A Chosen Messages Attack on the ISO/IEC 9796–1 Signature Scheme

François Grieu

Innovatron, 1 rue Danton, 75006 Paris, France
fgrieu@innovatron.fr

Abstract. We introduce an attack against the ISO/IEC 9796–1 digital signature scheme using redundancy, taking advantage of the multiplicative property of the RSA and Rabin cryptosystems. The forged signature of 1 message is obtained from the signature of 3 others for any public exponent v. For even v, the modulus is factored from the signature of 4 messages, or just 2 for $v = 2$. The attacker must select the above messages from a particular message subset, which size grows exponentialy with the public modulus bit size. The attack is computationally inexpensive, and works for any modulus of $16z$, $16z \pm 1$, or $16z \pm 2$ bits. This prompts the need to revise ISO/IEC 9796–1, or avoid its use in situations where an adversary could obtain the signature of even a few mostly chosen messages.

1 Introduction

ISO/IEC 9796–1 [1] [2] is an international standard specifying a digital signature scheme giving message recovery, designed primarily for the RSA and Rabin public key cryptosystems.

To sign a message M, it is first transformed by inserting redundant information obtained by simple transformations of individual bytes of M, producing the expanded message $\tilde{M}$; then the private key function $\mathcal{S}$ of the cryptosystem is applied, producing the signature $\ddot{M} = \mathcal{S}(\tilde{M})$.

To verify an alleged signature $\ddot{M}'$, the public key function $\mathcal{V}$ of the cryptosystem is applied, producing an alleged expanded message $\tilde{M}' = \mathcal{V}(\ddot{M}')$; then the alleged message M' is recovered from $\tilde{M}'$ by straightforward extraction, and it is checked $\tilde{M}'$ is what it should be under the signature production process.

ISO/IEC 9796–1 expansion makes it highly improbable that a randomly generated value is an acceptable signature. It meets precise design criterias in order to guard against a variety of other attacks, see [3] and [2].

The recently introduced Coron–Naccache–Stern forgery strategy of [4] is effective on a slightly simplified variant of ISO/IEC 9796–1. Motivated by this breakthrough and unaware of an extension to the full standard in [6], the author made an independent effort to attack ISO/IEC 9796–1 and discovered a new, simple and effective method.

In a nutshell, we efficiently construct many message pairs A, B with $\tilde{A}/\tilde{B}$ equal to a common ratio. Forgery follows from the multiplicative property of the cryptosystem used: $\mathcal{S}(xy) = \mathcal{S}(x)\mathcal{S}(y)$.

2 Definitions

When there is no ambiguity, we assimilate a bit string of fixed length and the integer having this binary representation. Following ISO/IEC 9796–1 unless stated otherwise, we use the notations

$x \parallel y$	Concatenation of bitstrings x and y.
$x \oplus y$	Bitwise exclusive OR of bitstrings x and y.
$[x]_i$	The bitstring of exactly i bits with $[x]_i \equiv x \bmod 2^i$.
$\mathrm{lcm}(x, y)$	Least Common Multiple of x and y.
$\gcd(x, y)$	Greatest Common Divisor of x and y.
v	Public verification exponent.
k	Number of bits in public modulus.
	NB: the standard [1] often use $k_s = k - 1$.
n	Public modulus of k bits, thus with $2^{k-1} \leq n < 2^k$.
p, q	Secret factors of n, with $n = p\,q$.
	if v is odd, $p - 1$ and $q - 1$ are prime with v.
	if v is even, $(p-1)/2$ and $(q-1)/2$ are prime with v,
	$p \equiv 3 \bmod 4$ and $q \equiv p + 4 \bmod 8$.
$(x\|n)$	Jacobi symbol of x with respect to n, used for even v only.
	$(x\|n) = (x\|p)(x\|q) = (x^{(p-1)/2} \bmod p)(x^{(q-1)/2} \bmod q)$.
	For even v the construction of p and q is such that $(2\|n) = -1$.
	$(x\|n)$ can be efficiently computed without knowledge of p and q.
s	Secret signing exponent.
	if v is odd, $s\,v \equiv 1 \bmod \mathrm{lcm}(p-1, q-1)$,
	and as a consequence $(x^s)^v \equiv x \bmod n$ for any x.
	if v is even, $s\,v \equiv 1 \bmod \mathrm{lcm}(p-1, q-1)/2$,
	and as a consequence $(x^s)^v \equiv x \bmod n$ if $(x\|n) = +1$.
z	Number of bytes a message fits in; $z \leq \lfloor (k+2)/16 \rfloor$.
M	Message to sign, which breaks up into the z bytes string
	$m_z \parallel m_{z-1} \parallel .. \parallel m_2 \parallel m_1$
$\tilde{M}$	Message as expanded according to ISO/IEC 9796–1 (see below).
	NB: $\tilde{M}$ is noted Ir in [1] and also Sr in [2].
$\ddot{M}$	The signature of M. NB: $\ddot{M}$ is noted $\Sigma(M)$ in [1] and [2].
	if v is odd, $\ddot{M} = \min(\tilde{M}^s \bmod n,\ n - \tilde{M}^s \bmod n)$
	if v is even, assuming $\gcd(\tilde{M}, n) = 1$ which is highly probable,
	$\ddot{M} = \min\left(\left(\frac{\tilde{M}}{2^{(1-(\tilde{M}\|n))/2}}\right)^s \bmod n,\ n - \left(\frac{\tilde{M}}{2^{(1-(\tilde{M}\|n))/2}}\right)^s \bmod n\right)$

We restrict our attack and our description of ISO/IEC 9796–1 to the cases $k \equiv 0, \pm 1$, or $\pm 2 \bmod 16$, which covers many common choices of moduli, and to messages of $z = \lfloor (k+2)/16 \rfloor$ bytes, the maximum allowed message size. With these restrictions, the construction of the redundant message amounts to the local transformation of each byte m_i of the message by an injection F_i, yielding the redundant message

$$\tilde{M} = F_z(m_z) \parallel F_{z-1}(m_{z-1}) \parallel .. \parallel F_2(m_2) \parallel F_1(m_1)$$

with the injections F_i transforming an individual byte m_i of two 4 bit digits $x \parallel y$ as defined by

$$\begin{array}{l} F_1(x \parallel y) = \Pi(x) \parallel \Pi(y) \parallel y \parallel [6]_4 \\ F_i(x \parallel y) = \Pi(x) \parallel \Pi(y) \parallel x \parallel y \quad \text{for } 1 < i < z \\ F_z(x \parallel y) = [1]_1 \parallel [\Pi(x)]_{k+2 \bmod 16} \parallel \Pi(y) \parallel x \parallel (y \oplus 1) \end{array} \tag{1}$$

and where Π is the permutation on the set of 4 bit nibbles given by

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$\Pi(x)$	E	3	5	8	9	4	2	F	0	D	B	6	7	A	C	1

or as an equivalent definition, if the nibble x consists of the bits $x_4 \parallel x_3 \parallel x_2 \parallel x_1$, $\Pi(x) = (x_4 \oplus x_2 \oplus x_1 \oplus 1) \parallel (x_4 \oplus x_3 \oplus x_1 \oplus 1) \parallel (x_4 \oplus x_3 \oplus x_2 \oplus 1) \parallel (x_3 \oplus x_2 \oplus x_1)$.

3 The new attack

We essentialy select a pair of small positive integers a, b and search all the message pairs A, B that yield redundant messages verifying

$$\frac{\tilde{A}}{\tilde{B}} = \frac{a}{b} \tag{2}$$

3.1 Choice of ratio a/b

Since the ratios a/b and b/a will uncover the same messages, we can restrict our choice of a, b to $a < b$ without missing any message pairs satisfying (2). Similarly, we can restrict ourselves to relatively prime a, b. Since $\tilde{A}$ and $\tilde{B}$ are strings of equal length with a 1 bit on the left, we must have $b < 2a$. We transform equation (2) into $\tilde{B}a = \tilde{A}b$, reduce $\bmod 16$, observe $[\tilde{A}]_4 = [\tilde{B}]_4 = 6$, get $6a \equiv 6b \bmod 16$, so we restrict ourselves to $a \equiv b \bmod 8$.

Thus in the following we restrict our choice for the ratio a/b to relatively prime integers a, b with $9 \leq a < b < 2a$ and $a \equiv b \bmod 8$.

3.2 Making the search manageable

Since the fraction a/b is chosen irreducible, for an hypothetical message pair A, B verifying (2), we can uniquely define the integer W such that

$$\tilde{A} = a\,W \quad \text{and} \quad \tilde{B} = b\,W \tag{3}$$

We break up A, B into z bytes, and, noticing that $9 \leq a < b$ implies $W < 2^{16z}$ for our choice of k, we break up W into z 16 bits strings

$$\begin{array}{l} A = a_z \parallel a_{z-1} \parallel .. \parallel a_2 \parallel a_1 \\ B = b_z \parallel b_{z-1} \parallel .. \parallel b_2 \parallel b_1 \\ W = w_z \parallel w_{z-1} \parallel .. \parallel w_2 \parallel w_1 \end{array}$$

We break up each of the two multiplications appearing in (3) into z multiply and add steps operating on each of the w_i, performed from right to left, with $z-1$ steps generating an overflow to the next step, and a last step producing the remaining left $(k+2 \bmod 16)+13$ bits. We define the *overflows*

$$\begin{array}{ll} \bar{a}_0 = \bar{a}_z = 0 & \bar{b}_0 = \bar{b}_z = 0 \\ \bar{a}_i = \lfloor (a\, w_i + \bar{a}_{i-1})/2^{16} \rfloor & \bar{b}_i = \lfloor (b\, w_i + \bar{b}_{i-1})/2^{16} \rfloor \qquad \text{for } 1 \le i < z \end{array} \tag{4}$$

so we can transform (3) into the equivalent

$$\begin{array}{ll} F_i(a_i) = aw_i + \bar{a}_{i-1} \bmod 2^{16} & F_i(b_i) = bw_i + \bar{b}_{i-1} \bmod 2^{16} \quad \text{for } 1 \le i < z \\ F_i(a_z) = aw_z + \bar{a}_{z-1} & F_z(b_z) = bw_z + \bar{b}_{z-1} \end{array} \tag{5}$$

The search for message pairs A, B satisfying (2) is equivalent to the search of w_i, a_i, b_i, $\bar{a}_i$, $\bar{b}_i$ satisfying (4)(5). This is z smaller problems, linked together by the overflows $\bar{a}_i, \bar{b}_i$.

3.3 Reducing overflows $\bar{a}_i, \bar{b}_i$ to one link l_i

Definition (4) of the overflows $\bar{a}_i, \bar{b}_i$ implies, by induction

$$\bar{a}_i = \left\lfloor \frac{a\,[W]_{16i}}{2^{16i}} \right\rfloor \quad \text{and} \quad \bar{b}_i = \left\lfloor \frac{b\,[W]_{16i}}{2^{16i}} \right\rfloor \quad \text{for } 1 \le i < z \tag{6}$$

Since $0 \le [W]_{16i} < 2^{16i}$ we have

$$0 \le \bar{a}_i < a \quad \text{and} \quad 0 \le \bar{b}_i < b \tag{7}$$

We also observe that $\bar{a}_i$ and $\bar{b}_i$ are roughly in the ratio a/b, more precisely equation (6) implies sucessively

$$a\frac{[W]_{16i}}{2^{16i}} - 1 < \bar{a}_i \le a\frac{[W]_{16i}}{2^{16i}} \quad \text{and} \quad b\frac{[W]_{16i}}{2^{16i}} - 1 < \bar{b}_i \le b\frac{[W]_{16i}}{2^{16i}}$$

$$\frac{\bar{a}_i}{a} \le \frac{[W]_{16i}}{2^{16i}} < \frac{\bar{a}_i + 1}{a} \quad \text{and} \quad \frac{\bar{b}_i}{b} \le \frac{[W]_{16i}}{2^{16i}} < \frac{\bar{b}_i + 1}{b}$$

$$a\frac{\bar{b}_i}{b} - 1 < \bar{a}_i < a\frac{\bar{b}_i + 1}{b} \quad \text{and} \quad b\frac{\bar{a}_i}{a} - 1 < \bar{b}_i < b\frac{\bar{a}_i + 1}{a}$$

so, as consequence of their definition, the $\bar{a}_i$, $\bar{b}_i$ must verify

$$-a < a\bar{b}_i - b\bar{a}_i < b \tag{8}$$

For a given $\bar{b}_i$ with $0 \le \bar{b}_i < b$, one or two $\bar{a}_i$ are solution of (8): $\lfloor a\,\bar{b}_i/b \rfloor$, and $\lfloor a\,\bar{b}_i/b \rfloor + 1$ if and only if $a\,\bar{b}_{i-1} \bmod b > b - a$.

It is handy to group $\bar{a}_i$, $\bar{b}_i$ into a single *link* defined as

$$l_i = \bar{a}_i + \bar{b}_i + 1 \qquad \text{with} \quad 1 \le l_i < a + b \tag{9}$$

so we can rearrange (8) into

$$\bar{a}_i = \left\lfloor \frac{a\,l_i}{a+b} \right\rfloor \quad \text{and} \quad \bar{b}_i = \left\lfloor \frac{b\,l_i}{a+b} \right\rfloor \tag{10}$$

3.4 Turning the problem into a graph traversal

For $1 \le i \le z$ we define the z sets of triples

$$T_i = \{(l_i, w_i, l_{i-1}) \quad \exists (a_i, b_i, \bar{a}_i, \bar{b}_i, \bar{a}_{i-1}, \bar{b}_{i-1}) \quad \text{verifying (4)(5)(7)(9)(10)}\}$$

and we define that $(l_i, w_i, l_{i-1}) \in T_i$ connects to $(l'_j, w'_j, l'_{j-1}) \in T_j$ when $j = i-1$ and $l_{i-1} = l'_j$. Solving (2) is equivalent to finding a connected path from an element of T_z to an element of T_1. If this can be achieved, a suitable W is obtained by concatenating the w_i in the path, and $\tilde{A}$, $\tilde{B}$ follow from (3).

3.5 Building and traversing the graph

The graph can be explored in either direction with about equal ease, we describe the right to left procedure.

Initialy we start with the only link $l_0 = 1$. At step $i = 1$ and growing, for each of the link at the previous step, we vary b_i in range $[0..2^8 - 1]$ and directly compute

$$w_i = \left(F_i(b_i) - \left\lfloor \frac{b\, l_{i-1}}{a+b} \right\rfloor \right) b^{-1} \bmod 2^{16} \tag{11}$$

Using an inverted table of F_i we can determine in one lookup if there exist an a_i such that

$$F_i(a_i) = a\, w_i + \left\lfloor \frac{a\, l_{i-1}}{a+b} \right\rfloor \bmod 2^{16} \tag{12}$$

and in that case we remember the new triple (l_i, w_i, l_{i-1}) with the new link

$$l_i = \left\lfloor \frac{a\, w_i + \left\lfloor \frac{a\, l_{i-1}}{a+b} \right\rfloor}{2^{16}} \right\rfloor + \left\lfloor \frac{b\, w_i + \left\lfloor \frac{b\, l_{i-1}}{a+b} \right\rfloor}{2^{16}} \right\rfloor + 1 \tag{13}$$

We repeat this process until a step has failed to produce any link, or we reach $i = z$ where we need to modify (11)(12)(13) by replacing the term 2^{16} by $2^{(k+2 \bmod 16)+13}$, and reject nodes where $l_z \neq 1$.

If we produce a link in the last step $i = z$, we can obtain a solution to (2) by backtracking any path followed, and the resulting graph covers every solutions.

Exploration for the simplest ratio 9/17 stops on the first step, but 11/19 is more fruitfull. For $k = 256$, and restricting to nodes belonging to a solution, we can draw the graph in figure 1.

Using this graph to produce solutions to (2) is childishly simple: message pairs are obtained by choosing a path between terminals nodes, and collecting the message bytes a_i (resp. b_i) shown above (resp. below) the nodes[1].

[1] As a convenience we have shown the bytes a_i, b_i of messages A, B instead of the triples (l_i, w_i, l_{i-1}).

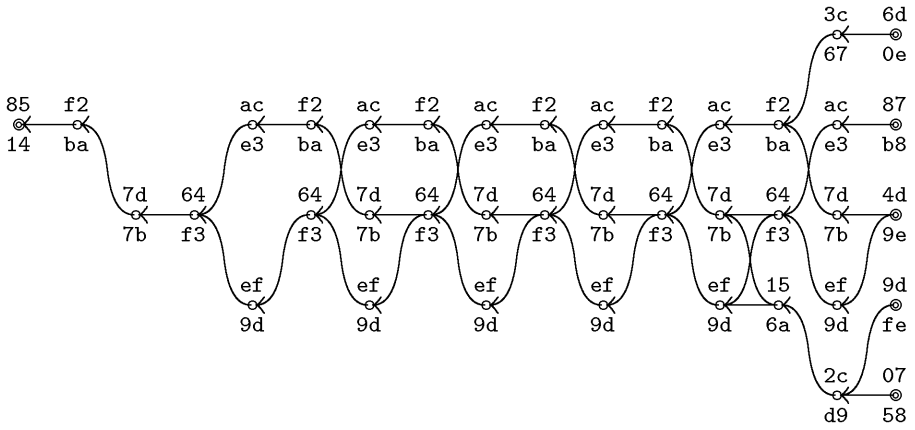


Fig. 1. Graph of solutions of (2) for $k = 256$ and $a/b = 11/19$

For example, if we follow the bottom link, the graph gives messages
A=85f27d64ef..64ef..64ef152c07
B=14ba7bf39d..f39d..f39d6ad958 thus
$\tilde{A}$=458515f2fa7d2964c1ef..2964c1ef..2964c1ef3415572cef76
$\tilde{B}$=78146bbaf67b18f3da9d..18f3da9d..18f3da9d2b6aadd94086
with indeed $\tilde{A}/\tilde{B} = 11/19$.

3.6 Counting solutions

It is easy to count the solutions: assign the count 1 to right nodes, and to all others the sum of the count of their right-linked nodes. The number of solutions to (2) is the sum of the count of the left nodes. This gives 42 for the graph above, which Douglas Adams fans will appreciate.

Since the center part of the graph has a period of two steps, it is trivial to extend it for higher k with $k \equiv 0 \bmod 32$. Asymptoticaly, this count grows by a factor $\frac{\sqrt{5}+1}{2}$ when the modulus is increased by 32 bits.

If we take $k = 1024$ bits and restrict to $b < 2^{10}$, there are 13264 ratios worth to explore. About 40% are eliminated on the first step, 9% have at least one solution to (2), 7% have at least two solutions. There are about $5.7\,10^{14}$ usable message pairs, among which 98% come from the ratio 389/525 which yields 2^{49} solutions. The code computing the above statistics runs in a minute on a personal computer, and can output thousands of messages per second.

Lower bounds on the number of pairs of solutions to (2) are derived by counting solutions for a good ratio

$2^{(k-164.7..)/16}$	solutions for $k \equiv -2 \bmod 16$	using ratio 389/525
$1.62177..^{(k-148.3..)/16}$	solutions for $k \equiv -1 \bmod 16$	using ratio 511/775
$2^{(k-240)/16}$	solutions for $k \equiv 0 \bmod 16$	using ratio 389/525
$1.62177..^{(k-227.6..)/16}$	solutions for $k \equiv 1 \bmod 16$	using ratio 511/775
$2^{(k-226)/16}$	solutions for $k \equiv 2 \bmod 16$	using ratio 389/525

3.7 Existential forgery from the signature of 3 chosen messages

By selecting a ratio a/b and finding two messages pairs A, B and C, D solutions of (2), we can now construct 4 messages A, B, C, D such that

$$\tilde{A}\,\tilde{D} = \tilde{B}\,\tilde{C} \tag{14}$$

With high probability, $\tilde{A}$ and n are relatively prime[2], so that

$$\tilde{D}^s \equiv \tilde{A}^{-s}\tilde{B}^s\tilde{C}^s \bmod n \tag{15}$$

and therefore, for odd v,

$$\ddot{D} = \min(\ddot{A}^{-1}\ddot{B}\,\ddot{C} \bmod n,\ n - \ddot{A}^{-1}\ddot{B}\,\ddot{C} \bmod n) \tag{16}$$

If we can obtain the three signatures $\ddot{A}$, $\ddot{B}$, $\ddot{C}$, it is now straightforward to compute $\ddot{D}$, using the extended Euclidian algorithm for the modular inversion of $\ddot{A} \bmod n$.

For even v, equation (15) implies

$$\ddot{D} = \min(2^{js}\ddot{A}^{-1}\ddot{B}\,\ddot{C} \bmod n,\ n - 2^{js}\ddot{A}^{-1}\ddot{B}\,\ddot{C} \bmod n) \tag{17}$$

$$\text{with } j = \frac{(\tilde{A}|n)-1}{2} + \frac{1-(\tilde{B}|n)}{2} + \frac{1-(\tilde{C}|n)}{2} + \frac{(\tilde{D}|n)-1}{2}$$

If $(a|n) = (b|n)$ then $(\tilde{A}|n) = (\tilde{B}|n)$ and $(\tilde{C}|n) = (\tilde{D}|n)$ thus j is always 0. If $(a|n) = -(b|n)$ then j is -2, 0 or 2, the case $j = 0$ has probability about $1/2$, and it is necessary to examine at most three message pairs before finding two such that $j = 0$. When $j = 0$, equation (17) reduces to (16) and again we obtain a forgery from three signatures.

In summary we have one forgery from three signatures for any public exponent. Using the terminology in [8], it is a chosen messages existential forgery, in that the adversary is bound to pick from a predefined subset the messages submited for signature and the bogus message. More generaly, f forgeries can be obtained from $f + 2$ signatures.

3.8 Total break from the signature of 4 chosen messages for even v

As pointed out in [7], for even public exponents v, finding a multiplicative relation among expanded messages can lead to factorisation of the public modulus n.

We select a ratio a/b such that $(a|n) = -(b|n)$, which for a given n occurs for about half the ratios. We then test solutions of (2) until we find two messages pairs A, B and C, D solutions of (2) verifying $(\tilde{A}|n) = 1$ and $(\tilde{C}|n) = -1$, with the probability of not finding a solution about halved after each trial. For even v, equation (15) implies

$$2^{2s} = u \quad \text{with } u = \pm\ddot{A}\ddot{B}^{-1}\ddot{C}^{-1}\ddot{D} \bmod n \tag{18}$$

[2] else we would get a prime factor of n by computing $\gcd(\tilde{A}, n)$

where the term u is known[3]. Taking the above to the known power $v/2$ and reducing $\bmod p$ gives

$$u^{v/2} \equiv 2^{vs} \equiv 2^{\frac{p-1}{2}\frac{q-1}{2}+1} \equiv (2|p)^{\frac{q-1}{2}}\ 2 \equiv (2|p)\ 2 \bmod p$$

and similarly

$$u^{v/2} \equiv (2|q)\ 2 \bmod q.$$

Noticing that one of p or q is 3 mod 8 and the other is 7 mod 8, we have $(2|p) = -(2|q)$. We deduce that $(u^{v/2}+2) \bmod n$ is a multiple of only one of p or q.

Therefore a prime factor of n is $\gcd(\ddot{A}^{v/2}\ddot{B}^{-v/2}\ddot{C}^{-v/2}\ddot{D}^{v/2}+2 \bmod n,\ n)$.

If we can obtain the four signatures $\ddot{A}$, $\ddot{B}$, $\ddot{C}$, $\ddot{D}$ we can thus factor the modulus n. Of course this let us compute a valid signing exponent s then sign any message just as easily as the legitimate signer, a total break using the terminology in [8].

3.9 Reducing the number of required signatures for small v

Assume we can find two messages A, B solution of

$$\frac{\tilde{A}}{\tilde{B}} = \frac{c^v}{d^v} \quad \text{with } c \neq d \tag{19}$$

This implies

$$\tilde{A}^s d^{vs} \equiv \tilde{B}^s c^{vs} \bmod n \tag{20}$$

For odd v, it follows that

$$\ddot{B} = \min(c^{-1}d\ \ddot{A} \bmod n,\ n - c^{-1}d\ \ddot{A} \bmod n) \tag{21}$$

and we obtain one forgery from a single signature.

For even v, we can similarly obtain forgery from a single signature if $(c|n) = (d|n)$, or factor the modulus from two signature if $(c|n) = -(d|n)$.

Solutions to (19) can be found for $v = 2$. For example with $v = 2$ and $k = 1024$, 21 among the 1933 irreducible ratios with $d < 2^{16}$ give 22645 message pairs, among which 16059 for the ratio $19^2/25^2$. An example for $k = 512$ is:
`ECE8F706C09CA276A3FC8F00803C821D90A3C03222C37DE26F5C3FD37A886FE4`
`CA969C94FA0B801DDEEA0C22932D80570F95A9C767D27FA8F06A56E7371B16DF`

For $v = 3$ the search becomes more difficult, with only 7 ratios and message pairs for $d < 2^{16}$ and $510 \leq k \leq 2050$, and many values of k without a solution. An example is $k = 510$ and ratio $49^3/57^3$ which gives the message pair:
`C6C058A3239EE6D5ED2C4D17588B02B884A30D92B5D414DDB4B5A6DA58B6901B`
`20768B854644F693DB1508DE0124B4457CD7261DF699F422D9634D5E4D5781A4`

[3] within sign; we could recover the sign, but it is not needed.

3.10 Constraints on signed messages

We have seen the number of usable message pairs is huge, and grows exponentialy with the modulus bit size, by a factor of 2 or 1.62177.. for every 16 bit of modulus. The attacker can, with remarkable ease, select among this mass those messages obeying a particular constraint, simply by restricting the range of bytes allowed at some stage in the graph construction.

For example with $k = 0 \bmod 16$ the ratio 389/525 generates many mostly ASCII message pairs, like

`2B0D59B00D060D8FF65300B56A3A3D3D3D3D3D3D3D3D3D3D3D3D3D..3D3D3D37`
`A50F7D50962A02BDE981A4B28D9F5A5A5A5A5A5A5A5A5A5A5A5A..5A5A5A26`

If we restrict all message bytes to $[32..126] \cup [160..255]$, a subset of what Windows considers displayable, it is still easy to generate messages pairs; for example with $k = 512$ the ratio 169/217 gives 682 message pairs like

`5374FC56DEA856DEA856DEA856DEA856DEA856DEA856DEA856DEA856439F22CF`
`27D36E26425A26425A26425A26425A26425A26425A26425A26425A26CD1EB6F1`

and

`53A856DEA856DEA856DEA856DEA856DEA856DEA856DEA856DE74FCA3C7711BAF`
`275A26425A26425A26425A26425A26425A26425A26425A2642D36E0D81C70B21`

3.11 Generality of the attack

The idea of searching solutions to (2) could apply to other redundancy schemes, though no actually used system comes to mind. The same search principle efficiently finds the solutions, *if they exist*, for any redundancy scheme that operate on independant segments of the message, regardless of the tranformations used, as long as the width of the individual segments do no prevent exhaustive search. The search can be conducted sequentialy in either direction, and works equally well if the redundancy added is dependant on the whole portion of the message on some fixed side of the segment, rather than on the segment alone.

Experimentally, the existence of solutions to (2) appears independant of the particular permutation $\Pi(x)$. It does depends to some degree on the repeated use of the same local injection, because that makes the center of the graph more regular. It does depend heavily on an amount of redundancy not markedly exceeding the message itself.

4 Future work

4.1 Other parameters

We have restricted our attack to $k \equiv 0, \pm 1$, or $\pm 2 \bmod 16$ and to messages of $z = \lfloor (k+2)/16 \rfloor$ bytes, the maximum allowed message size. The difficulty appears to increase quickly as the message gets shorter than half the modulus. The attack does works without modification for messages a few bits shorter, and maybe could be extended to any value of k.

4.2 Attack of 'massive mask changes' variants

As a countermeasure against the attack of [4], it has been envisioned in [5] to use not only three injections like in the original standard, but z injections F_i depending on i. Although the above search method applies, the author did not yet establish if (2) has solutions for some ratios a/b with the particular variants[4] proposed.

4.3 Combination with other attacks

Other attacks against ISO/IEC 9796–1 introduced in [4] then perfected in [6] construct messages M which expanded form $\tilde{M}$ is the product of a common constant Γ and small prime factors, then by gaussian elimination find a multiplicative relation similar to 14, although among thousands messages.

The technique we describe can be used to efficently find messages satisfying (2) where a and b only have small prime factors. This gives a relation readily usable in the gaussian elimination process. The combined attack can operate on a wider range of messages, yet still has modest computing requirements.

5 Conclusion

Our attack applies to the full ISO/IEC 9796–1 standard, with common parameters: public modulus of $16z$, $16z \pm 1$, or $16z \pm 2$ bits, and messages of $8z$ bits. Using an inexpensive graph traversal, we constructs 2 messages pairs which expansion are in a common ratio, giving 4 messages which signatures are in a simple multiplicative relation.

For any public exponent v, the attack obtains the forged signature of 1 such message from the legitimate signature of 3 chosen others, or asymptotically nearly one forgery per legitimate signature; it is a major concern for example if obtaining a signature is possible for a price, and forged signatures have a value for messages the attack applies to.

For even v, the attack is a total break in situations where an attacker can obtain the signature of 4 chosen messages (or just 2 for $v = 2$). It is a major concern for example if the attacker can gain limited access to a signing device accepting arbitrary messages, as likely with an off-the-shelf Smart Card implementation of ISO/IEC 9796–1.

The messages the attack can use are computationally easy to generate. Their number grows exponentialy with the modulus size. Messages can efficiently be found including with a small degree of constraint on the message structure.

This prompts the need to revise ISO/IEC 9796–1, or avoid its use in situations where an adversary could obtain the signature of even a few mostly chosen messages.

[4] Remarking that $\Pi(x \oplus y) = \Pi(x) \oplus \Pi(y) \oplus \Pi(0)$, two of the three variants differ only by the choice of arbitrary constants.

References

1. ISO/IEC 9796:1991. Information technology – Security techniques – Digital signature scheme giving message recovery, 1991.
   See also <http://www.iso.ch/jtc1/sc27/27sd799a.htm#9796>.
2. ISO/IEC 9796–1 Second edition Final Committee Draft. Information technology – Security techniques – Digital signature scheme giving message recovery – Part 1: Mechanisms using redundancy.
   Circulated as ISO/IEC JTC1/SC27 N2175 (1998).
3. Guillou, L. C. and Quisquater, J. J. and Walker, M. and Landrock, P. and Shaer, C.: Precautions taken against various potential attacks in ISO/IEC DIS 9796. Advances in Cryptology - EuroCrypt '90 (1990) 465–473.
4. Coron, J. S. and Naccache, D. and Stern, J. P.: A new signature forgery strategy applicable to ISO 9796–1/2, ECASH™, PKCS#1 V2.0, ANSI X9.31, SSL-3.02.
   Circulated as ISO/IEC JTC1/SC27 N2329 alias WG2 N429 (1999).
5. Coppersmith, D. and Halevi, S. and Jutla, C.: Some countermeasures against the new forgery strategy (Working Draft).
   Circulated as ISO/IEC JTC1/SC27 N2362 (1999).
6. Coppersmith, D. and Halevi, S. and Jutla, C.: ISO 9796–1 and the new forgery strategy (Working Draft) (1999).
   See <http://grouper.ieee.org/groups/1363/contrib.html>.
7. Joye, M. and Quisquater, J.J.: On Rabin-type Signatures (Working Draft)
   Circulated as ISO/IEC JTC1/SC27/WG2 N449 (1999).
8. Menezes, A. and van Oorschot, P. and Vanstone, S.: Handbook of Applied Cryptography (1997). CRC Press, ed.
   See <http://cacr.math.uwaterloo.ca/hac/>.