HASH-FUNCTIONS USING MODULO-N OPERATIONS

Marc Girault

Service d'Etudes communes des Postes et Télécommunications
42 rue des Coutures
B.P. 6243
14066 Caen-Cedex, France

INTRODUCTION

Today, there is a need for one-way hash-functions, particularly for use in digital signatures [1]. Following R.S. Winternitz [2], we define that H is a one-way hash-function if it maps messages of arbitrary length to some small fixed length, such that it is computationally infeasible to find two different messages M and M' hashing to the same value $H(M) = H(M')$. Now, if Alice wishes to sign M using for example the public-key system RSA, she submits $H(M)$ to her secret function S_{Alice} , and the signature of M is $\text{Sig} = S_{\text{Alice}}[H(M)]$. The functions H and P_{Alice} (her RSA public function) being public, anybody who received the plain message M along with its signature Sig , is able to verify the signature by matching $P_{\text{Alice}}(\text{Sig})$ against $H(M)$. The one-way property of H is not the only one required [3], but is the essential one: it prevents anybody (including Alice) from claiming that Sig is Alice's signature of a message M' , different from M .

Until now, much attention has been paid to hash-functions based on a conventional encipherment algorithm, generally DES [2,3,4,5,6,7]. Nevertheless, it is also useful to design hash-functions using modulo- n operations, particularly if (as above) the hash-result is intended to be signed by the secret function of a public-key system, because these systems use this type of operations. In 1984, D.W. Davies and W.L. Price proposed such a hash-function [8,9]. The main objective of this paper is to show the weakness of their scheme (referred to below as the DP-scheme) and to examine some variations of it. This weakness was pointed out (but without details) for the first time by A. Jung in [10].

I TWO BASIC MODULO-N SCHEMES

Let n be a public integer (the so-called modulus) and $(+)$ denote the bit-by-bit modulo 2 addition. By "message", we mean any pattern of

bits (issued from the natural language, a random number generator or any other source). As a consequence, a message may or may not have some internal redundancy.

I.1 TYPE 1 SCHEMES

A first basic scheme has been introduced by R.R. Jueneman [11], though used by him in a somewhat different way. It consists in dividing the message M to be hashed into k blocks B_i smaller than n and in performing:

$$\begin{aligned} H_0 &= I \text{ (an initialization vector)} \\ H_i &= (H_{i-1} (+) B_i)^2 \pmod n \quad 1 \leq i \leq k \end{aligned} \quad (E)$$

The hash-result is $H = H_k$.

Such a scheme is similar to the CBC mode of a block-cipher algorithm. But so described it is not secure, for the enemy can choose all the blocks as he likes and correct the resulting alteration by properly choosing the last one.

Precisely, he can replace $(B_1, \dots, B_{k-1})$ with $(B'_1, \dots, B'_{q-1})$, which lead to partial hash-results $(H'_1, \dots, H'_{q-1})$, then choose the last block B'_q equal to $H_{k-1} (+) H'_{q-1} (+) B_k$. In that way, the input of the last modulo- n squaring remains unchanged and the hash-result too. For commodity, we will call this attack the "correcting last block (CLB) attack".

The reply consists in forcing the blocks to have some explicitly defined (and sufficiently high) redundancy. So, the attack mentioned above becomes ineffective because the probability is close to 1 that the corrected block B'_q will not satisfy the redundancy rule. But the following parts of this paper show that some precautions must be taken when adopting such a rule, otherwise a CLB attack remains possible, though becoming a little less easy.

I.2 TYPE 2 SCHEMES

A second basic scheme consists in replacing the equations (E) with:

$$\begin{aligned} H_0 &= I \text{ (an initialization vector)} \\ H_i &= H_{i-1} (+) (B_i^2 \pmod n) \quad 1 \leq i \leq k \end{aligned} \quad (E')$$

The hash-result is $H = H_k$, equal to the "sum" of the squares modulo n of all the blocks of the message.

Of course, this scheme is also imperfect because the enemy can permute the B_i , insert a spurious B' an even number of times or insert

the null block any number of times without modifying H. He may also easily find blocks $(B'_1, \dots, B'_q)$ such that H is the "sum" of their real squares (e.g. $40 = 36(+)9(+)4(+)1$).

As for the previous scheme, we can reply to these attacks by imposing to the blocks some redundancy. But here, the redundancy will have to depend on the rank of each block, in order to avoid the first two mentioned attacks. For example, this rank can be inserted somewhere in the block.

The rest of the paper is dedicated to the type 1 schemes (equations (E)).

II THE DP-SCHEME DESCRIPTION

The Davies-Price scheme is a type 1 scheme with redundancy. The exact description is:

- choose a public 512 bit integer n;
- divide the message M into 448 bit blocks $B_1, B_2, \dots, B_k$ (that is equivalent to imposing 64 bits of redundancy at the head of each B_i , these bits being equal to 0);
- apply the equations (E).

The hope is that the probability of a successful CLB attack be equal to the probability that a random number satisfy the required redundancy (i.e. 2^{-64} , negligible). We now show that this hope must be abandoned.

III AN ATTACK ON THE DP SCHEME

Let us call "valid" a 512 bit block with the 64 m.s.b. (most significant bits) equal to zero. A successful attack consists in replacing the valid blocks $(B_1, \dots, B_k)$ with other valid blocks $(B'_1, \dots, B'_q)$ leading to the same hash-result.

For simplicity, we only deal here with the case: $k=q=2$ and $I=0$ (the general case is considered in section V); hence $H_1 = B_1^2 \pmod{n}$ and $H'_1 = B'^2_1 \pmod{n}$.

First, let us show that the trivial choice $B'_1 = B_1 + 1$ leads very often to the success. In that case, $H'_1 = H_1 + 2B_1 + 1 \pmod{n}$. If h denotes the integer composed of the 448 l.s.b. of H_1 , then the probability for H_1 and H'_1 to have the same 64 m.s.b is equal to the probability for $(h + 2B_1 + 1)$ to be smaller than 2^{448} , i.e. approximatively $1/4$, if we consider h and B_1 as uniformly distributed in the interval $[0, 2^{448}]$. If this event occurs, and since B_2 is valid, it

follows that $B'_2 = H_1 (+) H'_1 (+) B_2$ is valid too and the CLB attack succeeds, without any subtlety!

In a general manner, the enemy must find a block B'_1 , close to B_1 (here, "close" means "having the same 64 m.s.b.") and such that the squares modulo n of B_1 and B'_1 are close too. If he finds it, then B'_1 as long as $B'_2 = H_1 (+) H'_1 (+) B_2$ will be valid and the CLB attack will succeed again.

In order to find such a B'_1 , let d denote the small rational integer such that $B'_1 = B_1 + d$ (where $+$ denotes natural addition). H'_1 is equal to $(B_1^2 + 2B_1d + d^2) \bmod n$ and will be close to H_1 if (but not only if) $2B_1d \bmod n$ and $d^2 \bmod n$ are both small (here, "small" means approximatively "smaller than 2^{448} "). And $d^2 \bmod n$ will be small if (but not only if!) d^2 is itself small, i.e. d is a "very small" integer (approximatively smaller than 2^{224}).

So, the problem is now reduced to finding a very small integer d such that $2B_1d \bmod n$ is small. At this stage, we need a little arithmetic.

IV SOME ELEMENTARY ARITHMETIC

Let n be a positive (>0) integer and b, X, Y three positive integers smaller than n . Does the system

$$\begin{cases} bx = y \pmod{n} \\ \text{abs}(x) < X \\ \text{abs}(y) < Y \end{cases} \quad (S)$$

admit solutions in $\mathbb{Z}^2 - (0,0)$?

The answer is "certainly yes" if $XY > n$. The essential of the proof comes from [12], and uses the "pigeon-hole principle". The pigeons are the pairs (x, y) where $1 \leq x \leq X$ and $0 \leq y \leq Y-1$, and the holes are the integers $bx - y \pmod{n}$ calculated with these pairs. There are XY pigeons and at most n holes. Consequently, if $XY > n$, there are necessarily two pigeons in the same hole, i.e. two different pairs (x, y) and (x', y') such that $bx - y = bx' - y' \pmod{n}$. So, denoting $x' - x$ by x'' and $y' - y$ by y'' , we have found a pair (x'', y'') different from $(0,0)$ such that $bx'' = y'' \pmod{n}$. If we remark that x'' cannot be zero, it is clear that $0 < \text{abs}(x'') < X$ and $\text{abs}(y'') < Y$, hence that we found a solution to our system (and even two, since the opposite of a solution is another solution too).

Now, what about the effective construction of such solutions ? Of course, the above proof is of no help if n is 512 bit long! We should rather remark that finding a small y such that $bx = y \pmod{n}$ comes to

finding a good approximation of the fraction b/n . Consequently, the continued fractions theory should be useful here (see an introduction to the subject e.g. in [13]).

In a few words, it is possible with this theory to find "best" rational approximations of any real number x . These approximations, denoted b_i/n_i , are called convergents of x . For example, the first convergents of the square root of 2 are $1/1$, $3/2$, $7/5$, $17/12$, $41/29$, etc. This sequence is illimited except of course when x is itself a rational (say b/n). For example, if $x=72/91$, the convergents are $1/1$, $3/4$, $4/5$, $15/19$, $19/24$ and $72/91$. In the rational case, the convergents are very easy to calculate, because provided by the Extended Euclidean Algorithm, which is now described.

As all the pupils know, the basic Euclidian Algorithm is an effective way to calculate the gcd (greatest common divisor) of two positive integers (n,b) and works as follows:

Let $r_0=n$, $r_1=b$ and $r_{i+1} = r_{i-1} - q_i r_i$ ($i \geq 1$) where q_i is the quotient in the euclidean division of r_{i-1} by r_i (hence r_{i+1} is the remainder of this division). The sequence of remainders r_i is strictly decreasing and the algorithm stops when a remainder (say r_{k+1}) comes to be equal to zero. Then $\gcd(n,b)$ is equal to r_k , the last non-zero remainder.

Now, let us define two supplementary sequences (l_i) and (m_i) in the following way: $l_0=1$ and $l_1=0$; $m_1=0$ and $m_1=1$; $l_{i+1} = l_{i-1} - q_i l_i$ and $m_{i+1} = m_{i-1} - q_i m_i$ for $i \geq 1$. As the recurrent equations are the same for the three sequences (r_i) , (l_i) and (m_i) , it is easy to see that for all the i : $l_i n + m_i b = r_i$. Hence, for all the i , $m_i b = r_i \pmod{n}$. This algorithm is called Extended Euclidean Algorithm (EEA), and it can be shown that the absolute values of the coefficients l_i and m_i are equal to, respectively, the numerator b_i and the denominator n_i of the convergents of b/n (in fact, $b_i/n_i = -l_i/m_i$ for all the i).

Moreover, the continued fractions theory states that the absolute value of the difference between a number and one of its convergents b_i/n_i is smaller than $(1/n_i)^2$. So:

$$\begin{aligned} \text{abs}(b/n - b_i/n_i) &< (1/n_i)^2 \iff \text{abs}(b/n + l_i/m_i) < (1/m_i)^2 \\ &\iff \text{abs}(r_i/m_i n) < (1/m_i)^2 \iff \text{abs}(m_i r_i) < n \end{aligned}$$

Finally, we see that the EEA provides some pairs (m_i, r_i) such that:

$$\begin{cases} b m_i = r_i \pmod{n} \\ \text{abs}(m_i r_i) < n \end{cases}$$

If we come back now to the initial system (S) stated at the top of this section, it appears that the EEA provides some pairs (x,y) such that $bx = y \pmod n$ and $\text{abs}(xy) < n$ but not necessarily verifying $\text{abs}(x) < X$ and $\text{abs}(y) < Y$. However, if the product XY is not too close to n , the hope is great that at least one of these pairs be a solution.

V THE DP-SCHEME ATTACK ALGORITHM

Now, we can state our attack against the DP-scheme precisely. Recall that we presently suppose that the enemy wants to replace a message M composed of two blocks B_1 and B_2 with another message composed of two blocks B'_1 and B'_2 , without modifying the hash-result H , calculated from the message M and an initialization vector equal to zero. We stated in section III that the enemy has a good chance of success if he finds a very small integer d such that $2B_1d \pmod n$ is small (the terms "small" and "very small" being vaguely defined, because they depend on B_1). So his problem is a particular instance (S_1) of the system (S) stated in the section IV with $b=2B_1$, $x=d$, $X=2^{224}$, $Y=2^{448}$ (and n a 512 bit long integer). As XY is higher than n , we now that (S_1) has solutions. In addition, XY being much higher than n , we are almost sure that some of them can be found with the EEA applied to b and n .

So a simple algorithm to attack the DP scheme is:

- (1): let $b = 2B_1 \pmod n$ and $H_1 = B_1^2 \pmod n$
- (2): apply the EEA to (n,b) ; the i -th step is $l_i n + m_i b = r_i$
- (3): at each step of (2), let $B'_1 = B_1 + m_i$; $H'_1 = B_1'^2 \pmod n$;
 $B'_2 = H_1 (+) H'_1 (+) B_2$
- (4): if B'_1 and B'_2 are valid then output (B'_1, B'_2)

Such an algorithm, applied to some (n, B_1, B_2) picked at random, made us able to find 121 fraudulent pairs (B'_1, B'_2) . It is enough to prove the weakness of the DP scheme. But in reality, the number of pairs which could be calculated is far greater, because linear combinations of solutions provided by the EEA are very good candidates to be solutions too, as explained in the following paragraph. Consequently, because the product XY is much higher than n , the number of solutions is finally astronomic, at least about 2^{150} !

Suppose for example that the EEA provided (x_1, y_1) and (x_2, y_2) satisfying the system (S_1) with $x_1 \# 2^{142}$, $y_1 \# 2^{370}$ and $x_2 \# 2^{146}$, $y_2 \# 2^{366}$ (a very plausible situation). Then all the pairs $(u_1 x_1 + u_2 x_2, u_1 y_1 + u_2 y_2)$ with $\text{abs}(u_1)$ and $\text{abs}(u_2) < 2^{77}$ satisfy (S_1) and therefore probably

provide 2^{146} fraudulent pairs (B'_1, B'_2) to the enemy, who did not request so many!

Now, the attack has been described only with an initialization vector equal to zero. If I is supposed to be any natural integer smaller than n , the algorithm is modified as follows:

```
(0,5): let  $C_1 = I (+) B_1$ 
(1)  : let  $b = 2C_1 \pmod n$  and  $H_1 = C_1^2 \pmod n$ 
(2)  : as above
(3)  : at each step, let  $C'_1 = C_1 + m_i$ ;  $H'_1 = C'_1{}^2 \pmod n$ ;
       $B'_2 = H_1 (+) H'_1 (+) B_2$ 
(3,5): let  $B'_1 = I (+) C'_1$ 
(4)  : as above.
```

The trials that we made show that this algorithm is in average as effective as the previous one.

At last, it is easy to see that not only the first two blocks of a message M can be modified in an undetected manner. For example, the blocks B_i and B_{i+1} can be successfully altered by replacing

$(B_1, C_1, H_1, B_2, B'_1, C'_1, H'_1, B'_2 \text{ and } I)$ with, respectively,
 $(B_i, C_i, H_i, B_{i+1}, B'_i, C'_i, H'_i, B'_{i+1} \text{ and } H_{i-1})$ in the second algorithm.

VI IMPROVEMENTS OF THE DP-SCHEME

Many variations to the DP-scheme may be envisioned in order to make it stronger.

An exponent higher than 2 can be chosen. But it has a direct consequence on the performance (execution time) of the function. For this reason, the better choice seems to be a power of 2.

The redundancy length can be increased. If u redundancy bits are required (instead of 64), the system (S_1) becomes the system (S_2) with $X=2^v$ and $Y=2^{2v}$, where $v=(512-u)/2$. If XY becomes close to n , we know that the success of our attack is no more guaranteed. This corresponds to $3v=512$, i.e. about $u=v=170$. This suggests that a minimal number of redundancy bits be 256 (in order to preserve a safety margin).

The redundancy can be placed elsewhere. For example, the additional bits could be placed on the right of the block instead of on the left. But this suggestion is not satisfactory, because the attack described in section III can be adapted to such a case, as shown now:

Let us call "valid" a 512 bit block with the 64 l.s.b. (least significant bits) equal to zero. As in III, and for simplification, we only show here that a CLB attack is possible with $k=q=2$ and $I=0$. So, B_1

and B_2 being valid, we have to find valid blocks B'_1 and B'_2 leading to the same hash-result. The problem is hence to find a valid B'_1 such that H_1 and H'_1 have the same 64 l.s.b. . If we succeed, then $B'_2 = H_1 (+) H'_1 (+) B_2$ will be valid and the CLB attack will work.

In order to find such a B'_1 , let d denote the small integer such that $B'_1 = B_1 + 2^{64}d$. H'_1 is equal to $[B_1^2 + 2^{64} (2B_1d + 2^{64}d^2)] \bmod n$ and will be as wanted if (but not only if) $2B_1d$ and $2^{64}d^2 \bmod n$ are both small (approximatively smaller than 2^{448}). Now, $2^{64}d^2$ will be small if (but not only if) d is itself particularly small (approximatively smaller than 2^{192}).

So the problem is now reduced to solving a particular instance (S_2) of the problem (S) stated in the section IV, with $b=2B_1$, $X=2^{192}$, $Y=2^{448}$ (and n a 512 bit integer). As $XY \gg n$, we know that there are a lot of solutions, and the enemy will discover a lot of false messages. Let us just remark that the attack stops to be effective if the number of redundancy bits is 128 or more (instead of 170 for the original DP scheme).

So, if we think to move the additional bits, a better idea consists in placing them ...in the middle of the block! In that way, the attack seems to be no more effective. A still better idea consists in dispersing them in the block (for example: a few on the left, a few on the right, and the rest in the middle).

By combining two of the above suggestions (explicitly: 256 redundancy bits, which are regularly dispersed in the blocks), one obtains a hash-function proposed by the OSIS-TeleTrust project [10] for standardization as an annex in CCITT/SG VII [14] and ISO TC97/SC21 [15]. Again, it is a type 1 scheme with a number of redundancy bits equal to the half of the modulus size, these bits being equal to 1, and placed four at the time.

Precisely, and assuming a modulus size equal to 512, the message M is first divided into 256 bit blocks B_i . Then, each octet of B_i is split in half, each of the halves being headed by binary ones (moreover, the first octet of block B_i contains the low order octet of the length of the message and if the last block is incomplete, it is padded with "1"s).

Of course, such a hash-function is certainly robust enough to resist to any variation of our attack. But it is perhaps still possible to improve it (at least theoretically). For example, redundancy quartets could be dependent on the significant quartets they are associated to, instead of being fixed. Such a choice would make the enemy's work still more difficult, if it is possible.

VII OTHER MODULO-N SCHEMES

It has been told in section II that the type 1 schemes are viable only if some redundancy is inserted in the blocks. The sections which followed were only a discussion about the nature of this redundancy. But, whatever the choice may be, introduction of redundancy always implies an increase of the execution time. Particularly if there are as many redundancy bits as significant ones.

So, a question is: can we design secure schemes which do not impose redundancy to the blocks of the messages to be hashed? It seems as if the answer was not so easy. For example, in [16], the following scheme is proposed, very close to a type 1 scheme:

$$\begin{aligned} H_0 &= I \\ H_i &= ((H_{i-1} (+) B_i)^2 \bmod n) (+) B_i \quad (1 \leq i \leq k) \end{aligned}$$

Unfortunately, there is (at least) one weakness in this proposal, due to the fact that the operation "squaring modulo" has fixed points (0, 1 and perhaps others). If P is such a point, it is clear that we can insert the block $H_i (+) P$ between B_i and B_{i+1} without modifying the hash-result.

Another possibility, quite different, consists in squaring the blocks before adding them to the current hash-result:

$$\begin{aligned} H_0 &= I \\ H_i &= [H_{i-1} (+) B_i^2 \bmod n]^2 \bmod n \quad (1 \leq i \leq k) \end{aligned}$$

But there is no gain in execution time, compared to the TeleTrust scheme.

VIII CONCLUSION

This paper dealt with hash-functions using modulo-n operations. Two types have been presented, which are viable only if redundancy is inserted in the blocks of the message to be hashed. A proposal of the first type, by D.W. Davies and W.L. Price, was considered. It has been shown that an enemy can use the Extended Euclidean Algorithm to find more than about 2^{150} substitutes to a message of length superior to 448 bits, without modifying the hash-result. Some variations of this proposal have been analysed, in particular the OSIS-TeleTrust scheme which seems very solid (but perhaps perfectible). Other schemes, which do not use redundancy, have been also presented and criticized.

REFERENCES

- [1] Pinkas, D., "The need for a Standardized Compression Algorithm for Digital Signatures", presented at Eurocrypt 86.
- [2] Winternitz, R.S., "Producing a One-Way Hash Function from DES", *Advances in Cryptology, Proceedings of Crypto 83*, ed. D.Chaum, Plenum Press, New York, 1984, pp.203-207.
- [3] Denning, D.E., "Digital Signatures with RSA and Other Public-Key Cryptosystems", *CACM*, Vol. 27, N°4, April 84, pp. 388-392.
- [4] Rabin, M., "Digital Signatures", *Foundations of Secure Computation*, Academic Press, New York, 1978.
- [5] Davies, D.W., "Applying the RSA Signature to Electronic Mail", *Computer*, 1983.
- [6] Akl, S.G., "On the Security of Compressed Encodings", *Advances in Cryptology, Proceedings of Crypto 83*, ed. D.Chaum, Plenum Press, New York, 1984, pp.209-230.
- [7] Coppersmith, D., "Another Birthday Attack", *Advances in Cryptology, Crypto '85 Proceedings, Lecture Notes in Computer Science*, Vol.218, Springer-Verlag, Berlin, 1986, pp. 14-17.
- [8] Davies, D.W. and Price, W.L., "Digital Signatures, an update", *Proceedings of the International Conference on Computer Communications*, Sydney, October 1984, pp. 845-849.
- [9] "Modes of Operations and Hash-functions", *ISO/TC97/SC20/WG2/N31*, July 1985.
- [10] "OSIS Security Aspects", *OSIS European Working Group WG1, Final report*, October 1985.
- [11] Jueneman, R.R., Matyas, S.M. and Meyer, C.H., "Message Authentication with Manipulation Detection Codes", *Proceedings of Security & Privacy*, 1983, *IEEE Catalog N° 83CH1882-0*, pp. 33-54.
- [12] De Jonge, W. and Chaum, D., "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.
- [13] Riesel, H., "Prime numbers and computer methods for factorization", *Progress in Mathematics*, vol.57, 1985, Birkhäuser Ed., pp.300-317.
- [14] "The Directory-Authentication Framework", *CCITT/SGVII/Q35 Draft Recommendation X 509*, June 1987.
- [15] "The Directory-Authentication Framework", *ISO/TC97/SC21 DP 9594/8*, June 1987.
- [16] Pailles, J.C. and Girault, M., "The security processor CRIPT", *Preprints of the 4th IFIP'SEC, Monte-Carlo*, December 2-4, 1986, pp.127-139.