A Multiplicative Attack Using LLL Algorithm on RSA Signatures with Redundancy

Jean-François Misarsky

France Télécom - Branche Développement
Centre National d'Etudes des Télécommunications
42, rue des Coutures, B.P. 6243
14066 Caen Cedex, FRANCE
jeanfrancois.misarsky@cnet.francetelecom.fr

Abstract. We show that some RSA signature schemes using fixed or modular redundancy and dispersion of redundancy bits are insecure. Our attack is based on the multiplicative property of RSA signature function and extends old results of De Jonge and Chaum [DJC] as well as recent results of Girault and Misarsky [GM]. Our method uses the lattice basis reduction [LLL] and algorithms of László Babai [B]. Our attack is valid when the length of redundancy is roughly less than half the length of the public modulus. We successfully apply our attack to a scheme proposed for discussion inside ISO. Afterwards, we also describe possible adaptations of our method to attack schemes using mask or different modular redundancies. We explain limits of our attack and how to defeat it.

Keywords. Multiplicative attack, LLL algorithm, redundancy, RSA.

1 Introduction

Let n be a RSA modulus [RSA], e the public exponent, and d the secret exponent. We can define $P(x) = x^e \pmod n$ the public function and $S(x) = x^d \pmod n$ the secret one. The multiplicative property of RSA, i.e. the fact that $S(xy) = S(x)S(y) \pmod n$, leads to potential weaknesses, especially when used for signatures. We will make an extensive use of this property in our attack.

When a forger wants the signature of a message m, he generates two messages x and y that satisfy $m = xy \pmod n$. If he obtains the signatures of x and y, as exponentiation preserves the multiplicative structure of the input, he simply computes the signature of m as the product of $S(x)$ and $S(y)$, $S(m) = S(x)\,S(y) \pmod n$. This is a chosen-message attack.

Two standard ways exist to eliminate this potential weakness. One is to sign a hashed value of the message rather than the message itself. The other is to add some redundancy to the message to be signed. These different signature schemes are sometimes called, respectively, schemes with appendix and schemes with message recovery ([MOV], pp.428-432).

Only the redundancy solution is concerned by this paper. It is of particular interest when the message is short, because it prevents from specifying and implementing a hash-function (a rather delicate cryptographic challenge), and it allows to construct very compact signed messages, since messages can be recovered from the signatures themselves (and hence need not any longer be transmitted or stored). Let R be the invertible redundancy function. The signature of a message m is $\Sigma(m) = S[R(m)]$ and the signer only sends $\Sigma(m)$ of the receiver. The latter applies P to $\Sigma(m)$, and verifies that the result complies with the redundancy rule, i.e. is an element of the image set of R. Then he recovers m by discarding the redundancy, i.e. by applying R^{-1} to this result.
At Crypto'85 conference, De Jonge and Chaum [DJC] showed that simple redundancy does not avoid all the chosen-message attacks. In their paper, they show that it is not sufficient to append trailing '0' bits to the right or the left of the message. They study the case when redundancy is an affine function of m, i.e. the signature $\Sigma(m)$ to m is computed as $\Sigma(m) = S(\omega m + a)$. Their attack is based on Euclid's algorithm and is valid for any message m for:

- $a = 0$, and any value of ω such that the amount of redundancy is less than half the length of the public modulus n.
- $\omega = 1$, a small value of a, and when the amount of redundancy is less than one third of the length of the public modulus n.

Girault and Misarsky [GM] recently extended these results. Their attack uses an affine variant of Euclid's algorithm due to Okamoto and Shiraishi [OS]. It is valid for any constant ω, any constant a, any message m provided that the amount of redundancy is less than half the length of the public modulus n. Moreover, they study the case when modular redundancy is used, i.e. when the amount of redundancy is obtained by appending to m the remainder of m modulo some fixed value. In this case, the signature is still subject to a chosen-message attack when redundancy is less than half the length of the public modulus, minus the length of remainder. They give three solutions that prevent their attack; one of them consists in dispersing the message in different parts and another one in using two different modular redundancies.
We show in this paper that a multiplicative attack is feasible on signature scheme that uses dispersion of redundancy bits and fixed or modular redundancy. We precisely explain our attack in this case. But our attack is also valid on more simple schemes or schemes with mask or different modular redundancies.
Our method makes use of the lattice basis reduction, which has not been used in multiplicative attacks yet. But, lattice reduction has already been applied successfully in cryptoanalysis: against Merkle-Hellman public key cryptosystem [S], against Okamoto's cryptosystems [VGT1], against RSA cryptosystem with small exponent [H], or against RSA encryption with small exponents and random padding [C], for instance.
We successfully apply our method on ISO 9796 Part 3, Working Draft, December 1996 [ISO2], a scheme using dispersion of redundancy bits and modular redundancy. Afterwards, we explain limits of our attack and how to defeat it.
Throughout this paper, we call bitlength (or length in short) of an integer the number of bits of its binary representation. We denote by $|m|$ the bitlength of m.

2 Our Results

We describe a method using lattice basis reduction that finds solutions x and y of the equation $R(m)R(x) = R(y) \pmod{n}$ where:

- R is a redundancy function
- m is a message of which we want to forge a signature

If signatures of x and y can be obtained, i.e. respectively $\Sigma(x) = S(R(x)) \pmod{n}$ and $\Sigma(y) = S(R(y)) \pmod{n}$, then the signature of m can be easily forged:

$$\Sigma(m) = \frac{\Sigma(y)}{\Sigma(x)} \pmod{n}$$

In the sequel, we denote by:

$\omega_1, \omega_2, \ldots$: miscellaneous multiplicative redundancies constants

a : fixed redundancy constant

m : a message

k_1 : the number of parts of m

m_i : the i^{th} part of m. The message m is split up into k_1 parts which have not necessary the same length:

$m =$ | m_1 | m_2 | ..... | m_i | ..... | m_{k_1} |

$\varphi(m)$: modular redundancy of the message m i.e. the remainder of m modulo a fixed value

k_2 : the number of parts of $\varphi(m)$

$\varphi(m)_j$: the j^{th} part of $\varphi(m)$. The modular redundancy is split up into k_2 parts which have not necessary the same length:

$\varphi(m) =$ | $\varphi(m)_1$ | $\varphi(m)_2$ | ..... | $\varphi(m)_j$ | ..... | $\varphi(m)_{k_2}$ |

n : RSA modulus

m_r : redundancy modulus ($\varphi(m) = m \pmod{m_r}$)

The redundancy function R can take several forms, with increasing complexity:

i) $R(m) = \omega m + a$

ii) $R(m) = \omega_1 m + \omega_2 \varphi(m) + a$

iii) $R(m) = \sum_{i=1}^{k_1} m_i \omega_i + a$

iv) $R(m) = \sum_{i=1}^{k_1} m_i \omega_i + \sum_{j=1}^{k_2} \varphi(m)_j \omega_{j+k_1} + a$

The case iv) generalizes the others and we only study it in the sequel.

Example: when all ω_i are powers of two in the case iv), one could have:

$R(m) =$	10111...	m_1	$\varphi(m)_1$	..1001..	$\varphi(m)_i$	m_i	$\varphi(m)_{i+1}$	...	..1011..

with $a =$	10111...	0...0	0.....0	..1001..	0.....0	0...0	0.....0	...	..1011..

Remark: we call *the number of bits of redundancy* the length of n minus the length of m. Note that the number of bits of modular redundancy is included in the number of bits of redundancy.

Main result:

If a signature scheme uses this kind of redundancy function:

$$R(m) = \sum_{i=1}^{k_1} m_i \omega_i + \sum_{j=1}^{k_2} \varphi(m)_j \omega_{j+k_1} + a$$

then our attack is valid when the number of bits of redundancy is roughly less than half the length of the public modulus n, minus the number of bits of modular redundancy (when the latter is present):

$$|redundancy| < \frac{1}{2}|n| - |m_r|$$

Another version of our attack, requiring more computation and memory, is valid when the number of bits of redundancy is roughly less than half the length of the public modulus n.

3 System: Definition and Solution

Solving $R(m)R(x) = R(y) \pmod n$ is equivalent to finding the different parts of $R(x)$ and $R(y)$, i.e. respectively $(x_i)_{1 \le i \le k_1}$, $(\varphi(x)_j)_{1 \le j \le k_2}$, and $(y_i)_{1 \le i \le k_1}$, $(\varphi(y)_j)_{1 \le j \le k_2}$.
Let $(X_i)_{1 \le i \le k}$ be the different parts of $R(x)$ to find, i.e. all or only part of $(x_i)_{1 \le i \le k_1}$ and $(\varphi(x)_j)_{1 \le j \le k_2}$. Let $(Y_i)_{1 \le i \le k}$ be the different parts of $R(y)$ to find, i.e. all or only part of $(y_i)_{1 \le i \le k_1}$ and $(\varphi(y)_j)_{1 \le j \le k_2}$. The modular redundancy, the fact that $x = \varphi(x) \pmod{m_r}$ and $y = \varphi(y) \pmod{m_r}$, implies two equations:

$$(i) \qquad a_1X_1 + a_2X_2 + ... + a_{k-1}X_{k-1} = X_k + h_1 \pmod{m_r}$$
$$(ii) \qquad b_1Y_1 + b_2Y_2 + ... + b_{k-1}Y_{k-1} = Y_k + h_2 \pmod{m_r}$$

with $(a_i)_{1 \le i \le k-1}$, $(b_i)_{1 \le i \le k-1}$, h_1 and h_2 fixed integers.

Note that:

- h_1 and h_2 are present only when some parts of $R(x)$ and $R(y)$ are fixed, i.e. one or several x_i, $\varphi(x)_j$, y_i or $\varphi(y)_j$ are fixed.
- our method requires the coefficients of X_k and Y_k to be equal to one. It is easily obtained by a division modulo m_r. We have deliberately omitted to describe this step.

$R(m)R(x) = R(y) \pmod n$ also implies an other equation:

$$(iii) \qquad c_1X_1 + c_2X_2 + ... + c_kX_k + d_1Y_1 + d_2Y_2 + ... + d_{k-1}Y_{k-1} = Y_k + h_3 \pmod n$$

with $(c_i)_{1 \le i \le k}$, $(d_i)_{1 \le i \le k-1}$ and h_3 fixed integers.

Let (*SI*) be the system:

$$(SI)\quad \begin{cases} a_1X_1+\ldots+a_{k-1}X_{k-1} = X_k+h_1 \pmod{m_r} & (i)\\ b_1Y_1+\ldots+b_{k-1}Y_{k-1} = Y_k+h_2 \pmod{m_r} & (ii)\\ c_1X_1+\ldots+c_kX_k+d_1Y_1+\ldots+d_{k-1}Y_{k-1} = Y_k+h_3 \pmod{n} & (iii)\end{cases}$$

(*SI*) is a system with constraints on value of $(X_i)_{1\le i\le k}$ and $(Y_i)_{1\le i\le k}$.
We have for $1 \le i \le k$:

$$0 \le X_i < 2^{\text{Length of the part } X_i \text{ in bits}}$$
$$0 \le Y_i < 2^{\text{Length of the part } Y_i \text{ in bits}}$$

When modular redundancy is not used in the signature scheme, (*i*) and (*ii*) are useless. Only (*iii*) is necessary.
In the first part of our study, we define a lattice where all points give a solution to this system without second member, $h_1 = h_2 = h_3 = 0$, and without constraints on values of $(X_i)_{1\le i\le k}$ and $(Y_i)_{1\le i\le k}$. Next, we define a method to find a solution to (*SI*) without constraints on values of $(X_i)_{1\le i\le k}$ and $(Y_i)_{1\le i\le k}$ by using this lattice. After, we explain how to obtain solutions to the system (*SI*) with additional constraints on values of $(X_i)_{1\le i\le k}$ and $(Y_i)_{1\le i\le k}$. Finally we study the efficiency of our method.

3.1 First Step: Determination of the Lattice

We define an integer lattice L such that any element of this lattice is solution to (*S*). (*S*) is the system (*SI*) without second member and without constraints on values of $(X_i)_{1\le i\le k}$ and $(Y_i)_{1\le i\le k}$.

$$(S)\quad \begin{cases} a_1X_1+\ldots+a_{k-1}X_{k-1} = X_k \pmod{m_r}\\ b_1Y_1+\ldots+b_{k-1}Y_{k-1} = Y_k \pmod{m_r}\\ c_1X_1+\ldots+c_kX_k+d_1Y_1+\ldots+d_{k-1}Y_{k-1} = Y_k \pmod{n}\end{cases}$$

Hence, we define a lattice L of dimension $2k$ such that any vector $v = (v_1, v_2, \ldots, v_{2k-1}, v_{2k})$ verifies:

$$(a)\quad a_1v_1 + a_2v_2 +\ldots+ a_{k-1}v_{k-1} = v_k \pmod{m_r}$$
$$(b)\quad b_1v_{k+1} + b_2v_{k+2} +\ldots+ b_{k-1}v_{2k-1} = v_{2k} \pmod{m_r}$$
$$(c)\quad c_1v_1 + c_2v_2 +\ldots+ c_kv_k + d_1v_{k+1} + d_2v_{k+2} +\ldots+ d_{k-1}v_{2k-1} = v_{2k} \pmod{n}$$

Let M be the matrix of lattice L. Columns vectors of M are a basis of L, and for any element v of L, there is a column vector α with integer components such that:

$$M\,\alpha = v$$

Now, we construct this matrix M.
We denote by M_1 an identity matrix of dimension $2k$ where the row k is replaced by:

$$(a_1, a_2,\ldots,a_{k-1}, m_r, 0, 0,\ldots,0)$$

Then, for any vector α with integer components, $v = M_1\,\alpha$ is a vector with components satisfying (*a*). Remark that $v_i = \alpha_i$ with $1 \le i \le 2k$, $i \ne k$ and:

(*d*) $$v_k = a_1 v_1 + a_2 v_2 + \ldots + a_{k-1} v_{k-1} + \alpha_k m_r$$

We gather equations (*b*) and (*c*) together with the Chinese Remainder Theorem. It is possible because n is the public modulus of RSA and is prime with m_r (otherwise we have a factor of n !).
We denote by Chinese the Chinese remainder function:
Chinese(a (mod m), b (mod n)), with m and n relatively primes, returns x such that:

$$\begin{cases} x = a \pmod{m} \\ x = b \pmod{n} \end{cases}$$

Let $(f_i)_{1 \le i \le 2k}$ such that:

f_i = Chinese(0 (mod m_r), c_i (mod n)) when $1 \le i \le k$
f_i = Chinese(b_{i-k} (mod m_r), d_{i-k} (mod n)) when $k + 1 \le i \le 2k - 1$

We obtain:

(*f*) $$f_1 v_1 + f_2 v_2 + \ldots + f_{k-1} v_{k-1} + f_k v_k + f_{k+1} v_{k+1} + f_{k+2} v_{k+2} + \ldots + f_{2k-1} v_{2k-1} = v_{2k} \pmod{m_r n}$$

But α_k is different from v_k. We use (*d*) to replace v_k in (*f*). We finally obtain an equation (*e*), equivalent to (*b*) and (*c*), that has this form:

(*e*) $$e_1 v_1 + e_2 v_2 + \ldots + e_{k-1} v_{k-1} + e_k \alpha_k + e_{k+1} v_{k+1} + e_{k+2} v_{k+2} + \ldots + e_{2k-1} v_{2k-1} = v_{2k} \pmod{m_r n}$$

with:
$e_i = f_i + a_i f_k$ when $1 \le i \le k-1$
$e_k = f_k m_r$
$e_i = f_i$ when $k + 1 \le i \le 2k-1$

Finally, the matrix M is the matrix M_1 where the latest row is replaced by the vector:

$$(e_1, e_2, \ldots, e_{k-1}, e_k, e_{k+1}, \ldots, e_{2k-1}, m_r n)$$

We have:

$$M = \begin{pmatrix} 1 & 0 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & 0 \\ 0 & \ddots & \ddots & & & & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & & & & & \vdots \\ 0 & \ldots & 0 & 1 & \ddots & & & & \vdots \\ a_1 & \ldots & \ldots & a_{k-1} & m_r & 0 & & & \vdots \\ 0 & \ldots & \ldots & \ldots & 0 & 1 & \ddots & & \vdots \\ \vdots & & & & & \ddots & \ddots & 0 & \vdots \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots & 0 & 1 & 0 \\ e_1 & e_2 & \ldots & \ldots & \ldots & \ldots & e_{2k-2} & e_{2k-1} & m_r n \end{pmatrix}$$

A solution to the system (*S*) is obtained by multiplying matrix M by an integer vector α. The result v gives a solution to (*S*): $v_1, \ldots, v_k$ will be $X_1, \ldots, X_k$ and $v_{k+1}, \ldots, v_{2k}$ will be $Y_1, \ldots, Y_k$. The reciprocal can be easily demonstrated and consequently we have:

Proposition 3.1.1:

A vector is in L if and only if it is a solution to (S).

3.2 Second step: System with a Second Member

Let (*S'*) be the system (*S*) with a second member. (*S'*) is the initial system (*SI*) but without constraints on values of solutions. The same lattice *L* is used to solve (*S'*).

Proposition 3.2.1:

Let $v = (v_1, v_2, ..., v_{2k-1}, v_{2k})$ *be a vector of L.*

Let $P = (0, 0, ..., 0, p_k, 0, ..., 0, p_{2k})$ *with*

$p_k = h_1$

$p_{2k} = Chinese(h_2 \pmod{m_r}, h_3 \pmod{n}) + Chinese(0 \pmod{m_r}, c_k \pmod{n}).h_1$

Then $\beta = v - P$ *gives a solution to (S').*

$\beta_1, ..., \beta_k$ *will be* $X_1, ..., X_k$ *and* $\beta_{k+1}, ..., \beta_{2k}$ *will be* $Y_1, ..., Y_k$

Proof:

$\beta_i = v_i$ when $i \in \{1, 2, 3, ..., k-1, k+1, k+2, ..., 2k-1\}$

$\beta_k = v_k - p_k$

$= a_1v_1 + a_2v_2 + ... + a_{k-1}v_{k-1} + \alpha_k m_r - h_1$

$= a_1\beta_1 + a_2\beta_2 + ... + a_{k-1}\beta_{k-1} + \alpha_k m_r - h_1$

$\beta_{2k} = v_{2k} - p_{2k}$

$= e_1v_1 + ... + e_{k-1}v_{k-1} + e_k\alpha_k + e_{k+1}v_{k+1} + e_{k+2}v_{k+2} + ... + e_{2k-1}v_{2k-1} + \alpha_{2k} m_r n - p_{2k}$

And we have:

$\beta_{2k} \pmod{m_r} = b_1v_{k+1} + b_2v_{k+2} + ... + b_{k-1}v_{2k-1} - h_2$

$= b_1\beta_{k+1} + b_2\beta_{k+2} + ... + b_{k-1}\beta_{2k-1} - h_2$

$\beta_{2k} \pmod{n} = c_1v_1 + c_2v_2 + ... + c_kv_k + d_1v_{k+1} + ... + d_{k-1}v_{2k-1} - h_3 - c_kh_1$

$= c_1\beta_1 + c_2\beta_2 + ... + c_k(v_k - h_1) + d_1\beta_{k+1} + ... + d_{k-1}\beta_{2k-1} - h_3$

As $v_k - h_1 = v_k - p_k = \beta_k$, we have:

$\beta_{2k} \pmod{n} = c_1\beta_1 + c_2\beta_2 + ... + c_k\beta_k + d_1\beta_{k+1} + ... + d_{k-1}\beta_{2k-1} - h_3$

Thus, β gives a solution to (*S'*). ■

3.3 Third step: Additional Constraints

We always consider the system (*S'*), but we take into account the initial constraints on values of $(X_i)_{1 \le i \le k}$ and $(Y_i)_{1 \le i \le k}$. Hence, we solve (*SI*).

First case: same bounds

Let *B* be a positive integer. Find X_i and Y_i such that $0 \le X_i \le B$ and $0 \le Y_i \le B$ for any i such that $1 \le i \le k$.

Proposition 3.3.1:

Let HC be a ball of radius B/2, relative to the norm sup, centred on $Q = P + (B/2, B/2, ..., B/2)$*, where the point P is defined in the proposition 3.2.1. Let v be a vector of L inside HC, and* $\beta = v - P$.

Then β *gives a solution to (S') and satisfies additional constraints.*

Proof:

Proposition 3.2.1 shows that β gives a solution to (S').

v inside HC implies $0 \le v_i - p_i \le B$, i.e. $0 \le \beta_i \le B$, for any $1 \le i \le 2k$. ■

Second case: distinct bounds

Let $(B_i)_{1 \le i \le 2k}$ be a family of positive integers. Find X_i and Y_i such that: $0 \le X_i \le B_i$ and $0 \le Y_i \le B_{k+i}$ for any i such that $1 \le i \le k$.

We apply a method of expansion-contraction to the lattice L to obtain another lattice L', see [VGT1] and [VGT2] for more details. We denote by M' the matrix of lattice L'.

Define B as $B^{2k} = \prod_{i=1}^{2k} B_i$. Let $(\lambda_i)_{1 \le i \le 2k}$ such that $\lambda_i = \frac{B}{B_i}$. Then the product $\prod_{i=1}^{2k} \lambda_i$ is equal to 1. M' is the matrix M where each row i, $1 \le i \le 2k$, is multiplied by λ_i:

$$M' = \begin{pmatrix} \lambda_1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & 0 \\ 0 & \ddots & \ddots & & & & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & & & & & \vdots \\ 0 & \dots & 0 & \lambda_{k-1} & 0 & & & & \vdots \\ \lambda_k a_1 & \dots & \dots & \lambda_k a_{k-1} & \lambda_k m_r & 0 & & & \vdots \\ 0 & \dots & \dots & \dots & 0 & \lambda_{k+1} & 0 & & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 & \lambda_{2k-1} & 0 \\ \lambda_{2k} e_1 & \lambda_{2k} e_2 & \dots & \dots & \dots & \dots & \lambda_{2k} e_{2k-2} & \lambda_{2k} e_{2k-1} & \lambda_{2k} m_r n \end{pmatrix}$$

Remark that $\det(M) = \det(M') = m_r^2 n$.

Proposition 3.3.2:

Let $P' = (0, 0, \dots, 0, p_k\lambda_k, 0, \dots, 0, p_{2k}\lambda_{2k})$, where P is defined in the proposition 3.2.1. Let HC be a ball, relative to the norm sup, of radius B/2 centred on $Q = P' + (B/2, B/2, \dots, B/2)$. Let v' be an element of L' inside HC and $\beta' = v' - P'$. Then $\beta = (\lambda_1^{-1}\beta_1', \dots, \lambda_{2k}^{-1}\beta_{2k}')$ gives a solution to (S') and satisfies additional constraints.

Proof:

$\beta = (\lambda_1^{-1}\beta_1', \dots, \lambda_{2k}^{-1}\beta_{2k}') = (\lambda_1^{-1}(v_1' - p_1'), \dots, \lambda_{2k}^{-1}(v_{2k}' - p_{2k}'))$

$= (\lambda_1^{-1}v_1' - p_1, \dots, \lambda_{2k}^{-1}v_{2k}' - p_{2k})$

Let $v = (\lambda_1^{-1}v_1', \dots, \lambda_{2k}^{-1}v_{2k}')$ and $\beta = v - P$.

Then $v \in L$ and proposition 3.2.1 shows that β gives a solution to (S').

v' inside HC implies $0 \le v_i' - p_i' \le B$, i.e. $0 \le \beta_i \le B_i$, for any $1 \le i \le 2k$. ■

Remark: the first case is a particular case of the second. In the sequel, we will consider always the lattice L' and its matrix M'.

3.4 How to Generate a Solution?

Proposition 3.3.2 shows that a point of lattice L' inside HC gives a solution to the system (SI). To find one, take a point x inside HC and find a close lattice point inside HC.

First, apply the LLL algorithm [LLL] to the matrix M'. A reduced basis of L' is obtained. Next, apply one of two algorithms of László Babai, *Rounding Off* or *Nearest Plane*, described in [B] to find a solution.

Let u be the nearest lattice point of x and d the dimension of lattice L'.

- **ROUNDING OFF**: this algorithm finds a lattice point v' such that:

$$|x - v'| \leq C_d |x - u| \text{ with } C_d = 1 + 2d(9/2)^{d/2}$$

- **NEAREST PLANE**: this algorithm finds a lattice point v' such that:

$$|x - v'| \leq C_d |x - u| \text{ with } C_d = 2^{d/2}$$

Remark that, if the dimension d of lattice increases, then the probability that one of these algorithms finds a lattice point inside HC decreases.

3.5 Efficiency: Heuristic Approach

Heuristically, if the ratio of the HC volume to the lattice determinant is greater than 1, then there is at least one lattice point in HC.

The *Nearest Plane* algorithm certainly finds this point when the dimension d of the lattice is not too large. When d increases, the term $C_d = 2^{d/2}$ increases too, and the probability to obtain a point inside HC decreases.

We study the general case where the redundant version of m is:

$$R(m) = \sum_{i=1}^{k_1} m_i \omega_i + \sum_{j=1}^{k_2} \varphi(m)_j \omega_{j+k_1} + a$$

This is the most complicated case, and the solutions to the others can be derived from the following analysis.

We denote by:

t : the bitlength of n

b_i: the length of the part m_i of m

b : such that $\sum_{i=1}^{k_1} b_i = b$

c_j: the length of the part $\varphi(m)_j$ of $\varphi(m)$

c : such that $\sum_{j=1}^{k_2} c_j = c$

First method: modular redundancies are fixed

Modular redundancies $\varphi(x)$ and $\varphi(y)$ are fixed. Finding two messages x and y such that $R(m)R(x) = R(y) \pmod n$ is equivalent to solve (SI) with $k = k_1$.

Lattice dimension : $d = 2k_1$

Lattice determinant : $\det(L) = \det(L') = m_r^2 n < \left(2^c\right)^2 2^t$

HC volume : $\left(2^{\frac{2\sum_i b_i}{d}}\right)^d = 2^{2b}$

Heuristically, there is one point in *HC* if:

$$\begin{aligned} \frac{2^{2b}}{2^{2c+t}} &> 1 \\ 2b &> 2c+t \\ b &> \frac{t}{2}+c \\ t-b &< \frac{t}{2}-c \end{aligned}$$

$$\boxed{|redundancy| < \frac{1}{2}|n| - |m_r|}$$

Second method: modular redundancies are not fixed

Finding two messages x and y such that $R(m)R(x) = R(y) \pmod n$ is equivalent to solve (*SI*) with $k = k_1 + k_2$. But, there is a disadvantage when modular redundancies are not fixed. The dimension of lattice increases and therefore the probability to find a lattice point in *HC* with Babai's algorithms decreases.

Lattice dimension : $d = 2(k_1 + k_2)$

Lattice determinant : $\det(L) = m_r^2 n < \left(2^c\right)^2 2^t$

HC volume : $\left(2^{\frac{2\left(\sum_i b_i + \sum_j c_j\right)}{d}}\right)^d = 2^{2(b+c)}$

Heuristically, there is one point in *HC* if:

$$\begin{aligned} \frac{2^{2(b+c)}}{2^{2c+t}} &> 1 \\ 2(b+c) &> 2c+t \\ b &> \frac{t}{2} \\ t-b &< \frac{t}{2} \end{aligned}$$

$$\boxed{|redundancy| < \frac{1}{2}|n|}$$

4 Application

We applied our attack on a project of digital signature schemes giving message recovery ISO/IEC 9796-3, Working Draft, December 1996 [ISO2]. It is supposed to avoid the known attacks against RSA [GQLS]. This part of ISO/IEC 9796 specifies a digital signature scheme for messages of limited length, so that the message is completely recovered from the signature. It uses a check-function to save bits and computations. This check-function is a modular redundancy, it is the remainder of the message to be signed modulo $2^{79}+1$. The modular redundancy takes the form:

$$R(m) = \sum_{i=1}^{k_1} m_i \omega_i + \sum_{j=1}^{k_2} \varphi(m)_j \omega_{j+k_1} + a$$

with all $(\omega_i)_{1 \le i \le k_1+k_2}$ powers of two. We experiment our attacks on this scheme with a public modulus n of 640 bits of length. In this case the project defines an intermediate string IS:

Structure of the intermediate string IS (640-80 = 560 bits)

Header	Padding Field	Data field	Trailer
Three bits	640 - k_m - 87 bits	k_m bits	Four bits
Set at 010	640 - k_m - 88 bits set to 0 followed by one bit set to 1	Message m	Set to 0110

The structure of the valid message (640 bits) is:

Binary pattern (check-code in bold)

$$12 + \mathbf{4} + 28 + \mathbf{4} + 28 + \mathbf{4} + \ldots + 28 + \mathbf{4} + 28 + \mathbf{4} + 16 = 640 \text{ bits}$$

We applied the first method, i.e. we fixed the check-code. We found several solutions by using the *Rounding Off* algorithm. We give an example of solution:

Public modulus:

```
n =  ffffffff  78f6c555  06c59785  e871211e
     e120b0b5  dd644aa7  96d82413  a47b2457
     3f1be574  5b5cd995  0f6b389b  52350d4e
     01e90009  669a8720  bf265a28  65994190
     a661dea3  c7828e2e  7ca1b196  51adc2d5
```

Message m:

```
m =  fedcba98  76543210  fedcba98  76543210
     fedcba98  76543210  fedcba98  76543210
     fedcba98  76543210  fedcba98  76543210
     fedcba98  76543210  fedcba98  76543210
```

Check-code:

```
c =  0f6e 4af3 a0b1 3571 358b
```

Valid message of m:

$Sr(m)$ =

```
4bb0bbbb  bbbfafed  cba69876  543e210f
edc4ba98  765a4321  0fefdcba  98736543
210afedc  ba908765  432b10fe  dcb1a987
65433210  fed5cba9  87675432  10f1edcb
a9837654  32150fed  cba89876  543b2106
```

Message x:

x =

```
fedcba0e  2fff215a  00200b1f  17a18638
3212ac94  21061f58  0619a4f0  f912910d
bd3220e3  f4b8064c  89f15211  880c5445
6127d8c9  1a336791  5b962f17  a8386210
```

Message y:

y =

```
fedcba14  7597b137  39d20f85  33b07f20
cd1335d1  308be96c  14b053d1  4230e40f
02b2f14a  39f709a6  e6a0ede5  ae1f6313
50f4eaf1  1a2f2381  064c2f0f  f3ffa210
```

Valid message of x:

$Sr(x)$ =

```
4bb0bbbb  bbbfafed  cba60e2f  ff2e15a0
02040b1f  17aa1863  832f12ac  94231061
f58a0619  a4f00f91  291b0dbd  32210e3f
4b83064c  89f51521  18870c54  4561127d
8c931a33  679515b9  62f817a8  386b2106
```

Valid message of y:

$Sr(y)$ =

```
4bb0bbbb  bbbfafed  cba61475  97be1373
9d240f85  33ba07f2  0cdf1335  d13308be
96ca14b0  53d01423  0e4b0f02  b2f114a3
9f7309a6  e6a50ede  5ae71f63  13510f4e
af131a2f  23851064  c2f80ff3  ffab2106
```

We obtained this result within 30 minutes on a Pentium 166MHz by using GP/PARI CALCULATOR Version 1.39 (`ftp: megrez.math.u-bordeaux.fr/pub/pari`). It is the time necessary to apply LLL algorithm to the initial matrix. After, we can easily obtain different messages x and y in a few seconds by using *Rounding Off* or *Nearest Plane* algorithm on different points inside *HC*.

5 Extensions

We have described an attack on a signature scheme using one modular redundancy. But it is possible to increase the number of modular redundancies. If the different moduli are relatively prime, they can be gathered into one equation with the Chinese Remainder theorem and solved with the first method. If these moduli are not relatively prime, we use the second method, then the probability to find a solution is lower because the dimension is high.

We denote by mask a k_2-bit fixed string. Our attack also succeeds on a scheme that uses a modular redundancy and a mask, i.e. you apply the function exclusive OR between modular redundancy and the mask. In this case we use the first method.

6 How to defeat this forgery

If you want to use fixed or modular redundancy, it is recommended to have the same amount of redundancy as the number of bits of message m, and to have a big dispersion of redundancy bits. It is not sure that you cannot apply our attack but the probability of success will be small.
Another way to avoid this attack is to split the message and define bits of redundancy as parity bits (such as those determined by Hamming codes) of its different parts. ISO 9796 [ISO1] is another possible solution, but it doubles the length of the bit pattern you sign. Our attack cannot apply to the latter schemes because the redundancy depends on different bits of message m and we cannot adjust our attack to this case.

7 Conclusion

This paper describes two attacks to forge a signature of a message m when the bits of redundancy are dispersed and/or when a modular redundancy is used. The first one is valid when the length of redundancy is less than half the length of public modulus, minus the length of modular redundancy. The second attack is valid when the length of redundancy is less than half the length of public modulus, but the probability to find a forgery is smaller (because the lattice dimension grows); however, we have noticed that the *Nearest Plane* and *Rounding Off* algorithms [B] generally give better results than expected.
Afterwards, we have briefly described possible adaptations of our method to attack schemes using mask or different modular redundancies. Hence, we have shown the weakness of many attractive redundancy functions for the purpose of RSA digital signatures.
Finally, we advise to use, for RSA signature scheme with fixed or modular redundancy, the same length of redundancy that the length of the message and to disperse message bits in the valid message. But the best solution remains to use ISO 9796 [ISO1] or the parity bits scheme briefly described above, because they apparently cannot be attacked by our techniques.

Acknowledgments

We would like to thank Marc Girault for encouraging this research and for helpful comments on this paper. We are grateful to Brigitte Vallée for help on the lattice theory and for pointing out corrections to the initial draft. We also thank Louis Guillou for stimulating this research, Jacques Traoré and the referees for their useful comments.

References

[B] L. Babai, "On Lovász' lattice reduction and the nearest lattice point problem", Combinatorica 6, pp. 1-14.

[C] Don Coppersmith, "Finding a Small Root of Univariate Modular Equation", Proceedings of Eurocrypt '96, Lecture Note in Computer Science, vol. 1070, pp. 155-165.

[DJC] W. De Jonge, D. Chaum, "Attacks on some RSA Signatures", Advances in Cryptology, Crypto '85 Proceedings, Lecture Notes In Computer Science, vol. 218, Springer-Verlag, Berlin, 1986, pp. 18-27.

[GQLS] L.C. Guillou, J.J. Quisquater, P. Landrock, C. Shaer, "Precautions taken against various potential attacks in ISO/IEC DIS 9796, Digital signature scheme giving message recovery", Eurocrypt '90 Proceedings, Lecture Notes in Computer Science, vol.473, Springer-Verlag, pp 465-473.

[GM] M. Girault, J.F. Misarsky, "Selective Forgery of RSA Signatures Using Redundancy", Advances in Cryptology - Eurocrypt '97, Lecture Notes in Computer Science, vol. 1233, Springer-Verlag, pp 495-507.

[H] J. Hastad, "Solving simultaneous modular equations of low degree", SIAM J. Comput. vol.17, No.2, April 1988.

[ISO1] ISO/IEC 9796, December 1991, "Digital signature scheme giving message recovery".

[ISO2] ISO/IEC 9796-3, Working Draft, December 1996, "Digital signature schemes giving message recovery; Part 3: Mechanisms using a check-function".

[LLL] A. K. Lenstra, H. W. Lenstra, L. Lovász, "Factoring Polynomials with Rational Coefficients", Mathematische Annalen, vol. 261, n. 4, 1982, pp. 515-534.

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

[MOV] A.J. Menezes, P.C. Van Oorschot, S.A. Vanstone, "Handbook of Applied Cryptography", CRC Press.

[RSA] R.L. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", CACM, Vol. 21, n°2, Feb. 1978, pp. 120-126.

[S] A. Shamir, "A polynomial-time algorithm for breaking the basic Merkle-Hellman cryptosystem", Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science, pp 145-152. IEEE, 1982.

[VGT1] B. Vallée, M. Girault, P. Toffin, "How to break Okamoto's cryptosystems by reducing lattice bases", Proceedings of Eurocrypt'87, Lecture notes in Computer Science.

[VGT2] B. Vallée, M. Girault, P. Toffin, "How to guess L-th roots modulo n by reducing lattice bases", Proc. of Conference of ISSAC-88 and AAECC-6, Jul. 88.