RC 20318 (12/27/95)
Computer Science / Mathematics

IBM Research Report

Low-Exponent RSA with Related Messages

Don Coppersmith

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, New York

Matthew Franklin

AT&T Bell Laboratories
Murray Hill, New Jersey, USA

Jacques Patarin

CP8 Transac, France

Michael Reiter

AT&T Bell Laboratories
Murray Hill, New Jersey, USA

IBM Research Division
Almaden · T.J. Watson · Tokyo · Zurich

Low-Exponent RSA with Related Messages

Don Coppersmith* Matthew Franklin† Jacques Patarin‡ Michael Reiter§

1 Introduction

In this paper we present a new class of attacks against RSA [6] with low encrypting exponent. They enable the recovery of plaintext messages from their ciphertexts and a known polynomial relationship among the messages, provided that the ciphertexts were created using the same RSA public key with low encrypting exponent. These attacks differ from the low-exponent attacks described by Moore [5] and Hastad [4] and the common modulus attack identified by Simmons [8], which pertain only to ciphertexts encrypted under *different* public keys.

Given encryptions of k messages under the same RSA public key with exponent e, together with knowledge of a polynomial relation among the messages of degree δ, the goal of the attacks is to recover all messages. Our results were influenced by an attack presented by Franklin and Reiter [3] for the case $k = 2$, $e = 3$, $\delta = 1$. Starting with this case, we generalize the exponent e in Section 2, the degree δ in Section 3, and the number of messages k in Section 4. Implications of the attack are considered in Section 5.

2 Generalizing the exponent e

Suppose we have two messages m_1 and m_2 related by a known affine relation

$$m_2 = \alpha m_1 + \beta.$$

Suppose further that the messages are encrypted under RSA with an exponent of 3 using a single public modulus N.

$$c_i = m_i^3 \bmod N, \qquad i = 1, 2$$

Then from

$$c_1, c_2, \alpha, \beta, N$$

we can calculate the secret messages m_i algebraically as follows:

$$\frac{\beta(c_2 + 2\alpha^3 c_1 - \beta^3)}{\alpha(c_2 - \alpha^3 c_1 + 2\beta^3)} \equiv \frac{3\alpha^3\beta m_1^3 + 3\alpha^2\beta^2 m_1^2 + 3\alpha\beta^3 m_1}{3\alpha^3\beta m_1^2 + 3\alpha^2\beta^2 m_1 + 3\alpha\beta^3} \equiv m_1 \bmod N.$$

*IBM T.J. Watson Research Center, Yorktown Heights, New York, USA; copper@watson.ibm.com
†AT&T Bell Laboratories, Murray Hill, New Jersey, USA; franklin@research.att.com
‡CP8 Transac, France; patarin@hades.inria.fr
§AT&T Bell Laboratories, Murray Hill, New Jersey, USA; reiter@research.att.com

The algebra is more transparent if we assume (without loss of generality) that $\alpha = \beta = 1$.

$$\frac{(m+1)^3 + 2m^3 - 1}{(m+1)^3 - m^3 + 2} \equiv \frac{3m^3 + 3m^2 + 3m}{3m^2 + 3m + 3} \equiv m \bmod N. \tag{1}$$

So if the RSA exponent is $e = 3$ and we have $k = 2$ messages, satisfying a known polynomial relation of degree $\delta = 1$, we can recover the messages m_i algebraically from the ciphertexts and the coefficients of the polynomial relation.

When $e = 5$, we can find

$$\begin{aligned}
P(m) &= c_2^3 - 3c_1c_2^2 + 3c_1^2c_2 - c_1^3 + 37c_2^2 + 176c_1c_2 + 37c_1^2 + 73c_2 - 73c_1 + 14 \\
mP(m) &= 2c_2^3 - 1c_1c_2^2 - 4c_1^2c_2 + 3c_1^3 + 14c_2^2 - 88c_1c_2 - 51c_1^2 - 9c_2 + 64c_1 - 7 \\
m &= \frac{mP(m)}{P(m)}
\end{aligned}$$

For an arbitrary exponent e in the case of $k = 2$ messages subject to a linear relation, it will always be possible to write down an equation analogous to (1). Specifically, there will exist polynomials $P(m)$ and $Q(m)$ such that each can be expressed as rational polynomials in m^e and $(m+1)^e$, and such that $Q(m) = mP(m)$. Already for $e = 5$, however, this is fairly complicated. As e grows, this explicit expression of m as a ratio of two polynomials in c_1 and c_2 requires $\Theta(e^2)$ coefficients, and it is not immediately obvious how to calculate these coefficients efficiently.

Fortunately there is an easier method. Let z denote the unknown message m. Then z satisfies the following two polynomial relations:

$$\begin{aligned}
z^5 - c_1 &\equiv 0 \bmod N \\
(z+1)^5 - c_2 &\equiv 0 \bmod N.
\end{aligned}$$

where the c_i are treated as known constants. Apply the Euclidean algorithm to find the greatest common divisor of these two univariate polynomials over the ring $\mathbf{Z}/N$:

$$\gcd(z^5 - c_1, (z+1)^5 - c_2) \in \mathbf{Z}/N[z].$$

Except in rare cases, this gcd is the linear polynomial $z - m$:

$$\gcd(z^5 - c_1, (z+1)^5 - c_2) = z - m.$$

This attack applies for any value of e, but is limited by the costs required to compute the gcd of two polynomials of degree e. A straightforward implementation of Euclid's algorithm takes $O(e^2)$ operations in the ring $\mathbf{Z}/N$. More sophisticated techniques can be used to compute the gcd in $O(e \log^2 e)$ time [10]. Using these methods, the attack may be practical for all exponents of length up to around 32 bits. For example, the attack will be very efficient against $e = 2^{16} + 1$, a popular choice in many applications.

3 Generalizing the degree δ of the polynomial

One generalization is immediate. Suppose we have two messages m_1, m_2 satisfying a known polynomial relation of the form

$$m_2 = p(m_1), \qquad \deg(p) = \delta,$$

and we know p and the two ciphertexts c_i. Then as before, the two equations

$$\begin{aligned} z^e - c_1 &\equiv 0 \bmod N \\ (p(z))^e - c_2 &\equiv 0 \bmod N \end{aligned}$$

are both satisfied by $z = m_1 \bmod N$ so that

$$\gcd(z^e - c_1, (p(z))^e - c_2) \in \mathbf{Z}/N[z]$$

will be divisible by $z - m_1$, and except in rare cases we will have

$$\gcd(z^e - c_1, (p(z))^e - c_2) = z - m_1.$$

(One of the exceptional cases, in which this procedure fails, is when $p(z)$ is of the form $p(z) = z^h q(z^e)$, because then the ciphertext c_2 is easily derived from the ciphertext c_1, namely

$$c_2 = c_1^h (q(c_1))^e,$$

and we gain no new information.)

What if m_1 and m_2 satisfy an *implicit* polynomial relation?

$$p(m_1, m_2) = 0 \bmod N$$

In this case we have three polynomials relating two unknowns $\bmod N$:

$$\begin{aligned} P_1 &= p(m_1, m_2) &&= 0 \bmod N \\ P_2 &= m_1^e - c_1 &&= 0 \bmod N \\ P_3 &= m_2^e - c_2 &&= 0 \bmod N \end{aligned}$$

Now we need another algebraic tool: the resultant. The resultant of two multivariate polynomials, with respect to one of their variables x, is a third multivariate polynomial, in all the variables except the special variable x. For any setting of all the variables (including x) for which the two input polynomials are simultaneously 0, the resultant is also 0. This gives us a means of eliminating x from a system of equations.

In the example above, substitute x and y for the unknowns m_1 and m_2, respectively. We can take the resultant of $P_1(x, y)$ and $P_2(x)$ with respect to the variable x, over the ring $\mathbf{Z}/N$. This will yield a polynomial P_4 in y, whose degree is at most δe (the product of the total degrees of the original polynomials). Then we can take the gcd of this new polynomial P_4 with P_3 (as univariate polynomials in y over $\mathbf{Z}/N$) to yield (hopefully) the linear polynomial $y - m_2$. Substitute this value of m_2 which we have discovered, back into P_1, to find a polynomial in x alone, which we can combine with P_2 by the gcd to find the correct value of $x = m_1$.

The two operations, resultant and gcd, can be combined under the general heading of "Groebner basis." Indeed, fixing the coefficients of p and the ciphertexts c_i as constants, and computing the Groebner basis of $P_1(x, y)$, $P_2(x)$, $P_3(y)$, as polynomials over $\mathbf{Z}/N$, will generally produce the desired result:

$$[(x - m_1), (y - m_2)].$$

4 Generalizing the number of messages k

4.1 Arbitrary polynomial relationship among messages

Suppose that we have k messages $m_1, \ldots, m_k$, related by a polynomial $p(m_1, \ldots, m_k)$, and that we know the ciphertexts $c_i = m_i^e \bmod N$ and the coefficients of the polynomial p. As before, substitute variables x_i for the unknown messages m_i, and obtain the $k+1$ polynomials

$$\begin{array}{rcccl} P_0(x_1, \ldots, x_k) & = & p(x_1, \ldots, x_k) & = & 0 \bmod N \\ P_1(x_1) & = & x_1^e - c_1 & = & 0 \bmod N \\ P_2(x_2) & = & x_2^e - c_2 & = & 0 \bmod N \\ \ldots & & & & \\ P_i(x_i) & = & x_i^e - c_i & = & 0 \bmod N \\ \ldots & & & & \\ P_k(x_k) & = & x_k^e - c_k & = & 0 \bmod N \end{array}$$

which must be simultaneously satisfied. We can just compute

$$\text{Groebner}([P_0, P_1, \ldots, P_k])$$

and generally obtain the answer

$$[x_1 - m_1, \ldots, x_k - m_k].$$

Or, more explicitly, we can set

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

and iteratively evaluate

$$Q_i(x_{i+1}, \ldots, x_k) = \text{Resultant}_{x_i}(Q_{i-1}, P_i)$$

until we find

$$Q_{k-1}(x_k).$$

Then evaluating

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

we hope to find the linear polynomial $(x_k - m_k)$, from which we discover m_k. Finally, repeatedly back substitute:

$$\gcd(P_i(x_i), Q_{i-1}(x_i, m_{i+1}, m_{i+2}, \ldots, m_k)) = (x_i - m_i),$$

as i goes from $k-1$ to 1, to find all the messages m_i.

The complexity of this general attack is dominated by the computation of the resultants $Q_1, \ldots, Q_{k-1}$. Each Q_i is a polynomial of total degree $e^i\delta$, in $k-i$ variables. The number of monomial terms in each Q_i is potentially as large as $O(\delta^{k/2} e^{k^2/4})$. Thus, this attack may require $\delta^{\Theta(1)k} e^{\Theta(1)k^2}$ operations over $\mathbf{Z}/N$.

4.2 A special case: linearly related messages

As will be seen in Section 5.2, a special case of interest is that in which $P_0 = p$ is linear, say

$$p(x_1, \ldots, x_k) = x_1 + x_2 + \ldots + x_k - w$$

with w some known constant. The special form of p allows us to make the computations more efficient. In addition, the success of the attack differs depending on whether w is zero or nonzero.

4.2.1 Inhomogeneous case

In the inhomogeneous case $w \neq 0$, the attack proceeds as follows. Introduce unknowns

$$y_i = x_1 + x_2 + \ldots + x_i = w - x_k - x_{k-1} - \ldots - x_{i+1}$$

satisfying

$$\begin{aligned} y_{i-1} &= y_i - x_i \\ y_{i+1} &= y_i + x_{i+1} \end{aligned}$$

Set $h = \lfloor k/2 \rfloor$. Introduce polynomials:

$$\begin{aligned} R_1(y_1) &= P_1(y_1) \\ R_i(y_i) &= \text{Resultant}_{x_i}(R_{i-1}(y_i - x_i), P_i(x_i)), \qquad i = 1, 2, \ldots, h \\ S_{k-1}(y_{k-1}) &= P_k(w - y_k) \\ S_i(y_i) &= \text{Resultant}_{x_i}(S_{i+1}(y_i + x_{i+1}), P_{i+1}(x_{i+1})), \qquad i = k-2, k-3, \ldots, h \end{aligned}$$

Both $R_h(y_h)$ and $S_h(y_h)$ are univariate polynomials in y_h. Their gcd will hopefully be the linear polynomial

$$y_h - (m_1 + m_2 + \ldots + m_h)$$

and we can proceed from there by divide-and-conquer.

Each of the resultants in the above equation has the special form

$$\text{Resultant}_x(R(y - x), x^e - c).$$

A shortcut to computing this resultant is to evaluate $R(y - x)$ by Horner's rule, replacing each occurrence of x^e by c.

4.2.2 The homogeneous case

A difficulty arises in the homogeneous case $w = 0$. Because P_0 is homogeneous, and all the other polynomials are of the form $x^e - c$, given any solution $(x_1, x_2, \ldots, x_k)$ and any eth root of unity ϕ, the tuple $(\phi x_1, \phi x_2, \ldots, \phi x_k)$ is also a solution of all the P_i. Thus the attacker cannot solve for the individual x_i; the most it can hope for is to be able to solve for all homogeneous polynomials of degree e in the x_i. In particular, when the attacker attempts to compute y_h, the gcd yields at best something like

$$y_h^e - (m_1 + m_2 + \ldots + m_h)^e$$

rather than the desired linear polynomial. This is an inherent difficulty in our approach, and has a bearing on the application described in Section 5.2.

However, there are occasions when the attack succeeds even in the homogeneous case. For example, suppose

$$\begin{aligned} m_1 &= (m + \lambda_1 \beta) \bmod N \\ m_2 &= (m + \lambda_2 \beta) \bmod N \\ m_3 &= (m + \lambda_3 \beta) \bmod N \end{aligned}$$

where $\lambda_1, \lambda_2, \lambda_3$ are known, while m, β are unknown. Further suppose that β is known to satisfy $\beta^3 < N$. This is a homogeneous case, because the three plaintexts are known to satisfy the linear relation

$$(\lambda_3 - \lambda_2)m_1 + (\lambda_1 - \lambda_3)m_2 + (\lambda_2 - \lambda_1)m_3 \equiv 0 \bmod N.$$

From the three ciphertexts $c_i = m_i^3 \bmod N$ (i.e., $e = 3$) and the three coefficients $\lambda_1, \lambda_2, \lambda_3$, it is possible to solve for $B = \beta^3 \bmod N$: e.g., $B = \gcd(f, g)$, where

$$\begin{aligned} f &= \text{Resultant}_m((m + \lambda_1\beta)^3 - c_1, (m + \lambda_2\beta)^3 - c_2) \\ g &= \text{Resultant}_m((m + \lambda_1\beta)^3 - c_1, (m + \lambda_3\beta)^3 - c_3) \end{aligned}$$

Then β can be recovered by computing the real cube root of B (i.e., without modular reduction), from which m can be recovered using previous techniques.

5 Implications

Due to the widespread popularity of RSA with low encrypting exponent, our attacks potentially have implications to the security of a wide range of current and future cryptographic protocols. In this section we show how our attacks reveal vulnerabilities in two protocols.

5.1 The TMN protocol

In [11], Tatebayashi, Matsuzaki and Newman proposed a key distribution protocol. In this protocol, a passive eavesdropper sees $r_1^e \bmod N$, $r_2^e \bmod N$, and $r_1 + r_2 \bmod N$ exchanged among the protocol participants, where $e = 3$ and r_1, r_2 are randomly generated values. The techniques of the previous sections enable a passive eavesdropper to learn the shared session key r_2 distributed in the protocol.

Simmons [9] previously found an *active* attack on this scheme (requiring two conspirators), for which three counter measures were suggested in [11]. The first two countermeasures—incorporating structure into r_1 and r_2, and prepending timestamps to r_1 and r_2—do not prevent our passive attack. The third, which assumes a shared secret key between the server and each party, appears to withstand our passive attack, as do variations in which $r_1 + r_2 \bmod N$ is replaced by a non-polynomial function of the messages (e.g., $r_1 \oplus r_2 \bmod N$).

5.2 Verifiable signature sharing

In [2], Franklin and Reiter presented a scheme to efficiently share an RSA signature of a known message among $n \geq 5t + 1$ servers so that the servers could verify the signature relation, despite the malicious misbehavior of up to t of the servers. As part of this protocol, the holder of the signature shares the signature using Shamir's secret sharing scheme [7], i.e., by choosing at random a univariate polynomial

$$B(x) = \sum_{j=0}^{t} b_j x^j$$

over $\mathbf{Z}/N$ such that b_0 is the secret signature, and privately sending the share $B(i)$ to the i-th server for $1 \leq i \leq n$. The intention is to ensure that a subset of $t + 1$ servers can use their shares to recover B (by Lagrange interpolation), but to prevent t or fewer malicious servers from doing so. However, the holder of the signature also publishes the RSA encryption of each share under the same RSA public key with exponent $e = 3$, i.e., $\{B(i)^e \bmod N\}_{1 \le i \le n}$.

The techniques of this paper enable one server S_{i_0}, knowing its share $B(i_0)$ and the ciphertexts $B(i_1)^e \bmod N, \ldots, B(i_k)^e \bmod N$ of at least $k = t + 1$ other shares (i.e., $i_0 \ne i_j$ for any $1 \le j \le k$), to compute the secret polynomial B. Specifically, because the shares $B(i_0), \ldots, B(i_k)$ are linear combinations of the unknowns b_j with known coefficients, S_{i_0} can compute a linear relation that holds among the shares:

$$\sum_{j=0}^{k} p_j B(i_j) \equiv 0 \bmod N. \tag{2}$$

Since server S_{i_0} knows $p_0 B(i_0)$, which is nonzero with high probability, it can learn an inhomogeneous linear relation among the other k terms $\{p_j B(i_j)\}_{1 \le j \le k}$:

$$\sum_{j=1}^{k} p_j B(i_j) \equiv -p_0 B(i_0) \bmod N.$$

Using this information, and the easily computable ciphertext $p_j^e B(i_j)^e \bmod N$ of each term, it can use the techniques described before to recover each term $p_j B(i_j)$ and hence $B(i_j)$. Using Lagrange interpolation, it can then recover the secret polynomial B and the signature b_0.

It is interesting to note that this attack fails for a passive eavesdropper that is not one of the n servers. Such an eavesdropper sees only the published RSA encryptions of each share, i.e., $\{B(i)^e \bmod N\}_{1 \le i \le n}$. The eavesdropper can again find a linear equation of the form (2) among any $k + 1$ of the shares. However, since this equation is homogeneous, it can recover only homogeneous polynomials of degree e in the terms $p_j B(i_j)$ (see Section 4.2.2).

6 Conclusion

We have identified a new class attacks against RSA with low encrypting exponent, which exploit known polynomial relationships among the encrypted messages. This can lead to weaknesses in protocols for which such relationships can be inferred. When the relationships are essential to the correctness of a protocol, as in the two instances that we attack, the only repair seems to be increasing the size of the encrypting exponent. If the polynomial relationships are not essential, then another repair might be to transform the plaintexts so that those relationships no longer hold. One possible transformation would be to apply a public permutation, such as DES with a fixed key. Another transformation might be to pad the plaintext with random bits, though this may not always suffice; see [1]. In any case, a careful look at each specific use of RSA is in order.

References

[1] Finding a small root of a univariate modular equation. Anonymous submission to EUROCRYPT '96, 1996.

[2] M. K. Franklin and M. K. Reiter. Verifiable signature sharing. In *Advances in Cryptology—EUROCRYPT '95* (Lecture Notes in Computer Science 921), L. C. Guillou and J. Quisquater, Eds. 1995, pp. 50–63, Springer-Verlag.

[3] M. K. Franklin and M. K. Reiter. A linear protocol failure for RSA with exponent three. Presented at the CRYPTO '95 Rump Session, Aug. 1995.

[4] J. Hastad. Solving simultaneous modular equations of low degree. *SIAM Journal of Computing* 17:336–341, 1988.

[5] J. H. Moore. Protocol failures in cryptosystems. *Proceedings of the IEEE* 76(5), May 1988.

[6] R. L. Rivest, A. Shamir and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* 21(2):120–126, Feb. 1978.

[7] A. Shamir. How to share a secret. *Communications of the ACM* 22(11):612–613, Nov. 1979.

[8] G. Simmons. A "weak" privacy protocol using the RSA cryptoalgorithm. *Cryptologia* 7:180–182, 1983.

[9] G. Simmons. Proof of soundness (integrity) of cryptographic protocols. *Journal of Cryptology* 7:69–77, 1994.

[10] V. Strassen. The computational complexity of continued fractions. *SIAM J Comput* 12(1):1–27, 1983.

[11] M. Tatebayashi and N. Matsuzakai and D. B. Newman. Key distribution protocol for digital mobile communication systems. In *Advances in Cryptology—CRYPTO '89* (Lecture Notes in Computer Science 435), G. Brassard, Ed. 1990, pp. 324–333, Springer-Verlag.