=23\times17$

4) MPQS (Multiple Polynomial Quadratic Sieve)[30]

(가) 개 요

MPQS는 quadratic sieve의 변형으로 C. Pomerance의 방법에 기초하여 1987년에 Robert D. Silverman에 의해서 개발되었다. 일반적으로 큰 수를 인수분해할 경우 더 많은 relation이 필요하게된다. quadratic sieve 인수분해 알고리즘에서 quadratic 다항식 $q(x)=(x+m)2-n$은 x가 증가함에 따라 q(x)가 증가하게 되고, smoothness의 가능성도 줄어들게 된다. MPQS는 이러한 문제들을 해결하기 위하여 하나의 quadratic 다항식을 사용하는 대신, 여러 개의 quadratic 다항식을 구하여 각 다항식으로부터 qi(x)를 추출하여 sieving을 수행하는데, 각각의 다항식에 고정된 길이의 sieving 구간을 설정하여

q(x)가 증가되는 범위는 제한하였다. 그 결과 이전의 quadratic sievie의 경우보다 $2\sqrt{n}$ 배 더 작은 구간에서 sieving 단계를 수행하면서, 더 많은 relation들을 찾을 수 있도록 하였다. 또한 여러 개의 quadratic 다항식들을 사용하기 때문에 여러 컴퓨터에서 병렬로 수행될 수 있으므로, 현재 많은 인수분해 알고리즘의 구현에 사용되고 있다.(예, [8], [20])

(나) 알고리즘

- 입 력 : 합성수 n
- 출 력 : n의 non-trivial 소인수 d
- 과 정 :
  ① Factor base $S=\{p_1, p_2, p_3, \cdots, p_t\}$를 선택한다.
  ② [Montgomery의 quadratic 다항식 생성 알고리즘]을 이용하여 quadratic 다항식 $q_1(x), q_2(x), \cdots, q_k(x)$을 생성한다.
  ③ [Sieving 알고리즘]을 이용하여, 각 qi(x)를 만족하는 모든 값들 중에서 smoothness를 검사할 값들을 선택한다.
  ④ 과정 ⑤-⑧을 수행하여 t+1개의 (a_i, b_i)의 쌍을 수집한다.
  ⑤ 선택된 q(x)들 trial division을 이용하여 p_t - smooth한지를 검사한다.
  ⑥ p_t - smooth하면, $b_i=\prod_{j=1}^{t} p_j^{v_{ij}}$로, $a_i=(x+m)$ 로 설정한다.
  ⑦ $1\le j\le t$에 대해 $e_{ij}=v_{ij} \bmod 2$인 $v_i=(v_{i1}, v_{i2}, \cdots, v_{it})$를 생성한다.
  ⑧ Gaussian 소거법을 사용하여, $T\subseteq\{1, 2, \cdots, t+1\}$에서 $\prod_{i\in T} v_i=0$인 것들을 찾는다.
  ⑨ $x=\prod_{i\in T} a_i$, $y=\prod_{i\in T} \sqrt{b_i}$를 계산한다.
  ⑩ 만일 $x\equiv \pm y \bmod n$ 이면 과정 ⑦로 되돌아간다. 그렇지 않으면 d=gcd(x - y, n)계산하고 d를 출력한다.

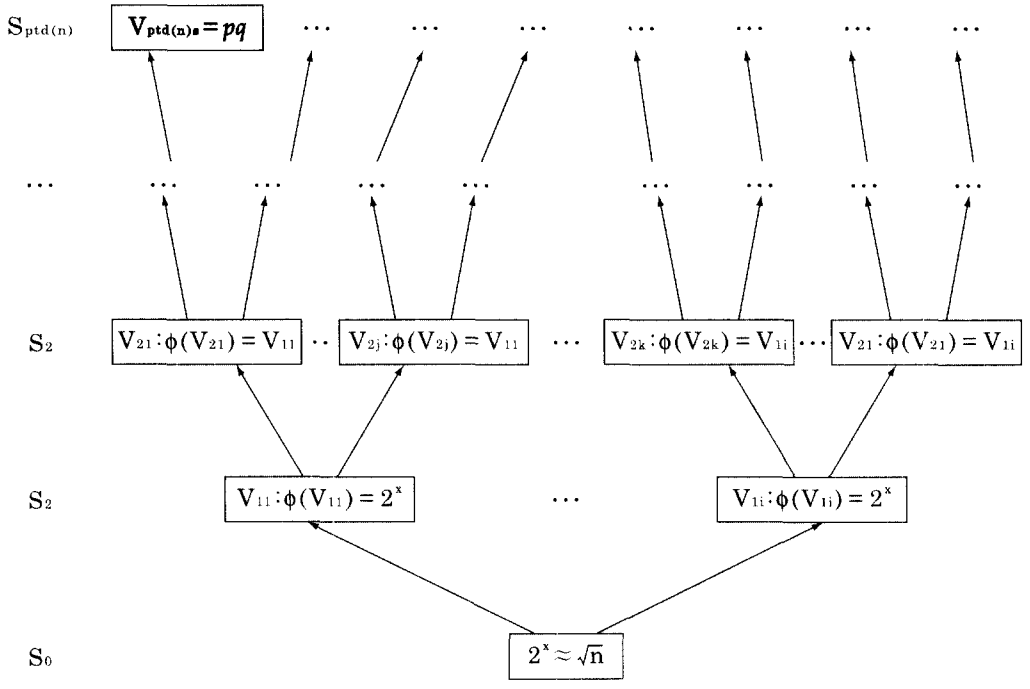
[Montgomery의 Quadratic 다항식 생성 알고리즘]

- 입 력 : $F(x)=ax^2+2bx+c$
- 출 력 : a, b, c (단, $n\mid b^2-ac$)
- 절 차 :
  ① Sieving 구간의 크기를 설정한다.
  $$2M$$
  ② F(x)의 값을 적게 하기 위하여 구간 I를 다음과 같이 결정
  $$I=(-b/a-M,\ -b/a+M)$$
  ③ $-F(-b/a)\approx F(-b/a-M)=F(-b/a+M)$인 a, b, c를 과정 ③, ④를 이용하여 찾는다.
  ④ $aF(-b/a-M)=aF(-b/a+M)=a^2M^2-n$ 이므로, $a\approx\sqrt{2n}/M$을 선택한다.
  $$(\text{단, } (\frac{n}{a})=1)$$
  ⑤ $b^2-ac=n$을 만족하는 b, c를 선택한다.
  $b^2\equiv n \bmod a$, $c=(b^2-n)/a$

5) Huber의 cryptanalytic 인수분해 알고리즘[12]

(가) 개 요

n에 Euler함수 $\varphi(n)$를 되풀이하여 적용하면, 2의 멱승으로 표현된다. RSA의 n은 비슷한 크기의 p와 q의 곱으로 이루어져 있으므로, $\sqrt{n}$에 가까운 2의 멱승에 [개념도 3.6]와 같이 $\varphi(x)$함수의 역을 재귀적으로 되풀이하면 n의 소인수 $p\cdot q$ 를 얻을 수 있다.



[개념도 3.6] Huber의 cryptanalytic 인수분해 알고리즘

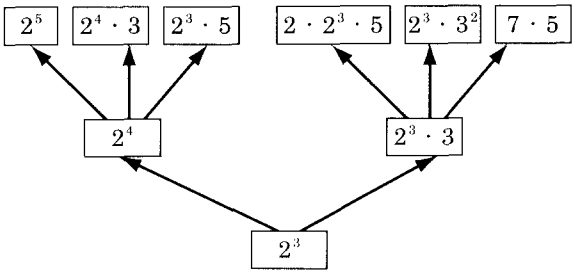
(나) 알고리즘

- 입 력 : n
- 출 력 : n의 소인수들의 곱
- 조 건 :
  ① $\varphi(n)=|\{i \mid \gcd(i, b)=1, 1\le i\le n\}|=n\cdot\prod_{p_i \mid n}\frac{p_i-1}{p_i}$
  ② $ptd(n)=\min\{j \mid \varphi^{(j)}(n)=2^i, i=1, 2, \cdots\}$
  ③ $E\{ptd(n)\}\approx\frac{\log_2 n}{\ln\ln n}$
  ④ $\Psi_r=\{i \mid \varphi(i)=r\}$

- 과 정 :
  ① 초기값 : $\sqrt{n}$에 가까운 2^k을 선택한다.
  ② Ψ_{2^k}인 모든 값들을 구한다.
  ③ 단계가 $E\{ptd(n)\}\approx\frac{\log_2 n}{\ln\ln n}$가 넘지 않는 동안 단계 ④ - ⑤를 수행한다.
  ④ 전 단계에서 생성된 값들 v에 대하여 Ψ_v을 계산한다.
  ⑤ 생성된 값 중에서 n이나 p, q를 포함하면 성공으로 알고리즘을 마친다.
  ⑥ 단계가 $E\{ptd(n)\}\approx\frac{\log_2 n}{\ln\ln n}$을 넘으면, 새로운 2^k을 선정하고 단계 ③으로 간다.

(라) 예 제

Huber의 Cryptanalytic 인수분해 알고리즘의 간단한 예로, n=7 · 5에 대한 인수분해는 다음과 같다.



4. RSA 시스템에 대한 공격

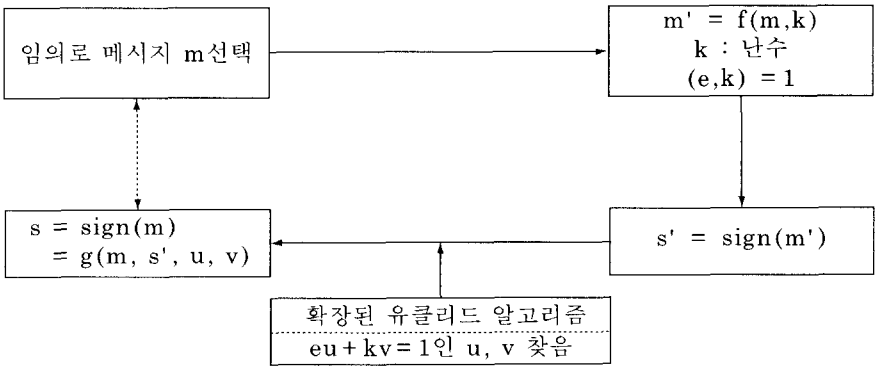
4.1 Homomorphic attack

RSA 시스템은 기본적으로 homomorphic 성질을 가지는데, 이 성질로 인하여 암호시스템과 서명시스템이 동일한 형태를 가지는 장점이 있다. 반면에 오히려 이 성질로 인하여 RSA 시스템은 시스템 자체가 공격당하는 약점을 가지게 된다. 본 절에서는 RSA의 homomorphic 성질을 이용한 chosen-message 공격에 대하여 먼저 설명하고, chosen-message 공격을 기본으로 하여 구성되는 garbage-man-in-the-middle 공격을 다룬다. 또한 메시지에 redundancy를 추가하거나 서명 구조를 변형하여 homomorphic 성질로 인해 발생되는 문제점을 제거하기 위한 노력들이 있었지만, 이 역시 chosen-message 공격에 안전하지 않다. 마지막 절에서는 이에 대한 공격 방식을 다룬다.

1) Chosen-message 공격

(가) 개 요

Chosen-message 공격은 공격자가 임의로 선택한 메시지에 대해서 획득한 상대방의 서명으로부터 본래의 메시지를 복원하는 공격이다.



[개념도 4.1] Chosen-message 공격

(나) 예 제

(1) Homomorphic 시스템에 대한 chosen-message 공격 : RSA

- 입 력 : 임의로 선택된 메시지 m
- 출 력 : 메시지 m에 대한 서명 s
- 과 정 :

① 임의의 메시지 m과 $k \in_r Z_n$ 선택, (k, e)=1 ku+ve=1 을 만족하는 u, v∈Z 찾음

② $m' = m^k \bmod n$

③ m'에 대한 서명 획득

$s' = m'^d \bmod n$

④ $s = s'^u m^v \bmod n$

(2) Non-homomorphic chosen message 공격 : LUC

- 입 력 : 임의로 선택된 메시지 m
- 출 력 : 메시지 m에 대한 서명 s
- 과 정 :

① 메시지 m과, $k \in_r Z_n$ 선택, (k, e)=1 ku+ve=1 을 만족하는 u, v∈Z 찾음

② $m' = V_k(m, 1) \bmod n$

③ m'에 대한 서명 s' 획득

$s' = V_d(m, 1) \bmod n$

④ $V_{ukd}(m, 1) = V_u(s', 1) \bmod n$

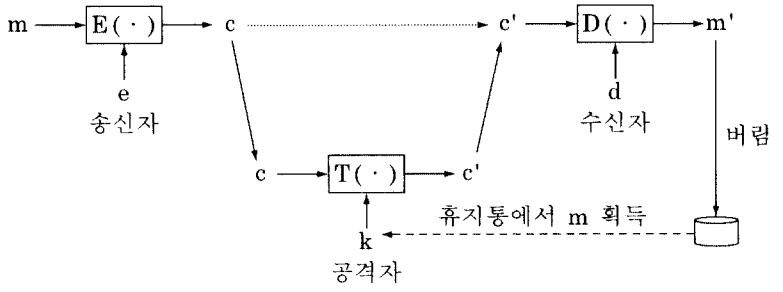
$$U_{ukd}(m, 1) = \frac{U_k(m, 1)U_u(s', 1)}{U_v(s', 1)} \bmod n$$

$$\therefore s \equiv V_d(m, 1) \equiv \frac{V_{ukd}(m, 1)V_v(m, 1)}{2} + \frac{(m^2 - 4)U_{ukd}(m, 1)U_v(m, 1)}{2} \bmod n$$

2) Garbage-man-in-the-middle 공격

(가) 개 요

송신자로부터 가로챈 메시지에 근거하여 새로운 메시지를 조작하여 이에 대한 수신자의 서명을 획득할 수 있다면, 이로부터 원래 메시지 m을 복원할 수 있다. garbage-man-in-the-middle attack은 기본적으로 chosen message attack을 이용하여 공격을 수행한다. Chosen message 공격 방식은 Davida가 1982년 제안하였다.[5] 하지만 이는 homomorphic 성질을 갖는 시스템에 대해서만 적용가능하였는데 1997년 Bleichenbacher가 non-homo-morphic 시스템(예, LUC)에 대한 공격 방식을 제안하였다.[1] 또한, M. Joye는 이를 일반화하여 모든 RSA 암호시스템에 대해서 garbage-man-in-the-middle 공격이 가능하도록 하였다.[15]



[개념도 4.2] Garbage-man-in-the-middle 공격

(나) 알고리즘

(1) Davida의 알고리즘

- 입 력 : $c \equiv m^e \pmod n$
- 출 력 : m
- 조 건 :

① Carol은 Bob으로 가는 메시지를 가로챌 수 있다.

② Carol은 Bob이 버린 메시지들을 획득할 수 있다.

- 과 정 :

① Alice가 Bob으로 보내는 암호문을 가로

챈다.

$$c \equiv m^e \pmod n$$

② 난수 k를 선택하여 c′을 계산한다.

$$c' \equiv ck^e \bmod n$$

③ c′에 복호화 과정을 적용한 메시지 m′을 Bob으로부터 획득한다.

$$m' \equiv (c')d \bmod n$$

④ 획득한 m′과 k로부터 m을 계산한다.

$$m \equiv m'k^{-1} \bmod n$$

Davida의 알고리즘은 non-homomorphic 시스템에 대해서는 적용할 수 없다. 이러한 시스템에 대해서는 다음의 Joye의 공격방식을 적용한다.

(2) Joye의 알고리즘

- 입 력 : $c=E_e(m)$
- 출 력 : m
- 조 건 :
  ① Carol은 Bob으로 가는 메시지를 가로챌 수 있다.
  ② Carol은 Bob이 버린 메시지들을 획득할 수 있다.
- 과 정 :
  ① Alice가 Bob으로 보내는 암호문을 가로챈다.

$$c=E^e(m)$$

  ② 난수 k를 선택하여 c′을 계산한다.

$$c'=T_k(c)$$

  ③ c′에 복호화 과정을 적용한 메시지 m′을 Bob으로부터 획득한다.

$$m'=D_d(c')$$

  ④ 획득한 m′과 k로부터 non-trivial relation을 이용하여 m을 계산한다.

(나) 예 제

(1) RSA 시스템에 대한 homomorphic 형태의 공격

- 입 력 : $c \equiv me \bmod n$
- 출 력 : m
- 과 정 :
  ① Carol은 Alice가 Bob으로 보내는 암호문을 가로챈다.

$$c \equiv m^e \bmod n$$

  ② 임의의 난수 k 선택, gcd(e, k)=1
  ② $c' \equiv c^k \bmod n$ 계산 후 Alice에게 전송③ c′에 복호화 과정을 적용한 메시지 m′을 Bob으로부터 획득한다.

$$m' \equiv c'^d \bmod n$$

  ④ $c^u m'^v=(m^e)^u(m^k)^v=m^{eu+vk} \equiv m \bmod n$

(2) LUC 시스템에 대한 homomorphic 형태의 공격

- 입 력 : $c=V_e(m, 1)$
- 출 력 : m
- 과 정 :
  ① Carol은 Alice가 Bob으로 보내는 암호문을 가로챈다.

$$c=V_e(m, 1)$$

  ② e와 서로소인 난수 k 선택
  ⇒ ue+vk=1을 만족하는 u, v∈Z 찾음
  ③ $c' \equiv V_k(c, 1) \bmod n$ 계산 후 Alice 에게 전송
  ④ c′에 복호화 과정을 적용한 메시지 m′을 Bob으로부터 획득한다.

$$m' \equiv V_d(c', 1) \bmod n$$

  ⑤ $V_{ukd}(c, 1) \equiv Vu(m', 1) \bmod n$

$$U_{ukd}(c, 1) \equiv \frac{U_k(c, 1)U_u(m', 1)}{U_e(m', 1)} \bmod n$$

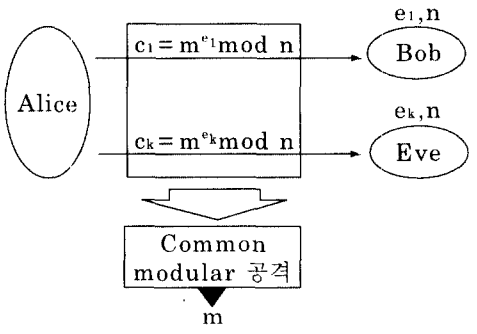
$$\therefore s \equiv \frac{V_{ukd}(c, 1)V_v(c, 1)}{2} + \frac{(c^2-4)U_{ukd}(c, 1)U_v(c, 1)}{2} \bmod n$$

3) Common modular 공격

(가) 개 요

이 공격 역시 RSA시스템의 homomorphic 성질을 이용한 공격으로 동일한 메시지를 암호화하고 동일한 모듈러 n을 사용하

는 경우, 이들 암호문들로부터 본래의 메시지 m을 얻을수 있다.[14]



[개념도 4.3] Common modular 공격

(나) 예 제

(1) RSA 시스템에 대한 공격

- 입 력 : $c_1 \equiv m^{e_1} \bmod n$, $c_2 \equiv m^{e_2} \bmod n$, $\gcd(e_1, e_2)=1$
- 출 력 : m
- 과 정 :
  ① $ue_1+ve_2=1$인 u, v∈Z를 찾음
  ② $m=m^{ue_1+ve_2} \equiv c_1^u c_2^v \pmod n$

(2) LUC 시스템에 대한 공격

- 입 력 : $c_1 \equiv V_{e_1}(m, 1) \bmod n$, $c_2 \equiv V_{e_2}(m, 1) \bmod n$, $\gcd(e_1, e_2)=1$
- 출 력 : m
- 과 정 :
  ① $ue_1+ve_2=1$인 u, v∈Z를 찾음
  ② $m=\frac{V_u(c_1, 1)V_v(c_2, 1)}{2}+\frac{(c_2^2-4)}{2}$

$$\frac{U_u(c_1, 1)U_v(c_2, 1)}{U_v(c_1, 1)} \bmod n$$

4) Redundancy를 사용하는 서명 시스템에 관한 공격

RSA의 multiplicative 성질로 인한 약점을 극복하기 위해 서명될 메시지에 redundancy를 추가한다. 그러나 redundancy는 안전성(security)과 그 확장률(redundancy expansion rate)에 대한 trade-off를 가지므로 redundancy의 길이에 따라 공격이 가능할 수 있다. 1985년 W. De Jonge과 D. Chaum이 redundancy를 사용한 서명에 대한 공격 방식을 제안하였고[12] 1997년 M. Girault와 J. F. Misarsky가 이를 확장하여 일반적인 형태의 redundancy를 사용한 서명 시스템에 대한 공격 방식을 제안하였다[9].

가)W.D.Jonge과 D.Chaum의 공격 방법

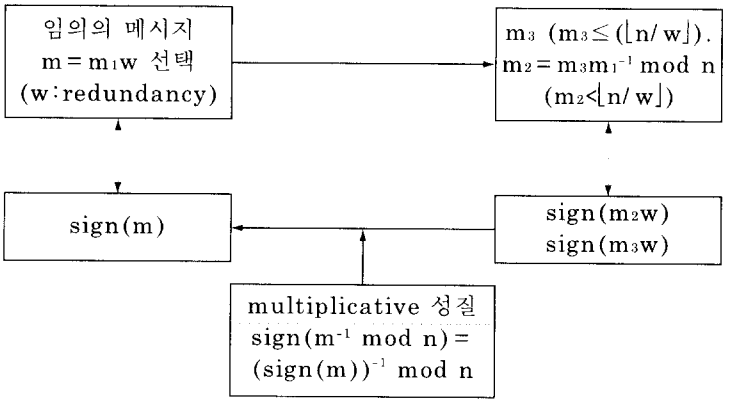
(1) Right-padded redundancy의 경우

(가) 개 요

RSA의 특성인 $f(m-1 \bmod n)=f(m)-1 \bmod n$을 이용한다. m_1에 대한 서명을 획득하기 위하여 chosen message 공격을 적용하여 다음 식을 만족하도록 m_2와 m_3을 선택한 후 서명 $sign(M_2)$와 $sign(M_3)$을 획득하여 $sign(M_1)$을 계산한다. ([개념도 4.4]참조)

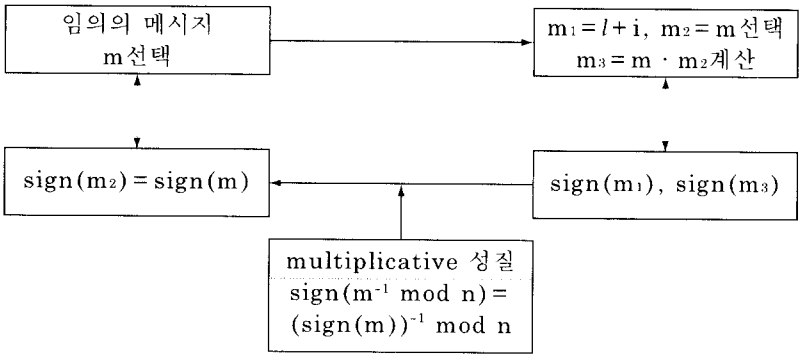
(나) 알고리즘

- 입 력 : $m_1w=m$ (w:redundancy, m_1:임의의 메시지)
- 출 력 : m에 대한 서명
- 과 정 :
  ① $m_1<n \text{ div } w$
  유효한 메시지 $m=m_1w$ ($m<n$, $m \bmod w=0$)
  $x \equiv (m_1w)^{-1} \bmod n$이다
  ② $(cx) \bmod n \le \lfloor n/w \rfloor$이고 $0<c \le \lfloor n/w \rfloor$인 c를 구하여 이를 m_3이라 놓는다
  ③ $m_2 \equiv (m_3x) \bmod n$라 놓는다
  ④ m_2, m_3에 대한 서명 $sign(m_2w)$와 $sign(m_3w)$를 획득한다
  ⑤ $sign(m_2w)$와 $sign(m_3w)$의 역수를 곱하여 m_1에 대한 서명을 구한다
  $sign(m)=sign((x^{-1}(m_3w)(m_3w)^{-1}) \bmod n)$



단, $m = m_1 w$

[개념도 4.4] Right-padded redundancy인 시스템에 대한 공격



단, $m = (l+i)m$

[개념도 4.5] Left-padded redundancy인 시스템에 대한 공격

$$= (\mathrm{sign}(m_3 w)\mathrm{sign}((m_3 x w)^{-1}))\bmod n$$
$$= (\mathrm{sign}(m_3 w)(\mathrm{sign}(m_2 w))^{-1})\bmod n$$

(2) Left-padded redundancy의 경우

(가) 개 요

유효한 메시지의 범위 [l, u]에 대하여, l이 $\sqrt{n}$보다 작으면 $[l, u^{1/2}]$밖의 두 유효한 메시지 M_1, M_2의 곱은 [l, u] 사이의 유효한 메시지가 되는 성질을 이용하여 chosen message 공격을 적용한다. ([개념도 4.5] 참조)

(나) 알고리즘

- 입 력 : m
- 출 력 : m에 대한 서명
- 과 정 :
  ① $l : l < u^{1/2}$, $m \le u - l$
  ② $(l+i)m \bmod n$이 $[l, u]$사이에 위치할 경우, 다음과 같은 3개의 메시지를 찾는다.
  $m_1 = l+i \bmod n$, $m_2 = m \bmod n$
  ③ m_1과 m_3에 대한 서명을 획득한다.
  $\mathrm{sign}(m_3)$

$= \mathrm{sign}(m_1 \cdot m_2) \bmod n$
$= \mathrm{sign}(l+i \cdot m) \bmod n$
$= \mathrm{sign}(m_1) \cdot \mathrm{sign}(m_2) \bmod n$

④ $\mathrm{sign}(m_3) \cdot \mathrm{sign}(m_1)^{-1}$로 $m_2(=m)$에 대한 서명을 얻는다
$\mathrm{sign}(m)=\mathrm{sign}(m_3)/\mathrm{sign}(m_1)$

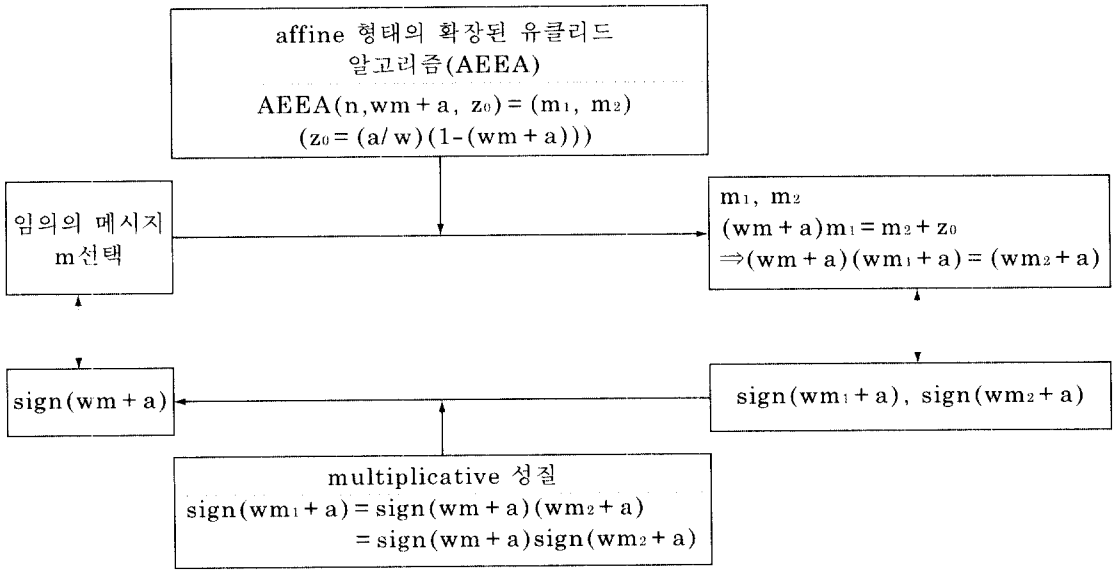
나) M.Girault와 J.F.Misarsky의 공격 방법

기존의 De Jonge과 Chaum의 공격은 redundancy가 포함된 메시지를 affine 형태인 wm+a라고 했을 때, w=1이거나 a=0인 경우로 제한했었다[13]. Girault와 Misarsky는 기존의 De Jonge과 Chaum의 chosen message 공격을 확장하여 affine 형태인 wm+a에 대한 일반적인 chosen message 공격 방법을 제안하였다[16]. 또한 사용되는 redundancy의 형태를 서명될 메시지에 고정된 크기의 비트들을 붙이는 형태와 메시지에 모듈러 연산을 통해 얻은 값을 붙이는 형태로 나누어 설명하였다. 사용되는 기본 알고리즘은 Okamoto-Shiraishi가 제안한 affine 형태의 확장된 유클리드 알고리즘으로 양의정수 n, d, $z_0(<n)$, X, $Y(<n)$가 주어졌을 때 $dx=y+z_0 \pmod n$인 특정 범위($|x|<X$, $|y|<Y$) 내의 x와 y를 찾는 알고리즘이다[22].

(1) 일정한 크기의 redundancy를 갖는 유효한 메시지에 대한 공격 방법

(가) 개 요

유효한 메시지 M은 실제 메시지의 양쪽에 redundancy가 붙은 형태이고 M=wm+a(a, w는 상수)로 나타낸다. Chosen message 공격의 적용을 위해 m이 주어졌을 때 affine 형태의 확장된 유클리드 알고리즘을 이용하여 적절한 m_1과 m_2를 얻고 그 각각의 서명을 통하여 M에 대한 서명 sign(wm+a)을 얻는다. ([개념도 4.6] 참조)



단, m = wm + a

[개념도 4.6] 일정한 크기의 redundancy를 갖는 시스템에 대한 공격

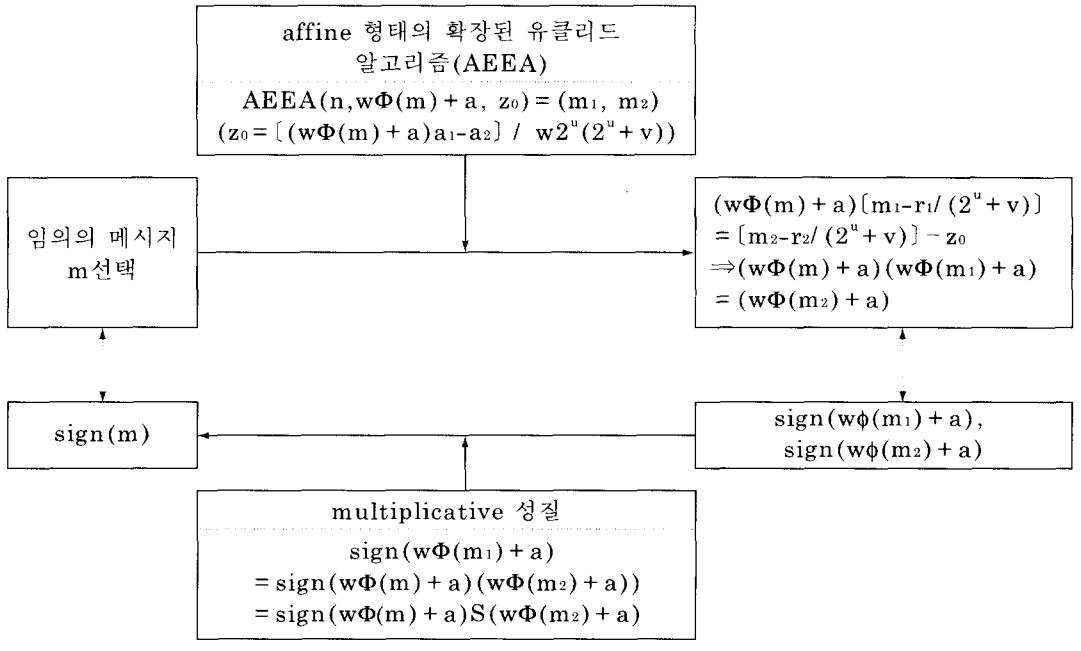
(나) 알고리즘

- 입 력 : m
- 출 력 : m에 대한 서명
- 과 정 :
  ① $a \le m < b < n$
  w:상수, β:max{|m|}
  메시지 집합:정수 $\{m \mid 0 \le m < (b-a)\beta\}$
  유효한 메시지 집합:$\{wm+a \mid 0 \le m < (b-a)\beta\}$
  ② 위조된 서명을 얻고자 하는 메시지 m($0 \le m < (b-a)\beta$)을 선택
  ③ $z_0 \equiv (a/w)[1-(wm+a)] \bmod n$라 놓는다
  ④ affine 형태의 확장된 유클리드 알고리즘을 사용하여 m_1, m_2($0 \le m < (b-a)\beta$)를 구한다 $(wm+a)m_1 \equiv m_2 + z_0 \bmod n$
  ⑤ 식 $(wm+a)(wm_1+a)=(wm_2+a)$을 얻는다
  ⑥ 메시지 m_1과 m_2에 대한서명 $sign(wm_1+a)$, $sign(wm_2+a)$를 얻는다.
  $sign(wm+a)=sign(wm_1+a)/sign(wm_2+a)$

(2) 크기가 일정하고 모듈러 redundancy를 갖는 메시지에 대한 공격 방법

(가) 개 요

모듈러 redundancy는 modular 연산을 통 해 얻 은 redundancy를 말 하 며 $H(m)=m \pmod{2u+v} \oplus \text{Mask}$ (여기서 Mask는 u-bit의 고정된 스트링)로 나타낸다. 유효한 메시지 M은 실제 메시지에 모듈러 redundancy H(m)를 붙이고 그 양쪽에 redundancy를 붙인 형태이고 $w\Phi(m)+a$ (a, w는 상수)로 나타낸다. Chosen message 공격의 적용을 위해 m이 주어졌을 때 affine 형태의 확장된 유클리드 알고리즘을 이용하여 적절한 m_1과 m_2를 얻고 그 각각의 서명을 통하여 M에 대한 서명 $sign(w\Phi(m)+a)$을 얻는다 ([개념도 4.7] 참조)



단, $m = (w\Phi(m)+a)$, $\Phi(m) = (m-r)2^u + r2^u + (r \oplus \text{Mask})$

[개념도 4.7] 모듈러 redundancy를 갖는 시스템에 대한 공격

(나) 알고리즘

- 입 력 : m
- 출 력 : m에 대한 서명
- 과 정 :

① a, w (<n) : 정수

실제 메시지 집합 : $\{m \mid 0 \le m < 2^{\alpha}\}$

유효한 메시지 집합 : $\{w\Phi(m)+a \mid 0 \le M < 2^{\alpha}$

$C(r)=r2^{u}+(r \oplus Mask)$

$\Phi(m)=q(2^{u}+v)2^{u}+C(r)$

② m ($0 \le m < 2^{\alpha}$)을 선택 (위조된 서명을 얻고자 하는 메시지)

③ r_1과 r_2를 선택 ($<2^{u}+v$)

④ $a_1=C(r_1)w+a$

$a_2=C(r_2)w+a$

$z_0=[(w\Phi(m)+a)a_1 - a_2]/w2^{u}(2^{u}+v)$라 놓는다

⑤ affine 형태의 확장된 유클리드 알고리즘을 사용하여 $q_1<(2^{\alpha}-r_1)/(2^{u}+v)$, $q_2<(2^{\alpha}-r_2)/(2^{u}+v)$를 푼다. $(w\Phi(m)+a)\ q_1 \equiv q_2 - z_0 \bmod n$

⑥ $m_1=q_1(2^{u}+v)+r_1$ $m_2=q_2(2^{u}+v)+r_2$라 놓으면 식 $(w\Phi(m)+a)(w\Phi(m_1)+a) = (w\Phi(m_2)+a) \bmod n$을 얻는다

⑦ 메시지 m_1과 m_2에 대한서명 $sign(w\Phi(m_1)+a)$, $sign(w\Phi(m_2)+a)$를 얻는다. $sign(w\Phi(m)+a)=sign(w\Phi(m_1)+a)/sign(w\Phi(m_2)+a)$

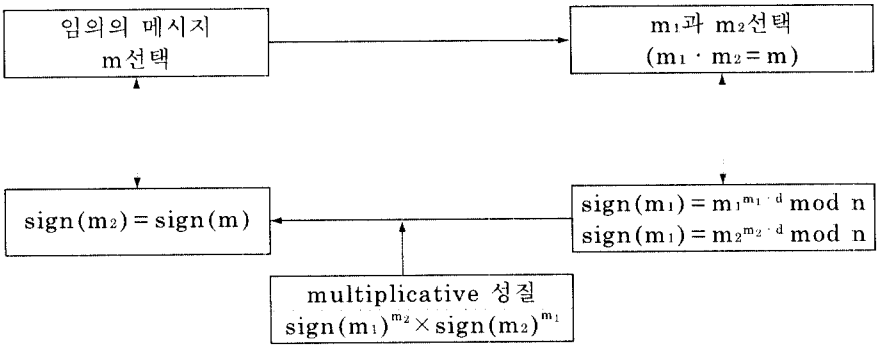
5) 변형된 서명 구조를 갖는 시스템에 대한 공격

RSA 시스템이 갖고 있는 고유의 특성인 reciprocal 성질과 multiplicative 성질에 대한 약점을 극복하기 위한 방법으로 서명식의 구조 자체를 변형시키는 방법이 있다[13](여기서 reciprocal 성질이란 RSA의 키 쌍을 공개함수와 비밀함수의 쌍 (P, S)라 할 경우 P S = SP = 단위함수(identity function)가 되는 성질을 말하며 multiplicative 성질이란 S(xy) = S(x)S(y)가 되는 성질을 말한다). 이를 위한 모델로 RSA 지수 부분을 m, n의 함수로 표현한 GES(Generalized Exponentiation Signature) 스킴을 사용한다(예를 들어, sign(서명문) = $F_1(m, n)F_2(m, n) \bmod n$)). 여기서는 함수 F_1과 F_2가 서로 다른 3개의 스킴에 대하여 chosen message 공격을 적용해본다 (주 - RSA는 GES의 특별한 경우이다. 즉 F_2가 상수함수로 $d = e^{-1} \bmod \varphi(n)$이다).

가) 메시지 m에 대한 서명이 $sign(m)= m^{m \cdot d} \bmod n$인 경우

(가) 개 요

지수부분(F_2)을 $(m \cdot d) \bmod \varphi(n)$로 변형한 형태이다. ([개념도 4.8] 참조)



단, $sign(m) = m^{m \cdot d} \bmod n$

[개념도 4.8] $sign(m) = m^{m \cdot d} \bmod n$인 시스템에 대한 공격

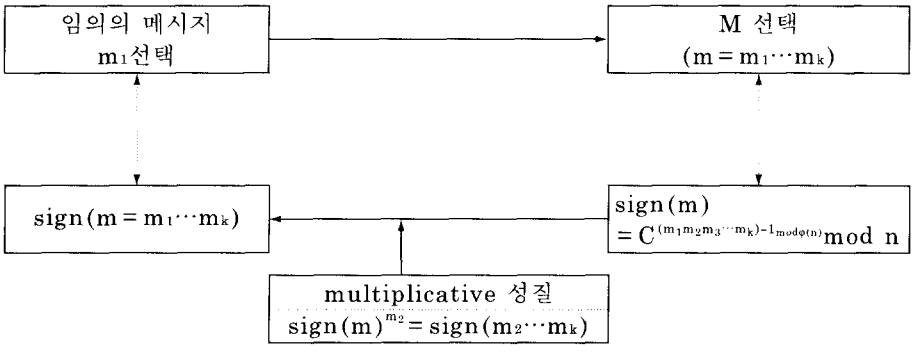
(나) 알고리즘

- 입 력 : m
- 출 력 : m에 대한 서명
- 과 정 :
  ① 다음을 만족하는 m_1과 m_2 선택
  $m_1 \cdot m_2 = m$
  ② 각각 m_1과 m_2의 서명 $sign(m_1)=m_1^{m_1^{-1}d} \bmod n$과 $sign(m_2)=m_2^{m_2^{-1}d} \bmod n$을 얻는다
  ③ $sign(m_1)^{m_2}$와 $sign(m_2)^{m_1}$을 구하여 곱하면
  $(m1^{m_1 m_2 d} \bmod n) \cdot (m_2^{m_1 m_2 d} \bmod n)$
  ④ $(m1m2)^{(m_1 m_2)d} \bmod n = sign(m)$을 얻을 수 있다.

나) 메시지 m에 대한 서명이 $sign(m)=C^{m^{-1} \bmod \varphi(n)} \bmod n$인 경우

(가) 개 요

F_1을 상수 C로, 지수부분(F_2)을 $m-1 \bmod \varphi(n)$으로 변형한 형태이다. ([개념도 4.9] 참조)



단, $sign(m) = C^{(m^{-1} \bmod \varphi(n))} \bmod n$, C는 상수

[개념도 4.9] $sign(m) = Cm^{-1} \bmod \varphi(n) \bmod n$인 시스템에 대한 공격

(나) 알고리즘

- 입 력 : m_1
- 출 력 : 메시지 $m_2 \cdots m_k$의 서명 $sign(m_2 \cdots m_k)$
- 과 정 :
  ① 다음을 만족하는 m을 선택
  $m = m_1 \cdots m_k$
  ② m의 서명 $sign(m)=C^{(m_1 m_2 m_3 \cdots m_k)^{-1} \bmod \varphi(n)} \bmod n$을 얻는다
  ③ $sign(m)m_1 = sign(m_2 \cdots m_k)$이므로 메시지 $m_2 \cdots m_k$의 서명 $sign(m_2 \cdots m_k)$을 얻음

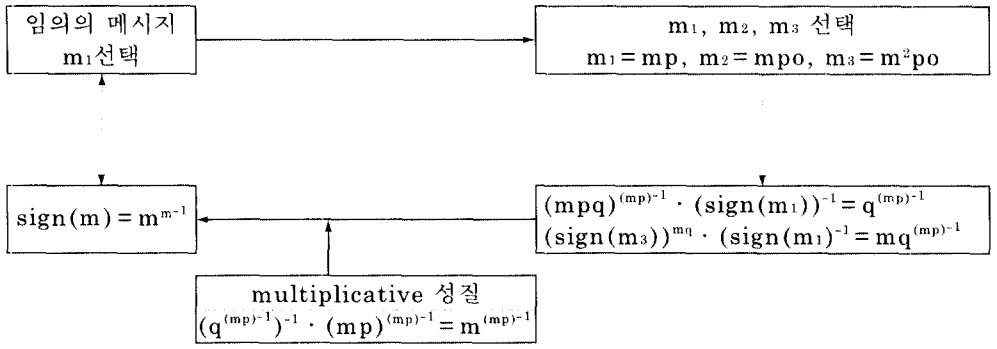
다) 메시지 m에 대한 서명이 $sign(m)=m^{m^{-1} \bmod \varphi(n)} \bmod n$인 경우

(가) 개 요

지수부분(F_2)을 $m^{-1} \bmod \varphi(n)$으로 변형한 형태이다. ([개념도 4.10] 참조)

(나) 알고리즘

- 입 력 : m
- 출 력 : m에 대한 서명
- 과 정 :
  ① 다음을 만족하는 m_1, m_2, m_3 선택
  $m_1 = mp$, $m_2 = mpq$, $m_3 = m^2pq$
  ② $(sign(m_2))^q \bmod n = (mpq)^{(mp)^{-1}}$, $sign(m_1)^{-1} = mp^{(mp)}$이다. 그러므로,
  $(mpq)^{(mp)^{-1}} \cdot (sign(m_1))^{-1} = Q^{(mp)}$ ------(*)
  $(sign(m_3))^{mq} = m^2pq^{(mp)^{-1}}$, $sign^{(m_3)^{-1}} = mp^{(mp)}$이다. 그러므로,



단, $sign(m) = m^{m^{-1} \bmod \varphi(n)} \bmod n$

[개념도 4.10] $sign(m) = m^{m^{-1} \bmod \varphi(n)} \bmod n$인 시스템에 대한 공격

$(sign(m_3))^{mq} \cdot (sign(m_1)^{-1} = mq^{(mp)^{-1}}$ ------ (**)

③ (*)과 (**)에서 $(q^{(mp)^{-1}})^{-1} \cdot (mq)^{(mp)!} = m^{(mp)^{-1}}$ 이 된다. 이것은 $sign(m))^p$이므로 $sign(m) = mm\text{-}1$을 얻을 수 있다.

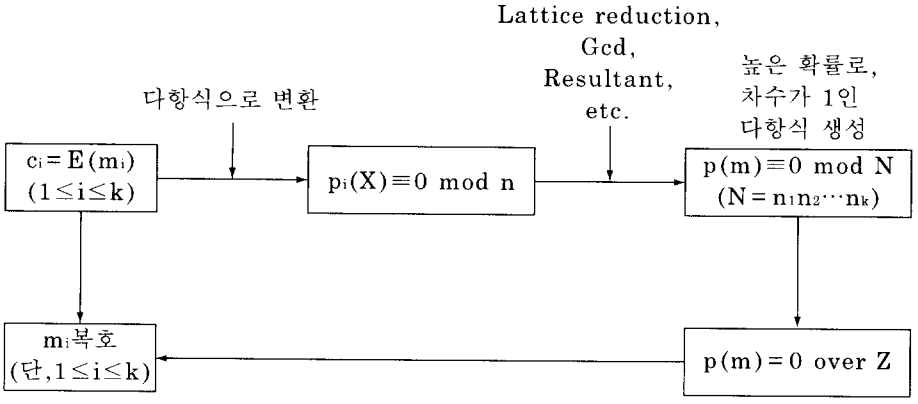
4.2 다항식 공격

RSA의 암호화(서명)된 메시지들은 본래의 메시지를 미지수로 갖는 하나의 다항식들로 변환할 수 있다. 동일한 메시지를 암호화한 경우 각각의 암호문에 대한 다항식들은 모두 동일한 근을 갖는다. 다항식 공격은, 먼저 이러한 다항식들을 lattice, gcd, resultant 등의 방법을 사용하여 동일한 근을 갖는 하나의 다항식으로 묶어 이 다항식의 근을 구하는 방식이다. Lattice 방법은, 다항식들의 계수로부터 가공된 행렬에 대해 LLL 알고리즘을 적용하여 다항식의 계수를 결정하는 벡터를 구하는 방법이다. 그리고 다항식들의 gcd로 생성되는 새로운 다항식이 이전의 다항식들과 동일한 근을 가지는 1차 다항식이 될 확률이 높다는 사실을 이용하는 방법이다. 상호 관련성이 있는 다항식의 수가 크면 gcd방식을 사용하기 힘든데. 이 경우 resultant방식을 이용하여, 다항식들의 미지수들을 줄여 하나의 미지수에 대해서 다항식을 정리하고, 이 결과에 gcd 방식을 적용한다.기존의 다항식들로부터 새로 생성된 하나의 다항식은 모듈러 n 상에서의 연산보다 계산량이 훨씬 적어 공격자가 근을 구하기 쉽다. 또한, 기본적으로 다항식을 기반으로 하여 구축되는 시스템은 공격방식 또한 다항식 형태를 가지게 되는데. RSA시스템의 homo morphic 성질을 이용하여 공격하는 garbage-man-in-the-middle 공격이 LUC 시스템에 적용되었을 경우, 이는 다항식 공격 형태를 가지게 된다.

1) Hastad의 공격

(가) 개 요

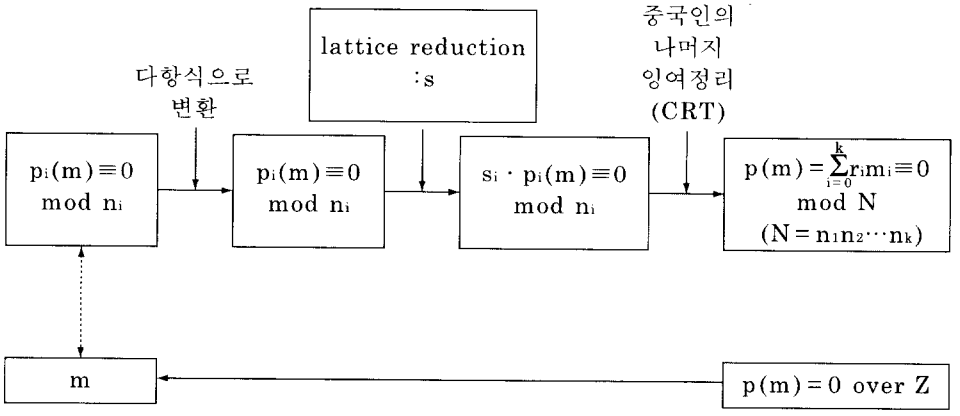
Blum, Liebereur 그리고 William은, 동일한 메시지를 e로 암호화하여 여러 수신자에게 배포할 경우, 메시지 m이 공격자에게 노출될 수 있음을 보였다. 하지만 이 공격은 메시지를, 일반적인 형태인, am+b로 구성함으로써 방지할 수 있다. Hastad는 이러한 일반적인 형태의 메시지에 대한 공격방법을 획기적으로 제안하였다.[10] 이 방식은 Takagi



단, E()는 암호화알고리즘

[개념도 4.11] 다항식 공격 알고리즘

와 Naito에 의해서 일반화되었고 이후 M. Joye가 다시 이를 개선하였다.[16] 본 절에서는 일반적인 형태의 메시지에 대한 다항식 공격 방식의 근간을 이루는 Hastad 의 공격 방식을 중심으로 설명하고자 한다.



단, E():암호화 알고리즘, $k \ge e(e+1)/2$, e:암호화키, s:$(s_1, \cdots, s_k, \cdots, s_{k+e+1})$
r_i : CRT로 묶은 다항식 P(m) 계수

[개념도 4.12] Hastad의 다항식 공격

정리 5.1 (Hastad의 정리)
$N = \prod_{i=1}^{k} n_i$, $n = \min n_i$ (단, $\gcd(n_i, n_j) = 1$ $(i \ne j)$ $\forall i$, $\gcd(\langle a_{i,j} \rangle_j^{\delta} = 0, n_i) = 1$)라 하자.
만약 아래의 조건
$N > n^{\frac{e(e+1)}{2}} (k+e+1)^{\frac{(k+e+1)}{2}} 2^{\frac{(k+e+1)^2}{2}} (e+1)^{(e+1)}$
을 만족하면, 다항식 시간 안에 x를 복원할 수 있다.

따름 정리 5.2
$N = \prod_{i=1}^{k} n_i$, $n = \min n_i$ 그리고 $m_i = \alpha_i \cdot m + \beta_i$

(단, α, β는 알려진 상수)라 하자. k개의 e로 암호화된 메시지 $m_i^e \pmod{n_i}$이 주어질 때, 만약 아래의 조건

$$k>\frac{e(e+1)}{2},\ N>n^{\frac{e(e+1)}{2}}(k+e+1)^{\frac{(k+e+1)}{2}}2^{\frac{(k+e+1)^2}{2}}$$
$$(e+1)^{(e+1)}$$

을 만족하면, 다항식 시간 안에 x를 복원할 수 있다.

(나) 알고리즘

- 입 력 : c_i ($=m_i^e \bmod n_i$), i=1, ⋯, k
- 출 력 : m
- 조 건 :
  ① k>e(e+1)/2
  ② n_i는 서로 소
  ③ $n=\min(n_i)$
  ④ $N=\prod_{i=1}^{k} n_i$
  ⑤ $m_i(=m\|t_i)<\min(n_i)$, i=1, ⋯, k
  t_i : time stamp
- 과 정 :
  ① 각각의 식을 전개
  $$\sum_{i=0}^{e} a_{ij}m^i \equiv 0 \pmod{n_i}$$
  ② $\overrightarrow{b_1}=(a_{10}u_1, na_{11}u_1, n^2a_{12}u_1, \cdots, n^ea_{1e}u_1, \frac{N}{n_1(e+1)}, 0, \cdots, 0)$
  $\overrightarrow{b_2}=(a_{20}u_2, na_{21}u_2, n^2a_{22}u_2, \cdots, n^ea_{2e}u_2, 0, \frac{N}{n_2(e+1)}, \cdots, 0)$
  $\overrightarrow{b_k}=(a_{k0}u_k, na_{k1}u_k, n^2ak_2u_{nk}, \cdots, n^e a_{ke}u_k, 0, 0, \cdots, \frac{N}{n_k(e+1)})$
  $\overrightarrow{b_{k+1}}=(N, 0, 0, \cdots, 0, 0, 0, \cdots, 0)$
  $\overrightarrow{b_{k+1}}=(0, nN, 0, \cdots, 0, 0, 0, \cdots, 0)$
  $\overrightarrow{b_{k+1}}=(0, 0, 0, \cdots, n^dN, 0, 0, \cdots, 0)$

위의 vector들을 base로 갖는 lattice L로부터 다음을 만족하는 vector $\overrightarrow{b}$를 찾는다. (lattice reduction 사용)

$$\|\overrightarrow{b}\|<\frac{N}{e+1}$$

③ $\overrightarrow{b}=\sum_{i=0}^{k+d+1} s_i\overrightarrow{b_i}$ 으로부터 $s_1, \cdots, s_k$를 추출

④ ①의 각 식에 상수 s_i를 곱함

$$\sum_{i=0}^{e} m^i \sum_{j=1}^{k} a_{ij}s_jm^i \equiv 0 \pmod{n_i}\ i=1, \cdots, k$$

⑤ CRT로 다항식들을 하나의 다항식으로 구성

$$\sum_{i=0}^{e} m^i \sum_{j=1}^{k} a_{ij}s_ju_i = \sum_{i=0}^{e} m^ic_i \equiv 0 \pmod{N}$$

$u_i \equiv \delta_i \pmod{n_i}$ ($\delta_i=1$ if i=j, otherwise $\delta_i=0$)

⑥ $\sum_{i=0}^{e} m^ic_i=0$ over Z
⇒ 다항식 시간 안에 m 계산

2) Coppersmith의 공격

(가) 개 요

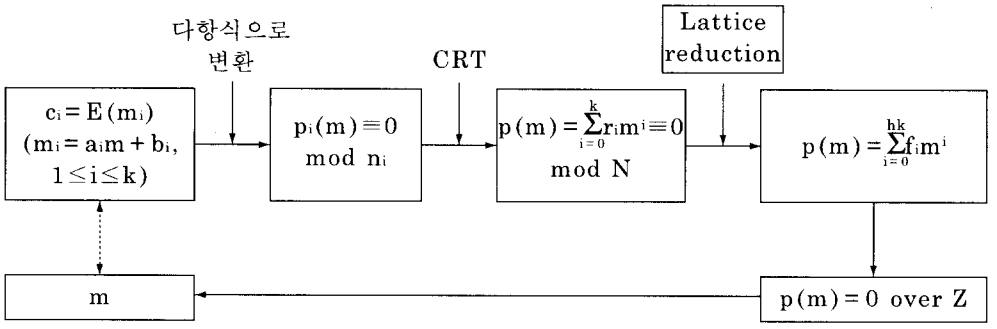
4.1 1)에서 언급한 Hastad 방식관련 공격방식들은 계산량이나 공격을 위해 요구되는 메시지들의 수 등의 측면에서 최적화되지 않았다. Coppersmith는 Eurocrypt'96에서 다항식 공격방식에 대해 최적화된 공격방식을 제안하였는데, 이 방식으로 인해, 공격을 위해 요구되는 메시지의 수는 이전의 Blum 등이 제안한 방식과 동일하면서, 일반적인 형태의 메시지들에 대해서 공격이 가능해졌다.[3]

정리 5.2 (Coppersmith의 정리)

p(x)가 k차 monic 다항식이고, N은 인수분해를 모르는 양의 정수일 때, 다항식 시간 안에 $p(x_0) = 0 \bmod N$ 인 모든 x_0을 계산할 수 있다.

따름 정리 5.4

RSA, LUC 시스템에서, 선형적으로 관련된 e개 이상의 $\alpha_i \cdot m+\beta_i \bmod n_i$ 메시지들을 획득하면, 다항식 시간 안에 m을 계산할 수 있다.



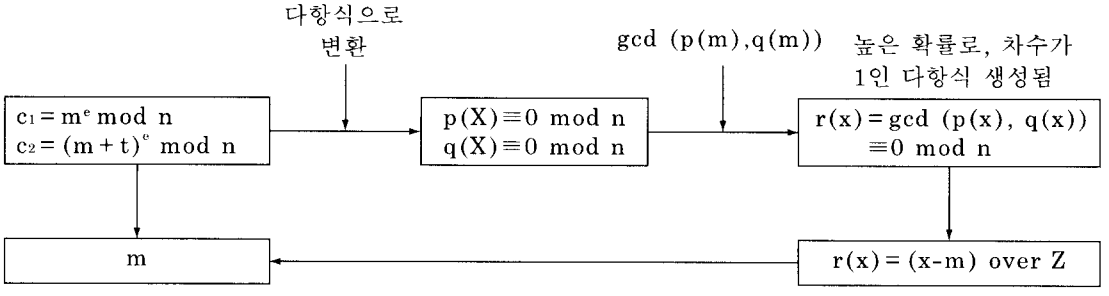
단, E():암호화 알고리즘, $k \ge e$, e=암호화키, f:$(f_i, \cdots, f_{hk}, \cdots, f_{k2hk-k})$, N:$n_1 n_2 \cdots n_k$

[개념도 4.13] Coppersmith의 다항식 공격

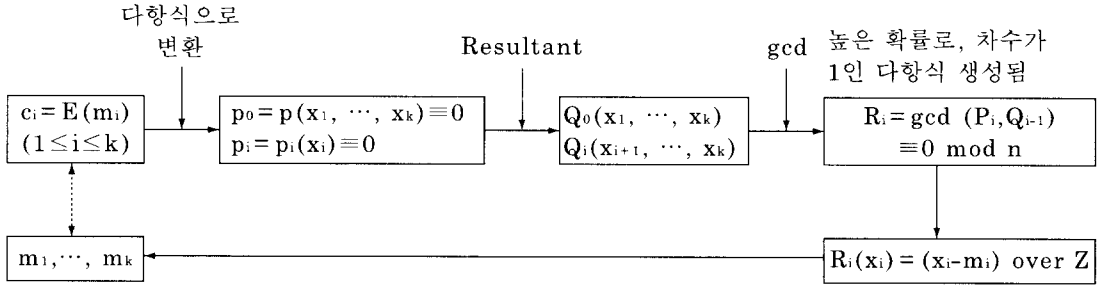
3) Gcd 공격

(가) 개 요

Gcd 공격은 동일한 근을 갖는 두 개의 다항식의 gcd를 계산하여 새로운 하나의 다항식을 생성한 후 이로부터 근을 계산하는 방식이다. 이는 관련된 메시지의 수가 적을 경우에 적용 가능하지만, 관련된 메시지의 수가 큰 경우 다항식들로부터 gcd를 계산하는 것이 힘들다. 1996년 Coppersmith는 gcd 공격을 일반화한 공격방식을 제안하였다.[4]



[개념도 4.14 (1)] 기본적인 gcd 공격



단, E()는 암호화 알고리즘

[개념도 4.14(2)] 일반화된 gcd 공격

(나) 알고리즘

- 입 력 :

① k개의 메시지 $m_1, \cdots, m_k$ 에 대한 다항식 관계 $p(m_1, \cdots, m_k)$

② 암호문 $c_i=m_i^e \bmod n$ $(i=1, \cdots, k)$

- 출 력 : m_i $(i=1, \cdots, k)$
- 과 정 :

① 메시지 m_i를 x_i라 놓으면 다음과 같이 k+1개의 다항식을 얻을 수 있다.

$P_0(x_1, \cdots, x_k)=p(x_1, \cdots, x_k)=0 \bmod n$

$P_i(x_i)=x_i^e - c_i=0 \bmod n$ $(i=1, \cdots, k)$

② 다음과 같이 Resultant(·)를 반복한다.

$$Q_0(x_1, \cdots, x_k)=P_0$$

$$Q_i(x_i+1, \cdots, x_k)=\text{Resultant}_{x_i}(Q_i - 1, P_i)$$

$$\cdots$$

$$Q_k - 1(xk)$$

③ gcd를 이용하여 다음과 같은 선형다항식 $(x_k - m_k)$을 얻는다

$\gcd(Q_{k-1}(x_k), P_k(x_k))=(x_k - m_k)$

⇒ 다항식 시간 안에 m_k 계산 가능

④ 동일한 방법으로 k - 1부터 1까지 반복.

$\gcd(Q_i - 1(x_i, m_i+1, m_i+2, \cdots, m_k), P_i(x_i))=(x_i - m_i)$

⇒ 다항식 시간 안에 m_i $(i=1, \cdots, k-1)$ 계산 가능

4) Garbage-man-in-the-middle 공격 (II)

(가) 개 요

Garbage-man-in-the-middle attack은 기본적으로 RSA의 homomorphic 성질을 이용하는 공격방식이지만 다항식을 이용한 공격방식 형태로도 공격이 가능하다.

(나) 예 제

(1) LUC 시스템에 대한 다항식 형태의 garbage-man-in-the-middle 공격

- 입 력 : $c=E_e(m)$
- 출 력 : m
- 조 건 :

① Function

$E_e(m) \equiv V_e(m, 1) \bmod n=c$

$D_d(c') \equiv V_d(c', 1) \bmod n=m'$

$T_k(c) \equiv V_k(k, 1) \bmod n=c'$

② non-trivial relation

$P(x) \equiv D_e(x, 1) - c \bmod n$

$Q(x) \equiv D_k(x, 1) - m' \bmod n$

(단, $P(m)=Q(m) \equiv 0 \bmod n$

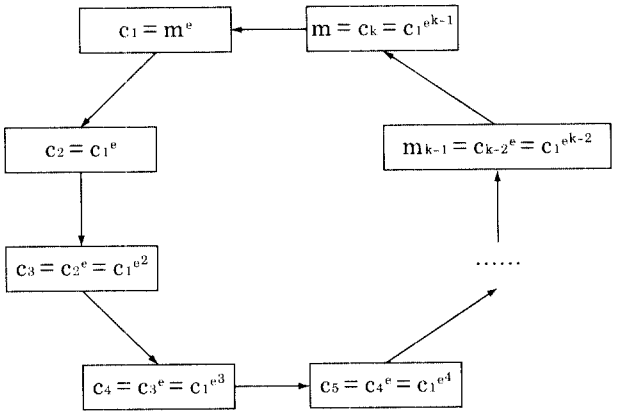
$D_e(x, 1)$, $D_k(x, 1)$: Dickson의 다항식)

- 과 정 :

① $c=E_e(m) \equiv V_e(m, 1) \bmod n$ 을 가로챔

② $k \in_r Z_n$, $(k, e)=1$

③ $c' \equiv T_k(c, 1) \bmod n$ 계산 후, 수신자에게



[개념도 4.15] Cycling 공격 방식

전송

④ 수신자로부터 복호화 과정이 적용된 의미 없는 메시지를 획득

$m' \equiv V_d(c', 1) \bmod n \equiv V_k(m, 1) \bmod n$

⑤ non-trivial relation으로부터 다음 다항식의 근을 계산

$R(x)=gcd(P(x), Q(x))$

4.3 기타 RSA시스템 공격방식

1) Cycling 공격방식[31]

(가) 개 요

Cycling 공격방식은 Simmons와 Norris에 의해서 제안된 암호문으로부터 평문을 구하는 공격방법으로, RSA의 평문과 암호문이 동일한 메시지 공간 {0, 1, 2, …, n-1}를 가지므로, $c^{e^k} \equiv c \pmod n$인 k가 존재하며, $c^{e^k} - 1 \equiv m \bmod n$이다.

(나) 알고리즘

- 입 력 : c, e, n
- 출 력 : c의 평문 m
- 과 정 :
  ① 해독하고자 하는 m에 대한 c, e, n을 획득한다.
  ② $c^{e^k}=c \bmod n$일 때까지 암호화를 수행한다.
  ③ $m=c^{e^k} - 1 \bmod n$을 계산한다.

5. 결 론

본 논문에서는 RSA 공개키 암호 시스템에 관하여 소인수분해 알고리즘의 안전성과 시스템 자체의 안전성 등으로 나누어 각각 그 문제점들을 고찰하였다. 본 논문에서 살펴본 바와 같이 안전한 암호 시스템의 설계를 위해서는 파라미터들의 신중한 선택이나 시스템 자체의 고유 특성에 관한 안전성, 그리고 프로토콜상의 안전성 등이 충분히 고려되어야 한다. 그러므로, 본 연구는 기존의 제안된 RSA 암호 시스템의 안전성 분석 및 앞으로 개발될 새로운 암호 시스템의 설계에 많은 도움을 줄 것으로 기대된다.

〈참고문헌〉

[1] D. Bleichenbacher, M. Joye, and J.-J. Quisquater, "A new and optimal chosen-message attack on RSA-type cryptosystems", Information and Communications Security (Y. Han, T. Okamoto, and S. Qing, eds.), Lecture Notes in Computer Science, vol. 1334, Springer-Verlag, 1997, pp. 302-313.

[2] D. Coppersmith, "Finding a small root of a bivariate integer equation; factoring with high bits known", Advances in Cryptology - Eurocrypt '96 (U. Maurer, ed.), Lecture Notes in Computer Science, vol. 1070, Springer-Verlag, 1996, pp. 178-189.

[3] D. Coppersmith, "Finding a small root of a univariate modular equation, Advances in Cryptology" - Eurocrypt '96 (U. Maurer, ed.), Lecture Notes in Computer Science, vol. 1070, Springer-Verlag, 1996, pp. 155-165.

[4] D. Coppersmith, M. Franklin, J. Patarin, and M. Reiter, "Low exponent RSA with related messages", Advances in Cryptology - Eurocrypt '96 (U. Maurer, ed.), Lecture Notes in Computer Science, vol. 1070, Springer-Verlag, 1996, pp. 1-9.

[5] G. Davida, " Chosen signature cryptanalysis of the RSA (MIT) public

key cryptosystem", Tech. Report TR-CS-82-2, Dept. of Electrical Engineering and Computer Science,university of Wisconsin, Milwaukee,uSA, October 1982.

[6] T. Denny, B. Dodson, A.K. Lenstra, M.S. Manasse, "On the Factorization of RSA-120", Crypto '93, 1993, pp. 166-174.

[7] W. Diffie and M. E. Hellman, "New directions in cryptography", IEEE Transaction on Information Theory IT-26 (1976), no. 6, pp. 644-654.

[8] Brandon Dixon, Arjen K. Lenstra, "Factoring Integers Using SIMD Sieves", Euro Crypt '93, 1993, pp. 28-39.

[9] M. Girault and J.-F. Misarsky, "Selective forgery of RSA signatures using redundancy", Advances in Cryptology-Eurocrypt '97 (W. Fumy, ed.), Lecture Notes in Computer Science, vol. 1233, Springer-Verlag, 1997, pp. 493-507.

[10] J. Hastad, "On using RSA with low exponent in a public key network, Advance in Cryptology" - Crypto '85 (H. C. Williams, ed.), Lecture Notes in Computer Science, vol. 218, Springer-Verlag, 1986, pp. 404-408.

[11] Klaus Huber, "Some Considerations concerning the Selection of RSA Moduli", Euro Crypt '91, 1991.

[12] W. de Jonge and D. Chaum, "Attacks on some RSA signatures", Advances in Cryptology - Crypto '85 (H. C. Williams, ed.), Lecture Notes in Computer Science, vol. 218, Springer-Verlag, 1986, pp. 18-27.

[13] W. de Jonge and D. Chaum, "Some variations on RSA signatures & their security", Advances in Cryptology - Crypto '86 (A. M. Odlyzko, ed.), Lecture Notes in Computer Science, vol. 263, Springer-Verlag, 1987, pp. 49-59.

[14] M. Joye, "Common modulus attack against Lucas-based cryptosystem." Tech. Report CG-1996/10, UCL Crypto Group, Louvain-la-Neuve, December 1996.

[15] M. Joye, "On the importance of securing your bins: The garbage-man-in-the-middle attack", Proc. of the 4th ACM Conference on Computer and Communications Security (T. Matsumoto, ed.), ACM Press, 1997, pp. 135-141.

[16] M. Joye, "Takagi/Naito's algorithm revisited", Tech. Report CG-1997/3, UCL Crypto Group, Louvain-la-Neuve, March 1997.

[17] N. Koblitz, A course in number theory and cryptography, 2nd ed., Graduate Texts in Mathematics, vol. 114, Springer-Verlag, 1994.

[18] D. E. Knuth, The art of computer programming: Volume 2/seminumerical algorithms, 2nd ed., Addison-Wesley, 1981.

[19] Arjen K. Lenstra, "Factoring with two large primes", Euro Crypto '90, 1990, pp. 72-82.

[20] Arjen K. Lenstra, Mark S. Manasse, "Factoring by electronic mail", Euro Crypt '89, 1989, pp. 1-10.

[21] A. Menezes, P. Oorschot, and S. Vanstone, Handbook of applied

cryptography, CRC Press, 1996.

[22] M. A. Morrison and J. Brillhart, "A method of factoring and factorization of F7", Math.Comp. 29 (1975), pp. 183-205.

[23] T. Okamoto and A. Shiraishi, "A fast signature scheme based on quadratic inequalities", Proc of the 1985 Symposium on Security and Privacy, Apr. 1985, Oakland, CA.

[24] J. M. Pollard, "A Monte Carlo Method for Factorization", BIT, 16 (1975), pp. 331-334.

[25] J. M. Pollard, "Theorems on factorizatin and primality testing", Proceeding of the Cambridge Philosophical Society, 76 (1975), pp 521-528.

[26] C. Pomerance, "Analysis and comparison of some integer factoring algorithm, in Computational Method in Computational Methods in Number Theory", H.W. Lenstra, Jr. and R. Tijdeman, eds., Math. Centrum 154 (1982), pp. 89-139.

[27] C. Pomerance, "The Quadratic Sieve Factoring Algorithm", Euro Crypt '84, 1984, pp. 169-182.

[28] L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM 21 (1978), no. 2, pp. 120-126.

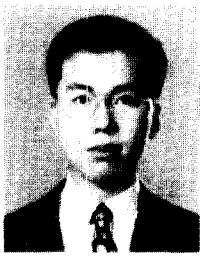
[29] B. Schneier, Applied cryptography: protocols, algorithms, and source code in C, 2nd ed., John Wiley & Sons, 1996.

[30] R.D. Silverman, "The multiple polynomial quadratic sieve, Math.Comp 84 (1987), pp. 327-339.

[31] G. J. Simmons and M. J. Norris, "Preliminary comments on the M.I.T. public-key cryptosystem", Cryptologia, 1 (1977), pp. 406-414.

[32] Michael J. Wiener, "Cryptanalysis of Short RSA Secret Exponents", Euro Crypt '89, 1989, pp. 1-15.

□ 著者紹介



조 동 욱

1998년 2월 성균관대학교 정보공학과 졸업(공학사)
1998년 3월 ~ 현재 성균관대학교 전기전자 및 컴퓨터공학부 석사과정

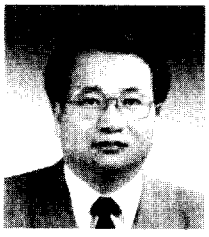
김 영 수

1998년 2월 성균관대학교 정보공학과 졸업(공학사)
1998년 3월 ~ 현재 성균관대학교 전기전자 및 컴퓨터공학부 석사과정

정 권 성

1996년 8월 성균관대학교 수학과 졸업(이학사)
1997년 3월 ~ 현재 성균관대학교 전기전자 및 컴퓨터공학부 석사과정

□ 著者紹介



원 동 호

1976년 2월 성균관대학교 전자공학과 졸업(공학사)
1978년 2월 성균관대학교 대학원 졸업(공학석사)
1988년 2월 성균관대학교 대학원 전자공학과(공학박사)
1978년 4월 ~ 1980년 3월 한국전자통신연구소 연구원
1985년 9월 ~ 1986년 8월 일본 동경공대 객원 연구원
1982년 3월 ~ 현재 성균관대학교 전기전자 및 컴퓨터공학부 교수
1991년 ~ 현재 한국통신정보보호학회 편집이사
1996년 4월 ~ 1998년 4월 정보화 추진위원회 자문위원

※ 주관심분야 : 암호이론, 정보이론