Report on Integer Factorization

René Peralta [*]

December 14, 2001

1 Introduction

This report describes the state of knowledge about the computational complexity of the integer factorization problem for integers of special form. The form is $P^kQ \quad (k \geq 1)$ where P and Q are primes chosen uniformly at random from spaces Ω_P, Ω_Q, respectively. It is assumed that $\Omega_P = \Omega_Q = \omega$-bit integers for a security parameter ω. Of special interest are

- the case $k = 1$, which we shall call "RSA numbers";
- the case $k = 2$, which we shall call "Okamoto" numbers since they are proposed for several applications in papers by T. Okamoto and various co-authors.

Per IPA's contract specifications, this report includes discussion of three specific questions:

1. The effectiveness of the elliptic curve (EC) and number field sieve (NFS) methods in factoring Okamoto numbers.
2. The effectiveness of the lattice-based method in factoring Okamoto numbers.
3. Given
   - the state of knowledge in computational number theory;
   - the computational resources currently available;
   - the expectations regarding the increase in computational resources in future years

   estimate the bit-length of N sufficient to ensure that factoring RSA numbers (Okamoto numbers, respectively) is infeasible.

[*] The author is with the Department of Computer Science, Yale University.

2 What is feasible and what is not

We will express the number of operations an algorithm performs in terms of powers of 2. This is so that we may more easily correlate these values with machine speeds. A supercomputer's speed is currently measured in Terahertz ($\approx 2^{40}$ cycles per second). It takes many cycles to perform each of the arithmetic operations involved in factoring algorithms. However, it is currently possible for a large organization to build a special-purpose machine which could perform arithmetic operations over the ring Z_N at speeds in the Terahertz range. There are under 2^{30} seconds in 20 years. Thus, such a machine would be able to perform no more than 2^{70} operations in twenty years. Allowing for a safety factor of about 2^{15} to take into account possible increase in computer power during two decades yields a target security parameter of 2^{85}. That is, we will consider a problem infeasible if it takes on the order of 2^{85} arithmetic operations.

The above can be considered a very strong security requirement in the short range. In the medium range, say 5 to 10 years, we have very high confidence that 2^{85} arithmetic operations will remain infeasible. The statement becomes more speculative when one considers a horizon of twenty years. Thus, one should reconsider the above statements every five years or so. Furthermore, we have assumed that no dramatically different (as opposed to just faster) machines will be built in the next few decades. This may not be a safe assumption. In particular, close attention should be payed to developments in the area of quantum computation. A large enough quantum computer, if ever built, would be able to factor both RSA and Okamoto numbers in polynomial time. Most experts currently think that we are many decades away from being able to build such a machine. Many other experts believe such a machine will never be built. My own opinion on this matter is that it is more likely that progress in algorithms force us to revise the above security parameters sometime in the next two decades. In particular, combinatorial approaches to integer factorization (as opposed to algebraic ones) have not been adequately studied.

3 Relative effectiveness of EC and NFS algorithms on Okamoto numbers

Let $|N| = n$ be number of bits of N. Both EC and NFS methods run in a number of steps which is exponential in n. For Okamoto numbers only, the approximate asymptotic running time EC is given by

$$2^{0.981 * \sqrt{n \ln(0.231n)}}.$$

For either Okamoto or RSA numbers, the approximate asymptotic running time of NFS is given by

$$2^{2.428 n^{1/3} \ln(0.693n)^{2/3}}.$$

For Okamoto numbers, NFS is asymptotically faster than EC. However, a plot of these curves shows that EC is faster than NFS for n smaller than 2800.

At $n = 2800$ the number of steps of both algorithms is about 2^{132}. This is well beyond our security goal parameter of 2^{85}. The conclusion is that, for Okamoto numbers in the range of practical interest to cryptographic applications, we may restrict our attention to the EC method of factorization.

4 Lattice-based methods

Methods based on applications of LLL (the lattice reduction algorithm due to Lenstra, Lenstra and Lovasz) can be used to factor numbers of the form P^kQ in polynomial time when k is of order $log(P)$. These methods do not pose a direct threat to Okamoto numbers. The fastest known lattice-based method for factoring Okamoto numbers performs about $N^{1/9}$ lattice reduction steps. This is slower than most factoring algorithms. However, if about one-third of the bits of P are somehow leaked, then an Okamoto number can be easily factored by a lattice based method. This is not to be considered a significant weakness of Okamoto numbers: any cryptographic application based on the difficulty of factoring should be careful not to leak bits of the prime factors of the modulus. This holds both for Okamoto numbers and for RSA numbers.

As the exponent k in P^kQ grows, lattice-based methods quickly become relevant. Therefore careful evaluation should precede any cryptographic application of numbers of this form with k greater than 2.

5 Secure lengths for RSA and for Okamoto moduli

The problem here is to estimate the relative sizes of RSA and Okamoto moduli which would provide similar levels of security. The fastest known algorithm for factoring RSA numbers is NFS. In the case of an Okamoto number P^2Q, it turns out that computing Jacobi symbols $\left(\frac{x}{Q}\right)$ is feasible. Being able to do the same for RSA numbers would violate the well-known Quadratic Residuosity Assumption, and therefore would be an unlikely development. This small advantage can be exploited to speed up the elliptic curve algorithm in the case of Okamoto numbers. With the speed-up, the EC algorithm has running time approximately

$$2^{0.981*\sqrt{n\ln(0.231n)}-1.44\ln(n)}.$$

A joint plot of the exponent of this expression with the exponent

$$2.428n^{1/3}\ln(0.693n)^{2/3}$$

of the expression for the running time of NFS on RSA numbers shows that to achieve comparable levels of security, Okamoto numbers must be between 500 and 600 bits longer than RSA numbers. This holds throughout the range of sizes of interest to cryptographic applications.[1] In particular, our stated security

[1] See attached plot.

goal of 2^{85} operations is achieved with either 1000-bit RSA numbers or 1600-bit Okamoto numbers.

It is widely believed, although it has never been proven, that the RSA cryptosystem and it's various applications are secure if factoring RSA numbers is hard. The equivalent statement regarding the digital signature system ESIGN is that it is secure if factoring Okamoto numbers is hard. We see no reason to doubt either of these commonly held beliefs. It is highly unlikely that RSA will be broken without factoring RSA integers or that ESIGN will be broken without factoring Okamoto numbers.