Factoring Unbalanced Moduli with Known Bits

Eric Brier[1], David Naccache[2], and Mehdi Tibouchi[2]

[1] Ingenico
1, rue Claude Chappe, BP 346, F-07503 Guilherand-Granges, France
eric.brier@ingenico.com
[2] École normale supérieure
Équipe de cryptographie, 45 rue d'Ulm, F-75230 Paris CEDEX 05, France
{david.naccache, mehdi.tibouchi}@ens.fr

Abstract. Let $n = pq > q^3$ be an RSA modulus. This note describes a LLL-based method allowing to factor n given $2\log_2 q$ contiguous bits of p, irrespective to their position. A second method is presented, which needs fewer bits but whose length depends on the position of the known bit pattern. Finally, we introduce a somewhat surprising *ad hoc* method where two different known bit chunks, totalling $\frac{3}{2}\log_2 q$ bits suffice to factor n.

1 Introduction

The problem of factoring using partial information was introduced by Rivest and Shamir [9] in 1986. Factoring using partial information relates both to the (very theoretical) *oracle complexity* of factoring and to the (very practical) *side channel* analysis of public-key implementations.

In most past works [9, 3–5, 10] the attacker knows some of the bits of one of the factors, usually the most significant bits (MSBs) or chunks of bits spread over one of the factors [6]. In other settings (*e.g.* [7]), the opponent is given access to an oracles answering yes/no questions. Recently, May and Ritzenhofen considered the factoring of integers whose factors feature a common, yet unknown, bit-pattern [8]. Finally, [1] tackles the factorization of numbers of the form $p^r q$.

In this note we show that when $n = pq > q^3$ (figure 1), one can factor n given $2\log_2 q$ contiguous bits of p. The technique is interesting because it does not appear to relate directly to other LLL-based results. Furthermore, the amount of bits to be known does not depend on the size of n but rather of the size of its smaller factor q.

Conventions: Throughout this paper, capital letters will denote the bit-size of lowercase variables. In addition, we will illustrate the different factoring techniques using black rectangles for known (given) bit blocs and white rectangles for unknown bit blocks.

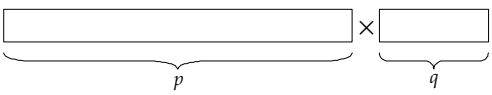

Fig. 1. The factoring problem: Hard.

2 An Initial Observation

Factoring given $p' = p \bmod 2^Q$, the Q least significant bits (LSBs) of p, is trivial:

$$\frac{n}{p'} \bmod 2^Q = q \bmod 2^Q = q$$

Fig. 2. Factoring knowing the Q LSBs of p: Easy.

It is easy to observe that factoring unbalanced moduli is also easy when p presents a pattern of Q zeros at positions $[2Q, Q]$.

If p is of the form $p = u2^{2Q} + y$ where $Y \leq Q$ then:

$$\gcd\left(n, n \bmod 2^{2Q}\right) = \gcd\left(pq, yq \bmod 2^{2Q}\right) = \gcd\left(pq, yq\right) = q$$

Fig. 3. Factoring knowing that bits $[2Q, Q]$ of p are zeros: Easy.

$p = u2^{2Q} + y$ is a particular case of the general form $p = u2^{W+L} + v2^W + y$ where v is a known L-bit pattern.

Given v and setting $a = v2^W$, $b = n \bmod 2^{W+L}$ and $q = x$, factoring n boils-down to solving the equation:

$$b = x(a + y) \bmod 2^{W+L} \tag{1}$$

for $\{x, y\}$, with x of size Q and y of size W.

The two following sections focus on solving this equation.

3 Applying Lattice Reduction

The most straightforward approach to solve equation (1) is to set $z = xy$. The new variable z being of size $Q + W$. The equation becomes:

$$b = ax + z \bmod 2^{W+L}$$

which is a bivariate linear modular equation. In [4], Coppersmith gives an LLL-based heuristic algorithm, to solve such equations when the sum of the sizes of variables is less than the modulus divided by the equation's degree. In our case, this means that:

$$Q + (Q + W) < W + L$$

The solution is thus found as soon as $L > 2Q$. This means that n can be factored as soon as $2Q$ contiguous bits of p are known, no matter where their position is (figure 4).

Fig. 4. Factoring given any $2Q$-bit block of p: Easy.

4 Using Fewer Bits

We also notice that this equation is very similar to Boneh and Durfee's *Small Inverse Problem* (section 4 of [2]). This problem amounts to solving the equation:

$$1 = x(a + y) \bmod e.$$

Replacing 1 by an arbitrary integer b does not change anything in the algorithm's analysis, since the diagonal of the triangular basis of the lattice used to solve the equation is independent of b.

The main difference is that [2] handles only the case $2Y = E$, which is not necessarily the case in our setting.

We will focus on solving $b = x(a + y) \bmod e$ for $\{x, y\}$ with $X = e^\delta$ and $Y = e^\alpha$. The lattice is built as in section 4 of [2] but the choice of the optimizing parameter t will differ.

For convenience, we set $t := \tau m$. In the inequality $\det(L) < e^{mw}$, we consider only dominant terms, *i.e.* terms in m^3, we get the inequality:

$$3\alpha\tau^2 + 3(\alpha + \delta - 1)\tau + \alpha + 2\delta - 1 < 0.$$

Solutions to this inequality exist if and only if the discriminant of the quadric equation in τ is positive. The condition on α and δ is:

$$3\delta^2 - \alpha^2 - 2\alpha\delta - 6\delta - 2\alpha + 3 > 0.$$

This result is of an independent interest, generalizing [2].

To harness this to our initial problem, one needs to set

$$e = 2^{W+L}, \quad \delta = \frac{Q}{W + L}, \quad \text{and} \quad \alpha = \frac{W}{W + L}.$$

Then, the length L of the known bit pattern must satisfy

$$3L^2 + (4W - 6Q)L + 3Q^2 - 8QW > 0.$$

This quadric admits two solutions, the smaller of which corresponds to parameters $\delta > 1$, which makes no sense. Taking the larger solution into account, the final result is:

$$L > Q + \frac{2}{3}(\sqrt{W^2 + 3QW} - W).$$

In other words, as the position of the known bit block slides from the LSBs to the MSBs (*i.e.* when W increases from 0 to ∞), the amount of known bits increases from Q to $2Q$ (figure 7). This method is always better than the one presented in section 3.

Fig. 5. Factoring given progressively bigger chunks of p: Easy.

5 *Ad Hoc* Configurations

In addition to the previously presented techniques, it appears possible to obtain better results in a number of specific cases. We illustrate two such instances in this section.

5.1 Disjoint LSB Blocks

We first turn our attention to the case, very similar to the observation in the introduction, where we know a pattern of Q bits in the prime factor p starting from the Q-th bit. As per the previous section's results, this is not enough since we would need $5Q/3$ bits to factor. Thus, we suppose that we also know the L LSBs of p. Evidently, we can now get the L LSBs of q as well by division modulo 2^L. In other words, we have the following representation of the factors:

$$p = u2^{Q+L} + v2^Q + y2^L + w$$
$$q = x2^L + w',$$

where v, w and w' are known and $ww' = n \bmod 2^L$.

Expand the equation $pq = n$ and reduce it modulo 2^{2Q}. Obviously, one can factor 2^L since we properly selected w'. We get a quadratic bivariate equation in the variables x and y. The variables are of size 2^{Q-L} and the equation must be satisfied modulo 2^{2Q-L}. Note that the only quadratic term is $xy2^L$, hence the equation becomes linear modulo 2^L (and easy to solve). We use lattice reduction techniques to present the general solution under the form:

$$x = x_0 + rx_1 + sx_2$$
$$y = y_0 + ry_1 + sy_2,$$

where r and s are unknown integers. The linear equation is to be understood modulo 2^L and thus the numbers x_i and y_i can be chosen of approximate size $2^{L/2}$. Since x and y are of size 2^{Q-L}, we infer that r and s are of size $2^{Q-3L/2}$. We now plug the parameterizations of x and y into our original equation and get a quadratic equation in the variables r and s. It is clear that we can factor 2^L and get an equation modulo 2^{2Q-2L}. Again, we use Coppersmith's heuristic for bivariate equations to compute r and s. For this to be possible, the product of the size of the variables must be less than half the modulus' size:

$$2(Q - \frac{3L}{2}) \leq Q - L,$$

which is easily transformed into $Q \geq 2L$. From the values of r and s, we get back to the values of x and y and subsequently find q.

All in all, if we know a pattern of Q bits of p in the range $[Q, 2Q]$ and the $Q/2$ LSBs of p (or q), we can factor n in heuristic polynomial time (figure 6). Note that the number of bits needed in this section is less than the number claimed by the previous section. This stems from the fact that p's LSBs of p leak direct knowledge on q's LSBs.

Fig. 6. Factoring given bits $[2Q, Q]$ and $[Q/2, 0]$ of p: Easy.

5.2 Particular n Formats

We now adapt to our purpose the finer analysis of [2], section 5 (instead of section 4). Consider again:

$$b = x(a + y) \bmod e$$

with $|x| \leq X = e^{\delta}$ and $|y| \leq Y = e^{\alpha}$, where α and δ are such that $\alpha + \delta < 1$.

Unfortunately, in general, the resulting matrix M_y of y-shifts is not geometrically progressive in Boneh-Durfee's sense.

However, M_y does become geometrically progressive if we further assume that $|b| \leq e^r$ for some constant $r \in \mathbb{R}$ satisfying:

$$0 < r < \alpha + \delta \quad \text{and} \quad r < 2 - \frac{1-\delta}{\alpha}$$

The implications of this assumption will be examined at the end of this section.

We can readily verify that M_y is geometrically progressive with parameters $(4^m, e, m, \alpha + \delta - r, \alpha - 1, r - 1, 1, (1-r)/(\alpha + \delta - r))$.

Setting the parameter t to $(1 - \alpha - \delta)m/\alpha$, we find that $\det(L_1)e^{-mw} = e^{u(m)}$ with

$$\begin{aligned} u(m) &= \left[2 + \alpha + 2\delta + \frac{1}{\alpha}(1-\alpha-\delta)(2+\alpha+\delta)\right]\frac{m^3}{6} - \frac{1-\delta}{2\alpha}m^3 + o(m^3) \\ &= \left(\alpha - (1-\delta)^2\right)\frac{m^3}{6\alpha} + o(m^3) \end{aligned}$$

It follows that lattice reduction can be applied for large enough m as soon as:

$$(1-\delta)^2 > \alpha$$

This result may, yet again, be of independent interest.

Returning to our particular setting in which

$$e = 2^{W+L}, \quad \delta = \frac{Q}{W+L}, \quad \text{and} \quad \alpha = \frac{W}{W+L}$$

we see that the length L must satisfy $(L + W - Q)^2 > W(L + W)$, which gives the following bound:

$$L > Q + \frac{1}{2}(\sqrt{W^2 + 4QW} - W)$$

Although this bound also increases from Q to $2Q$ as W grows, it is always (slightly) tighter than the bound obtained it section 4. To assess the best possible improvement we denote $W = \lambda Q$ and seek to maximize:

$$f(\lambda) = \frac{2}{3}(\sqrt{\lambda^2 + 3\lambda} - \lambda) - \frac{1}{2}(\sqrt{\lambda^2 + 4\lambda} - \lambda)$$

We have $f'(\lambda_0) = 0$ for $\lambda_0 \approx 0.716$, the positive root of the polynomial $\lambda^4 + 7\lambda^3 + 12\lambda^2 - 9$ corresponding to a maximal gain of $f(\lambda_0) \approx 0.049$.[3]

[3] *e.g.* for $Q = 400$ bits the attack requires ≈ 20 fewer bits.

However, this 5% improvement is only obtained under a very costly assumption: the condition on b implies that n has pattern of $Q + L/W$ zero bits before position $W + L$. Note that:

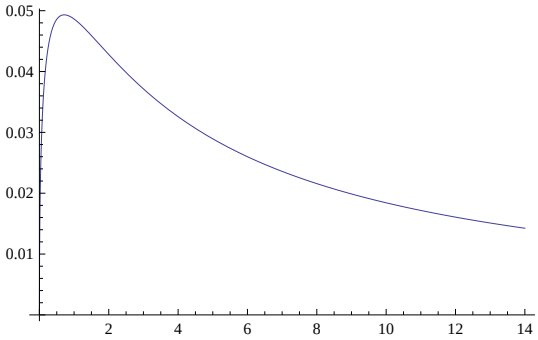

Fig. 7. A plot of $f(\lambda)$. Note that $\lim_{\lambda \to \infty} f(\lambda) = 0$

The technique will therefore only apply to n values having this special form.

6 Conclusion

This paper showed that the knowledge of a pattern of contiguous bits in the larger factor of an unbalanced modulus is sufficient to factor as soon as the length of this pattern is twice the size of the smaller factor.

A deeper analysis showed that fewer bits are required, depending on the known bit-chunk's position.

The existence of a variety of *ad hoc* configurations, of which we gave two examples seems to indicate that a systematic exploration of topic is an interesting further research direction.

References

1. D. Boneh, G. Durfee and N. Howgrave-Graham, *Factoring $N = p^r q$ for large r*, Advances in Cryptology - Crypto'99, Lecture Notes in Computer Science Vol. 1666, Springer-Verlag, pp. 326-337, 1999.
2. D. Boneh and G. Durfee, *Cryptanalysis of* RSA *with private key d less than* $n^{0.292}$, IEEE Transactions on Information Theory, vol. 46, pp. 1339-1349, 1999.
3. D. Coppersmith, *Factoring with a hint*, IBM Research Report RC 19905, 1995.
4. D. Coppersmith, *Finding a small root of a univariate modular equation*, Advances in Cryptology (Eurocrypt'96), Lecture Notes in Computer Science Vol. 1070, Springer-Verlag, pp. 155-165, 1996.
5. C. Crépeau and A. Slakmon, *Simple backdoors for* RSA *key generation*, Topics in Cryptology (CT-RSA 2003), Lecture Notes in Computer Science Vol. 2612, pp. 403-416, Springer-Verlag, 2003.
6. M. Herrmann, A. May, *Solving linear equations modulo divisors: On factoring given any bits*, In Advances in Cryptology (Asiacrypt 2008), Lecture Notes in Computer Science Vol. 5350, pp. 406-424, Springer-Verlag, 2008.
7. U. Maurer, *Factoring with an oracle*, Advances in Cryptology (Eurocrypt'92), Lecture Notes in Computer Science Vol. 658, pp. 429-436, Springer-Verlag 1993.

8. A. May and M. Ritzenhofen, *Implicit Factoring: On polynomial time factoring given only an implicit hint*, Practice and Theory in Public Key Cryptography (PKC 2009), Lecture Notes in Computer Science Vol. 5443, pp. 1-14, Springer-Verlag, 2009.
9. R. Rivest and A. Shamir, *Efficient factoring based on partial information*, Advances in Cryptology (Eurocrypt'85), Lecture Notes in Computer Science Vol. 219, pp. 31-34, Springer-Verlag, 1986.
10. B. Santoso, N. Kunihiro, N. Kanayama and K. Ohta, Factorization of square-free integers with high bits known, Progress in Cryptology (Vietcrypt 2006), Lecture Notes in Computer Science Vol. 4341, pp. 115-130, 2006.