Sixth International Workshop on Optimal Codes and Related Topics
June 16-22, 2009, Varna, Bulgaria *pp. 157-162*

Hiding information in products of integers

Constantinos Patsakis, Nikolaos Alexandris

{kpatsak, alexandr}@unipi.gr
Department of Informatics, University of Piraeus
Karaoli & Dimitriou 80, Piraeus, 185 34 Greece

Abstract. In this work we present a new SETUP for public key encryption algorithms that use products as part of their keys. Using the decomposition of an integer as a sum of four squares, enables us to create a backdoor on the products that if properly used, may lead their polynomial time factorization.

1 Introduction

Public key cryptography is based on hard mathematical problems. One of the most widely used problems is integer factorisation, which means decomposing a large integer n into a product prime numbers. While it is not yet known in which complexity class this problem belongs to, it is known to belong in both NP and co-NP classes. The best algorithms for integer factorisation are Pollard's rho algorithm, the General and the Special number field sieves, the continued fraction algorithm, the Quadratic sieve and the elliptic curve method. In table 1 we give the theoretical complexity of these algorithms.

Method	Complexity
General field sieve	$O(exp(\frac{64}{9}N)^{\frac{1}{3}}(\log N)^{\frac{2}{3}})$
Quadratic sieve	$O(exp(N^{\frac{1}{2}}(\log N)^{\frac{1}{2}})$
Elliptic curve	$O(exp((1+o(1))\sqrt{(\ln p \ln\ln p)}))$
Pollard rho	$O(exp(N^{\frac{1}{4}}(\log N)^{\frac{1}{2}})$

Table 1: Factoring methods and their complexity, N represents the number of bits of the integer n we want to factor

Based on the hardness of factoring, many public key encryption algorithms have been developed, like RSA [14], Rabin [16] and others. In order to find an integer that cannot be factored trivially or that the methods above cannot factor the integer fast enough, we pick two large prime numbers, p and q and compute their product n. To make it more secure, we use p and q for which $p < q < 2p$, with $q - p > 2^{60}$, so that Fermat factoring method cannot work.

In most cases, we are given the integer n as well as the other parameters from a black box, an application that we don't know how it works and we

seem to trust.Yet, the creator of this black box may want to offer security to all that use this black box, but in the same time be able to decrypt all the encrypted messages of users without their knowledge or approval. We can imagine a company that offers secure communication to its employees, but on the same time wants to see that important documents or information do not leak outside. We call these mechanisms SETUPs, which stands for Secretly Embedded Trapdoor with Universal Protection. Some well known SETUPs are Anderson's [11], Young and Yung SETUP [12] and Crepeau's and Slakmon's hidden prime factor method [13].

In this work we present a new SETUP which can be used in order to create integers n which are hard to factor, yet the generator of these integers has parametrized their creation so that the decomposition of n as a sum of four squares can be used factor integer n in feasible time. The trapdoor that is created can be only be traced and exploited by the authority that created these integers. The following section contains the mathematical background needed for the algorithm to be explained.

2 Background

The algorithm is based on a Lagrange's theorem [2, 4] and [7].

Theorem 1 (Four square theorem) *Every integer n can be written as a sum of the squares of four integers.*

Example 1 $15 = 3^2 + 2^2 + 1^2 + 1^2$, $5413654687 = 73099^2 + 8378^2 + 1^2 + 1^2$.

The Euler's identity:

Theorem 2 *For every* $x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4$ *we have:*

$$(x_1^2 + x_2^2 + x_3^2 + x_4^2)(y_1^2 + y_2^2 + y_3^2 + y_4^2) = a^2 + b^2 + c^2 + d^2,$$

where:

$$\begin{aligned} a &= x_1y_1 + x_2y_2 + x_3y_3 + x_4y_4, \\ b &= x_1y_2 - x_2y_1 + x_3y_4 - x_4y_3, \\ c &= x_1y_3 - x_3y_1 + x_4y_2 - x_2y_4, \\ d &= x_1y_4 - x_4y_1 + x_2y_3 - x_3y_2. \end{aligned}$$

Using a Hardy's conjecture, proved by Bateman [8] and later improved by Landau [9, 3], we have that

Theorem 3 *Let $r_3(n)$ represent the number of representations of an integer n as a sum of three squares then*

$$\sum_{n\leq x} r_3(n) \sim \frac{4}{3}\pi x^{\frac{3}{2}}.$$

From the theorem above we have:

Theorem 4 *Let $r_3(n)$ represent the number of representations of an integer n as a sum of three squares then*

$$r_3(n) \cong 2\pi n^{\frac{1}{2}}.$$

Proof Let $F(x) = \sum_{n\leq x} r_3(n)$ then :

$$F(x+1) - F(x) = \sum_{n\leq x+1} r_3(n) - \sum_{n\leq x} r_3(n) = r_3(x+1).$$

Moreover we have that:

$$\begin{aligned}
F(x+1) - F(x) &= \sum_{n\leq x+1} r_3(n) - \sum_{n\leq x} r_3(n) \\
&= \frac{4}{3}\pi(x+1)^{\frac{3}{2}} - \frac{4}{3}\pi x^{\frac{3}{2}} = \frac{4}{3}\pi((x+1)^{\frac{3}{2}} - x^{\frac{3}{2}}) \\
&= \frac{4}{3}\pi(\frac{(x+1)^{\frac{3}{2}} - x^{\frac{3}{2}}}{(x+1)^{\frac{3}{2}} + x^{\frac{3}{2}}}((x+1)^{\frac{3}{2}} + x^{\frac{3}{2}})) \\
&= \frac{4}{3}\pi\frac{(x+1)^3 - x^3}{(x+1)^{\frac{3}{2}} + x^{\frac{3}{2}}} \\
&= \frac{4}{3}\pi\frac{x^3 + 3x^2 + 3x + 1 - x^3}{(x+1)^{\frac{3}{2}} + x^{\frac{3}{2}}} \\
&= \frac{4}{3}\pi\frac{3x^2 + 3x + 1}{(x+1)^{\frac{3}{2}} + x^{\frac{3}{2}}}.
\end{aligned}$$

For big n we have that $3x^2+3x+1 \cong 3(x+1)^2$ and $(x+1)^{\frac{3}{2}}+x^{\frac{3}{2}} \cong 2(x+1)^{\frac{3}{2}}$,

$$F(x+1) - F(x) \cong \frac{4}{3}\pi\frac{3(x+1)^2}{2(x+1)^{\frac{3}{2}}}$$

$$= 2\pi(x+1)^{\frac{1}{2}} = r_3(x+1).$$

Thus

$$r_3(n) \cong 2\pi n^{\frac{1}{2}}.$$

For more reading on sums of squares of integers, the reader may refer to [15, 4, 5].

3 The algorithm

The main idea of the algorithm is to find primes p and q in order to have their product easy factorable by the creator of the blackbox using Lagrange four squares theorem and Euler's identity. To achieve this, we create a keyed algorithm for the calculation of the product $n = pq$.

Our black box selects p, q so that integer n has the security properties referred above. From the four squares theorem we have that n can be written as a sum of squares of four integers a, b, c and d, so

$$n = a^2 + b^2 + c^2 + d^2.$$

Thanks to Rabin and Shallit [10] we have two randomized algorithms for decomposing integers as sum of squares, the first with complexity of $O(\log^2 n)$ for decomposing integers as sum of two squares and the second one with complexity $O(\log^2 n \log\log n)$ for decomposing integers as sum of four squares. Using these algorithms, instead of picking random integers in a given interval, we use pseudo-random sequences, with initial values $K = (k_1, k_2, k_3)$ for the three pseudo-random number generators needed. This way, we create a keyed algorithm for the decomposition of an integer as sum of four squares which we call RS_K, figure 1.

We can now form the following set of equations:

$$\begin{aligned}
&(x_1^2 + x_2^2 + x_3^2 + x_4^2)(y_1^2 + y_2^2 + y_3^2 + y_4^2) = a^2 + b^2 + c^2 + d^2,\\
&a = x_1y_1 + x_1y_1 + x_1y_1 + x_1y_1,\\
&b = x_1y_2 - x_2y_1 + x_3y_4 - x_4y_3,\\
&c = x_1y_3 - x_3y_1 + x_4y_2 + x_2y_4,\\
&d = x_1y_4 - x_4y_1 + x_2y_3 - x_3y_2,\\
&p = x_1^2 + x_2^2 + x_3^2 + x_4^2,\\
&q = y_1^2 + y_2^2 + y_3^2 + y_4^2.
\end{aligned}$$

We set a base $B = (b_1, b_2, b_3)$ which determines the maximum values that variables x_1, x_2 and y_1 can have.

In order to get a, b, c and d we use our keyed algorithm RS_K, taking every possible value of x_1, x_2 and y_1 from base B and solve the remaining equations for x_3, x_4, y_2, y_3 and y_4. The generated system of equations can easily be solved using a mathematical application like Mathematica or Matlab. The set of possible solutions has been omitted due to lack of space. In case that the equations do not meet our constrains, we ask from our black box a new pair of p and q repeat the algorithm until a pair that fulfills our constrains is found.

$$\begin{array}{c} (k_1, k_2, k_3) \\ \downarrow \\ (b_1, b_2, b_3) \rightarrow \boxed{RS_k} \rightarrow n \end{array}$$

Figure 1. Keyed Rabin-Shellit algorithm

We should note that there are enough integers that meet our constrains so that this keying is possible. For example if we set $0 \leq y_1 \leq 100$ then $p - 100^2 \cong p$ and from the second theorem, there are about $100 * 2\pi\sqrt{p}$ such representations, because we would have to write $p - 100^2 \cong p$ as a sum of three squares

$$q = y_1^2 + y_2^2 + y_3^2 + y_4^2$$

for which we have $n = pq$.

In most applications, a base B close to (100,1000,100) which involves in worst case scenario 10.000.000 tests, can be considered in most cases an affordable calculation cost with small cost of time as well. The proposed base suggests that p can be expressed as a sum of a square of an integer of at most 100 and a square of an integer of at most 1000, while in it's sum of squares representation has an integer of at most 100, restrictions that can be easily be fulfilled.

Example 2 126545211218217382173812712131245131212121265474818 1
$= 355520819784723035243945672 + 1225389398582622803962216^2 + 6^2 + 0^2$

4 Conclusion

The proposed SETUP is mainly based on running the keyed Rabin's and Shallit's algorithm, so it runs in polynomial time $O(log^2 n\ loglogn)$. Furthermore, the proposed SETUP creates integers that don't seem to follow a specific pattern, the distribution of n, p, q is indistinguishable from an honest one, while no embedding of bits of a prime happen as in [13] while there is no need to interfere with the other parameters of the encryption algorithm. The fact that the algorithm is keyed through sets K and B, gives it the flexibility to be adopted by many entities, maintaining the security of users and giving them the ability to decrypt their messages. It should be noted that the only way for the trapdoor to be traced is only by knowing both the base B and the set of initial values K which produced a, b, c and d, so the the SETUP cannot be reversed engineered by a third party or even be detected.

The only drawback of this scheme seems to be the set of extensive equations that are needed to be stored for the calculation of $x_3,\ x_4,\ y_2,\ y_3$ and y_4. From one point of view this may slow down the whole process of creating n, yet this cost can be reduced using proper precalculation in the black box.

References

[1] H. Davenport, *Analytic Methods for Diophantine Equations and Diophantine inequalities*, University of Michigan.

[2] G. H. Hardy, E. M. Wright, *An Introduction to the Theory of Numbers*, Fifth Edition, Oxford University Press, 1998.

[3] E. Landau, *Elementary Number Theory*, Chelsea Publishing Company, 1958.

[4] G. H. Hardy, On the expression of a number as the sum of two squares, *Quart. J. Math* 46, 1915, 263-283.

[5] E. Grosswald, *Representations of Integers as Sums of Squares*, Springer-Verlag, 1985.

[6] T. Estermann, On the representation of a number as a sum of squares, *Acta Arithmetica*, 2, 1936, 47-49.

[7] L. E. Dickson, *History of the Theory of Numbers, Vol. 2: Diophantine Analysis*, Dover, 2005.

[8] P. T. Bateman, On the representations of a number as the sum of three squares, *Trans. Amer. Math. Soc.* 71, 1951, 70-101.

[9] E. Landau, *Collected Works*, Thales-Verlag, 1986.

[10] M. O. Rabin , J. O. Shallit, Randomized algorithms in number theory, *Commun. Pure Appl. Math.* 39, 1985, S239 - S256.

[11] R. J. Anderson, A practical RSA trapdoor, *Electr. Lett.* 29, 1993, 995

[12] A. Young, M. Yung, *Malicious Cryptography, Exposing Cryptovirology*, Wiley, 2004.

[13] C. Crepeau, A. Slakmon, Simple backdoors for RSA key generation, *Cryptographers Track* at RSA Conference, volume LNCS 2612, 2003, 403- 416.

[14] R. L. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Commun. ACM* 21, 1978, 120-126.

[15] C. J. Moreno, S. S. Wagstaff Jr, *Sums of Squares of Integers*, Chapman & Hall/CRC, 2005

[16] M. Rabin, Digitalized signatures and public-key functions as intractable as factorization, MIT Laboratory for Computer Science, Jan. 1979.