ATTACKS ON SOME RSA SIGNATURES

Wiebren de Jonge[1] and David Chaum [2]

[1]Department of Mathematics and Computer Science
Vrije Universiteit P.O. Box 7161
1007 MC Amsterdam The Netherlands

[2]Centre for Mathematics and Computer Science
Kruislaan 413
1098 SJ Amsterdam The Netherlands

ABSTRACT

Two simple redundancy schemes are shown to be inadequate in securing RSA signatures against attacks based on multiplicative properties. The schemes generalize the requirement that each valid message starts or ends with a fixed number of zero bits. Even though only messages with proper redundancy are signed, forgers are able to construct signatures on messages of their choice.

1. INTRODUCTION

The basic notions of redundancy in signatures and multiplicative attacks are introduced for completeness in this introductory section, along with an example which is used in subsequent sections. Next the two redundancy schemes are presented briefly. An algorithm is then described and used to construct attacks on the two schemes. Finally, a second kind of attack is presented which also compromises the two redundancy schemes.

1.1. THE NEED FOR REDUNDANCY

RSA used in its raw form does not protect against a forger choosing an integer S_c with $0 < S_c < n_A$, and computing $M_c = (S_c)^{e_A} \bmod n_A$ from it, where n_A and e_A are As public modulus and exponent in an RSA system. Subsequently, the forger could claim that S_c is the signature on M_c. Since exponentiation modulo n acts as a kind of one-way function when $\phi(n)$ is

H.C. Williams (Ed.): Advances in Cryptology - CRYPTO '85, LNCS 218, pp. 18-27, 1986.

unknown, this *chosen signature attack* can be used for finding signatures on "random" (i.e., unpredictable) messages only. Thus, it may be said that only the signer can form signatures on chosen messages, but anybody can determine which message corresponds to a chosen signature.

To prevent these unpredictable messages from having a reasonable chance of being accepted, *redundancy* will be required in signed messages. Hence, a distinction will be made between *messages* and *valid messages*: all numbers M with $0 \leq M < n$ are messages, but only a very small fraction of these will be valid messages. For instance, if 100 bits of redundancy are used, a chosen signature will have only a chance of 2^{-100} of corresponding to a valid message. Thus, finding a *false signature* (i.e., a signature on a valid message not actually signed by A) will cost 2^{99} trials on the average, which makes it infeasible to successfully guess a signature.

Some work has been based on the assumption that the signer would sign anything except some desired messages. ([DeMillo & Merritt 82] and [Denning 83] independently generalized and extended [Davida 82].) Under these assumptions, attackers were able to obtain signatures on desired messages simply by combining signatures on apparently unrelated messages. The seemingly more realistic and practical model assumed here, that the signer is only willing to sign valid messages, makes attacks more difficult—though not impossible—as will be shown.

1.2. MULTIPLICATIVE ATTACKS

Preventing chosen signature attacks not only requires a sufficient *quantity* of redundancy in valid messages; it also necessitates that the *nature* of the redundancy is appropriate, since RSA signatures are *multiplicative*.

For example, suppose that B can construct three valid messages M_1, M_2 and M_3 such that $M_3 = (M_1 \cdot M_2) \bmod n_A$. Then, if B succeeds in getting M_1 and M_2 signed by A, B can form the product (modulo n_A) of these signatures to get a false signature on M_3, denoted $S_A(M)$, since

$$S_A(M_3) = (M_1 \cdot M_2)^{d_A} \bmod n_A$$

$$= ((M_1^{d_A} \bmod n_A) \cdot (M_2^{d_A} \bmod n_A)) \bmod n_A$$

$$= (S_A(M_1) \cdot S_A(M_2)) \bmod n_A.$$

B can also use the inverse M^{-1} or the opposite $-M$ of a message M, assuming the corresponding signed version is known, as a factor in a product forming a new message, since $S_A(M^{-1} \bmod n_A) = (S_A(M))^{-1} \bmod n_A$ and $S_A(-M) = -S_A(M)$. (Notice that d_A is odd.)

Thus, if B knows A's signature on one or more valid messages M_i, B can easily forge signatures for valid messages that B can rewrite as a product of message(s) M_i, their opposite(s) $-M_i$,

or their inverse(s) M_i^{-1} (all in modulo n_A arithmetic). Note also that a message and/or its opposite and/or its inverse may occur in a product more than once. Therefore, the redundancy should make it infeasible to find such valid messages.

1.3. EXAMPLE CRYPTOSYSTEM

Rivest, Shamir and Adleman recommended that n be about 200 decimal digits, which amounts to about 664 bits [RSA 78]. We will use a particular example of an RSA system for illustrative purposes, in which n is 800 bits, thereby maintaining an ample margin of safety with respect to known factorization techniques for appropriate moduli. The amount of redundancy used in the examples will be 200 bits. One reason for this choice of amount of redundancy is to provide for sufficient protection against a chosen signature attack. Another is for efficiency, since one does not want to expand the messages to be signed too much, say, not more than one third. Since, for our choice of n, RSA limits signed messages to 800 bits, the redundancy should amount to at most 200 bits. As a consequence, only a fraction of 2^{-200} of all 800-bit messages are valid, and thus the original message to be signed, called the *actual* message, may comprise 600 bits.

An important assumption is that every bit pattern of 600 bits represents a meaningful message; the only redundancy is that explicitly included in the remaining 200 bits.

2. THE TWO REDUNDANCY SCHEMES

In the first redundancy scheme the redundant bits are combined with the actual message by multiplying that message with an agreed on constant w. That is, all messages M for which $M \bmod w = 0$ are defined to be valid. The actual message present in such a valid message is $m = M \operatorname{div} w$. For $w = 2^{200}$, this means that each valid message ends up with 200 zero bits.

In analogy to the special case where w is a power of two, the general scheme will be called the *right-padded redundancy* scheme. Figure 1 shows how the right-padded redundancy scheme spreads the valid messages over the interval $[0,n]$ for $n = 91$ and $w = 6$.

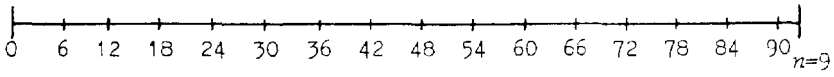

Fig. 1. The valid messages in case of right-padded redundancy for $n = 91$ and $w = 6$.
The valid messages are 0, 6, 12, ..., 90.
The actual messages are 0, 1, 2, ..., 15.

The counterpart of spreading valid messages over the interval $[0,n]$ is to concentrate them in some interval $[l,u]$ of appropriate size. The actual message contained in a valid message M then is $m = M - l$. For $l = 0$ and $u = 2^{600} - 1$, each valid message starts with a sequence of 200 zero bits. Accordingly, we call the general case of this scheme *left-padded redundancy*. Figure 2 illustrates the left-padded redundancy scheme for $n = 91$, $l = 19$ and $u = 34$.

3. A FIRST KIND OF ATTACK

Before treating our attacks in detail, we mention briefly an algorithm that will be used heavily.

3.1. A VARIATION OF EUCLID'S ALGORITHM

The algorithm finds, for a given x with $0 \leq x < n$, the smallest positive value c such that $(cx) \bmod n$ is less than some given threshold value t. It is very similar to Euclid's algorithm for computing the greatest common divisor. Indeed, Euclid's algorithm can be used to compute an increasing sequence of values c for which the corresponding values $(cx) \bmod n$ form a decreasing sequence. The only important difference is that processing with our algorithm stops as soon as a value below the given threshold is reached. Since Euclid's algorithm has a worst case (and average case) complexity of $O(\log n)$ [Knuth 69], our algorithm will certainly be fast (enough) too.

For our purposes, it is often important that the value found for c is reasonably small. Although our algorithm can be used to find for any given values x and t the smallest c for which $(cx) \bmod n \leq t$, there is no guarantee that c itself is smaller than some other threshold value. However, it is easy to show that there always exists some c with $0 < |c| \leq n / t$ for which $0 \leq (cx) \bmod n \leq t$.

Consider the integers /a/ and /b/ such that $(ax - b) \bmod n$ with $0 < a \leq \lceil n / t \rceil$ and $0 \leq b \leq t$. Since there are more than n different pairs (a,b), there exist two different pairs, say, (a_1,b_1) and (a_2,b_2), for which $(a_1x - b_1) \bmod n = (a_2x - b_2) \bmod n$. Since x usually will be co-prime with n (if not, one could factor n), we know that both $a_1 \neq a_2$ and $b_1 \neq b_2$. Therefore, we may safely assume that $b_1 > b_2$. Thus, for $c = (a_1 - a_2)$ it is true that $0 < |c| \leq n / t$ and

Fig. 2. The valid messages with left-padded redundancy for $n = 91$, $l = 19$ and $u = 34$. The valid messages are 19, 20, 21, ..., 34. The actual messages are 0, 1, 2, ..., 15.

$0 < (cx) \bmod n = (b_1 - b_2) < t$.

Since our algorithm searches for the smallest positive value c for which $(cx) \bmod n \leq t$, the value found for c may be larger than n / t. If so, the above shows that there exists some c, $0 < c \leq n / t$, such that $0 \leq (-cx) \bmod n \leq t$. This c can be found by applying our algorithm to $(-x) \bmod n$.

3.3. ATTACKING RIGHT-PADDED REDUNDANCY

If RSA is used in combination with the right-padded redundancy scheme, one attack proceeds as follows. First, choose an actual message m_1 (i.e., $m_1 < n_A \text{ div } w$) on which A's signature is desired. An attack, such as this, allowing a signature to be constructed for a chosen actual message will be called a *chosen message attack*. Now, $M_1 = m_1 w$ is a valid message, since $M_1 < n_A$ and $M_1 \bmod w = 0$. Next, compute $x = (m_1 w)^{-1} \bmod n_A$. If $w \leq n^{1/2}$, i.e., if the redundancy takes up less than half of the bits in a valid message, our algorithm of Figure 3 can be used to find a number $0 < c \leq n_A \text{ div } w$ such that $(cx) \bmod n_A \leq n_A \text{ div } w$ or $(-cx) \bmod n_A \leq n_A \text{ div } w$. Thus, one can find two actual messages m_2 and m_3 such that $m_2 = (m_3 x) \bmod n_A$ or $m_2 = (-m_3 x) \bmod n_A$.

If one succeeds in getting A's signature on m_2 and m_3 (i.e., $S_A(M_2)$ and $S_A(M_3)$), one can compute A's signature on m_1 by multiplying $S_A(M_3)$ with the inverse of $S_A(M_2)$ and taking the opposite in the case we used $-x$. Naturally, all arithmetic is done modulo n_A. In case we used just x, this works, because

$$S_A(M_1) = S_A((x^{-1} \cdot (m_3 w) \cdot (m_3 w)^{-1}) \bmod n_A)$$

$$= (\, S_A(m_3 w) \cdot S_A((m_3 x w)^{-1}) \,) \bmod n_A$$

$$= (\, S_A(M_3) \cdot (S_A(M_2))^{-1} \,) \bmod n_A;$$

in case $-x$, we have

$$S_A(M_1) = (\, S_A(M_3) \cdot (-S_A(M_2))^{-1} \,) \bmod n_A.$$

Of course, the attack makes sense only when $m_1 \neq m_2$ and $m_1 \neq m_3$. But if the found m_2 or m_3 happens to be equal to m_1, one simply searches for another value of c such that $(cx) \bmod n \leq t$ or $(-cx) \bmod n \leq t$. For example, one tries the next minimal value of $(cx) \bmod n$ or $(-cx) \bmod n$, respectively.

3.4. ATTACKING LEFT-PADDED REDUNDANCY

RSA's multiplicative properties are also useful for attacking the RSA signature system when left-padded redundancy is used. Recall that this scheme defines valid messages as those in the interval $[l,u]$.

As a first step it will be shown why, in the general case, l should be larger than $u^{1/2}$, and thus in our example should be larger than 2^{300}. If l would be smaller than $u^{1/2}$, then any two valid messages M_i $(i=1,2)$ out of $[l,u^{1/2}]$ have a product, say M_3, which lies in the interval $[l,u]$. This makes a multiplicative attack far too easy. Thus, the left-padded redundancy scheme should certainly not be used with $l=0$; i.e., just requiring each valid message to start with a certain number of zero-bits immediately appears to be unsuitable.

For $l>u^{1/2}$ there is a chosen message attack. Suppose that M is the valid message on which a false signature is desired. First, the attack will be shown for $M \leqslant u-l$, and later it will be extended for the more likely case that $M > u-l$.

Due to the large number of wrap-arounds, the number $(l \cdot M)$ mod n may be positioned anywhere in the interval $[0,n]$. Therefore, the chance that $l \cdot M$ mod n lies in the interval $[l,u]$ is negligibly small. (About 2^{-200} in the example.) However, it is easy to find a positive integer i such that $(l+i)M$ mod n is in $[l,u]$.

For example, suppose we have the situation as depicted in Figure 3, where $l \cdot M$ mod n is positioned somewhere to the right of $[l,u]$. Clearly, $(l+1)M$ mod n lies a (relatively small) step of size M to the right of $l \cdot M$ mod n, $(l+2)M$ mod n lies another such step further to the right, and so on. Thus, it is easy to compute i, the number of steps to the right needed to end up in the "next" interval $[l,u]$. Since M is supposed to be less than $u-l$, the step size is small enough to prevent the interval from being missed by jumping too far.

Thus, if l+i happens to be in $[l,u]$, we have found three valid messages M_i $(i=1,2,3)$ with $M_1 = l+i$, $M_2 = M$ and $M_3 = (l+i)M$ mod n for which $M_3 = (M_1 M_2)$ mod n. Thus, a false signature on M can be constructed from the signatures on M_1 and M_3.

To be sure that l+i indeed will be in $[l,u]$, i.e., that $i \leqslant s = u-l$, the step size should be large enough, i.e., $M \geqslant n/s$. Because of our assumption that $M \leqslant s = u-l$, and the interval size s should be larger than n/s. Therefore, this attack works for all chosen messages M with

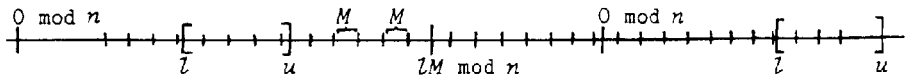

Fig. 3. An illustration of the basic idea of the attack.
Note that this figure is not drawn to scale!

$n/s \leq M \leq s$ if $s \geq n^{1/2}$, i.e., if the redundancy takes up less than half of the bits in a valid message.

If $M > s = u - l$, there is no guarantee that a "walk" to the right with steps of size M will end up in the "next" interval $[l,u]$. For example, if $u - l = 2^{600}$ and if $M \approx 2^{700}$ then the chance to hit the next first interval $[l,u]$ on a walk to the right is only about 2^{-100}.

However, as explained in §3.1, it is easy to find a value c for which $|c| \leq n/s$ and $cM \bmod n \leq s$. Starting with $x = l$ or $x = u$, one can use this new value $cM \bmod n$ as the step size (to the right or to the left) and can compute for which integer i the number $(x + ic)M \bmod n$ will be in the interval $[l,u]$. Since we want $x + ic$ to be a valid message, the product ic should be less than s. Assuming that each number less than s had equal probability of being the chosen step size $cM \bmod n$, the chance that the step size is, for a given p, larger than s/p is $1 - 1/p$. With a step size larger than s/p, i will be less than $(np)/s$. Thus, ic will then be less than $(n^2p)/s^2$. This upper bound on ic should be kept smaller than s, therefore, s should be such that $s^3 > (n^2p)$. In other words, the chance of success is very large roughly when the redundancy is less than one third of the bits in a valid message.

Consider our example with $l = 2^{700}$. The number c for which $cM \bmod n \leq 2^{600}$ or $(-cM) \bmod n \leq 2^{600}$ will be less than 2^{200}. There is a high probability that the new step size, $cM \bmod n$ or $(-cM) \bmod n$, will be larger than, say, 2^{500}. This means that the required number of steps, i, almost certainly will be less than 2^{300}. In our example, ic thus may be expected to be less than 2^{500}. This means that we could have started with almost any x in $[l,u]$.

4. A SECOND STYLE OF ATTACK

Another kind of attack is based on an approach called Multiplying-In-Dividing-Out (MIDO). It is used below to break the same two redundancy schemes.

4.1. RIGHT-PADDED REDUNDANCY AGAIN

Suppose that the actual message m on which a false signature is desired, can be written as the product of two numbers a_1 and a_2. Thus, $M = mw = a_1a_2w < n$. Now choose numbers b_1 and b_2 such that $M_1 = a_1b_1w < n$, $M_2 = a_2b_2w < n$, and $M_3 = b_1b_2w < n$ (e.g., choose any b_1 and b_2 with $b_1 < a_2$ and $b_2 < a_1$). Clearly, the three messages M_1, M_2 and M_3 are all valid, and $M = \dfrac{M_1 M_2}{M_3}$ (hence the name MIDO). Thus, if one succeeds in getting A's signature on the valid messages M_i $(i = 1,2,3)$, one can also construct a false signature on the chosen message M.

One difference with the attack of §3.3 is that this MIDO attack works for any amount of redundancy. On the other hand, this MIDO attack will not work for all chosen messages m, since it may be infeasible or even impossible to factor the integer m. Of course, one could mani-

pulate chosen factors to construct an appropriate actual message m, but this does not change the fact that there is only limited freedom in choosing m.

4.2. LEFT-PADDED REDUNDANCY REVISITED

The following method illustrates how the MIDO approach can be used for attacking left-padded redundancy. It works for all valid messages M that can be written as a product $a_1 a_2$ with $a_1 \geq a_2 \geq 2$, such that $a_1 \neq a_2 + 1$ and either (a) $M - l > a_2$ and $u - M > a_1$ or (b) $M - l > a_1$ and $u - M > a_2$.

In case condition (a) holds, take

$$M_1 = (a_1 - 1)a_2 = M - a_2,$$
$$M_2 = a_1(a_2 + 1) = M + a_1,$$
$$\text{and } M_3 = (a_1 - 1)(a_2 + 1) = M + a_1 - a_2 - 1.$$

Thus, $M = \dfrac{M_1 M_2}{M_3}$, while condition (a) assures that all three messages M_i $(i = 1,2,3)$ are valid. The condition $a_1 \neq a_2 + 1$ assures that $M_3 \neq M$. For the case that condition (b) is true, a_1 and a_2 should be exchanged in the above description. Figure 4 illustrates how M_1, M_2 and M_3 are positioned in $[l,u]$ if condition (a) holds.

Clearly, the chance of success with this method depends on the size and placement of the interval $[l,u]$, and thus on the amount of redundancy. Furthermore, this method does not work for all chosen messages. However, it is easy to adapt this attack to work for almost any chosen valid message M. The only restriction will be that M should not be chosen too close to l or u. Such a restriction is not very severe, since, for example, $u - M$ and $M - l$ are both larger than $2^{-10}(u - l)$ for 99.8 percent of all valid messages.

Once M is chosen, one searches for "factors" a_1 and a_2 such that $M = (a_1 a_2) \bmod n$. (The important difference with the attack above is the addition of "mod n".) This can easily be accomplished by freely choosing one factor, say a_1, and then computing the other factor, a_2, as $(a_1^{-1} M) \bmod n$. Having fixed a_1 and a_2 one computes the numbers c_1 and c_2 with $|c_1| \leq 2n / (u - M)$ and $|c_2| \leq 2n / (M - l)$ such that $(c_1 a_1) \bmod n < (u - M)/2$ and $(c_2 a_2) \bmod n < (M - l)/2$. In the following, we only treat the case that both c_1 and c_2 are positive. Take

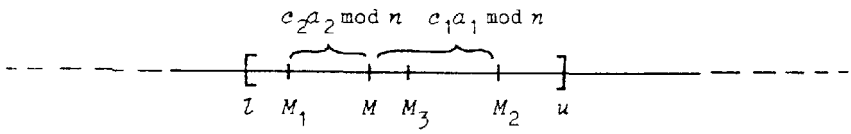

Fig. 4.

$$M_1 = a_2(a_1 - c_2) \bmod n = M - (c_2 a_2 \bmod n)$$
$$M_2 = a_1(a_2 + c_1) \bmod n = M + (c_1 a_1 \bmod n)$$
$$\text{and } M_3 = (a_1 - c_2)(a_2 + c_1) \bmod n = (M + c_1 a_1 - c_2 a_2 - c_1 c_2) \bmod n.$$

Thus, M_1 and M_2 are valid messages. Define z to be the minimum of $u - M$ and $M - l$. M_3 is also a valid message if $c_1 c_2$ is appropriately small, i.e., if $c_1 c_2$ is less than $z / 2$. (See Figure 5 for an illustration.) Since $c_1 c_2$ is known to be less than $4n^2 / z^2$, this product is certainly smaller than $z / 2$ if $8n^2 < z^3$. Thus, the attack works essentially when the redundancy amounts to less than one third of a valid message.

In our example, both $u - M$ and $M - l$ are numbers of almost 600 bits. Therefore, c_1 and c_2 may be expected to be numbers of a good 200 bits. Thus, their product may be estimated to be a number of something like 400 bits, which usually will be negligibly small compared to M's distance to l and u. As a consequence, the chance that M_3 is not in the interval $[l,u]$ is negligibly small.

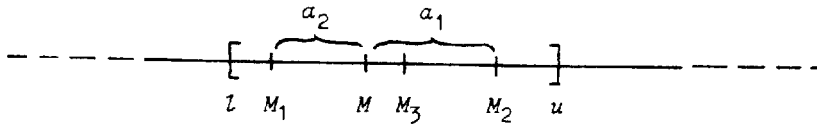

Fig. 5.

CONCLUDING REMARKS

The attacks presented use signatures obtained on messages having a redundancy property that are chosen to allow derivation of false signatures on other messages also having the redundancy property. The attacks are quite powerful, since they allow the derived message to be chosen freely or almost freely.

One obvious way to protect against attacks such as those shown here in practice, which has been known in the "folklore" of cryptography for some time, is to apply some sort of one-way function to actual messages before signing them. This approach can be quite practical for long messages. But for short messages, it may have the disadvantage of data expansion and may be unnecessarily computationally expensive.

There are of course signature schemes that do not appear to have the kinds of multiplicative structures used in the attacks presented here. These schemes generally have received less attention than RSA and most of those currently unbroken appear more expensive than RSA in various ways. An interesting and potentially attractive variation on RSA signatures, however, came out of this work [de Jonge 85].

Multiplicative properties of RSA and its variants should not necessarily be regarded as undesirable shortcomings to be avoided in improved systems, however, since they allow various

powerful and often desirable functionality, such as blind signatures [Chaum 85]. Motivation for embarking on this line of inquiry in fact came from consideration of the needs for secure blind signature systems. In such systems, any message may be signed; only messages with the redundancy property are accepted; and the primary security requirement, called conservation of signatures, is that it should not be possible to construct more signatures than are issued. Thus such systems do require redundancy properties robust in the presence of multiplicativity. The simple schemes considered here demonstrate that such redundancy properties must be chosen with care.

ACKNOWLEDGEMENTS

We are grateful to Evert Wattel and Jan-Hendrik Evertse for some stimulating discussions.

REFERENCES

(1) Chaum, D., "Security Without Identification: Transaction Systems to make Big Brother Obsolete," Communications of the ACM, Vol. 22, No. 10, October 1985, pp. 1030-1044.

(2) Davida, G.I., "Chosen Signature Cryptanalysis of the RSA (MIT) Public Key Cryptosystem," Technical Report TR-CS-82-2, University of Wisconsin, Milwaukee WI, October 1982.

(3) de Jonge, W., "Attacks on RSA Signatures and Countermeasures," in Security and Privacy in Information Systems: some technical aspects, Ph.D. Thesis, June 1985.

(4) DeMillo, R.A. and Merritt, M.J., "Chosen Signature Cryptanalysis of Public Key Cryptosystems," Technical Memorandum, School of Information and Computer Science, Georgia Institute of Technology, Atlanta GA, October 25, 1982.

(5) Denning, D.E., "The Many-Time Pad: Theme and Variations" Proceedings of the 1983 Symposium on Security and Privacy, April 25-27, 1983; the relevant part also appeared as "Digital Signatures with RSA and Other Public-Key Cryptosystems," Communications of the ACM, Vol. 27, No. 4, April 1984, pp. 388-392.

(6) Knuth, D.E., The art of computer programming, Volume 2, Seminumerical Algorithms, Addison-Wesley, 1969.

(7) Rivest, R.L., Shamir, A., and Adleman, L., "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Communications of the ACM, Vol. 21, No. 2, February 1978, pp. 120-126.